LAPORAN

PRAKTIKUM KOMPUTASI BIOMEDIS

Chapter 5 : System of Linier Equation: Jacobi & Gauss Seidel Iteration

Pelaksanaan Praktikum:

Hari: Selasa | Tanggal: 10 September 2019 | Jam ke: 9-10

Oleh:

Nama : M. Thoriqul Aziz E

NIM : 081711733002

Dosen Pembimbing : Endah Purwanti, S. Si, M. T.

LABORATORIUM KOMPUTER

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS AIRLANGGA

SURABAYA

2019

A. TUJUAN

Mahasiswa dapat menemukan solusi sistem persamaan linier menggunakan Metode Iterasi Jacobi dan Metode Iterasi Gauss Seidel.

B. DASAR TEORI

Metode untuk mencari nilai akar sebuah persamaan, selain menggunakan metode eliminasi Gauss, maka dapat juga menggunakan iterasi Jacobi atau metode iterasi Gauss-Seidel. Perbedaan metode iterasi dengan metode eliminasi yaitu pada metode iterasi menggunakan tebakan akar awal yang kemudian dimasukkan dalam perhitungan berulang dengan batas toleransi eror atau banyak iterasi. Sedangkan pada metode eliminasi, menggunakan eliminasi anggota matriks sedemikian sehingga terbentuk matriks segitiga atas yang kemudian melakukan subtitusi balik untuk mencari nilai tiap akarnya.

Metode iterasi Jacobi dan Gauss-Seidel secara algortima sama, akan tetapi berbeda pada penggunaan nilai tebakan akar. Pada metode iterasi Jacobi, tebakan akar yang digunakan akan akar dari iterasi sebelumnya, sedangkan pada metode iterasi Gauss-Seidel, akar yang digunakan adalah akar hasil baru yang telah diupdate. Sehingga, metode Gauss-Seidel mampu menemukan nilai tebakan akar jumlah iterasi lebih sedikit dibanding metode Jacobi pada nilai toleransi eror yang sama. Berikut adalah rumus matematika yang digunakan untuk metode Jacobi dan metode Gauss Seidel:

$$x_i^{k+1} = \frac{b_i - \sum_{j=i,j\neq i}^{n} a_{ij}x_j^k}{a_{ii}}\ , k = 0,1,2,3,\ldots (Rumusan\ Iterasi\ Jacobi)$$

$$x_i^{k+1} = \frac{b_i - \sum_{j=i,j\neq i}^{n} a_{ij}x_j^{k+1} - \sum_{j=i+1}^{n} a_{ij}x_j^k}{a_{ii}},\ k = 0,1,2,3,\ldots (Rumusan\ Iterasi\ Gauss - Seidel)$$

C. TUGAS

1. Define a biomedical problem of “Drug development and toxicity studies: animal-on-a-chip”.(Reference:King M.R and Mody N.A, 2010, “Numerical and Statistical Methods for Bioengineering”, Cambridge University Press, New York, page 48) by using Jacobi iteration method and Gauss-Seidel Iteration method! Analyze the advantages and disadvantages of both methods!

2. Contruction of Diet

   A doctor suggest a patient to follow a diet program based on the table below.

Amounts(gr) supplied per 100 gr of ingredients				Amounts(gr) supplied by Cambridge Diet in One Day
Nutrient	Non-fat milk	Soy flour	Whey	
Protein	36	51	13	33
Carbohydrate	52	34	74	45
Fat	0	7	11	3

   Please find how much non-fat milk-soy flour, and whey that are needed to fulfill the amounts of protein carbohydrate, and fat each day ideally?

3. Electrical Circuits

   Please define i_{12}, i_{25}, i_{32}, i_{56}, i_{54}, i_{43}, V_2, V_3, V_4, V_5, if the following information is known.

