BAB 4 SISTEM PERSAMAAN LINIER

4.2 Sistem Persamaan Linier Segitiga Atas

Sistem persamaan linier yang mempunyai matriks koefisien berupa matriks segitiga atas, disebut sistem persamaan linier segitiga atas. Sistem persamaan linier seperti itu dapat dituliskan dalam bentuk:

$$
\begin{array}{ccccccccc}
a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & c_1 \\
 & & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & c_2 \\
 & & \vdots & & & & \vdots & & \vdots \\
 & & & & a_{n-1,n-1}x_{n-1} & + & a_{n-1,n}x_n & = & c_{n-1} \\
 & & & & & & a_{nn}x_n & = & c_n
\end{array}
$$

Dengan asumsi elemen-elemen diagonal tak nol, $a_{kk} \neq 0$ untuk $k = 1, 2, \ldots, n$, maka terdapat suatu solusi tunggal dari sistem persamaan linier di atas. Kondisi $a_{kk} \neq 0$ ini sangat penting karena persamaan tersebut melibatkan pembagian oleh a_{kk}. Jika persyaratan ini tidak terpenuhi maka solusinya tidak ada atau terdapat takhingga banyaknya solusi.

Penyelesaian sistem persamaan linier segitiga atas mudah dicari dengan mempergunakan <u>substitusi mundur</u> (*backward substitution*).

Langkah umum dari proses tersebut adalah:

$$\boxed{x_k = \frac{c_k - \sum_{j=k+1}^{n} a_{kj}x_j}{a_{kk}}}$$

untuk $k = n-1, n-2, \ldots, 1$.

Algoritma substitusi mundur untuk sistem persamaan linier segitiga atas

Masukan: n, a_{ij}, c_i, i,j = 1, 2, ..., n.

Langkah-langkah:

$x_n = c_n/a_{nn}$

Untuk k = n-1, n-2, ..., 1 lakukan:

jumlah ⟸ 0

Untuk j = k+1, k+2, ..., n lakukan:

jumlah ⟸ jumlah + $a_{kj}.x_j$

x_k ⟸ (C_k – jumlah) / a_{kk}

4.3 Sistem Persamaan Linier Segitiga Bawah

Sistem persamaan linier yang mempunyai matriks koefisien berupa matriks segitiga bawah, disebut sistem persamaan linier segitiga bawah. Sistem persamaan linier seperti itu dapat dituliskan dalam bentuk:

$$
\begin{aligned}
a_{11}x_1 & & & & &= c_1 \\
a_{21}x_1 &+ a_{22}x_2 & & & &= c_2 \\
\vdots \\
a_{n1}x_1 &+ a_{n2}x_2 &+ \cdots &+ a_{nn}x_n & &= c_n
\end{aligned}
$$

Penyelesaian sistem persamaan linier ini dicari dengan substitusi maju.

Langkah umum dari proses tersebut ialah:

$$x_k = \frac{c_k - \sum_{i=1}^{k-1} a_{ki} x_i}{a_{kk}}$$

untuk k = 2, 3, ... , n.

Algoritma substitusi maju untuk sistem persamaan linier segitiga bawah

Masukan : n , a_{ij}, c_j , i , j = 1, 2, ... , n.

Langkah-langkah:

$x_1 = c_1/a_{11}$

Untuk k = 2, 3, ..., n lakukan:

- jumlah $\Longleftarrow$ 0
- Untuk i = 1, 2, ..., n-1 lakukan:
  - jumlah $\Longleftarrow$ jumlah + $a_{ki}.x_i$
- $x_k \Longleftarrow$ (c_k – jumlah) / a_{kk}

4.4 Metode Eliminasi Gauss dan Pivoting

Menyelesaikan sistem persamaan linier AX = C dengan n persamaan dan n peubah. Intinya adalah membangun suatu sistem persamaan segitiga atas UX = Y yang setara kemudian solusi X diselesaikan memakai metode substitusi mundur.

Bila kepada suatu sistem persamaan linier dipakai operasi-operasi berikut, akan dihasilkan sistem yang setara. Operasi-operasi tersebut ialah:

a) Pertukaran urutan diantara dua persamaan.

b) Penskalaan yaitu perkalian sebuah persamaan dengan konstanta tak nol.

c) Penggantian yaitu sebuah persamaan dapat digantikan oleh jumlah persamaan itu dengan suatu kelipatan sebarang persamaan lainnya.

