Derivative Recap

Derivative	Integral (Antiderivative)
$\frac{d}{dx}n=0$	$\int 0\,dx=C$
$\frac{d}{dx}x=1$	$\int 1\,dx=x+C$
$\frac{d}{dx}x^n=nx^{n-1}$	$\int x^n\,dx=\frac{x^{n+1}}{n+1}+C$
$\frac{d}{dx}e^x=e^x$	$\int e^x\,dx=e^x+C$
$\frac{d}{dx}\ln x=\frac{1}{x}$	$\int \frac{1}{x}\,dx=\ln x+C$
$\frac{d}{dx}n^x=n^x\ln x$	$\int n^x\,dx=\frac{n^x}{\ln n}+C$
$\frac{d}{dx}\sin x=\cos x$	$\int \cos x\,dx=\sin x+C$
$\frac{d}{dx}\cos x=-\sin x$	$\int \sin x\,dx=-\cos x+C$
$\frac{d}{dx}\tan x=\sec^2 x$	$\int \sec^2 x\,dx=\tan x+C$
$\frac{d}{dx}\cot x=-\csc^2 x$	$\int \csc^2 x\,dx=-\cot x+C$
$\frac{d}{dx}\sec x=\sec x\tan x$	$\int \tan x\sec x\,dx=\sec x+C$
$\frac{d}{dx}\csc x=-\csc x\cot x$	$\int \cot x\csc x\,dx=-\csc x+C$

Product/Quotient Rule

$$\frac{d}{dx}[f(x)g(x)]=f(x)g'(x)+g(x)f'(x)$$

$$\frac{d}{dx}\left[\frac{f(x)}{g(x)}\right]=\frac{g(x)f'(x)-f(x)g'(x)}{[g(x)]^2}$$

Find precision using digits

$\varepsilon s=5*10^{-t}$ t: sig. digits

We want to stop when $\varepsilon a<\varepsilon s$

Accuracy vs precision

Precision: how close it is to the previous one.

Accuracy: how close to true value

Sig Fig	Truncation	Rounding
1.23 = 3 s.f. 0.006 = 1 s.f.	3.45**6** = 3.45	3.45**6** = 3.46

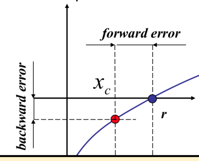
Error types

r: real root

x_c: approximation or r

backward: $|f(x_c)|$

Forward: $|r-x_c|$



Error definitions

Absolute error:

$$E_t=|\,true\,value-approximation\,|$$

True rel. error:

$$\varepsilon_t=\frac{|\,true\,value-approximation\,|}{|\,true\,value\,|}$$

Estimated rel. error

$$\varepsilon_a=\frac{|\,current\,approx-previous\,approx\,|}{|\,current\,approx\,|}$$

Taylor Serie

Method used to estimate a function at a certain point given that we know the value of a point of the function

a: point given

f(a) is known. We want f(x)

$$f(x)=f(a)+f'(a)(x-a)+\frac{f''(a)(x-a)^2}{2!}+...+\frac{f^{(n)}(a)(x-a)^n}{n!}$$

Multiple roots

A function has root multiplicity M if **$f(r)=0$, $f'(r)=0$, $f^{(M-1)}(r)=0$**, ..., **$f^{(M)}(r)\neq 0$**

Impact: Newton-Raphson becomes linearly convergent. Becomes challenging

Naive Gauss

Convert system to augmented matrix. Make the matrix upper triangular by using combinations of rows to form new ones. Once it's upper triangular, back propagate to find the point.

Gaus partial pivoting

Also called maximum pivoting. Pick the pivot with the highest absolute value. swap rows to put pivot to the top. make things below the pivot equal to 0 repeat for every pivot. Then solve.

Bisection algorithm

* function has to be continuous

bracketing around the root where we divide the search region by 2.

