Error:

Abs. $|\vec{x}-\vec{x}|$

relative $\frac{|\vec{x}-\vec{x}|}{|\vec{x}|}$

LU:

$A\cdot\vec{x}=\vec{b}$

$L\cdot\vec{C}=\vec{b}$ (Forward Sub.)

$U\cdot\vec{x}=\vec{C}$ (backward sub.)

$L\cdot U=A$

$L\cdot U\cdot\vec{x}=L\cdot\vec{C}$

Iterative methods:

$x_i^{(k+1)}=\frac{b_i-\sum_{j=1,j\neq i}^{n} a_{i,j}\,x_j^{k}}{a_{i,i}}$

Parial pivoting: If $|a_{i,i}|<|a_{j,i}|$ then switch our rows i & j

Diagonally dominant: $|a_{i,i}|\geq\sum_{j=1,j\neq i}^{n} a_{i,j}$ For all rows, & at least one row must be > the sum.

Interpolation

Lagrange: $\sum_{i=0}^{n}\left[Y_i\cdot\prod_{\substack{j=0\\ j\neq i}}^{n}\frac{x-x_j}{x_i-x_j}\right]$

Newton: $P(x)=\sum_{i=0}^{n}\left[a_i\prod_{j=0}^{i-1}(x-x_j)\right]$

$P(x)=a_0+a_1(x-x_0)+\cdots$

Horner's Algorithm:

$P=a_n$

for $k=n-1:-1:0$

$P=a_k+x\cdot P$

end

Differentiation via Lagrange:

$f_n^{(m)}\approx\frac{d^m}{dx^m}\sum_{i=0}^{n} Y_i\prod_{\substack{j=0\\ j\neq i}}^{n}\frac{x-x_j}{x_i-x_j}$

for $n=m\Rightarrow=\sum_{i=0}^{n} y_i\frac{(-1)^{n-i}}{h^n}\binom{n}{i}$

$\binom{n}{k}=\frac{n!}{(n-k)!\,k!}$

$h=\frac{b-a}{n}$

$f'(x_0)$ Forward diff. $\rightarrow f'(a)\approx\frac{f(a+h)-f(a)}{h}$

$f'(x)$ Centered diff. $\rightarrow f'(a)\approx\frac{f(a+h)-f(a-h)}{2h}$

$f'(x_i)$ backward diff. $\rightarrow f'(a)\approx\frac{f(a-h)-f(a)}{-h}$

Via Taylor:

$f(x)=f(a)+f'(a)(x-a)+\frac{f''(a)}{2!}(x-a)^2+\frac{f'''(a)}{3!}(x-a)^3+\cdots=\sum_{n=0}^{\infty}\frac{f^{n}(a)}{n!}(x-a)^n$

$f'(a)\approx C_1 f(a)+C_2 f(a+h)+C_3 f(a+2h)$ $\quad C_1=\frac{-3}{2h}$, $C_2=\frac{2}{h}$, $C_3=\frac{-1}{2h}$

$f(a+h)=f(a)+f'(a)\cdot h+\frac{f''(a)}{2!}h^2+\cdots$

$f(a+2h)=f(a)+f'(a)\,2h+\frac{f''(a)}{2!}(2h)^2+\cdots$

$\Rightarrow f'(a)\approx\frac{-3f(a)+4f(a+h)-f(a+2h)}{2h}$

Integration: Newton-Cotes:

$I=\int_a^b f(x)\,dx\approx\tilde{I}=\sum_{i=0}^{n} f(x_i)\cdot\int_a^b\prod_{\substack{j=0\\ j\neq i}}^{n}\frac{x-x_j}{x_i-x_j}\,dx$ $\quad x=hs+a$

$\Rightarrow\sum_{i=0}^{n} f(x_i)\cdot h\cdot\int_0^n\prod_{\substack{j=0\\ j\neq i}}^{n}\frac{s-j}{i-j}$

Simpson: (min: n=2)

$\tilde{I}(x)=\frac{h}{3}\left[f(x_0)+4f(x_{odd})+2f(x_{even})+f(x_n)\right]$

Trapezoidal:

Single: (min: n=1) $\Rightarrow\frac{h}{2}\left(f(a)+f(b)\right)$

Universal: $\frac{h}{2}\left[f(a)+f(b)+2\sum_{i=1}^{n+1} f(a+ih)\right]$ $\quad h=\frac{b-a}{n}$