Pandang sistem persamaan linier yang berbentuk:

$$\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= c_1 \\
a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= c_2 \\
\vdots \qquad\qquad\qquad &\quad \vdots \\
a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= c_n
\end{aligned}$$

Matriks lengkap sistem persamaan linier itu adalah:

$$[A,C] = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & c_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & c_2 \\ \vdots & & & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & c_n \end{bmatrix} \quad \cdots (*)$$

Sistem AX = C dengan matriks lengkap tersebut di atas dapat diselesaikan dengan melakukan operasi-operasi bais elementer (OBE). Dengan OBE itulah dihasilkan sistem yang setara yang berbentuk sistem persamaan linear segitiga atas. Bentuk terakhir ini diselesaikan dengan substitusi mundur (*backward substitution*). Proses inilah yang disebut eliminasi Gauss.

4.4.1 Metode Eliminasi Gauss Naif

Disebut metode eliminasi Gauss naif karena ia tidak menghindari kemungkinan pembagian oleh nol. Untuk itulah diperbaiki dengan strategi pivoting. Jika $a_{kk} = 0$, perlu mencari baris r, dengan $a_{rk} \neq 0$ dan $r > k$, kemudian mempertukarkan baris k dengan baris r sehingga diperoleh elemen tumpuan tak nol.

Operasi pada baris ke i :

$P \leftarrow a_{i1}/a_{11}$

Untuk j = 2, 3, ..., n+1

$\quad a_{ij} \leftarrow a_{ij} - p.a_{1j}$

$a_{i1} \leftarrow 0$

Untuk i = 2, 3, ..., n.

Setelah berakhirnya langkah 1 diperoleh hasil sebagai berikut:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & a_{1,n+1} \\ 0 & a'_{22} & \cdots & a'_{2n} & a'_{2,n+1} \\ \vdots & & & & \vdots \\ 0 & a'_{n2} & \cdots & a'_{nn} & a'_{n,n+1} \end{bmatrix}$$

Secara umum, langkah 2 tersebut dapat dinyatakan sebagai berikut:

Operasi baris ke i:

$P \longleftarrow a_{i2}/a_{22}$

Untuk $j = 3, 4, \ldots, n+1$

$\quad a_{ij} \longleftarrow a_{ij} - p.a_{2j}$

$a_{i2} \longleftarrow 0$

Untuk $i = 3, 4, \ldots, n.$

Setelah berakhirnya langkah 2 diperoleh hasil sebagai berikut:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} & a_{1,n+1} \\ 0 & a'_{22} & a'_{23} & \cdots & a'_{2n} & a'_{2,n+1} \\ 0 & 0 & a''_{33} & \cdots & a''_{3n} & a''_{3,n+1} \\ \vdots & & & & & \vdots \\ 0 & 0 & a''_{n3} & \cdots & a''_{nn} & a''_{n,n+1} \end{bmatrix}$$

Untuk memperoleh matriks segitiga atas diperlukan (n-1) langkah. Langkah ke k dapat dinyatakan sebagai berikut:

Untuk $k = 1, 2, \ldots, n-1$

$\quad$ Untuk $i = k+1, k+2, \ldots, n$

$\quad\quad P \longleftarrow a_{ik}/a_{kk}$

$\quad\quad$ Untuk $j = k+1, k+2, \ldots, n+1$

$\quad\quad\quad a_{ij} \longleftarrow a_{ij} - P.a_{kj}$

$\quad\quad a_{ik} \longleftarrow 0$

Algoritma Eliminasi Gauss Naif

Masukan: n, a(i,j), i = 1, 2, . . ., n

j = 1, 2, . . ., n+1

Keluaran: x(i), i = 1, 2, . . ., n.

Langkah-langkah:

1. Untuk k = 1, 2, . . ., n-1, lakukan:
   - Jika $a_{kk} \neq 0$ maka ke langkah 7
   - Jika tidak, maka baris $\leftarrow$ k
2. Untuk i = k+1, k+2, . . ., n, lakukan:
   - Jika $a_{ik} \neq 0$, maka ke langkah 4
   - Jika tidak, ke langkah 3
3. Cetak "Matriks Singular", selesai
4. Baris $\leftarrow$ i
5. Untuk i = k, k + 1, . . ., n+1, lakukan:
   - D $\leftarrow$ a_{ki}
   - $a_{ki} \leftarrow a_{baris,\, i}$
   - $a_{baris,i} \leftarrow$ D
6. Kembali ke langkah 5
7. Untuk i = k+1, k+2, . . ., n, lakukan:
   - P $\leftarrow$ a_{ik}/a_{kk}
   - Untuk j = k+1, k+2, . . ., n+1, lakukan:
     - $a_{ij} \leftarrow a_{ij} - P.a_{kj}$
   - $a_{ik} \leftarrow 0$
8. Jika a_{nn} = 0, maka matriks singular. Selesai.

