LAPORAN PRAKTIKUM
METODE NUMERIK

Judul: Sistem Persamaan Linier

DISUSUN OLEH

ILHAM NUR ROMDONI M0520038

PROGRAM INFORMATIKA

FAKULTAS MIPA

UNIVERSITAS SEBELAS MARET

2021

SCREENSHOT

A. Screenshot Praktikum

1. Metode Jacobi

```
% Ilham Nur Romdoni, M0520038

function [xl, g, H]= jacobi (A,b,X0,T,N)
H = X0';
n = length(b);
X1 = X0;
for k=1:N,
    for i = 1:n,
        S = b(i)-A(i, [1:i-1.i+1:n])*X) ([1:i-1,i+1:n]);
        X1 (i) =S/A(i,i);
    end
    g = abs(X1-X0);
    err = norm(g);
    relerr = err/(norm(X1)+ eps);
    X0 = X1;
    H = [H;X0'];
if (err<T)|(relerr<T),break,end
end
```

2. Metode Gauss-Seidel

```
% Ilham Nur Romdoni, M0520038

function [X1,g,H] = seidel(A,b,X0,T,N)
H = X0';
n = length(b);
X1 = X0 ;
for k=1:N,
    for i =1:n,
        S=b(i)-A(i,1:i-1)*X1(1:i-1) A(i,i+1:n)*X0(i+1:n);
        X1(i)=S/A(i,i);
    end
    g=abs(X1-X0);
    err=norm(g);
    relerr=err/(norm(X1)+eps);
    X0=X1;
    H=[H,X0'};
    if(err<T)\(relerr<T),break,end
end
```

3. Metode Eliminasi Gauss (OBE)

```
% Ilham Nur Romdoni, M0520038

function x = EliminasiGauss (A,b)
[n,l] = size(A);
for i = 1 : n-1,
    [pivot,k] = max(abs(A(i:n, i)));
```

```
    if (k > 1)
        temp1 = A(i, :);
        temp2 = b(i, :);
        A(i,:) = A(i+k-1,:);
        b(i,:) = b(i+k-1,:);
        A(i+k-1,:) = temp1;
        b(i+k-1,:) = temp2;
    end
    for (h = i+1 : n),
        m = A(h,i)/A(i,i);
        A(h,:) = A(h,:) - m*A(i,:);
        b(h,:) = b(h,:) - m*b(i,:);
    end
end
x(n,:) = b(n,:) / A(n,n);
for (i = n:-1:1),
    x(i,:) = (b(i,:)-A(i,i+1:n)*x(i+1:n,:)) / A(i,i);
end
```

4. Metode Eliminasi Gauss-Jordan

```
%Ilham Nur Romdoni, M0520038

function x = EliminasiGaussJordan(A,b)
[n,m] = size(A);
for i = 1 : n-1,
    [pivot,k] = max(abs(A(i:n, i)));
    if (k > 1)
        temp1 = A(i, :);
        temp2 = b(i, :);
        A(i,:) = A(i+k-1,:);
        b(i,:) = b(i+k-1,:);
        A(i+k-1,:) = temp1;
        b(i+k-1,:) = temp2;
    end
    for (h = i+1 : n),
        m = A(h,i)/A(i,i);
        A(h,:) = A(h,:) - m*A(i,:);
        b(h,:) = b(h,:) - m*b(i,:);
    end
end

for i = n:-1:2
    for h = i-1:-1:1
        m = A(h,i)/A(i,i);
        A(h,:) = A(h,:)-m*A(i,:);
        b(h,:) = b(h,:)-m*b(i,:);
    end
end

for i = 1:n
    x(i,:) = b(i,:)/A(i,i);
end
```