   R_{12}=5Ω; R_{23}=10Ω; R_{34}=5Ω;

   R_{45}=15Ω; R_{52}=10Ω; R_{12}=20Ω

   V_1=200V; V_6=0V

D. PEMBAHASAN

1. Permasalahan yaitu dari soal yang diberikan di buku tersebut, terdapat sebuah permasalahan tentang peredaran obat dalam sel yang mana menggunakan persmaan linier. Dari persamaan tersebut, kemudian disederhanakan dan disusun dalam bentuk matriks. Hasil dari matriks tersebut kemudian dilakukan pencarian nilai solusi persamaannya menggunakan metode Iterasi Jacobi dan metode Iterasi Gauss-Seidel. Algortima dari kedua metode hampir sama, yang membedakan adalah penggunaan nilai hasil ter-update dalam skema perhitunganya. Hasil hitungan matematis yang sudah diketahui penyederhanaan persmaannnya adalah :

$$C_{lung} = x_1 \; ; \; C_{liver} = x_2$$

$$(2 - 1.5R)x_1 - 0.5Rx_2 = 779.3 - 780R(Eq. 1)$$

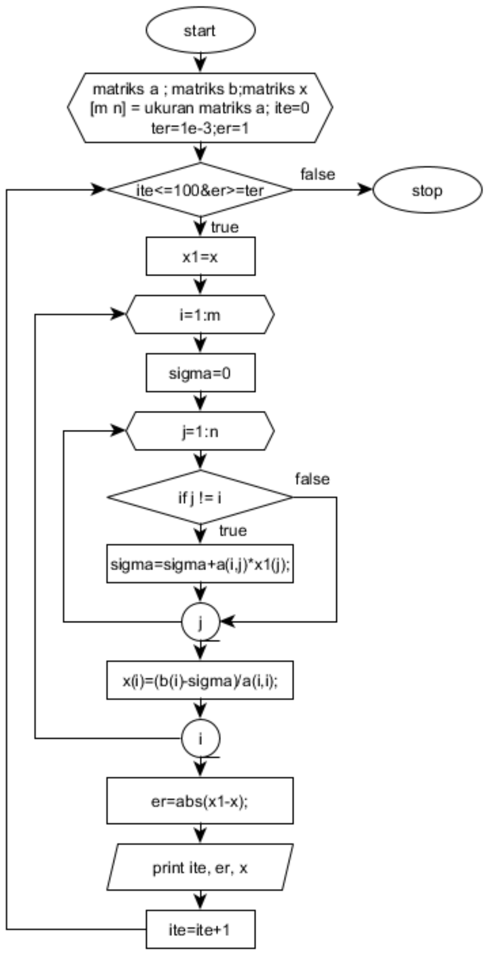
$$x_1 - x_2 = 76 \; (Eq. 2)$$

Dari persamaan tersebut, kemudian disusun matriks seperti dibawa ini :

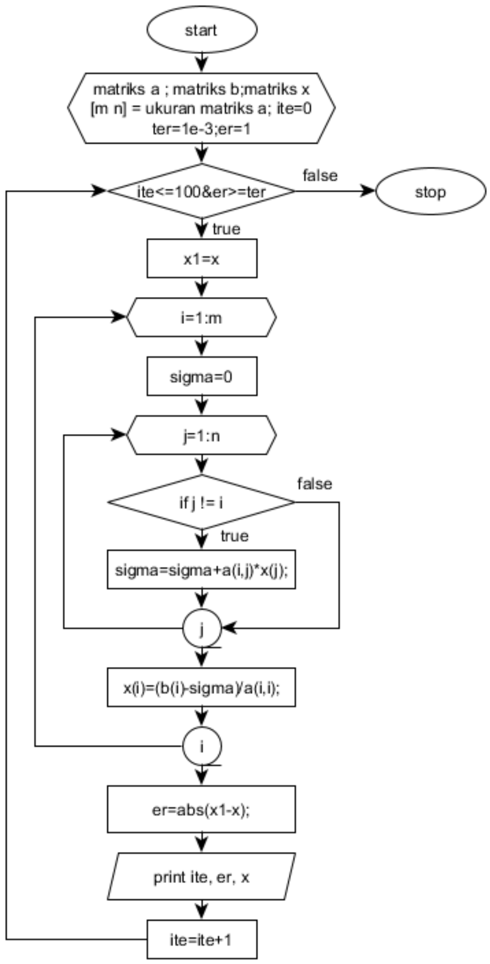
$$a = \begin{bmatrix} (2-1.5R) & -0.5R \\ 1 & -1 \end{bmatrix}$$

$$b = \begin{bmatrix} 779.3 - 780R \\ 76 \end{bmatrix}$$

Dan berikut adalah flowchart penyeselesaian menggunakan metode Jacobi dan Gauss-Seidel:



Flowchart Iterasi Jacobi



flowchart Iterasi Gauss Seidel

Berikut kode program dalam IDE Octave 5.10:

Metode iterasi Jacobi:

```
R=0.6;
a=[(2-1.5*R) -0.5;1 -1];
b=[779.3-780*R 76];
x=[0 0];
[m n]= size(a);
ite=0;
ter=1e-3;
er=1;
fprintf("ite       eror x1         eror x2              x1                x2\n")
 while (ite<=100&er>=ter)
 x1=x;
 for i=1:m
   sigma=0;
   for j=1:n
     if j!=i
       sigma=sigma+a(i,j)*x1(j);
     endif
   endfor
   x(i)=(b(i)-sigma)/a(i,i);
 endfor
 er=abs(x1-x);
 ite++;
 fprintf(" %d      %f      %f        %f      %f     \n", ite,er,x(2),x(1));
endwhile
```

Metode Iterasi Gauss-Seidel:

```
R=0.6;
a=[(2-1.5*R) -0.5;1 -1];
b=[779.3-780*R 76];
x=[0 0];
[m n]= size(a);
ite=1;
ter=1e-3;
er=1;
fprintf("ite       eror x1         eror x2              x1                x2\n")
 while (ite<=100&er>=ter)
 x1=x;
   for i=1:m
     sigma=0;
     for j=1:n
       if j!=i
         sigma=sigma+a(i,j)*x(j);
       endif
     endfor
     x(i)=(b(i)-sigma)/a(i,i);
   endfor
 er=abs(x1-x);
 ite++;
 fprintf(" %d      %f      %f        %f      %f     \n", ite,er,x(2),x(1));
endwhile
```

Pada syntax

```
R=0.6;
a=[(2-1.5*R) -0.5;1 -1];
b=[779.3-780*R 76];
x=[0 0];
```

menunjukkan nilai awala dari matrix yang akan dioprasikan dalam metode Jacobi maupun Gauss-Seidel. Variable R adalah variable ketebalan kompartemen menurut persoalan dengan nilai literature 0,6. Variable a adalah matriks 2x2 yang merupakan matriks koerfisien, variable b adalah matriks hasil dan variable x merupakan matriks tebakan awal dari nilai akar.

Pada syntax:

```
[m n]= size(a);
ite=0;
ter=1e-3;
er=1;
fprintf("ite       eror x1        eror x2            x1
x2\n")
```

matriks a terlebih dahulu harus didefinisikan ordonya dengan bantuan variable m dan n sedemikian sehingga memudahkan dalam pemberian batas perhitungan nantina. Kemudian variable ite diberikan nilai 0 sebgai nilai mula mula iterasi. Variable ter adalaha toleransi eror yang diberikan nilai 0,001 sedemikian sehingga akan menjadi syarat dari proses preulangan dibawahnya dan variable er adalah nilai eror mula mulayang bernilai 1. Syntax `fprintf` berfungsi dalam untuk pembuatan kolom tabel hasil iterasi.

Pada syntax:

```
while (ite<=100&er>=ter)
  x1=x;
```

menunjukan perulangan batas iterasi dan nilai eror, dimana perulangan `while` akan terus berulang sedemikian sehingga iterasi telah berjumlah lebih dari sama dengan 100 atau nilai eror sudah lebih kecil dari toleransi erornya. Syntax ini dalam algoritma menunjukan nilai akar pada tiap iterasinya. Kemudian `x1=x;`

menunjukan nilai matriks sebelum iterasi saat ke n dengan nilai matriks setelahnya.