Romberg:

$R(1,1)=$ Single Trapezoidal

$R(i,1)=$ Universal Trapezoidal $(n=2^{i-1})$

$R(i,j)=\frac{4^{j-1}\cdot R(i,j-1)-R(i-1,j-1)}{4^{j-1}-1}$

Non-linear Equ.: Newton's Method:

$X^{k+1}=X^k-\frac{f(x^k)}{f'(x^k)}$

$J(\vec{X}^k)\cdot\Delta\vec{X}^k=-\vec{f}(\vec{x}^k)$

$\begin{bmatrix}\frac{d}{dx}f_1(x,y) & \frac{d}{dy}f_1(x,y)\\ \frac{d}{dx}f_2(x,y) & \frac{d}{dy}f_2(x,y)\end{bmatrix}\cdot\begin{bmatrix}x^{k+1}-x^k\\ y^{k+1}-y^k\end{bmatrix}=\ominus\begin{bmatrix}f_1(x^k,y^k)\\ f_2(x^k,y^k)\end{bmatrix}$

Fixed Point:

Start $f(x)=0$ $\rightarrow x=g(x)$

LHS: x^{k+1}

RHS: x^k

Convergence: $|g'(x)|<1$ with our interval (a,b)

$\frac{|x^{k+1}-x^k|}{|x^{k+1}|}<\varepsilon$

$Y_{i+1}^{k+1}=Y_{i+1}^{k}-\frac{L(Y_{i+1}^{k})}{\hat{L}(Y_{i+1}^{k})}$

$L(Y_{i+1})=Y_{i+1}-Y_i-h\cdot f(x_{i+1},Y_{i+1})$

$\hat{L}(Y_{i+1})=1-h\cdot\frac{d}{dY_{i+1}}f(x_{i+1},Y_{i+1})$

Start Newton with $Y_{i+1}^{(0)}=Y_i$

Euler: Explicit:

$Y_{n+1}=Y_n+h\,f(x_n,Y_n)$

Forward method

Implicit: Backward method

$Y_{n+1}=Y_n+h\,f(x_{n+1},Y_{n+1})$

if too complex $\rightarrow$ use newton's

*IVP & h ✓

Heun Method:

$\tilde{y}_{n+1}=y_n+h\cdot f(x_n,y_n)$

$y_{n+1}=y_n+h\cdot\frac{f(x_n,y_n)+f(x_{n+1},\tilde{y}_{n+1})}{2}$

Super Slope

$x_{n+1}=x_n+h$

Runge-Kutta:

$S_1=f(x_n,y_n)$

$S_2=f(x_n+\frac{h}{2},y_n+S_1\cdot\frac{h}{2})$

$S_3=f(x_n+\frac{h}{2},y_n+S_2\cdot\frac{h}{2})$

$S_4=f(x_n+h,y_n+S_3\cdot h)$

$S=\frac{S_1+2S_2+2S_3+S_4}{6}$

Super Slope

If we have more than 1 ODE:

We need to have n num. of IVP (for each equation)

$\begin{cases}Y_1'=f_1\\ Y_2'=f_2\\ \vdots\\ Y_n'=f_n\end{cases}\Rightarrow\begin{cases}Y_{1,i+1}=Y_{1,i}+h\cdot f_1(x_i,y_{1,i},y_{2,i},\cdots,y_{n,i})\\ Y_{2,i+1}=Y_{2,i}+h\cdot f_2(x_i,y_{1,i},y_{2,i},\cdots,y_{n,i})\\ \vdots\\ Y_{n,i+1}=Y_{n,i}+h\cdot f_n(x_i,y_{1,i},y_{2,i},\cdots,y_{n,i})\end{cases}$

Differentiation via Taylor series:

$f'(a)\approx C_1 f(a)+C_2 f(a+h)+C_3 f(a+2h)$

$f'(a)\approx C_1 f(a)$

$+C_2\left[f(a)+f'(a)\cdot h+\frac{f''(a)}{2!}h^2+\frac{f'''(a)}{3!}h^3+\cdots\right]$

$+C_3\left[f(a)+f'(a)\cdot 2h+\frac{f''(a)}{2!}(2h)^2+\cdots\right]$

Stability: (of Implicit): $Y_{i+1}=Y_i+h\,f(x_{i+1},Y_{i+1})$

$y'=\lambda y$

$Y_{i+1}=Y_i+h(\lambda Y_{i+1})\Rightarrow(1-h\lambda)Y_{i+1}=Y_i\Rightarrow Y_{i+1}=\frac{1}{1-h\lambda}Y_i$