5. **Metode Dekomposisi LU**

```
%Ilham Nur Romdoni, M0520038

function [L,U] = Doolittle (A)
[n,m] = size(A);
U = zeros (n,n);
L = eye(n);
for k = 1:n,
    U(k,k) = A(k,k) - L(k, 1:k-1)*U(1:k-1,k);
    for j = k+1:n,
        U(k,j) = A(k,j) - L(k, 1:k-1)*U(1:k-1,j);
        L(j,k) = (A(j,k) - L(j, 1:k-1)*U(1:k-1,k))/U(k,k);
    end
end
```

```
%Ilham Nur Romdoni, M0520038

function x = LU_Solusi (L,U,b)
[n,m] = size(L);
z = zeros(n,1);
x = zeros(n,1);
z(1) = b(1)/L(1,1);
for i = 2:n,
    z(i) = (b(i) - L(i, 1:i-1)*z(1:i-1)) / L(i,i);
end
x(n) = z(n)/U(n,n);
for i = n-1:-1:1,
    x(i) = (z(i) - U(i,i+1:n)*x(i+1:n)) / U(i,i);
end
```

B. Screenshot Source Code

1. Kasus 1

Metode Gauss

```
% Ilham Nur Romdoni, M0520038

function x = EliminasiGauss (A,b)
[n,l] = size(A);
for i = 1 : n-1,
    [pivot,k] = max(abs(A(i:n, i)));
    if (k > 1)
        temp1 = A(i, :);
        temp2 = b(i, :);
        A(i,:) = A(i+k-1,:);
        b(i,:) = b(i+k-1,:);
        A(i+k-1,:) = temp1;
        b(i+k-1,:) = temp2;
    end
    for (h = i+1 : n),
        m = A(h,i)/A(i,i);
        A(h,:) = A(h,:) - m*A(i,:);
        b(h,:) = b(h,:) - m*b(i,:);
    end
end
```

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

Metode Gauss-Jordan

```
%Ilham Nur Romdoni, M0520038

function x = EliminasiGaussJordan(A,b)
[n,m] = size(A);
for i = 1 : n-1,
    [pivot,k] = max(abs(A(i:n, i)));
    if (k > 1)
        temp1 = A(i, :);
        temp2 = b(i, :);
        A(i,:) = A(i+k-1,:);
        b(i,:) = b(i+k-1,:);
        A(i+k-1,:) = temp1;
        b(i+k-1,:) = temp2;
    end
    for (h = i+1 : n),
        m = A(h,i)/A(i,i);
        A(h,:) = A(h,:) - m*A(i,:);
        b(h,:) = b(h,:) - m*b(i,:);
    end
end

for i = n:-1:2
    for h = i-1:-1:1
        m = A(h,i)/A(i,i);
        A(h,:) = A(h,:)-m*A(i,:);
        b(h,:) = b(h,:)-m*b(i,:);
    end
end

for i = 1:n
    x(i,:) = b(i,:)/A(i,i);
end
```

Metode LU Dekomposisi

```
%Ilham Nur Romdoni, M0520038

function [L,U] = Doolittle (A)
[n,m] = size(A);
U = zeros (n,n);
L = eye(n);
for k = 1:n,
    U(k,k) = A(k,k) - L(k, 1:k-1)*U(1:k-1,k);
    for j = k+1:n,
        U(k,j) = A(k,j) - L(k, 1:k-1)*U(1:k-1,j);
        L(j,k) = (A(j,k) - L(j, 1:k-1)*U(1:k-1,k))/U(k,k);
    end
end

%Ilham Nur Romdoni, M0520038

function x = LU_Solusi (L,U,b)
```

```
[n,m] = size(L);
z = zeros(n,1);
x = zeros(n,1);
z(1) = b(1)/L(1,1);
for i = 2:n,
    z(i) = (b(i) - L(i, 1:i-1)*z(1:i-1)) / L(i,i);
end
x(n) = z(n)/U(n,n);
for i = n-1:-1:1,
    x(i) = (z(i) - U(i,i+1:n)*x(i+1:n)) / U(i,i);
end
```

Metode Jacobi

```
% Ilham Nur Romdoni, M0520038

function x = IterasiJacobi(A,b)
[n m] = size(A);
for i = 1:n,
    xlama(i) = b(i)/A(i,i);
end
xlama = xlama';
C = -A;
for i = 1:n,
    C(i,i) = 0.0;
    C(i,:) = C(i,:)/A(i,i);
    d(i,1) = xlama(i);
end
i = 20;
e = 0,0001;
while (i <= 100)
    xbaru = C*xlama + d;
    if (norm(xbaru - xlama) <= e)
        x = xbaru;
        disp ('Jacobi method konverge');
        return
    else
        xlama = xbaru;
    end
    disp([i xbaru']);
    i = i + 1;
end;
disp ('Jacobi method not konverge');
```