Pada syntax:

```
for i=1:m
    sigma=0;
```

menunjukan perulangan pada baris ke i dari baris 1 hingga ke m dimana didalam perulangan tersebut didefinisikan terlebih dahulu nilai sigma=0. Nilai sigma ini akan berfungsi sebagai nilai jumlah pada rumus iterasi Jacobi maupun Gauss-Seidel.

Pada syntax:

```
for j=1:n
  if j!=i
   sigma=sigma+a(i,j)*x1(j);%untuk Jacobi
    sigma=sigma+a(i,j)*x(j);%untuk Gauss-Seidel
  endif
```

pada syntax `for j=1:n` menunjukan perulangan kolom matriks dari kolom pertama hingga kolom ke n yang mana perulangan ini terdapat didalam 2 perulangan sebeelumnya yaitu perulangan `while` dan perulangan baris. Kemudian dalam rumusan Jacobi maupun Gauss-Seidel mengharuskan adanya hitungan tanpa melibatkan nilai koefisiien matriks dari nilai akar yang dicari sehingga digunakan `if j!=i` yang berarti statement yang akan dilakukan ketika seleksi benar pada nilai j bukan sama dengan i. lalu, pada statement didalam if adalah syntax yang membedakan iterasi Jacobi dan iterasi Gauss-Seidel. Pada iterasi Jacobi, syntax program yang digunakan adalah `sigma=sigma+a(i,j)*x1(j);` dimana nilai sigma akan diupdate dengan mengkalikan angota matriks a pada baris i kolom j dengan nilai x1 atau nilai tebakan akar sebelumnya. Sedangkan pada iterasi Gauss-Seidel, syntax yang digunakan yaitu `sigma=sigma+a(i,j)*x(j);` dimana nilai sigma yang diupdate hasil perulangan adalah nilai sigma sebelumnya ditambah dengan hasil kali antaran anggota matriks a pada baris i kolom j dengan nilai tebakan akar dari hasil tebakan akar sebelumnya yaitu menggunakan nilai x ke j-1. Jika nilai x tersebut tidak ada dalam perulangan `for i=1:m` , maka akan digunakan adalah nilai dari x1 pada baris atau kolom yang sama.

Pada syntax:

```
x(i)=(b(i)-sigma)/a(i,i);
```

update nilai sigma sebelumnya pada perulangan kolom, kemudian akan digunakan untuk menebak nilai akar baru sesuai dengan rumusan iterasi Jacobi maupun Gauss-Seidel.

Pada syntax:

```
er=abs(x1-x);
  ite++;
  fprintf(" %d       %f       %f         %f      %f    \n",
ite,er,x(2),x(1));
```

sebelum perulangan while berakhir, maka sebelumnya harus diupdate terlebih dahulu nilai eror dan nilai iterasi baru, dimana kedua hal tersebut merupakan nilai penentu perulangan while. Apakah perulangan setelahnya berhenti atau tetap berlanjut. Kemudian syntax fprintf berfungsi untuk menampilkan semua nilai dalam tiap perulangan pada sebuah tabel. Dari algoritma tersebut diperoleh hasil pada *command window* iterasi Jacobi:

Command Window

ite	eror x1	eror x2	x1	x2
1	283.000000	76.000000	-76.000000	283.000000
2	34.545455	283.000000	207.000000	248.454545
3	128.636364	34.545455	172.454545	377.090909
4	15.702479	128.636364	301.090909	361.388430
5	58.471074	15.702479	285.388430	419.859504
6	7.137491	58.471074	343.859504	412.722014
7	26.577761	7.137491	336.722014	439.299775
8	3.244314	26.577761	363.299775	436.055461
9	12.080800	3.244314	360.055461	448.136261
10	1.474688	12.080800	372.136261	446.661573
11	5.491273	1.474688	370.661573	452.152846
12	0.670313	5.491273	376.152846	451.482533
13	2.496033	0.670313	375.482533	453.978566
14	0.304688	2.496033	377.978566	453.673879
15	1.134561	0.304688	377.673879	454.808439
16	0.138494	1.134561	378.808439	454.669945
17	0.515709	0.138494	378.669945	455.185654
18	0.062952	0.515709	379.185654	455.122702
19	0.234413	0.062952	379.122702	455.357116
20	0.028615	0.234413	379.357116	455.328501
21	0.106552	0.028615	379.328501	455.435053
22	0.013007	0.106552	379.435053	455.422046
23	0.048433	0.013007	379.422046	455.470478
24	0.005912	0.048433	379.470478	455.464566
25	0.022015	0.005912	379.464566	455.486581
26	0.002687	0.022015	379.486581	455.483894
27	0.010007	0.002687	379.483894	455.493900
28	0.001222	0.010007	379.493900	455.492679
29	0.004549	0.001222	379.492679	455.497227
30	0.000555	0.004549	379.497227	455.496672

>>

Sedangkan pada *command window* iterasi Gauss-Seidel:

Command Window

ite	eror x1	eror x2	x1	x2
2	283.000000	207.000000	207.000000	283.000000
3	94.090909	94.090909	301.090909	377.090909
4	42.768595	42.768595	343.859504	419.859504
5	19.440270	19.440270	363.299775	439.299775
6	8.836487	8.836487	372.136261	448.136261
7	4.016585	4.016585	376.152846	452.152846
8	1.825720	1.825720	377.978566	453.978566
9	0.829873	0.829873	378.808439	454.808439
10	0.377215	0.377215	379.185654	455.185654
11	0.171461	0.171461	379.357116	455.357116
12	0.077937	0.077937	379.435053	455.435053
13	0.035426	0.035426	379.470478	455.470478
14	0.016103	0.016103	379.486581	455.486581
15	0.007319	0.007319	379.493900	455.493900
16	0.003327	0.003327	379.497227	455.497227
17	0.001512	0.001512	379.498740	455.498740
18	0.000687	0.000687	379.499427	455.499427

\>>

dari kedua hasil iterasi tersebut, dapat diketahui bahwa dengan batas toleransi eror yang sama, iterasi Gauss-Seidel mampu memberikan jumlah perulangan lebih sedikit dibanding pada iterasi Jacobi dalam menentukan nilai tebakan akar. Sehingga metode iterasi Gauss-Seidel lebih cepat dalam proses penebakan akar dibanding dengan metode iterasi Jacobi. Akan tetapi, jika dilakukan secara matematis, metode iterasi Jacobi lebih mudah untuk dioprasikan karena lebih tidak membingungkan akibat dari semua nilai akar yang digunakan adalah nilai akar sebelumnya bukan nilai akar hasil iterasi baru.

2. Permasalahan dari soal nomor 2 tersebut ialah mencari nilai yang pas dari nutrient tiap tiap jenis makanannya sedemikian sehingga dapat memenuhi kebutuhan nutrisi secara ideal. Maka dari tabel tersebut dapat disusun persamaanan matematisnya seperti dibawah ini:

   Jika nilai jumlah pada tabel nutrient didapatkan dari 100 gram komposisi maka, persamaan menjadi:

$$0.36x_1 + 0.51x_2 + 0.13x_3 = 33$$
$$0.52x_1 + 0.34x_2 + 0.74x_3 = 45$$
$$0\ x_1 + 0.07x_2 + 0.11x_3 = 3$$

   Dari persamaan tersebut, terbentuk matriks:

$$a = \begin{bmatrix} 0.36 & 0.51 & 0.13 \\ 0.52 & 0.34 & 0.74 \\ 0 & 0.07 & 0.11 \end{bmatrix}$$

$$b = \begin{bmatrix} 33 \\ 45 \\ 3 \end{bmatrix}$$

Kemudian, dari hasil matriks tersebut dapat dicari solusi persamaan dengan salah satu metode yaitu metode Jacobi. Maka flowchart yang digunakan sama dengan flowchart Jacobi sebelumnya.