10. Untuk k = n-1, n-2, . . ., 1, lakukan:
    - jumlah $\leftarrow$ 0
    - Untuk j = k+1, k+2, . . ., n, lakukan:
      - jumlah $\leftarrow$ jumlah + a_{kj} * x_j
    - $x_k \leftarrow (a_{k,n+1}$ - jumlah) $/a_{kk}$

4.4.2 Metode Eliminasi Gauss Pivoting Parsial

Untuk menghindari pembagian dengan nol, strategi pivoting dapat juga diperluas untuk mengurangi galat pembulatan.

Pivoting parsial disarankan untuk memeriksa besarnya semua elemen di kolom k yang terletak pada atau di bawah diagonal, dan melokasikan baris r yang mempunyai elemen dengan nilai mutlak terbesar, yakni:

$$|a_{rk}| = maks\ \{|a_{kk}|, |a_{k+1,k}|, \ldots, |a_{n-1,k}|, |a_{nk}|\}$$

dan kemudian menukarkan baris r dan baris k jika r > k.
Dengan diambilnya elemen dengan nilai mutlak terbesar sebagai elemen tumpuan, akan menghasilkan perambatan galat yang kecil.

Algoritma Eliminasi Gauss Pivoting Parsial

Masukan : n, a_{ij}, i = 1, 2, ..., n
j = 1, 2, ..., n+1

Keluaran: x_i , i = 1, 2, ..., n

Langkah-langkah:

I. Untuk k = 1, 2, . . ., n-1
- l ⇐ k
- Untuk i = k+1, k+2, . . ., n
  - Jika $|a_{ik}| > |a_{lk}|$ maka l ⇐ i
- Jika $|a_{lk}| \le \varepsilon$ maka proses gagal. Selesai
- Jika l ≠ k maka
  - Untuk j = k, k + 1, . . ., n+1
    - $a_{lj} \Leftrightarrow a_{kj}$
- Untuk i = k+1, k+2, ..., n
  - P ⇐ a_{ik}/a_{kk}
  - Untuk j = k+1, k+2, . . ., n+1
    - $a_{ij} \Leftarrow a_{ij} - P.a_{kj}$
  - $a_{ik} \Leftarrow 0$

II. Jika $|a_{nn}| \le \varepsilon$, proses gagal (singular). Selesai.

$x_n \Leftarrow a_{n,n+1}/a_{nn}$

III. Untuk $k = n-1, n-2, \ldots, 1$, lakukan:

jumlah $\Leftarrow$ 0

Untuk $j = k+1, k+2, \ldots, n$, lakukan:

jumlah $\Leftarrow$ jumlah + $a_{kj} * x_j$

$x_k \Leftarrow (a_{k,n+1}$ - jumlah) $/a_{kk}$

4.5 Metode Dekomposisi/Faktorisasi Segitiga

Suatu matriks A taksingular dapat difaktorkan menjadi hasilkali suatu matriks segitiga bawah L dan matriks segitiga atas U. Metode ini dikenal dengan nama metode dekomposisi LU atau metode faktorisasi segitiga.

Ada dua macam pemfaktoran A menjadi LU yaitu:

1. Metode pemfaktoran Doolittle
2. Metode pemfaktoran Crout

4.5.1 Pemfaktoran Doolittle, mensyaratkan elemen diagonal L semuanya 1 dan elemen diagonal U taknol.

Misalkan untuk matriks A(3x3) dapat ditulis sebagai

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix}$$

4.5.2 Pemfaktoran Crout, mensyaratkan elemen diagonal L taknol dan semua elemen diagonal U bernilai 1.

Misalkan untuk matriks A(3x3) dapat ditulis sebagai

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = \begin{pmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{pmatrix} \begin{pmatrix} 1 & u_{12} & u_{13} \\ 0 & 1 & u_{23} \\ 0 & 0 & 1 \end{pmatrix}$$

Algoritma Pemfaktoran Doolittle

Masukkan : n, a_{ij} , i, j = 1, 2, ..., n.