Metode Gauss-Seidel

```
% Ilham Nur Romdoni, M0520038

function x = IterasiGaussSeidel(A,b)
[n m] = size(A);
for i = 1:n,
    if (i == 1)
        x(i) = b(i)/A(i,i);
    else
        x(i) = 0;
    end
end
```

```
x = x';
C = -A;
for i = 1:n,
    C(i,i) = 0.0;
    C(i,:) = C(i,:)/A(i,i);
    d(i,1) = x(i);
end
i = 20;
e = 0,0001;
while (i <= 100)
    xlama = x;
    for j = 1 : n,
        x(j) = C(j,:)*x + d(j);
    end
    if (norm(xlama - x) <= e)
        disp ('Gauss Seidel method konverge');
        return
    end
    disp([i x']);
    i = i + 1;
end
disp ('Gauss Seidel method not konverge');
```

2. Kasus 2

```
% Ilham Nur Romdoni, M0520038

langsung = EliminasiGaussJordan(A,b);

iteratif = IterasiJacobi(A,b);

e=abs(langsung - iteratif)
```

3. Kasus 3

```
% Ilham Nur Romdoni, M0520038

function x = EliminasiGauss (A,b)
tic;
[n,l] = size(A);
for i = 1 : n-1,
    [pivot,k] = max(abs(A(i:n, i)));
    if (k > 1)
        temp1 = A(i, :);
        temp2 = b(i, :);
        A(i,:) = A(i+k-1,:);
        b(i,:) = b(i+k-1,:);
        A(i+k-1,:) = temp1;
        b(i+k-1,:) = temp2;
    end
    for (h = i+1 : n),
        m = A(h,i)/A(i,i);
        A(h,:) = A(h,:) - m*A(i,:);
        b(h,:) = b(h,:) - m*b(i,:);
    end
```

```
end
x(n,:) = b(n,:) / A(n,n);
for (i = n:-1:1),
    x(i,:) = (b(i,:)-A(i,i+1:n)*x(i+1:n,:)) / A(i,i);
end
toc;
```

ANALISIS

A. Analisis Source Code

1. Kasus 1

Metode Gauss

Fungsi EliminasiGauss dengan parameter A dan b disimpan pada variabel x. Size (A,1) akan mengambil nilai dari size A pada kolom ke-1 dan disimpan pada variabel n. Dilakukan perulangan dari 1 hingga n-1. Terdapat nilai pivot yang merupakan nilai maksimum dari mutlak A baris I hingga n kolom i, letak pivot ditunjukkan dengan nilai k. Terdapat percabangan, jika k lebih dari 1 maka nilai A baris ke-i akan diganti dengan nilai A baris ke i+k-1 dan nilai B baris ke-i akan diganti dengan nilai B baris ke i+k-1. Terdapat perulangan nilai h dari i+1 hingga n. Nilai m adalah nilai dari matriks A baris ke h kolom ke i dibagi dengan matriks A baris ke i kolom ke i. Nilai dari matriks A baris ke h di semua kolom adalah A baris ke h di semua kolom dikurangi dengan m kali A baris ke i di semua kolom. Sedangkan nilai matriks B baris ke h di semua kolom adalah nilai B baris ke h di semua kolom dikurangi dengan m kali B baris ke i di semua kolom. Algoritma yang digunakan seperti pada *source code* di atas.

Metode Gauss-Jordan

Fungsi EliminasiGaussJordan dengan parameter A dan b disimpan pada variabel x. Size (A,1) akan mengambil nilai dari size A pada kolom ke-1 dan disimpan pada variabel n. Dilakukan perulangan dari 1 hingga n-1. Terdapat nilai pivot yang merupakan nilai maksimum dari mutlak A baris I hingga n kolom i, letak pivot ditunjukkan dengan nilai k. Terdapat percabangan, jika k lebih dari 1 maka nilai A baris ke-i akan diganti dengan nilai A baris ke i+k-1 dan nilai B baris ke-i akan diganti dengan nilai B baris ke i+k-1. Terdapat perulangan nilai h dari i+1 hingga n. Nilai m adalah nilai dari matriks A baris ke h kolom ke i dibagi dengan matriks A baris ke i kolom ke i. Nilai dari matriks A baris ke h di semua kolom adalah A baris ke h di semua kolom dikurangi dengan m kali A baris ke i di semua kolom. Sedangkan nilai matriks B baris ke h di semua kolom adalah nilai B baris ke h di semua kolom dikurangi dengan m kali B baris ke i di semua kolom. Algoritma yang digunakan seperti pada *source code* di atas.