$Y_1=\frac{1}{1-h\lambda}Y_0$

$Y_2=\frac{1}{1-h\lambda}Y_1=\left(\frac{1}{1-h\lambda}\right)^2Y_0$

$\vdots$

$Y_n=\left(\frac{1}{1-h\lambda}\right)^n Y_0$

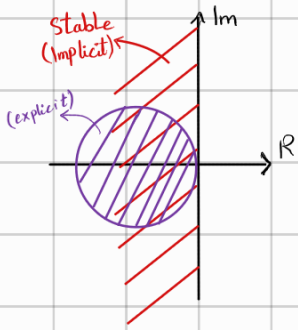
Stable if $\left|\frac{1}{1-h\lambda}\right|\leq 1$

$\Rightarrow h\lambda\leq 0$

$\Rightarrow h\geq\left|\frac{2}{\lambda}\right|$

Stable for $Re(\lambda)\leq 0$, $\lambda=a+ib$

$\left|\frac{1}{1-h\lambda}\right|=\frac{1}{|1-ha-hbi|}=\frac{1}{\sqrt{(1-ha)^2+(hb)^2}}$



$f'(a)\approx\underbrace{(C_1+C_2+C_3)}_{0}f(a)+\underbrace{(hC_2+2hC_3)}_{1}f'(a)+\underbrace{\left(\frac{C_2}{2!}h^2+\frac{C_3}{2!}4h^2\right)}_{0}f''(a)$

$+\left(\frac{C_2}{3!}h^3+\frac{C_3}{3!}8h^3\right)f'''(a)+\cdots$

Dominating term of the tyler series (if not equal to zero)

Need:

$C_1+C_2+C_3=0$

$h(C_2+2C_3)=1$

$h^2\left(\frac{C_2}{2}+2C_3\right)=0$

$\Rightarrow\left[\begin{array}{ccc|c}1&1&1&0\\0&1&2&1/h\\0&1&4&0\end{array}\right]\overset{III-II}{\Rightarrow}\left[\begin{array}{ccc|c}\textcircled{1}&1&1&0\\0&\textcircled{1}&2&1/h\\0&0&\textcircled{2}&-1/h\end{array}\right]$

$\rightarrow C_1=\frac{1}{h}$, $C_2=\frac{-1}{h}$, $C_3=\frac{-1}{2h}$

Stability of Explicit: $Y_{i+1}=Y_i+h\,f(x_i,Y_i)$

$Y_{i+1}=Y_i+h(\lambda Y_i)\Rightarrow Y_{i+1}=Y_i(1+h\lambda)$

$Y_1=Y_0(1+h\lambda)$

$Y_2=Y_1(1+h\lambda)=Y_0(1+h\lambda)^2$

$\vdots$

$Y_n=Y_0(1+\lambda h)^n$

Stable for $|1+\lambda h|\leq 1$

$\Rightarrow -1<1+\lambda h<1$

$-2<\lambda h<0$

$\frac{-2}{\lambda}<h<0$

PSeudCode: Gaus:

```
function X = gauss_Pivot(A,b)
  D = 1 % initizing D variable in case we need the det.
  n = Length(b)
  X = zero(n,1)
  % Forward elimination
  for K=1 to n-1 do
    % checking for larger pivits
    |Amax = A(K,K)
    |swap = K
    for i=K+1 to n do
      if |A(i,K)| > |Amax| then
        |Amax = A(i,K)
        |swap = i
      end if
    end for
    % Swap rows if necessary
    if swap ≠ K then
      |old_Pivot(1,:) = A(K,:)
      |old_b = b(K)
      |A(K,:) = A(swap,:)
      |A(swap,:) = old_Pivot(1,:)
      |b(K) = b(swap)
      |b(swap) = old_b
      |Sign_Pivot = -1
    else then
      Sign_Pivot = 1
    end if
    D = D * A(K,K) * Sign_Pivot % Update determinan
    % elemination:
    for i=K+1 to n do
      |m = A(i,K)/A(K,K)
      for j=K+1 to n do
        |A(i,j) = A(i,j) - m * A(K,j)
      end for
      |b(i) = b(i) - m * b(K)
    end for
  end for
  D = D * A(n,n)
  % Back substitution % last diagonal element
  |X(n) = b(n)/A(n,n)
  for i=n-1 downto 1 do
    |S = b(i)
    for j=i+1 to n do
      |S = S - A(i,j) * X(j)
    end for
    |X(i) = S/A(i,i)
  end for
end function
```

