TA
Aparajit Chandrasekhar
Rishabh Dubey

13/01/20

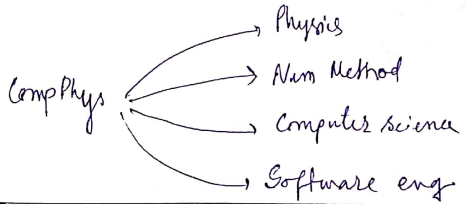
Topics

1. Linear algebra.
2. Differential equations
3. Fourier analysis
4. Random numbers

Physics Prob.
↓
Modelling.
↓
Algo & Code (Euler's algorithm; SPH...)

Reasons for Computation

- Analytical method don't work
- Problem size is big
- Large parameter space.
- Analysis of complex prob.
- Model Inference.
- Visualization
- Symbolic computation (like integration)



Algorithmic thinking.

Process of writing comp. phys. code:

- Identify problem
- Parameterization
- Scaling the quantities
- optimization depending on the machine.

GSL. Library in python —

NumPy, SciPy, NAG

15/01/20

Linear Algebra in Computation

Examples :-

- State spaces
- sensibility analysis
- Inference (fitting)
- Solving differential equations.

$$2w + x + 4y + z = -4$$
$$3w + 4x - y - z = 3$$
$$w - 4x + y + 5z = 9$$
$$2w - 2x + y + 3z = 7$$

$$\begin{pmatrix} 2 & 1 & 4 & 1 \\ 3 & 4 & -1 & -1 \\ 1 & -4 & 1 & 5 \\ 2 & -2 & 1 & 3 \end{pmatrix} \begin{pmatrix} w \\ x \\ y \\ z \end{pmatrix} = \begin{pmatrix} -4 \\ 3 \\ 9 \\ 7 \end{pmatrix}$$

$$A X = v$$

Method 1 → Taking inverse of A. (Not very good)

Method 2 Gaussian Elimination :-

We are allowed to

(a) Multiply any eqn by scalars
(b) Linearly combine any eqns.

Devide first row by 2

$$\begin{bmatrix} 1 & 0.5 & 2 & 0.5 \\ 0 & 2.5 & -7 & -2.5 \\ 0 & -4.5 & -1 & 4.5 \\ 0 & -3 & -3 & 2 \end{bmatrix} \begin{bmatrix} w \\ x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -2 \\ 9 \\ 11 \\ 11 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0.5 & 2 & 0.5 \\ 0 & 1 & -2.8 & -1 \\ 0 & 0 & 13.6 & 0 \\ 0 & 0 & -11.4 & -1 \end{bmatrix} \begin{pmatrix} w \\ x \\ y \\ z \end{pmatrix} = \begin{pmatrix} -2 \\ 3.6 \\ 27.2 \\ 21.8 \end{pmatrix}$$

↓

$$\left(\begin{array}{cccc|c} 1 & 0.5 & 2 & 0.5 & -2 \\ & 1 & -2.8 & -1 & 3.6 \\ & & 1 & 0 & -2 \\ & & & 1 & 1 \end{array}\right)$$

Get solution using back substitution

$z = 1$
$y = -2$
$x = 1$
$\omega = 2$

$$\begin{pmatrix} 1 & a_{01} & a_{02} & a_{03} \\ & 1 & a_{12} & a_{13} \\ & & 1 & a_{23} \\ & & & 1 \end{pmatrix} \begin{pmatrix} w \\ x \\ y \\ z \end{pmatrix} = \begin{bmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

$z = v_3$

$\Rightarrow \quad y + a_{23} z = v_2 \quad \Rightarrow \boxed{y = v_2 - a_{23} z}$

$\Rightarrow \boxed{y = v_2 - a_{23} v_3}$

$x + a_{12} y + a_{13} z = v_1$

$\Rightarrow x = v_1 - a_{12}(v_2 - a_{23} v_3) - a_{13} v_3$

$= v_1 - a_{12} v_2 - a_{13} v_3 + a_{12} a_{23} v_3.$

$\boxed{x = v_1 - a_{12} y - a_{13} z}$

$\boxed{w = v_0 - a_{01} x - a_{02} y - a_{03} z}$

$1 - \frac{8}{3}$
$-\frac{5}{3}$
$2 - \frac{2}{3}$

Exercise

$$\left(\begin{array}{ccc|c} 3 & -4 & 1 & 1 \\ 2 & 1 & 2 & 3 \\ 1 & 2 & -1 & 5 \end{array}\right) \rightarrow \left(\begin{array}{ccc|c} 1 & \frac{4}{3} & \frac{1}{3} & 1/3 \\ 0 & -\frac{5}{3} & & \end{array}\right)$$

$$\Rightarrow \left(\begin{array}{ccc|c} 1 & 2 & -1 & 5 \\ 3 & -4 & 1 & 1 \\ 2 & 1 & 2 & 3 \end{array}\right) \rightarrow \left(\begin{array}{ccc|c} 1 & 2 & -1 & 5 \\ 0 & +2 & -4 & +14 \\ 0 & -3 & 4 & -7 \end{array}\right)$$

$$\left(\begin{array}{ccc|c} 1 & 2 & 1 & 5 \\ 0 & -1 & -12 & +7 \\ 0 & 0 & 10 & 14 \end{array}\right)$$