Stable, always convergent and error is under control

Absolute error $e\leq\left|\frac{x_{lower}-x_{upper}}{2}\right|$

$$c=\frac{a+b}{2}$$

1. find two points around the root such that **f(a) < 0 < f(b)**
2. Make a table with **a**, **b**, **f(a)**, **f(b)**, **c**, **f(c)** and $\boldsymbol{\varepsilon_a}$
3. refine brackets based on sign of **f(c)** (change upper or lower term)
4. when $\boldsymbol{\varepsilon_a<\varepsilon_s}$ the value of **c** is the root found

Fix point iteration

Iteration algorithm, not bracketing. **Does not always converge but is linearly convergent**

Depends on starting point and how we made **g(x)**.

g(x) = x rewrite **f(x)** in terms of **g(x)**

ex: **$f(x)=x^2-x-2 \Rightarrow x^2=x+2 \Rightarrow x=1+2/x \Rightarrow g(x_n)=1+2/x$**

1. rewrite f(x) in g(x) and guess a first point (X_0)
2. make a table with **n**, **$x_n=g(x_{n-1})$**, **g(x)** and $\boldsymbol{\varepsilon_a}$

To check converge:

x: root found

$$|\,g'(x)\,|<1$$

False position algorithm

Similar to bisection except **Stable, Converges, error under control**

a and **b** must be on both side of the root

make table with **n**, **a**, **b**, **f(a)**, **f(b)**, **c**, **f(c)** and $\boldsymbol{\varepsilon_a}$

$$c=\frac{af(b)-bf(a)}{f(b)-f(a)}$$

Secant method

Iterative. Not always convergent. superlinear convergence (faster than linear but slower than quadratic). No need to find f'(x)

1. find initial guess x_0 and x_1 close to root. no need to bracket the root
2. make table using **n**, **x_{n-1}**, **x_n**, **x_{n+1}**, **$f(x_{n-1})$**, **$f(x_n)$** and $\boldsymbol{\varepsilon_a}$

$$x_{n+1}=x_n-f(x_n)\frac{x_n-x_{n-1}}{f(x_n)-f(x_{n-1})}$$

Newton Raphson method

Iterative. not always convergent. Quadratic convergence

1. make initial guess x0
2. make table with **n**, **x_n**, **$f(x_n)$**, **$f'(x_n)$**, **x_{n+1}**, $\boldsymbol{\varepsilon_a}$

$$x_{n+1}=x_n-\frac{f(x_n)}{f'(x_n)}$$

Modified (used when multiplicity > 0)

$$x_{n+1}=\frac{x_n-f(x_n)f'(x_n)}{[f'(x_n)]^2-f(x_n)f''(x_n)}$$

A = LU decomposition

A= $\begin{bmatrix}1&2&4\\3&8&14\\2&6&13\end{bmatrix}$ L = $\begin{bmatrix}1&0&0\\?&1&0\\?&?&1\end{bmatrix}$

-3 R_1+R_2 -> R_{2New} $L_{2,1}$ = -(-3) = 3

A= $\begin{bmatrix}1&2&4\\0&2&2\\2&6&13\end{bmatrix}$ L = $\begin{bmatrix}1&0&0\\3&1&0\\?&?&1\end{bmatrix}$

-2 R_1+R_3 -> R_{3New} $L_{3,1}$ = -(-2) = 2

A= $\begin{bmatrix}1&2&4\\0&2&2\\0&2&5\end{bmatrix}$ L = $\begin{bmatrix}1&0&0\\?&1&0\\?&?&1\end{bmatrix}$

-1 R_1+R_3 -> R_{3New} $L_{3,2}$ = -(-1) = 1

U= $\begin{bmatrix}1&2&4\\0&2&2\\0&0&3\end{bmatrix}$ L = $\begin{bmatrix}1&0&0\\3&1&0\\2&1&1\end{bmatrix}$

Matrix condition number

$$cond(A)=||\,A\,||_\infty||\,A^{-1}\,||_\infty$$

Infinity norm: row in matrix with highest sum (absolute of every element).

well conditioned >1

ill-conditioned ≈ 0 and **identity = 1**

Matrix indices

$$\begin{bmatrix}a_{11}&a_{12}&a_{13}\\a_{21}&a_{22}&a_{23}\\a_{31}&a_{32}&a_{33}\end{bmatrix}$$

Matrix Determinant

+ a_{11} * (a_{22} * a_{33} - a_{32} * a_{23})

- a_{21} * (a_{12} * a_{33} - a_{32} * a_{13})

+ a_{31} * (a_{12} * a_{23} - a_{22} * a_{13})

Norms

Euclidean-norm: $||A||_E=\sqrt{\Sigma each^2}$

Column-norm: $||A||_1$ = col with max sum

P-norm: $||A||_p=\sqrt[p]{\Sigma each^2}$

Pivoting

$\begin{bmatrix}1&1&1\\7&1&1\\1&3&1\end{bmatrix}$ -> $\begin{bmatrix}7&1&1\\1&1&1\\1&3&1\end{bmatrix}$ [P] = $\begin{bmatrix}0&1&0\\1&0&0\\0&0&1\end{bmatrix}$

