MM6 Lagrange Interpolation Method

- kl.8.15-9.00 review of MM5 and some examples
- kl.9.00 – 10.30 exercise (see notes)
- kl.10.40-11.30 MM6 lecture (I)

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

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

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

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

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

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

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

Secant Method (MM5)

- 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

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

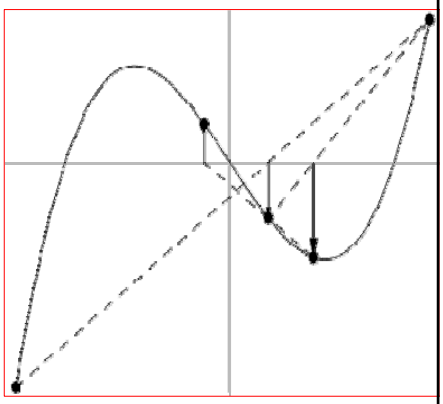
$$f(x_2)=0$$

$$\Downarrow$$

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

$$\Downarrow$$

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



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

Matlab Codes of Secant Method

```
function [y,its]=secant(f,x0,x1,tol)
format long
f0=feval(f,x0);
f1=feval(f,x1);
maxit=100;
its =0;
while (abs(x0-x1)>tol) & (its <maxit)
   x2=x1-(x1-x0)*f1/(f1-f0);
   x0=x1;
   x1=x2;
   f0=feval(f,x0);
   f1=feval(f,x1);
    its=its+1;

end

y=x1;
```

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

```
function y=example10(x);

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

```
>> [y,its]=secant(@example10,1,2,10e-6)

y =   1.256431208621747


its =     7
```

2 Equations in 2 Unknowns Using Newton's Method (MM5)

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

$$\Downarrow$$

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

set

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

$$\Downarrow$$

$$\begin{cases} h=\dfrac{-f_1f_{2y}+f_2f_{1y}}{f_{1x}f_{2y}-f_{1y}f_{2x}} \\ k=\dfrac{f_1f_{2x}-f_2f_{1x}}{f_{1x}f_{2y}-f_{1y}f_{2x}} \end{cases} \Rightarrow \begin{cases} x_{n+1}=x_n+h \\ y_{n+1}=y_n+k \end{cases}, \quad n=1,2,\cdots$$

Jacobian matrix:

$$\begin{bmatrix} \dfrac{\partial f_1(x,y)}{\partial x} & \dfrac{\partial f_1(x,y)}{\partial y} \\ \dfrac{\partial f_2(x,y)}{\partial x} & \dfrac{\partial f_2(x,y)}{\partial y} \end{bmatrix} = \begin{bmatrix} f_{1x} & f_{1y} \\ f_{2x} & f_{2y} \end{bmatrix}$$

Matlab Codes for Newton's Method (2 unknown, 2 equations)

```
function [sol,its]=newton2(fcn,Jac,v0,tol)

x0=v0(1); y0=v0(2);
f=feval(fcn,v0);
f1=f(1);
f2=f(2);
J=feval(Jac,v0);
f1x=J(1,1); f1y=J(1,2); f2x=J(2,1); f2y=J(2,2);
h=(-f1*f2y + f2*f1y)/(f1x*f2y - f1y*f2x);
k=(f1*f2x - f2*f1x)/(f1x*f2y - f1y*f2x);
x1=x0+h; y1=y0+k;
v1=[x1; y1];
its=1;

while abs(x1-x0)>tol | abs(y1-y0)>tol
   v0=v1;    x0=v0(1); y0=v0(2);
   f=feval(fcn,v0);
   f1=f(1); f2=f(2);
   J=feval(Jac,v0);
   f1x=J(1,1); f1y=J(1,2);f2x=J(2,1);f2y=J(2,2);
   h=(-f1*f2y + f2*f1y)/(f1x*f2y - f1y*f2x);
   k=(f1*f2x - f2*f1x)/(f1x*f2y - f1y*f2x);
   v1=[x0+h; y0+k];
   its=its+1;
end
sol=v1; its=its;
```

```
function f=eq2(v)
% here both f and v are vectors
x=v(1); y=v(2);
f(1)=4*x^2 + y^2-4;
f(2)=x^2*y^3-1;
```