$z = -7.$

$y = 7 + 2(-7)$

$y = -7$

$x = 5 - 2y + z$

$= 5 + 14 - 7$

$x = 12$

$12 \quad +7 \quad -14$

$=$

Pivoting

$$\left(\begin{array}{cccc|c} 0 & 1 & 4 & 1 & -4 \\ 3 & 4 & -1 & -1 & 3 \\ 1 & -4 & 1 & 5 & 9 \\ 2 & -2 & 1 & 3 & 7 \end{array}\right)$$ ← pivoting.

$$\left(\begin{array}{cccc|c} 3 & 4 & -1 & -1 & 3 \\ 0 & 1 & 4 & 1 & -4 \\ 1 & -4 & 1 & 5 & 9 \\ 2 & -2 & 1 & 3 & 7 \end{array}\right)$$

Partial pivoting

Put the maximum pivoting element in place of the zero.

complete pivoting when → columns are exchanged.

$Ax = \underset{\sim}{v}$ ← can be varied (a priori unknown)

LU decomposition.

LU decomposition.

$$A = \begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{pmatrix}$$

$$\frac{1}{a_{00}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ -a_{10} & a_{00} & 0 & 0 \\ -a_{20} & 0 & a_{00} & 0 \\ -a_{30} & 0 & 0 & a_{00} \end{pmatrix} \begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ \cdot & \cdot & & \cdot \\ \cdot & & \cdot & \cdot \\ a_{30} & - & - & a_{33} \end{pmatrix}$$

L_0

$$= \begin{pmatrix} 1 & b_{01} & b_{02} & b_{03} \\ 0 & b_{11} & & \\ 0 & b_{21} & \cdot & \\ 0 & b_{31} & & b_{33} \end{pmatrix}$$

$$L_1 = \frac{1}{b_{11}} \begin{pmatrix} b_{11} & 0 & 0 & 0 \\ 0 & b_{11} & 0 & 0 \\ 0 & -b_{21} & b_{11} & 0 \\ 0 & -b_{31} & 0 & b_{11} \end{pmatrix}$$

$$L_2 = \frac{1}{b c_{22}} \begin{pmatrix} c_{22} & 0 & 0 & 0 \\ 0 & c_{22} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -c_{32} & c_{22} \end{pmatrix}$$

$$L_3 = \frac{1}{d_{33}} \begin{pmatrix} d_{33} & 0 & 0 & 0 \\ 0 & d_{33} & 0 & 0 \\ 0 & 0 & d_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$L_3 L_2 L_1 L_0 A$ is upper triangular.

Suppose we have evaluated $L_3 L_2 L_1 L_0$, then

if $Ax = v$ is given,

$$L_3 L_2 L_1 L_0 Ax = L_3 L_2 L_1 L_0 v .$$

$U = L_3 L_2 L_1 L_0 A$

$L = L_0^{-1} L_1^{-1} L_2^{-1} L_3^{-1}$

& $A = LU$

$\Rightarrow LUx = v$

L is a lower triangular matrix.

$$L = \begin{pmatrix} a_{00} & & & \\ a_{10} & b_{11} & & \\ a_{20} & b_{21} & c_{22} & \\ a_{30} & b_{31} & c_{32} & d_{33} \end{pmatrix}$$

§ Libraries in Python — Numpy / Scipy / Matplotlib

numpy.linalg → solve

```
import numpy as np
from numpy.linalg import solve
```

20/1/20

$Ax = v$

$n \times m$

A, v known

x unknown.

Classical method

("Cramer's rule")

→ Evaluate cofactors and determinant.

$AX = V$ ← each of them 4×4 matrix.

A, V → known.

X → unknown.

$V \longrightarrow I$

$AX = I$

$\Rightarrow X = A^{-1}$.

(what are the other columns?)

Counting number of operations required for Gaussian Elimination

$$\begin{pmatrix} 2 & 1 & 4 & 1 \\ 3 & 4 & -1 & -1 \\ 1 & -9 & 1 & 5 \\ 2 & -2 & 1 & 3 \end{pmatrix} \begin{pmatrix} w \\ x \\ y \\ z \end{pmatrix} = \begin{pmatrix} -9 \\ 3 \\ 9 \\ 7 \end{pmatrix}.$$

Row 1: 5 divisions.
5 multiplications
5 subtractions. } ×3

Row 2: 4 divisions.
4 multiplications
4 subtractions } ×2

Row 3: 3 divisions.
3 multiplications
3 subtractions } ×1

Row 4: 2 divisions.

14 div
26 mult
26 subs. } 66 operations

Back substituting.

For z $\Rightarrow$ 0 mult, 0 subs.

for y $\Rightarrow$ 1 mult, 1 subs.

for x $\Rightarrow$ 2 mult, 2 subs.

for z $\Rightarrow$ 3 mult, 3 subs

$\Rightarrow$ 12 operations

In total, 78 operations.

$$\begin{bmatrix} 3 & -4 & 1 \end{bmatrix}_{3\times 3} \begin{pmatrix} \quad \end{pmatrix} = \begin{pmatrix} \quad \end{pmatrix}$$

$\Rightarrow$

4 div
4 mult } 2
4 subs

3 div
3 mult } 1
3 subs

2 div

9
22
31
+ 6
37

$(n+1)$
$(n+1)(n+1) \times (n-4)$

$\frac{78}{37} \sim 2$.