Berikut kode program dalam IDE Octave 5.10:

```
4  a=[0.36 0.51 0.13;
5    0.52 0.34 0.74;
6    0 0.07 0.11;];
7  b=[33 45 3];
8  x=[3 2 1];
9  [m n]=size(a);
10 ite=0;
11 ter=1e-3;
12 er=1;
13 fprintf("ite       eror x1        eror x2      eror x3           x1               x2                x3\n")
14   while (ite<20&er>=ter)
15   x1=x;
16   for i=1:m
17     sigma=0;
18     for j=1:n
19       if j!=i
20         sigma=sigma+a(i,j)*x1(j);
21       endif
22     endfor
23     x(i)=(b(i)-sigma)/a(i,i);
24   endfor
25   er=abs(x1-x);
26   ite++;
27   fprintf(" %d      %f      %f      %f     %f      %f       %f\n", ite,er,x(1),x(2),x(3));
28 endwhile
```

Dari hasil syntax tersebut, secara program sama dengan syntax pada iterasi Jacobi sebelumnya. Hanya berbeda pada nilai dari tiap variable awalannya saja. Kemudian hasil dari *command window* yang ditampilkan sebagai berikut:

```
Command Window
ite        eror x1          eror x2        eror x3           x1              x2               x3
 1      85.472222       123.588235      25.000000      88.472222       125.588235       26.000000
 2      184.111111       185.133987      78.647059      -95.638889       -59.545752       -52.647059
 3      290.673475       452.754710      117.812537      195.034586       393.208958       65.165478
 4      683.945922       700.974954      288.116633      -488.911336       -307.765996       -222.951155
 5      1097.089970       1673.112318       446.074971      608.178634       1365.346322       223.123816
 6      2531.325078       2648.771361       1064.707839      -1923.146444       -1283.425039       -841.584023
 7      4136.903925       6188.743651       1685.581775      2213.757480       4905.318612       843.997752
 8      9376.069146       9995.648689       3938.291414      -7162.311666       -5090.330078       -3094.293662
 9      15582.663098       22911.445890       6360.867348      8420.351432       17821.115812       3266.573686
 10      34754.861553       37676.548966       14580.011021       -26334.510121       -19855.433154       -11313.437335
 11      58640.115015       84887.459303       23975.985706       32305.604894       65032.026149       12662.548371
 12      128915.228851       141867.909500       54019.292284       -96609.623957       -76835.883351       -41356.743913
 13      220486.505117       314735.868507       90279.578773       123876.881160       237899.985157       48922.834860
 14      478476.772720       533705.502802       200286.461777       -354599.891560       -295805.517645       -151363.626918
 15      828408.462389       1167705.598617       339630.774510       473808.570829       871900.080972       188267.147593
 16      1776894.044392       2006174.039941       743085.380938       -1303085.473563       -1134273.958970       -554818.233346
 17      3110416.277478       4334906.132288       1276656.207235       1807330.803915       3200632.173319       721837.973890
 18      6602131.762243       7535711.934243       2758576.629638       -4794800.958328       -4335079.760924       -2036738.655748
 19      11671744.578658       16101338.889114       4795453.049064       6876943.620330       11766259.128189       2758714.393315
 20      24541921.471739       28288065.991791       10246306.565800       -17664977.851410       -16521806.863602       -7487592.172484
warning: Matlab-style short-circuit operation performed for operator &
warning: called from
    tugas2 at line 27 column 3
>> |
```

Dari hasil tersebut, dapat diketahui bahwa nilai iterasi berhenti akibat jumlah iterasi mencapi batasnya yaitu 20 karena jika diamati bahwa nilai akar tabakannya semakin besar,. maka dapat diketahui bahwa hasil akar tersebut bersifat divergen sehingga nilai eror yang di update akan semakin besar.