* avoid division by zero or ill condition

p is pivot matrix and now **Ax = Pb**

Max scaling=pivoting with max val on top

Solving with A = LU

X= $\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}$ Z= $\begin{bmatrix}z_1\\z_2\\z_3\end{bmatrix}$

A*X = b

1. find Z using L*Z = b
2. find X using U*X = Z

Matrix inverse using A = LU

A*X_1 = $\{I\}_1$

A*X_2 = $\{I\}_2$

A*X_3 = $\{I\}_3$

Solve them

$\begin{bmatrix}1\\0\\0\end{bmatrix}$ $\begin{bmatrix}0\\1\\0\end{bmatrix}$ $\begin{bmatrix}0\\0\\1\end{bmatrix}$

I_1 I_2 I_3

Group X_1, X_2, X_3 to form A^{-1}

A=LU decomposition with Crout

$$L=\begin{bmatrix}L_{11}&0&0\\L_{21}&L_{22}&0\\L_{31}&L_{32}&L_{33}\end{bmatrix}\quad U=\begin{bmatrix}1&U_{12}&U_{13}\\0&1&U_{23}\\0&0&1\end{bmatrix}$$

[a_{11} = L_{11}] [a_{12} = L_{11} * U_{12}] [a_{13} = L_{11}*U_{13}]

[a_{21} = L_{21}] [a_{22} = L_{21} * U_{12} + L_{22}] [a_{31} = L_{31}]

[a_{23} = L_{21} * U_{13} + L_{22} * U_{23}] [a_{32} = L_{31} * U_{12} + L_{32}] [a_{33} = L_{31} * U_{13} + L_{32} * U_{23} + L_{33}]

Jacobi (iterative method)

isolate x_1, x_2, x_3 & set initial guess ex{1,1,1}

put initial test in each equation. Find the value of the new x_1, x_2, x_3 Those will be used in the next iteration. Rel error = $\frac{||x^{(n+1)}-x^{(n)}||_2}{||x^{(n+1)}||_2}$

isolate x_1 in f_1(x).x_2 in f_2...

Gaus seidel (faster than jacobi)

Similar to jacobi but instead of solving all three variables, solve x_1 and use its value to find x_2 then use the new x_1, x_2 to find x_3

Make it diagonally dominant if possible

The rel error is calculated just like jacobi

Least square approximation

Ex: 4 points = > **n = 3 => biggest polynomial is n-1** = 2 order => **$P(x)=a_0+a_1x+a_2x^2$**

Use the value of the points to fill in the values in the following matrix. Then solve using the method of your choice. This will give the coefs. to write polynomial approx

$$\begin{bmatrix}n+1&\Sigma x&\Sigma x^2\\\Sigma x&\Sigma x^2&\Sigma x^3\\\Sigma x^2&\Sigma x^3&\Sigma x^4\end{bmatrix}\begin{bmatrix}a_0\\a_1\\a_2\end{bmatrix}=\begin{bmatrix}\Sigma y\\\Sigma xy\\\Sigma x^2y\end{bmatrix}$$

Total Error (Square residual): $E=S_r=\Sigma[f(x)-P(x)]^2$

use every points to calculate. You can find error at a point too

$S_{y/x}=\sqrt{\frac{S_r}{n-1}}$

Standard error of estimate

ȳ = average value of y

$$S_t=\Sigma(y_i-\bar{y})^2$$

coef of determination $r^2=\frac{S_t-S_r}{S_t}$

Perfect fit Sr = 0 and r^2 = 1. Sr should reduce when increasing order

Normal Equation

Use to solved inconsistent system of equations (6 equations,3 vars). Use a model eg. Poly 2nd order. find the value a_0, a_1, a_2 for all point and make the matrix [A] = (6row, 3 col).