for $(n \times n)$ matrix

i^{th} row $(n+2-i^0)$ div. | i start from 1

$(n+2-i^0)$ mult
$(n+2-i^0)$ subs
} $\times (n-i^0)$

$$2\left(1+2+\cdots+(n-1)\right) = n(n-1)$$

$$= 2 \cdot \frac{1}{2}\left[(n-1)+1\right](n-1)$$

$$\sum_{i=1}^{n} (n+2-i^0) + 2(n-i^0)(n+2-i^0)$$

$$= \sum_{i=1}^{n} (n+2-i^0) + 2(n^2+2n-ni^0-2i^0n-4i^0+2i^{0+2}$$

$$= \sum_{i=1}^{n} (n+2+2n^2+4n) + \sum_{i=1}^{n} (-i^0-4ni^0-4i^0) + 2\sum_{i=1}^{n} i^{0+2}$$

$$= n\left(2+2n^2+5n\right) + -(5+4n)\frac{1}{2}(n+1)n + 2\left(\frac{n(n+1)(2n+1)}{6}\right)$$

$$= 2n+2n^3+5n^2 - \frac{1}{2}(5+4n)(n^2+n) + \frac{1}{3}(n^2+1)(2n+1)$$

$$= 2n+2n^3+5n^2 - \frac{1}{2}(5n^2+4n^3+4n^2+5n) + \frac{1}{3}(2n^3+n^2+2n+1)$$

$$= \frac{12n^3 - 18n^3 + 4n^3}{6} + \frac{30n^2 - 27n^2 + 2n^2}{6}$$

$$+ \frac{12n - 15n + 4n}{6} + \frac{1}{3}$$

$$=$$

$$\frac{2}{3}n^3 + \frac{5n^2}{6} + \frac{n}{6} + \frac{1}{3}$$

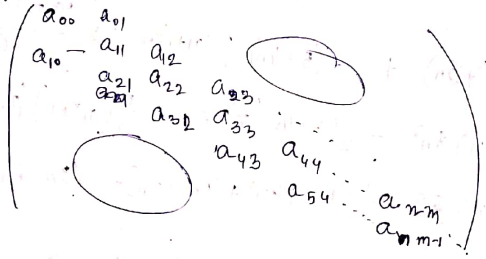
$$= \frac{1}{6}\left[4n^3 - 43n^2 + 6n + 2\right].$$

Total $an^3 + bn^2 + cn + d$

For large n, $\sim O(n^3)$

In Cramer's rule, $O(n!) \sim O(n^n)$

Tridiagonal matrices



Thomas Algorithm.

numpy.linalg

→ solve
→ inv

27/01/20

Round-off errors

Real numbers are of the form,

$$x = \alpha_n \cdots \alpha_2 \alpha_1 \alpha_0 . \alpha_{-1} \alpha_{-2} \cdots$$

$$= \alpha_n \times 10^n + \cdots + \alpha_2 \times 10^2 + \alpha_1 \times 10 + \alpha_0 \times 10^0 + \alpha_{-1} \times 10^{-1} + \alpha_{-2} \times 10^{-2} + \cdots$$

This representation is though non-unique

0.999... is same as 1.000...

In computer we only have finitely many digits.

30.421 in comp. 0.30421×10^2

For 3-digit decimal comp.

0.304×10^2

$x = a \times 10^b$ a - mantissa, b - exponent

→ finite # real number → ("machine numbers").

There cannot be unique rep. (without any error)

$5420 \qquad 0.542 \times 10^4$

0.0542×10^5

To avoid this, $|\text{mantissa}| \geq 0.1$

With finiteness of machine numbers comes the Round-off error.

Relative round-off error :

$$\frac{|\text{Actual number} - \text{machine number}|}{\text{Actual number}}$$

0.06 in 3-digit rep.

0.600×10^{-1}, relative error $= 0$.

$10^{-5} \qquad 0.100 \times 10^{-4}$

0.6987

0.699×10^0 relative error $\frac{0.0013}{0.6987}$

$= 0.04\ \%$.

Ill-condition

Where round off errors grow through an algorithm

Worst error

$$0.d_1 d_2 \cdots d_t \,|\, d_{t+1} \cdots$$

Let a t-digit computer

if $d_{t+1} < 5 \rightarrow 0.d_1 d_2 \cdots d_t$

if $d_{t+1} \geq 5 \rightarrow 0.d_1 d_2 \cdots d_t + 10^{-t}$

Relative error $\left|\frac{x - rd(x)}{x}\right| \leq \frac{5 \times 10^{-(t+1)}}{10^{-1}}$

(10^{-1} ← min. mantissa)

$\leq 5 \times 10^{-t}$ ← max round off error for t-digit

Round off error ~~in~~ properties of Gaussian Elimination :—

$10^{-5}x + y = 1$ $\quad y = 2 - x$

$x + y = 2$

$\therefore$ $10^{-5}x + (2-x) = 1$

$\Rightarrow x(10^{-5} - 1) = -1$

$\Rightarrow x = \frac{1}{1-10^{-5}}$

1.000000
0.000001
0.999999

re $= \frac{.00001}{.99999} \approx 1$

$$\begin{pmatrix} 10^{-5} & 1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} 1 & 10^{5} \\ 1 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 10^{5} \\ 2 \end{pmatrix}$$

$\Rightarrow (10^5 - 1)\, y = (10^5 - 2)$

$\Rightarrow (10^5 - 0.000001 \times 10^5)\, y = 10^5 - 0.000002 \times 10^5$

For 4-digit case $\Rightarrow 10^5 y = 10^5 \Rightarrow y = 1$

From back substitution

Then $x = 0$. which is wrong.