Keluaran : L_{nxn} , U_{nxn}

Langkah-langkah:

I. Untuk j = 1, 2, ... , n

$u_{1j} \leftarrow a_{1j}$

$l_{jj} \leftarrow 1$

Untuk i = 2, 3, ..., n

Untuk j = 1, 2, ..., i-1

Suku $\leftarrow$ 0

Untuk k = 1, 2, ... , j-1

Suku $\leftarrow$ suku + l_{ik} . u_{kj}

lij $\leftarrow$ (a_{ij} - suku)/u_{jj}

Untuk k = i, i+1, ... , n

Suku $\leftarrow$ 0

Untuk m = 1, 2, ... , i - 1

Suku $\leftarrow$ suku + l_{im} . u_{mk}

$u_{ik} \leftarrow a_{ik}$ - suku

II. LY = C dengan substitusi maju

III. UX = Y dengan substitusi mundur

4.6 Metode Iterasi Jacobi dan Gauss-Seidel

SPL $AX = C$ dapat diselesaikan dengan metode ini sehingga konvergen, apabila matriks koefisien A memenuhi syarat cukup yaitu *dominan secara diagonal:*

$$|a_{ii}| > \sum_{j=1, j\neq i}^{n} |a_{ij}|$$

untuk $i = 1, 2, 3, \ldots, n$.

Contoh 4.5

Matriks $\begin{pmatrix} 5 & 1 & 1 \\ 1 & 4 & 2 \\ -1 & 2 & 6 \end{pmatrix}$ adalah dominan secara diagonal

Karena $|5| > |1| + |1|$

$|4| > |1| + |2|$

$|6| > |-1| + |2|$

Pandang SPL:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &= b_2 \\ \vdots \qquad\qquad & \quad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n &= b_n \end{aligned}$$

dengan matriks koefisiennya dominan secara diagonal.

Untuk mencari hampiran solusinya disarankan bentuk-bentuk iteratif berikut:

$$a_{11}x_1 = b_1 - (a_{12}x_2 + a_{13}x_3 + \ldots + a_{1n}x_n)$$

$$x_1 = \frac{b_1 - (a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n)}{a_{11}}$$

$$a_{22}x_2 = b_2 - (a_{21}x_1 + a_{23}x_3 + \ldots + a_{2n}x_n)$$

$$x_2 = \frac{b_2 - (a_{21}x_1 + a_{23}x_3 + \cdots + a_{2n}x_n)}{a_{22}}$$

$$\vdots$$

$$a_{nn}x_n = b_n - (a_{n1}x_1 + a_{n2}x_2 + \; . \; . \; . \; + a_{n,n-1}x_{n-1})$$

$$x_n = \frac{b_n - (a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{n,n-1}x_{n-1})}{a_{nn}}$$

Misalkan diberikan nilai awal ($x_1^{\circ}, x_2^{\circ}, \ldots, x_n^{\circ}$), persamaan iterasi Jacobi dan Gauss-Seidelnya sebagai berikut.

4.6.1 Metode Iterasi Jacobi

Bentuk umum proses iteratif Jacobi adalah

$$x_i^{k+1} = \frac{b_i - \sum\limits_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} x_j^k}{a_{ii}} \qquad \textit{untuk} \;\; i = 1,2,\ldots,n \;\; \textit{dan} \;\; k = 0,1,2,\ldots$$

4.6.2 Metode Iterasi Gauss-Seidel

Kekonvergenan metode iterasi Jacobi agak lambat. Kekonvergenan ini dapat dipercepat bila setiap harga x_i yang baru dihasilkan segera dipakai pada persamaan berikutnya untuk menentukan harga x_{i+1} yang lainnya. Teknik inilah yang dipakai pada metode iterasi Gauss-Seidel.

Bentuk umum proses iteratif Gauss-Seidel adalah

$$x_i^{k+1} = \frac{b_i - \sum\limits_{j=1}^{i-1} a_{ij} x_j^{k+1} - \sum\limits_{j=i+1}^{n} a_{ij} x_j^k}{a_{ii}}$$

untuk $i = 1, 2, \ldots, n$ dan $k = 0, 1, 2, \ldots$