{x} = {a_0, a_1, a_2} and b is a 1 col 6 row matrix with all the values of y. Transpose A to have A^T (a_{12} becomes a_{21}). $A^TA\bar{x}=A^Tb$ (x bar is to identify that it's approximation. A^TA will give a 3x3 and b will now be a 3 row 1 col. You can solve regularly after that

Divided Difference Interpolation (equally spaced points)

x_0 -> **f[x_0]**   **f[x_1 x_0]**   **f[x_2 x_1 x_0]**

x_1 -> f[x1]   f[x_2 x_1]

x_2 -> f[x2]

Interpolation method that can be use up to the n^{th} order (nb points -1).

n^{th} otder polynomial: $f_n(x) = b_0 + b_1(x-x_0) + b_2(x-x_0)(x-x_1)$

$f[x_0] = b_0$ $f[x_1 x_0] = b_1$ $f[x_2x_1x_0] = b_2$ $f[x_1x_0] = (f[x_1]-f[x_0])/(x_1-x_0)$

$f[x_2x_1x_0] = (f[x_2x_1] - f[x_1x_0]) / (x_2 - x_0)$ **Only use biggest and smallest x_i for denominator**

Trapezoidal rule for numerical integration O(h^3)

h: step size. **n**: number of interval. **[a,b]**: limits of integral. I_n: numerical integration **h = (b-a)/n**

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

A smaller step will lead to smaller error

Single trapezoid: I=(b-a)*(f(a)+f(b))/2

Simpson 1/3 O(h^4) and 3/8 standard O(h^4)

1/3 Three points needed

$$I = \frac{b-a}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right]$$

3/8. Four points needed

$$I = \frac{b-a}{8}\left[f(a) + 3f\left(\frac{2a+b}{3}\right) + 3f\left(\frac{a+2b}{3}\right) + f(b)\right]$$

Gauss Quadrature (2 points) weighted coefficients

$$\int_a^b f(x)dx = \sum_{i=0}^{n} c_i f(x_i)$$

c_i and x_i are values from table. Table is usually from -1 to 1 so we need to change bounds from [a,b] to [-1,1]. **x = m*t+r**

m = (b-a)/2   **r = (b+a)/2**   **Ex**: 8≤x ≤ 30. with the change x = 11t+ 19 and **dx = m dt = 11dt** substitute x everywhere and dx so now f(x)dx [8,30] = f(11t+19)(11)dt [-1,1]. **N.B.** If the bounds were changed, you need to tweak the summation. It would now be 11 ∑ c_i* f(11x_i + 19)

Gerschgorin Circle theorem (find location of eigenvalues using circles)

$$\sum_{\substack{j=1\\ j\neq k}}^{n} |a_{kj}|$$

add all elements on a row except diagonal. Total is **radius** and position of center of circle is the diagonal element on the row. Place it on a line. The union of all circles is the area where the eigenvalues can be found.

Characteristic equation to find eigenvalues

$$\begin{bmatrix} -4 & 14 & 0\\ -5 & 13 & 0\\ -1 & 0 & 2\end{bmatrix} \rightarrow \begin{bmatrix} -4-\lambda & 14 & 0\\ -5 & 13-\lambda & 0\\ -1 & 0 & 2-\lambda\end{bmatrix}$$

Switch diagonal to be **(value)-λ.** Calculate determinant of new matrix and make it = 0. Solve for **λ**. All values of **λ** are eigen values

Find eigenvector from eigenvalue

Ex you found the eigenvalue **λ=2.387426** you put it in the matrix, Set one of the x equal to 1 and solve for others

$$\begin{bmatrix} 2-2.387426 & -1 & 0\\ -1/2 & 1-2.387426 & -1/2\\ 0 & -1/3 & -2/3-2.387426\end{bmatrix}\begin{bmatrix} x_a\\ x_b\\ x_c\end{bmatrix} = \begin{bmatrix}0\\0\\0\end{bmatrix}$$

System of non-linear equations

1. Start with n equations with x_1 to x_n

$$\left\{\begin{matrix} 12x_1 - 3x_2^2 - 4x_3 = 7.17\\ x_1^2 + 10x_2 - x_3 = 11.54\\ x_2^3 + 7x_3 = 7.631\end{matrix}\right\} \rightarrow$$

2. Make them = 0

$$f = \begin{bmatrix} 12x_1 - 3x_2^2 - 4x_3 - 7.17\\ x_1^2 + 10x_2 - x_3 - 11.54\\ x_2^3 + 7x_3 - 7.631\end{bmatrix}$$

Jacobian

$$J_{ij} = \frac{\partial f_i(x)}{\partial x_j}$$

3. Create the jacobian

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_n}\\ \frac{\partial f_2}{\partial x_1} & \ddots & \frac{\partial f_2}{\partial x_n}\\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \frac{\partial f_n}{\partial x_n}\end{bmatrix} \quad J = \begin{bmatrix} 12 & -6x_2 & -4\\ 2x_1 & 10 & -1\\ 0 & 3x_2^2 & 7\end{bmatrix}$$