3. Permasalahan sama dengan permasalah sebeleumnya yaitu menentukan solusi persamaan akan tetapi dengan kasus pada rangkaian listrik. Dilakukan pencarian solusi persamaan sedemikian dapat diketahui nilai arus dan tegangan pada beberapa komponen/ titik. Dari gambar tersebut, dapat dibuat persamaan menggunakan mesh analisis untuk menentukan nilai arus. Persamaan matematisnya sebagai berikut:

$$i_{12} + i_{52} + i_{32} = 0$$
$$i_{65} - i_{52} - i_{54} = 0$$
$$i_{43} - i_{32} = 0$$
$$i_{54} - i_{43} = 0$$
$$-15i_{54} - 5i_{43} - 10i_{32} + 10i_{52} = 0$$
$$-20i_{65} - 10i_{52} - 5i_{32} += 200$$

Dari persamaan tersebut dapat dibentuk matriks sebagai berikut:

$$a = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 10 & -10 & 0 & -15 & -5 \\ 5 & -10 & 0 & -20 & 0 & 0 \end{bmatrix}; b = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 200 \end{bmatrix}; x = \begin{bmatrix} 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{bmatrix}$$

Kemudian dengan metode iterasi Gauss-Seidel, Maka flowchart yang digunakan sama dengan flowchart Gauss-Seidel sebelumnya.

Berikut kode program dalam IDE Octave 5.10:

```
1  clc
2  history -c
3  
4  a=[1 1 1 0 0 0; 0 -1 0 1 -1 0; 0 0 -1 0 0 1;
5    0 0 0 0 1 -1; 0 10 -10 0 -15 -5; 5 -10 0 -20 0 0];
6  b=[0 0 0 0 0 200];
7  x=[2 3 5 2 10 7];
8  [m n]= size(a);
9  ite=1;
10 ter=1e-3;
11 er=1;
12   while (ite<=100&er>=ter)
13   x1=x;
14     for i=1:m
15       sigma=0;
16       for j=1:n
17         if j!=i
18           sigma=sigma+a(i,j)*x(j);
19         endif
20       endfor
21       x(i)=(b(i)-sigma)/a(i,i);
22     endfor
23   er=abs(x1-x);
24   ite++;
25 endwhile
```

Hasil *command window* dari algoritma program tersebut :

Command Window

```
warning: division by zero
warning: called from
    tugas3gauseid at line 21 column 11
warning: division by zero
warning: called from
    tugas3gauseid at line 21 column 11
ite =  2
er =

    10     11      2    Inf    NaN    NaN

x =

    -8     -8      7   -Inf    NaN    NaN

>> |
```

Dari hasil pemrograman tersbut, dapt diketahui bahwa program tidak dapat menampilkan hasil dengan maksimal akibat terjadinnya pembagian 0 dalam susunan program sedemikian sehingga nilai hasil bagi tidak dapat didefinisikan.

E. KESIMPULAN

Dari hasil metode iterasi Jacobi dan Gauss-Seidel pada persoalan pertama ditemukan nilai x1 dan x2 berturut turut 379,499 dan 455,49 dengan jumlah iterasi berbeda yaitu 30 untuk metode Jacobi dan 18 untuk metode Gauss-Seidel. Sedangkan pada persoalan kedua tidak dapat ditemukan nilai pasti akibat persamaan bersifat divergen dan sedangkan pada soal nomor 3 tidak dapat ditemukan nilai akar pasti akibat adanya nilai imaginer sedemikian sehingga algoritma tidak bisa berjalan semestinya.

F. DAFTAR PUSTAKA

Capra, Steven C and Canale.1991. "**Numerical Methods for Engineers with Personal Computers Applications**". MacGraw-Hill Book Company.

Zulkarnain, Egi , Bayu Prihandono, Ilhamsyah.2015." **Algoritma Eliminasi Gauss Interval dalam Mendapatkan Nilai Determinan Matriks Interval dan Mencari Solusi Sistem Persamaan Interval Linier**". Buletin Ilmiah MathStat dan terapannya. Diakses pada 8 September 2019.

King M.R and Mody N.A .2010. "**Numerical and Statical Methods for Bioengineering**".Cambridge University Press. New York.

Patel VA. 1994. "**Numerical Analysis**" .Saunders College Publishing