Jika saya munculkan indeks i pada bagian proses triangularisasi

```
%==== Proses Triangularisasi ====
%---- menghilangkan variabel x1 dari P2, P3 dan P4 ----
i = 1;
for j = i+1:4
     m=A(j,i)/A(i,i);
          for k = 1:5
               A(j,k) = A(j,k)-m*A(i,k);
          end
end
%---- menghilangkan variabel x2 dari P3 dan P4 ----
i = 2;
for j = i+1:4
     m=A(j,i)/A(i,i);
          for k = 1:5
               A(j,k) = A(j,k)-m*A(i,k);
          end
end
%---- menghilangkan variabel x3 dari P4 ----
i = 3;
for j = i+1:4
     m=A(j,i)/A(i,i);
          for k = 1:5
               A(j,k) = A(j,k)-m*A(i,k);
          end
end
```

**3.4. MATRIK DAN ELIMINASI GAUSS**

maka saya bisa gabungkan semua i tersebut menjadi

```
%==== Proses Triangularisasi ====
for i = 1:3
     for j = i+1:4
          m=A(j,i)/A(i,i);
               for k = 1:5
                    A(j,k) = A(j,k)-m*A(i,k);
               end
     end
end
```

Sehingga hasil optimasi sampai tahapan ini telahmengecilkan jumlah baris statemen dari semula70 baris menjadi hanya 34 baris saja. Inilah hasilnya

```
clear all
clc

%---- inisialisasi matrik A ----
A = [1 1 0 3
2 1 -1 1
```

```
3 -1 -1 2
-1 2 3 -1];

%---- inisialisasi vektor b ----
b = [4 ; 1 ; -3 ; 4];

%---- membentuk matrik augmentasi ----
dim = size(A);
n = dim(1);
for i = 1:n
A(i,n+1) = b(i);
end

%==== Proses Triangularisasi ====
for i = 1:3
     for j = i+1:4
          m=A(j,i)/A(i,i);
               for k = 1:5
                    A(j,k) = A(j,k)-m*A(i,k);
               end
     end
end

%==== Proses Substitusi Mundur ====
x(4,1)=A(4,5)/A(4,4);
x(3,1)=(A(3,5)-A(3,4)*x(4,1))/A(3,3);
x(2,1)=(A(2,5)-(A(2,3)*x(3,1)+A(2,4)*x(4,1)))/A(2,2);
x(1,1)=(A(1,5)-
(A(1,2)*x(2,1)+A(1,3)*x(3,1)+A(1,4)*x(4,1)))/A(1,1);
```

**3.4.4.2 Optimasi *source code* bagian substitusi-mundur**

OK, sekarang kita beralih ke bagian substitusi-mundur. Saya mulai dengan memodifikasi bagian tersebut menjadi seperti ini

```
%==== Proses Substitusi Mundur ====
x(4,1)=A(4,5)/A(4,4);
S = 0;
S = S + A(3,4)*x(4,1);
x(3,1)=(A(3,5)-S)/A(3,3);
S = 0;
S = S + A(2,3)*x(3,1);
S = S + A(2,4)*x(4,1);
x(2,1)=(A(2,5)-S)/A(2,2);
S = 0;
S = S + A(1,2)*x(2,1);
S = S + A(1,3)*x(3,1);
S = S + A(1,4)*x(4,1);
x(1,1)=(A(1,5)-S)/A(1,1);
```

Dari situ, saya modifikasi kembali menjadi seperti ini

```
%==== Proses Substitusi Mundur ====
x(4,1)=A(4,5)/A(4,4);
S = 0;
     for k = 4:4
          S = S + A(3,k)*x(k,1);
     end
x(3,1)=(A(3,5)-S)/A(3,3);
S = 0;
     for k = 3:4
          S = S + A(2,k)*x(k,1);
     end
x(2,1)=(A(2,5)-S)/A(2,2);
S = 0;
     for k = 2:4
          S = S + A(1,k)*x(k,1);
     end
x(1,1)=(A(1,5)-S)/A(1,1);
```