4. Find **f(x_0)** by plugging initial guess **x_0** in **f**
5. Find **J(x_0)** by plugging initial guess **x_0** in **J**

$$x_0 = \begin{bmatrix}1\\1\\1\end{bmatrix} \quad f(x_0) = \begin{bmatrix}-2.17\\ -.54\\ 0.369\end{bmatrix} \quad J(x_0) = \begin{bmatrix} 12 & -6 & -4\\ 2 & 10 & -1\\ 0 & 3 & 7\end{bmatrix}$$

6. Find **[J(x_0)]$^{-1}$** using the cofactor method

$$A = \begin{bmatrix} a & b & c\\ d & e & f\\ g & h & i\end{bmatrix}$$

$$cofactor(A) = \begin{bmatrix} \begin{vmatrix} e & f\\ h & i\end{vmatrix} & -\begin{vmatrix} d & f\\ g & i\end{vmatrix} & \begin{vmatrix} d & e\\ g & h\end{vmatrix}\\ -\begin{vmatrix} b & c\\ h & i\end{vmatrix} & \begin{vmatrix} a & c\\ g & i\end{vmatrix} & -\begin{vmatrix} a & b\\ g & h\end{vmatrix}\\ \begin{vmatrix} b & c\\ e & f\end{vmatrix} & -\begin{vmatrix} a & c\\ d & f\end{vmatrix} & \begin{vmatrix} a & b\\ d & e\end{vmatrix}\end{bmatrix}$$

[J(x_0)]$^{-1}$ = cofactor(J(x_o))/ det(J(x_0))

$$[J(x_0)]^{-1} = \begin{bmatrix} 0.078 & 0.0321 & 0.0491\\ -0.015 & 0.0897 & 0.0043\\ 0.0064 & -0.0385 & 0.1410\end{bmatrix}$$

7. Find x_{k+1} and iterate until precision is met

$$[x_{k+1}] = [x_k] - [J(x_k)]^{-1} f(x_k)$$

$$x_1 = \begin{bmatrix}1\\1\\1\end{bmatrix} - \begin{bmatrix} 0.078 & 0.0321 & 0.0491\\ -0.015 & 0.0897 & 0.0043\\ 0.0064 & -0.0385 & 0.1410\end{bmatrix}\begin{bmatrix}-2.17\\ -1.54\\ 0.369\end{bmatrix} = \begin{bmatrix}1.2005\\ 1.104\\ 0.9026\end{bmatrix}$$

8. Find rel. error

$$\frac{||x^{(n+1)} - x^{(n)}||_2}{||x^{(n+1)}||_2}$$

Rel error vectors (step 8)

x_1 = [a,b,c] & x_2 = [d,e,f]

$$\frac{\sqrt{(d-a)^2 + (e-b)^2 + (f-c)^2}}{\sqrt{(d)^2 + (e)^2 + (f)^2}}$$

matrix det.

$$\begin{bmatrix} + & - & +\\ - & + & -\\ + & - & +\end{bmatrix}$$

matrix product

$$\begin{bmatrix} a & b\\ c & d\end{bmatrix}\begin{bmatrix} e & f\\ g & h\end{bmatrix} = \begin{bmatrix} ae+bg & af+bh\\ ce+dg & cf+dh\end{bmatrix}$$

Errors (some definitions)

$\bar{f'} = max|f'(x)|$ **N.B. truncation error single trapezoid, use lagrange instead**

Simpson 1/3

$$\varepsilon_a = \frac{-(b-a)h^5}{90} f^5(x) = O(n^4)$$

Composite Simpson 1/3

$$\varepsilon_a = \frac{-(b-a)h^5}{180} f^5(x) = O(n^4)$$

Simpson 3/8

$$\varepsilon_a = \frac{-(b-a)h^4}{80} f^4(x) = O(n^4)$$

Composite 3/8

$$\varepsilon_a = \frac{-(b-a)h^4}{180} f^4(x) = O(n^4)$$

Lagrange Interpolation (not equally spaced points + satisfy data pts exactly)

$$P(x) = \sum_{i=0}^{n} L_i(x) f(x_i)$$

$$L_i(x) = \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{(x-x_j)}{(x_i-x_j)}$$

$$P_2(x) = L_0(x)f(x_0) + L_1(x)f(x_1) + L_2(x)f(x_2)$$

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

This example is with a second order and three points (n = 2)