Metode LU Dekomposisi

Fungsi LU Dekomposisi di atas menggunakan metode Doolittle untuk menemukan nila L dan U menggunakan parameter A. Algoritma yang digunakan seperti pada *source code* di atas. LU_Solusi digunakan setelah nilai L dan U ditemukan dengan fungsi Doolitle. Fungsi LU_Solusi menggunakan parameter L, U, dan b. Algoritma yang digunakan seperti pada *source code* di atas.

Metode Jacobi

Fungsi IterasiJacobi dengan parameter A, b dengan e didefinisikan dengan 0,0001 dan N adalah iterasi maksimal yaitu 100 disimpan pada variabel x. Size (A,1) akan mengambil nilai dari *size* A pada kolom ke-1 dan disimpan pada variabel n. Dilakukan perulangan dari 1 hingga n. Tiap perulangan akan mengeksekusi nilai b pada baris ke -i dibagi dengan nilai A pada baris ke-i kolom i dan hasilnya disimpan pada variabel xlama. Nilai xlama akan ditransposisi. Kemudian terdapat fungsi perulangan dimulai dari i hingga n. Setiap perulangan akan mengeksekusi nilai C pada baris ke-i kolom i = 0,0. Nilai C pada baris i untuk setiap kolom adalah Nilai C pada baris i untuk setiap kolom dibagi nilai A baris ke-i kolom i. Nilai d baris i kolom i adalah nilai dari xlama baris ke-i. Fungsi perulangan dimulai dari 1 hingga N. Nilai xbaru merupakan nilai C dikali nilai xlama ditambah dengan d. Kemudian terdapat fungsi percabangan jika nilai absolut dari xbaru-xlama kurang dari sama dengan nilai e, maka nilai x=xbaru akan menampilkan *output* “Jacobi method converge” dan perulangan berhenti. Namun, jika tidak memenuhi, maka akan menampilkan *output* “Jacobi method not converge”.

Metode Gauss-Seidel

Fungsi IterasiGaussSeidel dengan parameter A, b dengan e didefinisikan dengan 0,0001 dan N adalah iterasi maksimal yaitu 100 disimpan pada variabel x. Size (A,1) akan mengambil nilai dari *size* A pada kolom ke-1 dan disimpan pada variabel n. Dilakukan perulangan dari 1 hingga n. Tiap perulangan akan mengeksekusi nilai b pada baris ke -i dibagi dengan nilai A pada baris ke-i kolom i dan hasilnya disimpan pada variabel xlama. Nilai xlama akan ditransposisi. Kemudian terdapat fungsi perulangan dimulai dari i hingga n. Setiap perulangan akan mengeksekusi nilai C pada baris ke-i kolom i = 0,0. Nilai C pada baris i untuk setiap kolom adalah Nilai C pada baris i untuk setiap kolom dibagi nilai A baris ke-i kolom i. Nilai d baris i kolom i adalah nilai dari xlama baris ke-i. Fungsi perulangan dimulai dari 1 hingga N. Nilai x merupakan nilai C dikali nilai xlama ditambah dengan d. Kemudian terdapat fungsi percabangan jika nilai absolut dari xlama-x kurang dari sama dengan nilai e, maka nilai x akan menampilkan *output* “Gauss Seidel method converge” dan perulangan berhenti. Namun, jika tidak memenuhi, maka akan menampilkan *output* “Gauss Seidel method not converge”.

2. **Kasus 2**

   Perhitungan nilai *error* menggunakan nilai mutlak dari nilai salah satu metode langsung – salah satu metode iteratif. Pada praktikum ini didefinisikan langsung adalah nilai dai EliminasiGaussJordan (A,b) – IterasiJacobi (A,b). Kemudian hasil *error* akan ditampilkan.