Lalu saya munculkan indeks i, coba perhatikan dengan teliti

```
%==== Proses Substitusi Mundur ====
x(4,1)=A(4,5)/A(4,4);
i = 3;
S = 0;
for k = i+1:4
S = S + A(i,k)*x(k,1);
end
x(i,1)=(A(i,5)-S)/A(i,i);
i = 2;
S = 0;
for k = i+1:4
S = S + A(i,k)*x(k,1);
end
x(i,1)=(A(i,5)-S)/A(i,i);
i = 1;
S = 0;
for k = i+1:4
S = S + A(i,k)*x(k,1);
end
x(i,1)=(A(i,5)-S)/A(i,i);
```

dengan demikian saya bisa ringkas menjadi seperti ini

```
%==== Proses Substitusi Mundur ====
```

```
x(4,1)=A(4,5)/A(4,4);
for i = 3:-1:1
     S = 0;
          for k = i+1:4
               S = S + A(i,k)*x(k,1);
          end
     x(i,1)=(A(i,5)-S)/A(i,i);
end
```

Dan inilah hasil optimasi sampai tahapan yang terakhir

```
clear all
clc

%---- inisialisasi matrik A ----
A = [1 1 0 3
     2 1 -1 1
     3 -1 -1 2
     -1 2 3 -1];

%---- inisialisasi vektor b ----
b = [4 ; 1 ; -3 ; 4];

%---- membentuk matrik augmentasi ----
dim = size(A);
n = dim(1);
for i = 1:n
     A(i,n+1) = b(i);
end

%==== Proses Triangularisasi ====
for i = 1:3
     for j = i+1:4
          m=A(j,i)/A(i,i);
               for k = 1:5
                    A(j,k) = A(j,k)-m*A(i,k);
               end
     end
end

%==== Proses Substitusi Mundur ====
x(4,1)=A(4,5)/A(4,4);
for i = 3:-1:1
     S = 0;
          for k = i+1:4
               S = S + A(i,k)*x(k,1);
          end
     x(i,1)=(A(i,5)-S)/A(i,i);
end
```

### 3.4.5 Pentingnya nilai *n*

Pada baris ke-15, nilai n adalah nilai ukuran matrik A yang berbentuk bujursangkar. Dalamcontoh ini, *n* bernilai 4. Dengan menggunakan angka 4 (atau n) sebagai acuan, maka sourcecode hasil optimasi terakhir dimodifikasi kembali menjadi seperti ini

```
clear all
clc

%---- inisialisasi matrik A ----
A = [1 1 0 3
     2 1 -1 1
     3 -1 -1 2
     -1 2 3 -1];

%---- inisialisasi vektor b ----
b = [4 ; 1 ; -3 ; 4];

%---- membentuk matrik augmentasi ----
dim = size(A);
n = dim(1);
for i = 1:n
     A(i,n+1) = b(i);
end

%==== Proses Triangularisasi ====
for i = 1:n-1
     for j = i+1:n
          m=A(j,i)/A(i,i);
               for k = 1:n+1
                    A(j,k) = A(j,k)-m*A(i,k);
               end
     end
end

%==== Proses Substitusi Mundur ====
x(n,1)=A(n,n+1)/A(n,n);

for i = n-1:-1:1
S = 0;
for k = i+1:n
S = S + A(i,k)*x(k,1);
end
x(i,1)=(A(i,n+1)-S)/A(i,i);
end
```

13 Mei 2019

Sekarang, source code di atas akan bisa memprosesmatrik bujursangkar yang ukurannya sembarang;tidak hanya 4x4. Demikianlah akhir dari proses optimasi yang cukup melelahkan.

### 3.4.6 Jangan puas dulu..

Walaupun memiliki jumlah baris statemen yang lebih sedikit, *source-code* ini masih mengandung*bug* yang bisa berakibat fatal. Sekarang coba anda ganti angka-angka pada bagian inisialisasi matrik menjadi angka-angka baru yang disesuaikan dengan sistem persamaan linearberikut ini

$$P_1 : x_1 - x_2 + 2x_3 - x_4 = -8$$
$$P_2 : 2x_1 - 2x_2 + 3x_3 - 3x_4 = -20$$
$$P_3 : x_1 + x_2 + x_3 = -2$$
$$P_4 : x_1 - x_2 + 4x_3 + 3x_4 = 4$$

Saya jamin *source code* yang tadi akan berhenti sebelum tugasnya selesai. Artinya ia gagal menjalankan tugas mencari solusi sistem persamaan linear. Mengapa bisa begitu?