1.000000×10^0
1.000000×10^0
0.999999×10^0

Solution $\rightarrow$ Pivoting

$x + y = 2$

$10^{-5}x + y = 1$

$$\Rightarrow \begin{bmatrix} 10^{-5}x + 10^{-5}y = 2 \times 10^{-5} \\ 10^{-5}x + y = 1 \end{bmatrix}$$

$\Rightarrow (1 - 10^{-5})\, y = (1 - 2 \times 10^{-5})$

$\Rightarrow y = 1$ for four digit.

Using back substitution we get $x = 1$.

Scaling

$10^{-5}x + y = 1$

$x + y = 2$

Scaling by 10^5,

$x + 10^5 y = 10^5$

$x + y = 2$

The properties of roundoff error for Gaussian elimination and back substitution is we are not known properly.

Eigenvalue Problems

$$Ax = \lambda x$$

QR decomposition

$n \times n$ real symmetric matrix

$A = QR$ — Q: orthogonal, R: upper triangular

$A = Q_1 R_1$

$Q_1^T A = R_1$

$A_1 = R_1 Q_1$

$= Q_1^T A Q_1$

$A_1 = Q_2 R_2$

$A_2 = R_2 Q_2 = Q_2^T A_1 Q_2$

$= Q_2^T Q_1^T A Q_1 Q_2$

After k times repeating

$$A_k = \left(Q_k^T Q_{k-1}^T \cdots Q_1^T\right) A \left(Q_1 \cdots Q_k\right)$$

A_k is diagonal for large k !!

$$V = \prod_{i=1}^{k} Q_i$$

V is orthogonal.

$A_k = V^T A V$

$\Rightarrow V A_k = A V$

A_k → diagonal (D) $\Rightarrow AV = VD$

$\Rightarrow$ Columns of V are eigenvectors.

& Diagonal elements of D are eigenvalues.

03/02/20

$Ax = \lambda x$ ($n \times n$) Solve for x and λ.

$\Rightarrow (A - \lambda I) x = 0$

non trivial sol only when $(A - \lambda I)$ is singular.

Then, $\det |A - \lambda I| = 0$

n^{th} degree polynomial. $\Rightarrow$ n complex roots.

Finding roots, numerically is very hard.

We need other approach.

$A \nearrow \lambda_1 \dots \lambda_k$ eigenvalues

$A \searrow x_1, \dots x_n$ eigen vectors.

$k < n$ A is "defective". #Geometric multiplicity.
(when $k<n$ for the eigenvectors)

$$\begin{pmatrix} 2 & 0 & 0 \\ 1 & 1 & 2 \\ 1 & -1 & 4 \end{pmatrix}$$

eigenvalues are 3 and 2.

eigenvectors $\underbrace{\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}}_{\lambda=3}, \underbrace{\begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} -2 \\ 0 \\ 1 \end{pmatrix}}_{\lambda=2}$

Spans $\mathbb{R}^3$.

$$\begin{pmatrix} 2 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}$$

eigenvalues 3 & 2.

eigenvectors $\underbrace{\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}}_{\lambda=3}, \underbrace{\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}}_{\lambda=2}$

For $D_n = \lambda I_n$ eigenvalues $= \lambda$

eigenvectors $\rightarrow$ basis of $\mathbb{R}^n$

Jacobi matrix

$$C_n = \begin{bmatrix} \lambda & 1 & & 0 \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ 0 & & & \lambda \end{bmatrix}; \quad C_3 = \begin{bmatrix} \lambda & 1 & \\ & \lambda & 1 \\ & & \lambda \end{bmatrix}$$

eig. val λ

eig. vectors $\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$ $\Rightarrow$ not basis of $\mathbb{R}^n$.

Set of ~~eigenvalues~~ eigenvalues is called "Spectrum"

Under similarity transformation the spectrum remains the same.

If $A = S^{-1}BS$ then

If λ is an eigenvalues of A, then λ is also an eigenvalue of B.

In finding eigenvalues numerically, this property is used very much.

If x is an eigen vector of A then Sx is eigen vector of B.

A is not defective ~~iff A is similar~~

$\Leftrightarrow$ A is similar to a diagonal matrix.

If λ is an eigenvalue of A, then $P(\lambda)$ is an eigenvalue of $P(A)$, where P is a polynomial.

If λ is an eigenvalue of A then λ is an eigenvalue of A^T.

$$Ax = \lambda x \iff A^T y = \lambda y$$

x not equal y in gen.

If A is symmetric or Hermitian then eigenvalues are real.

If A is symmetric $\iff$ A is diagonalisable.

$$A = Q^T D Q \qquad Q - \text{orthogonal}.$$

The algorithm is / repeatedly apply similarity transform to A.

$$A^{(0)} = A$$
$$A^{(1)} = T^{(0)^{-1}} A^{(0)} T^{(1)}$$
$$A^{(2)} = T^{(2)^{-1}} A^{(1)} T^{(2)}$$
$$\vdots$$
$$A^{(k)} = T^{(k)^{-1}} A^{(k-1)} T^{(k)}$$

We are considering matrices A to be diagonalizable (i.e A is not defective). Even if A is defective, still following the ~~approach~~ algorithm we will get a nicer form of A (~~is~~ similar matrix of A).

* We will take A to be real.