Simpson 1/3 and 3/8 Composite (both O(h^4))

1/3:

$$I = \frac{h}{3}\left[f(a) + f(b) + 4\sum_{\substack{j=1\\ odd}}^{n-1} f(x_j) + 2\sum_{\substack{j=2\\ even}}^{n-2} f(x_j)\right]$$

h = (b -a) /n

n: nb. segments (points -1)

even segments. Min 4

3/8:

$$I = \frac{3h}{8}\left[f(a) + f(b) + 3\sum_{\substack{j=1\\ j\neq 3,6,9,\ldots}}^{n-1} f(x_j) + 2\sum_{j=3,6,9,\ldots}^{n-3} f(x_j)\right]$$

n must be a multiple of 3. Min 6 segments

Backward, forward & central diff (2 & 3 points) numerical differentiation

central O(h^2)

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h} \qquad f''(x_i) = \frac{f(x_{i+1}) - 2f(x_i) + f(x_{i-1})}{h^2}$$

O(h^4) -> $f'(x_i) = \dfrac{f(x_{i-2} - 8f(x_{i-1}) + 8f(x_{i+1}) - f(x_{i+2})}{12h}$ h is interval size (x_1-x_0)

forward O(h)

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_i)}{h} \qquad f''(x_i) = \frac{f(x_{i+2}) - 2f(x_{i+1}) + f(x_i)}{h^2}$$

O(h^2) -> $f'(x_i) = \dfrac{-3f(x_i) + 4f(x_{i+1}) - f(x_{i+2})}{2h}$

backward O(h)

$$f'(x_i) = \frac{f(x_i) - f(x_{i-1})}{h} \qquad f''(x_i) = \frac{f(x_i) - 2f(x_{i-1}) + f(x_{i-2})}{h^2}$$

O(h^2) -> $f'(x_i) = \dfrac{f(x_{i-2}) - 4f(x_{i-1}) + 3f(x_i)}{2h}$

Power method to find max eigenvalue and eigenvector (iterative)

$$A = \begin{bmatrix} 10 & -4 & 0\\ -6 & 9 & -3\\ 0 & -6 & 6\end{bmatrix} \quad x_0 = \begin{bmatrix}1\\1\\1\end{bmatrix}$$

Pick initial guess x_0 and multiply to matrix A

$$A * x_0 = \begin{bmatrix}6\\0\\0\end{bmatrix} \rightarrow \lambda_0 * x_1 = 6 * \begin{bmatrix}1\\0\\0\end{bmatrix}$$

λ_0: eigenvalue

x_1: eigenvector

To find **λ** and **x** normalize vector (divide it by highest element. Iterate until real error is small enough

Ordinary differential equations

$$\left\{\begin{matrix} \frac{dy}{dx} = f(x,y)\\ y(x_0) = y_0\end{matrix}\right\}$$

Euler (1^{st} RK):

$$y_{i+1} = y_i + k_1 h \qquad k_1 = f(x_i, y_i)$$

Heuns: (2^{nd} RK)

$$y_{i+1} = y_i + \frac{k_1 + k_2}{2}h \qquad k_1 = f(x_i, y_i) \qquad k_2 = f(x_{i+1}, y_{i+1})$$

Midpoint: (2^{nd} RK)

$$y_{i+1} = y_i + k_2 h \qquad k_1 = f(x_i, y_i) \qquad k_2 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_1\right)$$

Ralston (2^{nd} order RK)

$$k_2 = f\left(x_i + \tfrac{3}{4}h, y_i + \tfrac{3}{4}h\right) \qquad k_1 = f(x_i, y_i)$$

$$y_{i+1} = y_i + f\left(\tfrac{1}{3}k_1 + \tfrac{2}{3}k_2\right)h$$

Classical (4^{th} order RK)

$$y_{i+1} = y_i + f(k_1 + k_2 + k_3 + k_4)h$$

$$k_1 = f(x_i, y_i) \qquad k_3 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_2\right)$$

$$k_2 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_1\right) \qquad k_4 = f(x_i + h, y_i + k_3 h)$$

Note: RK is not iterative, at each step, the approximate deteriorates, not refining it. Also, using RK4 on 3^{rd} order polynomial yields 100% accuracy