### 3.4.7 *Pivoting*

Pada baris ke-23, yang merupakan bagian dari proses triangularisasi dalam *source code* di atas,terdapat

```
m=A[j,i]/A[i,i]
```

elemen A[i, i] tentunya tidak boleh bernilai nol. Jika itu terjadi, maka proses triangularisasi otomatis akan berhenti dan itu sekaligus menggagalkan metode eliminasi Gauss. Dilihat dari indeks-nya yang kembar yaitu [i, i], maka tidak diragukan lagi bahwa ia pasti menempati posisidi elemen diagonal dari matrik A. Nama lain elemen ini adalah elemen *pivot*. Jadi apa yangharus dilakukan jika secara tidak disengaja didalam aliran proses terdapat elemen *pivot* yangbernilai nol?

Salah satu cara untuk mengatasinya adalah dengan menukar seluruh elemen yang sebaris dengan elemen diagonal bernilai nol. Ia harus ditukar posisinya dengan baris yang adadibawahnya, sampai elemen diagonal matrik menjadi tidak nol, $a_{ii} \neq 0$. Cara ini disebut *pivoting*.

Penambahan proses *pivoting* kedalam *source code* eliminasi Gauss dimulai dari baris ke-23 sampai baris ke-30 berikut ini

```
clear all
clc

%---- inisialisasi matrik A
----
A = [1 -1 2 -1
     2 -2 3 -3
     1  1 1  0
     1 -1 4 3];

%---- inisialisasi vektor b
----
b = [-8 ; -20 ; -2 ; 4];

%---- membentuk matrik
augmentasi ----
dim = size(A);
n = dim(1);
for i = 1:n
A(i,n+1) = b(i);
end

%==== Proses Triangularisasi
====
for i = 1:n-1
%---- awal proses pivoting -
----
if A(i,i) == 0
     for s = 1:n+1
          v = A(i,s);
          u = A(i+1,s);
          A(i,s) = u;
          A(i+1,s) = v;
     end
end
%---- akhir proses pivoting
-----

for j = i+1:n
     m=A(j,i)/A(i,i);
          for k = 1:n+1
               A(j,k) =
          A(j,k)-m*A(i,k);
          end
end
end

%==== Proses Substitusi
Mundur ====
x(n,1)=A(n,n+1)/A(n,n);

for i = n-1:-1:1
S = 0;
for k = i+1:n
S = S + A(i,k)*x(k,1);
end
x(i,1)=(A(i,n+1)-S)/A(i,i);
end
```

### 3.5 Function Eliminasi Gauss

Pendefinisian *function* eliminasi gauss, yang akan diberi nama *elgauss* merupakan langkah palingakhir dari proses optimasi *source code* ini. Berdasarkan *source code* di atas, *function* eliminasigauss bisa dimulai dari baris ke-13 hingga baris ke-50. Berikut ini adalah cara pendefinisiannya

```
function x=elgauss(A,b)
%---- membentuk matrik
augmentasi ----
dim = size(A);
n = dim(1);
for i = 1:n
A(i,n+1) = b(i);
end
%==== Proses Triangularisasi
====
for i = 1:n-1
%---- awal proses pivoting -
----
if A(i,i) == 0
for s = 1:n+1
v = A(i,s);
u = A(i+1,s);
A(i,s) = u;
A(i+1,s) = v;
end
end
%---- akhir proses pivoting
-----

for j = i+1:n
m=A(j,i)/A(i,i);
for k = 1:n+1
A(j,k) = A(j,k)-m*A(i,k);
end
end
end

%==== Proses Substitusi
Mundur ====
x(n,1)=A(n,n+1)/A(n,n);

for i = n-1:-1:1
S = 0;
for k = i+1:n
S = S + A(i,k)*x(k,1);
end
x(i,1)=(A(i,n+1)-S)/A(i,i);
end
```

Dengan adanya *function elgauss*, maka *source-code* untuk menyelesaikan sistem persamaan lineardengan metode eliminasi gauss dapat ditulis secara sangat sederhana. Berikut ini contohnya..

```
clear all
clc

%---- inisialisasi matrik A ----
A = [1 -1 2 -1
2 -2 3 -3
1 1 1 0
1 -1 4 3];

%---- inisialisasi vektor b ----
b = [-8 ; -20 ; -2 ; 4];

x=elgauss(A,b)
A
b
x
```

22 Juni 2020