Power Method (not following the above mentioned algo).

$A \longrightarrow$ dominant eigenvalue
$(n \times n)$ corresponding eigenvector.
diagonalizable. (n linearly indep. eigenvectors)

$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$

↓ strictly greater than.

$A = 2, 5, 0, -7, -2 \Rightarrow -7$ is the dominant eigenvalue.

A gen. vector $x_0 \in \mathbb{R}^n$

$$x_0 = c_1 v_1 + c_2 v_2 + \cdots + c_n v_n$$

$$A x_0 = c_1 \lambda_1 v_1 + \cdots + c_n \lambda_n v_n$$

$$\Rightarrow A^k x_0 = c_1 \lambda_1^k v_1 + \cdots + c_n \lambda_n^k v_n$$

$$\Rightarrow A^k x_0 = \lambda_1^k \left(c_1 v_1 + \cdots + c_n \left(\frac{\lambda_n}{\lambda_1}\right)^k v_n \right)$$

For large enough k

$$A^k x_0 = \lambda_1^k c_1 v_1$$

Let $y \in \mathbb{R}^n$

Take inner product with $A^k x_0$

$$A^{(k)} x_0 \cdot y = \lambda_1^k c_1 v_1 \cdot y$$

And,

$$A^{k+1} x_0 \cdot y = \lambda_1^{k+1} c_1 v_1 \cdot y$$

$$\Rightarrow \frac{1}{\lambda_1^k} A^k x_0 \cdot y = \frac{1}{\lambda_1^{k+1}} A^{k+1} x_0 \cdot y$$

$$\Rightarrow \lambda_1 = \frac{A^{k+1} x_0 \cdot y}{A^k x_0 \cdot y}$$

Example

$$A = \begin{pmatrix} 1 & 3 \\ 2 & 2 \end{pmatrix}$$

$$x_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$\Rightarrow A x_0 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

$$A^2 x_0 = \begin{pmatrix} 1 & 3 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 7 \\ 6 \end{pmatrix}$$

$7 + 18$

$14 + 12$

$$A^3 x_0 = \begin{pmatrix} 1 & 3 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} 7 \\ 6 \end{pmatrix} = \begin{pmatrix} 25 \\ 26 \end{pmatrix}$$

$25 + 78$

$$A^4 (x_0) = \begin{pmatrix} 1 & 3 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} 25 \\ 26 \end{pmatrix} =$$

$$A^6 (x_0) = \begin{pmatrix} 2254 \\ 2252 \end{pmatrix}$$

$$y = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$\Rightarrow \lambda_1 = \frac{A^6 x_0 \cdot y}{A^5 x_0 \cdot y}$$

$$= \frac{2254}{562} = 4.01$$

$\Rightarrow$ eigenvalue ≈ 4.

$$y = A^m x_0$$

$$\lambda_1 = \frac{A^{m+1} x_0 \cdot A^m x_0}{A^m x_0 \cdot A^m x_0}$$

$$= \frac{A\left(A^m x_0 \cdot A^m x_0\right)}{\left(A^m x_0 \cdot A^m x_0\right)}$$

$$= \frac{Av \cdot v}{v \cdot v} = \lambda$$

$$\Rightarrow \quad Av = \lambda v$$

Then eigen vector corr. λ_1 is $A^m x_0$.

Example $A = \begin{pmatrix} -2 & -3 \\ 6 & 7 \end{pmatrix}$ $\quad \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 1 \end{pmatrix}$

$$A x_0 = \begin{pmatrix} -2 \\ 6 \end{pmatrix}$$

$$A^2 x_0 = \begin{pmatrix} -2 & -3 \\ 6 & 7 \end{pmatrix}\begin{pmatrix} -2 \\ 6 \end{pmatrix}$$

$$= \begin{pmatrix} -14 \\ 30 \end{pmatrix}$$

-12
42
$\overline{30}$

$$A^3 x_0 = \begin{pmatrix} -2 & -3 \\ 6 & 7 \end{pmatrix}\begin{pmatrix} -14 \\ 30 \end{pmatrix}$$

$$= \begin{pmatrix} -62 \\ 126 \end{pmatrix}$$

90
$28\,2$
$\overline{-62}$
210
-84
$\overline{126}$

$$A^4 x_0 = \begin{pmatrix} -2 & -3 \\ 6 & 7 \end{pmatrix}\begin{pmatrix} -62 \\ 126 \end{pmatrix} \qquad (1\ 0)$$

$$= \begin{pmatrix} -254 \\ 510 \end{pmatrix}$$

378
124
$\overline{254}$

882
372
$\overline{510}$

$$\lambda = \frac{-254}{-62}$$

$$= 4.01$$

31) 127(

$\lambda = 4$ $\qquad x = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$

$$\begin{pmatrix} -2 & -3 \\ 6 & 7 \end{pmatrix}\begin{pmatrix} -1 \\ 2 \end{pmatrix} = \begin{pmatrix} -4 \end{pmatrix}$$

LINPACK To b do subroutine.
EISPACK about eigenvalues & vectors.
LAPACK (by Jack Dongarra)

GSL
Numpy } ← LAPACK

BLAS → Level 1 (Scalar vector)
(basic linear alg. subroutine)
Level 2 (vector matrix)
Level 3 (matrix mult)

numpy.linalg.eigh

→ computes eigenvalues of real symmetric matrices.