3. **Kasus 3**

   Ketikkan tic dan toc yang berguna untuk menghitung waktu dari eksekusi program atau running time. Cara penulisaannya tic; algoritma toc;.

B. Analisis Jalannya Program

1. **Kasus 1**

   a. **2x – 6y – z = -38**

      -3x – y + 7z = -34

      -8x + y – 2z = -20

```
>> A = [2,-6,-1;-3,-1,7;-8,1,-2]

A =

     2    -6    -1
    -3    -1     7
    -8     1    -2

>> b = [-38;-34;-20]

b =

   -38
   -34
   -20
```

Metode Gauss

```
>> EliminasiGauss (A,b)

ans =

     4
     8
    -2
```

Metode Gauss-Jordan

```
>> EliminasiGaussJordan(A,b)

ans =

     4
     8
    -2
```

Metode LU Dekomposisi

```
>> [L,U] = Doolittle (A)

L =

    1.0000         0         0
   -1.5000    1.0000         0
   -4.0000    2.3000    1.0000


U =

    2.0000   -6.0000   -1.0000
         0  -10.0000    5.5000
         0         0  -18.6500

>> LU_Solusi (L,U,b)

ans =

    4.0000
    8.0000
   -2.0000
```

Metode Jacobi

```
>> IterasiJacobi(A,b)

e =

     0

    20    88   161   103

   1.0e+55 *

    0.0000   -8.3111   -5.8243    1.4261

Jacobi method not konverge
```

Metode tak konvergen maka matriks diubah menjadi *diagonally dominant*.

```
>> A = [-8,1,-2;2,-6,-1;-3,-1,7]

A =

    -8     1    -2
     2    -6    -1
    -3    -1     7
```

```
>> b = [-20;-38;-34]

b =

   -20
   -38
   -34

>> IterasiJacobi(A,b)

e =

     0

   20.0000    4.5060    7.9762   -2.8810

   53.0000    4.0000    8.0000   -2.0000

    54     4     8    -2

Jacobi method konverge

ans =

     4
     8
    -2
```

Penyelesaian SPL ditemukan pada iterasi ke-54.

Metode Gauss-Seidel

```
>> IterasiGaussSeidel(A,b)

e =

     0

   20.0000    2.5000    0.8333    1.1905

  100.0000    2.3056    0.5898    1.0724

Gauss Seidel method not konverge

ans =

    2.3056
    0.5898
    1.0724
```

Walaupun sudah pada iterasi maksimum, penyelesaian belum ditemukan.

b. **2a – b + 10c = -11**

3b – c + 8d = -11

10a – b + 2c = 6

-a + 11b – c + 3d = 25

```
>> A = [2,-1,10,0;0,3,-1,8;10,-1,2,0;-1,11,-1,3]

A =

     2    -1    10     0
     0     3    -1     8
    10    -1     2     0
    -1    11    -1     3

>> b = [-11;-11;6;25]

b =

   -11
   -11
     6
    25
```

Metode Gauss

```
>> x = EliminasiGauss (A,b)

x =

    1.1039
    2.9965
   -1.0211
   -2.6263
```

Metode Gauss-Jordan

```
>> x = EliminasiGaussJordan(A,b)

x =

    1.1039
    2.9965
   -1.0211
   -2.6263
```

Metode LU Dekomposisi

```
>> [L,U] = Doolittle (A)

L =

    1.0000         0         0         0
         0    1.0000         0         0
    5.0000    1.3333    1.0000         0
   -0.5000    3.5000   -0.1607    1.0000


U =

    2.0000   -1.0000   10.0000         0
         0    3.0000   -1.0000    8.0000
         0         0  -46.6667  -10.6667
         0         0         0  -26.7143
```

```
>> x = LU_Solusi (L,U,b)

x =

    1.1039
    2.9965
   -1.0211
   -2.6263
```