auxiliary variable swap

```
function [L,U] = get_LU_from_A(A) 2
  n = size(A,1)
  U = zeros(n)
  L = eye(n)
  for i=1 to n do
    for j=i to n do
      U(i,j) = A(i,j)
    end for
  end for
  for i=2 to n do
    for j=1 to i-1
      L(i,j) = A(i,j)
    end for
  end for
end function
```

```
function X = gauss_Seidel(A,b,X)
  % The input X is a vector of initial guess.
  n = length(b)
  eps: 10^-8 % Accepted tolerance
  Count = 0
  Count_max = 100
  err = norm(A*X-b) % residual error
  while err > eps do
    i=1
    S = b(i)
    for j=2 to n do
      S = S - A(i,j)*X(i)
    end for
    X(1) = S/A(i,i) % Here always i=1
    for i=2 to n-1 do
      S = b(i)
      for j=1 to i-1 do
        S = S - A(i,j)*X(j)
      end for
      for j=i+1 to n do
        S = S - A(i,j)*X(j)
      end for
      X(i) = S/A(i,i)
    end for
    i=n
    S = b(i)
    for j=1 to n-1 do
      S = S - A(i,j)*X(j)
    end for
    X(n) = S/A(i,i)
    Count = Count + 1
    if Count > Count_max do
      X = 0 % did not converge!
      break
    end if
    err = norm(A*X-b)
  end while
end function
```

```
function A = LU_decomp(A) 1
  % Forward elimination
  for K=1 to n-1 do
    for i=K+1 to n do
      m = A(i,K)/A(K,K)
      A(i,K) = m % store the entry for L in the lowest part of U (just where the zeros are).
      for j=K+1 to n do
        A(i,j) = A(i,j) - m * A(K,j)
      end for
    end for
  end for
end function
```

```
function X = LU_forward_back(L,U,b) 3
  n = Length(b)
  % Forward Substitution
  Y = zeros(n,1)
  Y(1) = b(1)
  for i=2 to n do
    Y(i) = b(i)
    for j=1 to i-1 do
      Y(i) = Y(i) - L(i,j) * Y(j)
    end for
  end for
  % Back Substitution
  X = zeros(n,1)
  X(n) = Y(n)/U(n,n)
  for i=n-1 downto 1 do
    S = Y(i)
    for j=i+1 to n do
      S = S - U(i,j) * X(j)
    end for
    X(i) = S/U(i,i)
  end for
end function
```

```
function A = newtonInterpol(X,f)
  % X (n-by-1) vector of horizontal positions
  % f (n-by-1) vector of function values
  % A (n-by-n) matrix, whose lower triangular matrix
  % is full of Newton-interpolation coefficients.
  n = Length(X)
  A = zeros(n)
  A(:,1) = f
  for i=2 to n do
    for j=i to n do
      A(j,i) = (A(j,i-1) - A(j-1,i-1)) / (X(j) - X(j+i-1))
    end for
  end for
end function
```

$R = @(x)([f_1(x_1,x_2); f_2(x_1,x_2)])$

$J = @(x)([\frac{d}{dx_1}f_1, \frac{d}{dx_2}f_1; \frac{d}{dx_1}f_2, \frac{d}{dx_2}f_2])$

$X = [0; 0]$

```
function X = newtonRaphson(R,J,X_init,err)
  % R: is an n column vector of the function
  % J: nxn Jacobian matrix
  % X_init: is an n column vector of the initial geusses
  % err: accepted error
  x = X_init
  while (norm(R(X)) > err) do
    delta = J(X) \ (-R(X))
    X = delta + X
  end while
end function
```

Explicit Euler:

```
Y = 0;  Y_hist = Y;
X = 0;  X_hist = X;
h = 0.001;
for i = 1:250
    Y = Y + h * Sin(X * cos(Y));   % f(X_n, Y_n)
    X = X + h;
    Y_hist(end+1) = Y;
    X_hist(end+1) = X;
end for
Plot(X_hist, Y_hist)
```