$\begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}\begin{pmatrix} -1 \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 1 \\ -1 \end{pmatrix}$

$\begin{pmatrix} 1 & 2 \end{pmatrix}$

05/02/20

Power method → gives only the dominant eigenvalue.

$A \longrightarrow \lambda_1 > \lambda_2 \geq \cdots \geq \lambda_n$

construct

$B \longrightarrow \lambda_2 \geq \cdots \geq \lambda_n$

$\downarrow$

$>$

Constructing B is called 'Deflation'.

QR decomposition

$A \rightarrow n \times n$ real symmetric $\Rightarrow$ n real eigenvalue.

$\lambda_1, \lambda_2, \ldots \lambda_n$ } eigenvalue
$v_1, v_2 \ldots v_n$ } eigenvector

$$V = \begin{bmatrix} | & | & & | \\ v_1 & v_2 & \cdots & v_n \\ | & | & & | \end{bmatrix}$$

$AV = VD$ → diagonal matrix.

$A = QR$

Q → orthogonal
R → upper triangular

$A_1 = Q_1 R_1$ ($A_1 \rightarrow$ given matrix)

$A_2 = R_1 Q_1$

$Q_1^T A_1 = R_1$

$\Rightarrow A_2 = Q_1^T A_1 Q_1$

$A_2 = Q_2 R_2$

$A_3 = R_2 Q_2$

$A_3 = Q_2^T A_2 Q_2$

$\Rightarrow A_3 = Q_2^T Q_1^T A_1 Q_1 Q_2$

$A_k = Q_{k-1}^T Q_{k-2}^T \cdots Q_1^T A_{k-1} Q_1 \cdots Q_{k-1}$

$A_k = \left[Q_{k-1}^T \cdots Q_1^T \right] A \left[Q_1 \cdots Q_{k-1} \right]$

$V = \prod_{i=1}^{k-1} Q_i$

$\Rightarrow A_k = V^T A V$

$V A_k = A V$

A_k is diagonal. for large k.

$\Rightarrow$ Diagonal elements of A_k will be the eigen values of A. And the columns of V will be the eigen vectors.

$$A = \begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 1 \\ 0 & 1 & 3 \end{bmatrix}$$

$$R_1 = \begin{bmatrix} \sqrt{10} & (3/5)\sqrt{10} & 1/\sqrt{10} \\ 0 & 2.7203 & 1.9851 \\ 0 & 0 & 2.4412 \end{bmatrix}$$

$$Q_1 = \begin{bmatrix} 0.94868 & -0.29409 & 0.11625 \\ 0.31623 & 0.88226 & -0.34874 \\ 0 & 0.36761 & 0.92998 \end{bmatrix}$$

$A_2 = R_1 Q_1$

$$A_2 = \begin{bmatrix} 3.6 & 0.86024 & 0 \\ 0.86024 & 3.12973 & 0.89740 \\ 0 & 0.89740 & 2.27027 \end{bmatrix}$$

$$A_{13} = \begin{bmatrix} 4.4 & & \\ & & \\ & & \end{bmatrix}$$

np.linalg.qr.

np.matmul @

$A = QR$

$$A = \begin{pmatrix} | & | & \\ a_0 & a_1 & \cdots \\ | & | & \end{pmatrix}$$

$u_0 = a_0 \qquad q_0 = \frac{u_0}{|u_0|}$

$u_1 = (a_1 - q_0(q_0 \cdot a_1)) \qquad q_1 = \frac{u_1}{|u_1|}$

$u_2 = a_2 - q_0(q_0 \cdot a_2) - q_1(q_1 \cdot a_2)$

$q_2 = \frac{u_2}{|u_2|}$

$q_0, q_1, q_2 \ldots$ orthonormal

$a_0 = |u_0| q_0$

$a_1 = |u_1| q_1 + (q_0 \cdot a_1) q_0$

$a_2 = |u_2| q_2 + (q_0 \cdot a_2) q_0 + (q_1 \cdot a_2) q_1$

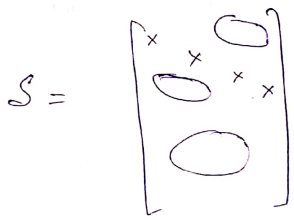
$$\begin{pmatrix} | & | & \\ a_0 & a_1 & \cdots \\ | & | & \end{pmatrix} = \begin{pmatrix} | & | & \\ q_0 & q_1 & \\ | & | & \end{pmatrix} \begin{pmatrix} |u_0| & q_0 \cdot a_1 & (q_0 \cdot q_2) \\ & |u_1| & (q_1 \cdot q_2) \\ & & |u_2| \end{pmatrix}$$

A Q R

Singular Value Decomposition (SVD)

$A = U S V^T$

$m \times n \quad (m \times m) \quad (m \times n) \quad (n \times n)$

$S =$  (as diagonal as possible)

$$S = \begin{bmatrix} s_1 & & \\ & s_2 & \\ & & \ddots \end{bmatrix}$$

$= s_1^2, s_2^2 \ldots$ are eigen values of

A^TA.

A^TA & AA^T are real symmetric. And both have same eigen values (~~different numbers maybe~~) & (~~diff~~ more 0 for one of them diag. as the order are different).

