Computational Methods and Applications

Scribe - III

Linear Systems and Interpolation

Equations of the form $Ax = b$; where

A is an $n \times n$ square matrix whose elements are a_{ij}

x and b are column vectors of dimension n

$Ax = b$ can be written as

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \quad \forall i \in \{1, 2, \dots n\}$$

b can also be interpreted as a linear combination of the

column matrix A weighted by vector x.

An example of pipe network was illustrated where in we would write a system of linear eqn to compute the pressure at each node.

→ Direct numerical methods are not ideal for very large system of linear equations

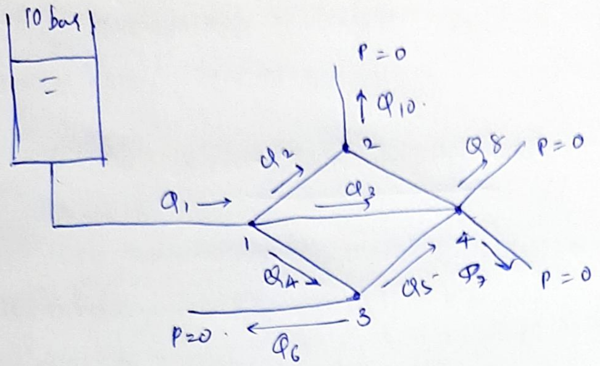
Iterative solution method

→ An iterative method for solution of linear ~~equation~~ system results in a sequence of vectors $\{x^{(k)}, k \geq 0\}$ of $\mathbb{R}^n$ that converges to the exact solution x^*, that is

$$\lim_{k \to \infty} x^{(k)} = x^*$$ for any given initial

vector $x^{(0)} \in \mathbb{R}^n$.

Construction



~~Re~~

$$Q_j = L_j \Delta P_j$$

Write a linear system of equations to compute the pressure at each node

$Q_1 = 10 - P_1$

$Q_2 = P_1 - P_2$

$Q_3 = P_1 - P_4$

$Q_4 = P_1 - P_3$

$Q_5 = P_3 - P_4$

$Q_6 = P_3$

$Q_7 = P_4$

$Q_8 = P_4$

$Q_9 = P_2 - P_4$

$Q_{10} = P_2$

$$10 - P_1 = P_1 - P_2 + P_1 - P_4 + P_1 - P_3 \quad -(1)$$

$$P_1 - P_2 = P_2 + P_2 - P_4 \quad -(2)$$

$$P_1 - P_3 = P_3 + P_3 - P_4 \quad -(3)$$

$$P_1 - P_4 + P_2 - P_4 + P_3 - P_4 = 2P_4 \quad -(4)$$

from (1) $\Rightarrow$ $10 = 4P_1 - P_2 - P_4 - P_3$

(2) $\Rightarrow$ $0 = 3P_2 - P_4 - P_1$

(3) $\Rightarrow$ $0 = 3P_3 - P_4 - P_1$

(4) $\Rightarrow$ $0 = P_1 + P_2 + P_3 - 5P_4$

Constructing iterative method

Select a suitable non singular matrix P, such that split matrix A

$$A = P - (P - A)$$

Then $$Px^* = b - (A - P)x^* \quad \text{——} (1)$$

Correspondingly for $k \geq 0$

$$Px^{(k+1)} = b - (A - P)x^{(k)} \quad \text{——} (2)$$

(2) - (1) gives.

$$x^{(k+1)} - x = (I - P^{-1}A)(x^{(k)} - x)$$

Convergence.

Let $e^{(k)} = x^{(k)} - x^*$ denote error at step k.

If $(I - P^{-1}A)$ is symmetric and positive definite (all eigenvalues positive)

$$\| e^{(k+1)} \|_2 = \| (I - P^{-1}A) e^{(k)} \|_2 \leq \rho(I - P^{-1}A) \| e^{(k)} \|_2$$

where $\rho(\cdot)$ is known as the spectral radius (maximum modulus of eigenvalues). If $\cancel{\rho(\cdot) \leq}$ $\rho(\cdot) < 1$ there is convergence.

The Jacobi Method

If diagonal entries of A are non zero, we can set $P = D$ (diagonal matrix containing ~~entries~~ diagonal entries of A. Then we get

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1, j \neq i}^{n} a_{ij} x_j^{(k)} \right) \quad \forall i \in \{1, 2, \dots n\}$$

Proposition

If the matrix A is strictly diagonally dominant by row, then the Jacobi method converges

- It may converge otherwise also
- The dominant diagonal elements becomes the denominator and drives the iterations towards convergence.

The Gauss - Siedel method

Faster convergence could be achieved if the new $(k+1)$ components already available are used

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

$$\forall\ i \in \{1, 2, \dots n\}$$

- There are no general results stating this method converges faster than Jacobi's

→ Pythons numpy.linalg module provide efficient low level implementations of standard linear algebra algorithms

Interpolation

In several applications we only know value of a function f at some given points $\{(x_i, y_i), i = 0, 1, 2, \dots n\}$. How do we determine f? We figure out an approximate function $\tilde{f}$ that satisfies

$$\tilde{f}(x_i) = y_i \quad \forall\ i \in \{0, 1, \dots n\}$$

→ 1 cubic functions exists passing through points (0, 1), (1, 4), (-1, 0) and (2, 15)

$$f(x) = x^3 + x^2 + x + 1$$

we got this solving

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 2 & 4 & 8 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ 0 \\ 15 \end{bmatrix}$$

Complexity $O(n^3)$

Different kinds of interpolation.

1) polynomial interpolation

$$\tilde{f}(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n.$$

2) trigonometric interpolation

$$\tilde{f}(x) = a_{-M} e^{-iMx} + \dots + a_0 + \dots + a_M e^{iMx}$$

3) rational interpolation

$$\tilde{f}(x) = \frac{a_0 + a_1 x + \dots + a_k x^k}{b_0 + b_1 x + \dots + b_m x^n}.$$

Lagrangian polynomial interpretation

for $j \in \{0, 1, \dots n\}$, define

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

Note that

$$\varphi_j(x_k) = \begin{cases} \prod_{i=0, i\neq j}^{n} \frac{x_j - x_i}{x_j - x_i} = 1 & \text{if } k = j \\ 0 & \text{otherwise} \end{cases}$$

Then required approximation is

$$\tilde{f}(x) = \sum_{j=0}^{n} y_j \varphi_j(x)$$

Eg: Function passing through the points $(0,1)$, $(1,4)$, $(-1,0)$ and $(2,15)$

$$\varphi_1(x) = \frac{x^3 - 2x^2 - x + 2}{2}.$$

$$\varphi_2(x) = \frac{-x^3 + x^2 + 2x}{2}$$

$$\varphi_3(x) = \frac{-x^3 + 3x^2 - 2x}{6}$$

$$\varphi_4(x) = \frac{x^3 - x}{6}.$$

$$\tilde{f}(x) = \varphi_1(x) + 4\varphi_2(x) + 15\varphi_4(x)$$

$$= x^3 + x^2 + x + 1$$