```
function J=Deq2(v)
% Jacobian matrix for eq2.m
x=v(1); y=v(2);
J(1,1)=8*x; J(1,2)=2*y;
J(2,1)=2*x*y^3;
J(2,2)=3*x^2*y^2;
```

```
>> [sol,its]=newton2(@eq2,@Deq2,[0.8,1.2],10e-8)
sol =
   0.821684259596287
   1.139889510230440
its =     2
>> [sol,its]=newton2(@eq2,@Deq2,[0.4,1.8],10e-8)
sol =
   0.404149564420627
   1.829386038547648
its =     2
```

Exercises (MM5)

Question One:

Consider the same equation as we used in MM3 and MM4 Exercise One, i.e.,

$$3x^3 - 5x^2 - 4x + 4 = 0 \tag{1}$$

- Create your m-file to obtain the solution of the above equation located within the interval $[0, 1]$, using secant method with tolerance 10^{-6};
- How many iterations would be needed to obtain this solution? Compare the result with that of Newton's method in Exercise MM4.

Question Two:

Consider the following two equations

$$\begin{aligned} 4x^2 + y^2 &= 4 \\ x^2 y^3 &= 1, \end{aligned} \tag{2}$$

- Find the coordinates of the intersections in the second quadrant of curves described by above equations using 2 unknown parameters Newton's Method.

MM6 Lagrange Interpolation Method

Reading material: section 4.1 and 4.2 in Textbook

Introduction to Interpolation

- **Motivation:**

  Approximate evaluation of a function which is known only by its values at a set of data points

- **Interpolation:**

  the process of fitting a function of a particular nature (typically a polynomial, or a collection of polynomials) through the data.

- **Calssification:**
  - Polynomial interpolation, C
  - Cubic spline approximation,
  - Least square approximation

Polynomial Interpolation

- Any continuous function can be approximated to any required accuracy by a polynomial (Weierstrass Theorem)
- Polynomials are the only functions, which we can, theoretically at least, evaluate exactly
- Efficient evaluation of a polynomial (Horner's Rule)

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$
$$p(x) = \{[(a_n x + a_{n-1})x + a_{n-2}]x + \cdots + a_1\}x + a_0$$

Lagrange Interpolation

- **Problem formulation:**

  Suppose we are given the values of a function *f* at *N+1* distinct points (nodes), we wish to find a polynomial *p* with minimum degree such as

  p(x_k)=f(x_k), k=0, 1,2 ,…N

$$p(x) = a_m x^m + a_{m-1} x^{m-1} + \cdots + a_1 x + a_0$$

$$\begin{cases} a_m x_0^{\,m} + a_{m-1} x_0^{\,m-1} + \cdots + a_1 x_0 + a_0 = f(x_0) \\ a_m x_1^{\,m} + a_{m-1} x_1^{\,m-1} + \cdots + a_1 x_1 + a_0 = f(x_1) \\ \vdots \\ a_m x_N^{\,m} + a_{m-1} x_N^{\,m-1} + \cdots + a_1 x_N + a_0 = f(x_N) \end{cases} \Leftrightarrow \quad AX = F$$

$$Vandermonde \quad \det er \min ant :$$

$$V = \begin{vmatrix} 1 & x_0 & \cdots & x_0^{\,N} \\ 1 & x_1 & \cdots & x_1^{\,N} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & \cdots & x_N^{\,N} \end{vmatrix}$$

Lagrange Polynomial

If we can find a number of polynomials $l_j(x)$ $(j = 0,1,...N)$ of degree at most N such that

$$l_j(x_k) = \delta_{jk} = \begin{cases} 1 & if \quad j = k \\ 0 & if \quad j \neq k \end{cases}$$

then the Lagrange polynomial is defined as

$$p(x) = \sum_{j=0}^{N} f(x_j) l_j(x)$$

The Lagrange basis polynomials $l_j(x)$ $(j = 0,1,...N)$ are defined as

$$l_j(x) = \frac{(x - x_0)\cdots(x - x_{j-1})(x - x_{j+1})\cdots(x - x_N)}{(x_j - x_0)\cdots(x_j - x_{j-1})(x_j - x_{j+1})\cdots(x_j - x_N)}$$