$\Rightarrow$ +ve eigen values.

$$\underbrace{s_1^2 \geq s_2^2 \geq \cdots s_k^2 \geq}_{k \text{ nonzero eig. values.}} \underbrace{s_{k+1}^2 = \cdots s_n^2 = 0}_{n-k \text{ zero eigen values.}}$$

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}_{5\times 3}.$$

$$A^TA = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 4 & 1 \\ 1 & 1 & 2 \end{bmatrix}_{3\times 3}$$

$s_1^2 = 5, \quad s_2^2 = 2, \quad s_3^2 = 1$

$$S = \begin{bmatrix} \sqrt{5} & 0 & \\ & \sqrt{2} & \\ 0 & & 1 \\ 0 & & \end{bmatrix}_{5\times 3}$$

$A = USV^T$

U & V^T are orthogonal.

$A^TA = V S^T U^T U S V^T$

$= V S^T S V^T$

$\Rightarrow$ V is constructed with the eigenvectors of A^TA.

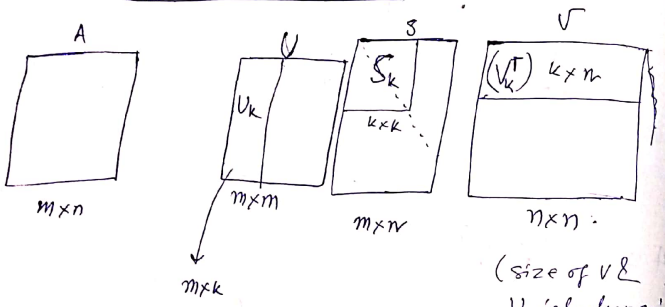
Similarly U is constructed with the eigenvectors of AA^T.

$A = U S V^T$ — stretching.

rotating

{ np.linalg.svd.

Used in data compression.



$A \approx U_k S_k V_k^T$ (where S_k containing the dominant elements only).

A has $m \cdot n$ numbers.

$U_k S_k V_k^T$ will have $(m \cdot k + k + k \cdot n)$ numbers

$= k(m+n+1)$

If $m = 2n$

then A will have $2n^2$

$U_k S_k V_k^T$ will have $k(3n+1)$

for $k = n/3$

$U_k S_k V_k^T$ will have, $\frac{n}{3}(3n+1)$

$\approx n^2$

PCA : Principle Component Analysis

Iterative Techniques

$Ax = b$ } "Direct techniques"
↳ Gauss elimination

x $Ax = \lambda x$

$x^{(0)}$ ← initial approximation of the sol^n of $Ax = b$
(guess sol)

$\{x^{(k)}\}_{k=0}^{\infty}$ ⟶ hopefully convergence true sol.

Gaussian elimination costs $O(n^3)$

The Iterative technique in best case can be $O(n^2)$

The ~~error~~ rounding error are smaller for (compared to Gaussian)

Large sparse matrices
↓ ↓
large dim. many zero elements.

Small matrix ← direct method
Large matrix ← Iterative method.

Jacobi method

$$Ax = b$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_i$$

$$x_i = -\sum_{j \neq i}^{n} \frac{a_{ij} x_j}{a_{ii}} + \frac{b_i}{a_{ii}}$$

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

Example

$$\begin{pmatrix} 10 & -1 & 2 & 0 \\ -1 & 11 & -1 & 3 \\ 2 & -1 & 10 & -1 \\ 0 & 3 & -1 & 8 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 6 \\ 25 \\ -11 \\ 15 \end{pmatrix}$$

$x^{(0)} = (0, 0, 0, 0)$

$x_1^{(1)} = \frac{6}{10}$

$x_2^{(1)} = \frac{25}{11}$

$x_3^{(1)} = -\frac{11}{10}$

$x_4^{(1)} = \frac{15}{8}$

$$x_1^{(2)} = -\frac{1}{10}\left[-\frac{25}{11} + -\frac{11}{5}\right] + \frac{6}{10}$$

$$= 1.047.$$

$$x_2^{(2)} = -\frac{1}{11}\left[0 - \frac{6}{10} + \frac{11}{10} + \frac{45}{8}\right] + \frac{25}{11}$$

$$= -1.716.$$

$x_3^{(2)} = -\frac{1}{10}[$ ...

Partial differential eqn can be converted into linear equations and solved in this method.

$x^{(k)} = T\, x^{(k-1)} + C$

T: matrix, C: vector

$A = D - L - U$

D: diagonal elements, L: lower diagonal, U: upper diagonal.

$Ax = b$

$\Rightarrow (D - L - U)x = b$

$\Rightarrow Dx = (L+U)x + b$

$\Rightarrow x = D^{-1}(L+U)x + D^{-1}b$

Now D^{-1} → T, $D^{-1}b$ → C

Gauss - Seidel

$$x_i^k = \frac{1}{a_{ii}}\left[-\sum_{j=1}^{i-1} a_{ij}\, x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}\, x_j^{(k-1)} + b_i\right]$$

Example,

$$\begin{pmatrix} 10 & -1 & 2 & 0 \\ -1 & 11 & -1 & 3 \\ 2 & -1 & 10 & -1 \\ 0 & 3 & -1 & 8 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 6 \\ 25 \\ -11 \\ 15 \end{pmatrix}$$