Metode Jacobi

Metode tak konvergen maka matriks diubah menjadi *diagonally dominant*.

```
>> A = [10,-1,2,0;-1,11,-1,3;2,-1,10,0;0,3,-1,8]

A =

    10    -1     2     0
    -1    11    -1     3
     2    -1    10     0
     0     3    -1     8

>> b = [6;25;-11;-11]

b =

     6
    25
   -11
   -11

>> x = IterasiJacobi(A,b)

e =

     0

   20.0000    1.0473    2.6023   -0.9927   -2.3648

   27.0000    1.1038    2.9965   -1.0212   -2.6261

   28.0000    1.1039    2.9964   -1.0211   -2.6263

   29.0000    1.1039    2.9965   -1.0211   -2.6263

   30.0000    1.1039    2.9965   -1.0211   -2.6263

  100.0000    1.1039    2.9965   -1.0211   -2.6263

Jacobi method not konverge
```

Walaupun sampai pada iterasi ke-29 dan seterusnya sudah menemukan hasil yang sangat mendekati penyelesaian, tetapi pada iterasi maksimal belum juga ditemukan hasil yang spesifik pada nilai penyelesaian.

Metode Gauss-Seidel

```
>> x = IterasiGaussSeidel(A,b)

e =

     0

   20.0000    0.6000    0.0545   -0.1145   -0.0348

   35.0000    0.6297    0.0561   -0.1203   -0.0361

Gauss Seidel method konverge

x =

    0.6297
    0.0561
   -0.1203
   -0.0361
```

Hasil penyelesaian ditemukan, tetapi sangat jauh dari hasil penyelesaian keempat metode lain.

c. **$6p - q - r = -1$**

$-p + 5q - r - s = 2$

$-p - q + 4r - s - t = 6$

$r + 4s - 2t = 2$

$q - r + s + 4t = -1$

Metode Gauss

```
>> x = EliminasiGauss (A,b)

x =

    0.2657
    0.8175
    1.7767
    0.0451
   -0.0215
```

Metode Gauss-Jordan

```
>> EliminasiGaussJordan(A,b)

ans =

    0.2657
    0.8175
    1.7767
    0.0451
   -0.0215
```

Metode LU Dekomposisi

```
>> [L,U] = Doolittle (A)

L =

    1.0000         0         0         0         0
   -0.1667    1.0000         0         0         0
   -0.1667   -0.2414    1.0000         0         0
         0         0    0.2816    1.0000         0
         0    0.2069   -0.2136    0.2165    1.0000


U =

    6.0000   -1.0000   -1.0000         0         0
         0    4.8333   -1.1667   -1.0000         0
         0         0    3.5517   -1.2414   -1.0000
         0         0         0    4.3495   -1.7184
         0         0         0         0    4.1585

>> LU_Solusi (L,U,b)

ans =

    0.2657
    0.8175
    1.7767
    0.0451
   -0.0215
```

Metode Jacobi

```
>> IterasiJacobi(A,b)

e =

     0

   20.0000    0.1500    0.7667    1.6208         0   -0.1000

   58.0000    0.2657    0.8175    1.7767    0.0451   -0.0215

Jacobi method konverge

ans =

    0.2657
    0.8175
    1.7767
    0.0451
   -0.0215
```

Pada iterasi ke-58 ditemukan hasil penyelesaian.

Metode Gauss-Seidel

```
>> IterasiGaussSeidel(A,b)

e =

     0

   20.0000   -0.1667   -0.0333   -0.0500    0.0125   -0.0073

   46.0000   -0.1836   -0.0456   -0.0558    0.0113   -0.0054

Gauss Seidel method konverge

ans =

   -0.1836
   -0.0456
   -0.0558
    0.0113
   -0.0054
```

Pada metode Gauss-Seidel ditemukan hasil penyelesaian tetapi dengan hasil yang jauh dari keempat metode lain.

Kesimpulan dari ketiga soal di atas himpunan penyelesaian dari metode Gauss-Seidel selalu ditemukan berbeda dengan keempat metode lain.

2. **Kasus 2**

   a. **2x – 6y – z = -38**

   -3x – y + 7z = -34

   -8x + y – 2z = -20

```
>> Hitung_error

e =

     0

   20.0000    4.5060    7.9762   -2.8810

    54     4     8    -2

Jacobi method konverge

e =

     0
     0
     0
```

Menunjukkan himpunan penyelesaian dari metode langsung dan iteratif sama.

b. **2a – b + 10c = -11**

3b – c + 8d = -11

10a – b + 2c = 6

-a + 11b – c + 3d = 25

```
>> Hitung_error

e =

     0

   20.0000    0.6000    0.0545   -0.1145   -0.0348

iteratif =

    0.6297
    0.0561
   -0.1203
   -0.0361


e =

    0.4742
    2.9404
    0.9008
    2.5902
```

Error ditemukan agak tinggi karena untuk metode iteratif menggunakan hasil penyelesaian metode Gauss-Seidel.

c. **6p – q – r = -1**

-p + 5q – r – s = 2

-p – q + 4r – s – t = 6

r + 4s – 2t = 2

q – r + s + 4t = -1

```
e =

   1.0e-15 *

    0.0555
    0.1110
         0
    0.0347
    0.0139
```

Error menunjukkan hasil penyelesaian mendekati tetapi tidak sama persis.

3. **Kasus 3**

   a. **$2x - 6y - z = -38$**

      $-3x - y + 7z = -34$

      $-8x + y - 2z = -20$

      Metode Gauss

```
>> EliminasiGauss (A,b)
Elapsed time is 0.013499 seconds.
```

      Metode Gauss-Jordan

```
>> EliminasiGaussJordan(A,b)
Elapsed time is 0.002681 seconds.
```

      Metode LU Dekomposisi

```
>> [L,U] = Doolittle (A)
Elapsed time is 0.009916 seconds.
```

```
>> LU_Solusi (L,U,b)
Elapsed time is 0.000829 seconds.
```

      Metode Jacobi

```
Elapsed time is 0.007788 seconds.
    54      4      8     -2
```

      Metode Gauss-Seidel

```
Gauss Seidel method not konverge
Elapsed time is 0.012029 seconds.
```

   b. **$2a - b + 10c = -11$**

      $3b - c + 8d = -11$

      $10a - b + 2c = 6$

      $-a + 11b - c + 3d = 25$

      Metode Gauss

```
>> EliminasiGauss (A,b)
Elapsed time is 0.002521 seconds.
```

      Metode Gauss-Jordan

```
>> EliminasiGaussJordan(A,b)
Elapsed time is 0.003030 seconds.
```

      Metode LU Dekomposisi

```
>> [L,U] = Doolittle (A)
Elapsed time is 0.002906 seconds.
```

```
>> LU_Solusi (L,U,b)
Elapsed time is 0.000663 seconds.
```

Metode Jacobi

```
Elapsed time is 0.012069 seconds.
  100.0000    1.1039    2.9965   -1.0211   -2.6263

Jacobi method not konverge
Elapsed time is 0.012345 seconds.
```

Metode Gauss-Seidel

```
Elapsed time is 0.005087 seconds.
Gauss Seidel method konverge

ans =

    0.6297
    0.0561
   -0.1203
   -0.0361
```

c. $6p - q - r = -1$

$-p + 5q - r - s = 2$

$-p - q + 4r - s - t = 6$

$r + 4s - 2t = 2$

$q - r + s + 4t = -1$

Metode Gauss

```
>> x = EliminasiGauss (A,b)
Elapsed time is 0.004318 seconds.
```

Metode Gauss-Jordan

```
>> EliminasiGaussJordan(A,b)
Elapsed time is 0.010241 seconds.
```

Metode LU Dekomposisi

```
>> [L,U] = Doolittle (A)
Elapsed time is 0.008157 seconds.
```

```
>> LU_Solusi (L,U,b)
Elapsed time is 0.004404 seconds.
```

Metode Jacobi

```
Elapsed time is 0.008461 seconds.
Jacobi method konverge

ans =

    0.2657
    0.8175
    1.7767
    0.0451
   -0.0215
```

Metode Gauss-Seidel

```
Elapsed time is 0.008580 seconds.
Gauss Seidel method konverge

ans =

   -0.1836
   -0.0456
   -0.0558
    0.0113
   -0.0054
```

Dapat disimpulkan dari semua metode, *running time* yang paling singkat adalah menggunakan metode Gauss dan yang paling lama adalah LU Dekomposisi karena menjalankan dua fungsi algoritma. Tetapi dalam singkat lamanya *running time*, sebenarnya ditentukan dari seberapa efektif algoritma yang disusun oleh pemrogram.