$x_1^{(1)} = \frac{6}{10}$

$x_2^{(1)} = \frac{1}{11}\left(25 - \left(-1 \times \frac{6}{10}\right)\right)$

$= \frac{1}{11}(25 + 0.6)$

$= 2.327$

$x_3^{(1)} = \frac{1}{10}\left[-11 - \left\{\frac{6}{5} - 2.327\right\}\right]$

$= -0.987$

$x_4^{(1)} = \frac{1}{8}\left[15 - \{3 \times 2.327 + 0.987\}\right]$

$= 0.879$

$$x_1^{(2)} = \frac{1}{10}\left[6 - \{-2.327 - 1.974\}\right]$$

$$= 1.0301$$

Again,

$$x_i^{(k)} = T x_i^{(k-1)} + C_i$$

$$T = (D-L)^{-1}U$$

$$C = (D-L)^{-1}b$$

Relaxation Method

Residual vector

$$r = b - A\tilde{x}$$

$\tilde{x}$: if approximation of true solution.

$$\tilde{x}_i^{(k)} = \begin{bmatrix} x_1^{(k)} \\ x_2^{(k)} \\ \vdots \\ x_{i-1}^{(k)} \\ x_i^{k-1} \\ \vdots \\ x_n^{k-1} \end{bmatrix} \begin{matrix} \} \text{ have been updated} \\ \\ \\ \\ \} \text{ Yet to be updated.} \end{matrix}$$

$$r_{mi}^{(k)} = b_m - \sum_{j=1}^{i-1}$$

$$r_i^{(k)} = b - A\tilde{x}_i^{(k)}$$

$$r_{mi}^{(k)} = b_m - \sum_{j=1}^{i-1} a_{mj} x_j^{(k)} - \sum_{j=i-1} a_{mj} x_j^{(k-1)} - a_{mi} x_i^{(k-1)}$$

$$r_{ii}^{(k)} = b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} - a_{ii} x_i^{(k-1)}$$

$$a_{ii} x_i^{(k-1)} + r_{ii}^{(k)} = b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)}$$

$$= a_{ii} x_i^{(k)}$$

$$\Rightarrow r_{ii}^{k} = a_{ii}\left(x_i^{(k)} - x_i^{(k-1)}\right)$$

Or,

$$\boxed{x_i^{(k)} = x_i^{(k-1)} + \frac{r_{ii}^{(k)}}{a_{ii}}}$$

Here comes the relaxation method,

$$x_i^{(k)} = x_i^{(k-1)} + \omega \frac{r_i^{(k)}}{a_{ii}}$$

multiply with ω.

$\omega > 1 \longrightarrow$ "over-relaxation"

$\omega < 1 \longrightarrow$ "under-relaxation"

SOR (Successive Over Relaxation)

$$x^{(k)} = T_\omega x^{(k-1)} + c_\omega$$

$$T_\omega = (D - \omega L)^{-1} \left[(1-\omega) D + \omega U \right]$$

&

$$c_\omega = \omega (D - \omega L)^{-1} b$$

Example

$$\begin{pmatrix} 4 & 3 & 1 \\ 3 & 4 & -1 \\ 0 & -1 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 24 \\ 30 \\ -24 \end{pmatrix}$$

$x^{(0)} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

$\omega = 1.25$

$$x_1^{(1)} = \frac{1}{4}\left[24 - \{ 3 + 1 \} \right]$$

$$= 0.5$$

$$x_2^{(1)} =$$

Conjugate Gradient Method

$Ax^* = b$ $\quad x^* \rightarrow$ true solution.

x, $v \neq 0$ } any vectors.

$t \longrightarrow$ a scalar.

$$\rightarrow g(x) = \langle x, Ax \rangle - 2 \langle x, b \rangle$$

$$g(x+tv) = \langle x+tv, Ax + tAv \rangle - 2 \langle x+tv, b \rangle$$

$$g(x+tv) = \langle x, Ax \rangle - 2\langle x, b \rangle + 2t \langle v, Ax \rangle - 2t \langle v, b \rangle + t^2 \langle v, Av \rangle$$

$\langle x, Av \rangle$

$\langle v, Ax \rangle$

$$= g(x) - 2t \langle v, b - Ax \rangle + t^2 \langle v, Av \rangle$$

h

$h(t) = g(x + tv) \rightarrow$ quadratic

$h'(t) = 0$ (for minima)

$\Rightarrow -2\langle v, b - Ax \rangle + 2t \langle v, Av \rangle = 0$

$$t_{(min)} = \frac{\langle v, b - Ax \rangle}{\langle v, Av \rangle}$$

$$\hat{t} = \frac{\langle v, b - Ax \rangle}{\langle v, Av \rangle}$$

$$h(\hat{t}) = g(x + \hat{t} v)$$

$$= g(x) - 2\hat{t}\langle v, b - Ax \rangle + \hat{t}^2 \langle v, Av \rangle$$

$$= g(x) - 2\frac{\langle v, b - Ax \rangle^2}{\langle v, Av \rangle} + \frac{\langle v, b - Ax \rangle^2}{\langle v, Av \rangle}$$

$$\boxed{g(x + \hat{t} v) = g(x) - \frac{\langle v, b - Ax \rangle^2}{\langle v, Av \rangle}}$$

for $x \rightarrow x^*$, $g(x^* + \hat{t} v) = g(x^*)$

So, for true solution no change in $g(x)$ when moving toward the tv. ($v \rightarrow$ arbitrary).

For x^* is a solution $\rightarrow$ x^* minimizes $g(x)$.

$$g(x + tv) = g(x) - \frac{\langle v, b - Ax \rangle^2}{\langle v, Av \rangle}$$ $\rightarrow$ use this with $v \propto -\nabla g$ method of steepest descent.