LAPORAN TUGAS AKHIR METODE NUMERIK

SOLUSI SISTEM PERSAMAAN LINIER MENGGUNAKAN METODE ITERASI JACOBI



Disusun Oleh:

AHMAD QOMARUDDIN / 145410028

SEKOLAH TINGG MANAJEMEN INFORMATIKA DAN KOMPUTER

**AKAKOM**

YOGYAKARTA

2016

**PENDAHULUAN**

1. **TUJUAN**
2. Tujuan pembuatan laporan ini untuk memenuhi tugas akhir matakuliah Metode Numerik yang diampu oleh Dra.F.Wiwiek Nurwiati,.M.T sebagai pengganti Ujian Akhir Semester matakuliah Metode Numerik.
3. Laporan ini membahas tentang Solusi Sistem Persamaan Linier Menggunakan Metode Iterasi Jacobi.
4. Metode iterasi jacobi digunakan untuk menyelesaikan persamaan linier berukuran besar dan proporsi koefisien nolnya besar.
5. **TEORI**

Metode Jacobi merupakan suatu teknik penyelesaian SPL berukuran *n x n*, *AX* = *b*, secara *iteratif*. Proses penyelesaian dimulai dengan suatu hampiran awal terhadap penyelesaian, *X0*, kemudian membentuk suatu serangkaian vector *X1*, *X2*, … yang konvergen ke *X*.

Teknik *iteratif* jarang digunakan untuk menyelesaikan SPL berukuran kecil karena metode-metode langsung seperti metode *eliminasi Gauss* lebih efisien dari pada metode iteratif. Akan tetapi, untuk SPL berukuran besar dengan persentase elemen nol pada matriks koefisien besar, teknik iteratif lebih efisien daripada metode langsung dalam hal penggunaan memori komputer maupun waktu komputasi. Metode *iterasi Jacobi*, prinsipnya: merupakan metode iteratif yang melakuakn perbaharuan nilai x yang diperoleh tiap iterasi (mirip metode substitusi berurutan, *successive substitution*).

Pada metode *iterasi Jacobi*, penyelesaian dilakukan secara iterasi, dimana proses iterasi dilakukan sampai dicapai suatu nilai yang konvergen dengan toleransi yang diberikan. Dari hasil pengujian dapat diketahui bahwa metode *Iterasi Jacobi* memiliki hasil ketelitian yang lebih baik dan waktu komputasi yang lebih cepat dari metode *Eliminasi Gauss* dan metode Dekomposisi LU.

Dipandang sistem dengan 3 persamaan dan 3 bilangan tak diketahui:

Persamaan 1:

*a*11 *x*1 + *a*12 *x*2 + *a*13 *x*3 = *b*1

*a*21 *x*1 + *a*22 *x*2 + *a*23 *x*3 = *b*2

*a*31 *x*1 + *a*32 *x*2 + *a*33 *x*3 = *b*3

Persamaan pertama dari sistem diatas dapat digunakan untuk menghitung *x*1 sebagai fungsi dari *x*2 dan *x*3. Demikian juga persamaan kedua dan ketiga untuk menghitung *x*2 dan *x*3 sehingga didapat:

Persamaan ke-2:



Hitungan dimulai dengan nilai perkiraan awal sembarang untuk variabel yang dicari (biasanya semua variabel diambil sama dengan nol). Nilai perkiraan awal disubstitusikan ke dalam ruas kanan dari sistem persamaan ke-2. Selanjutnya nilai variabel yang didapat tersebut disubstitusikan ke ruas kanan dari sistem persamaan ke-2 lagi untuk mendapatkan nilai perkiraan kedua. Prosedur tersebut diulangi lagi sampai nilai setiap variabel pada iterasi ke *n* mendekati nilai pada iterasi ke *n* − 1. Apabila indeks *n* menunjukkan jumlah iterasi, maka persamaan ke-2 dapat ditulis menjadi:

Persamaan ke-3:



Iterasi hitungan berakhir setelah:

 dan 

atau telah dipenuhi kriteria berikut:



dengan adalah batasan ketelitian yang dikehendaki.

**PEMBAHASAN**

1. **ALGORITMA**

Berikut adalah algoritma untuk metode iterasi jacobi:

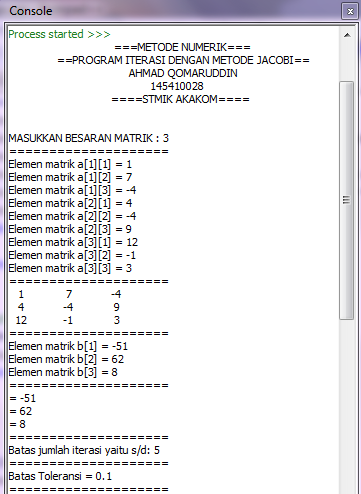
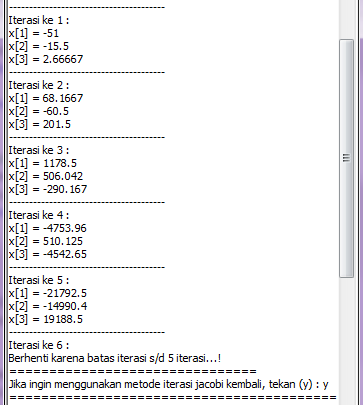
1. Mulai.
2. Masukkan besaran matrik yang akan dihitung.
3. Masukkan bilangan matrik dan besaran matrik hasil.
4. Masukkan batas jumlah iterasi yang akan dillakukan.
5. Masukkan batas toleransi.
6. Inisialisasi nilai awal.
7. Menentukan titik variabel x awal kemudian melakukan iterasi dengan  persamaan 3a hingga didapatkan nilai variabel x yang tidak berubah atau hampir tidak berubah  dari iterasi yang sebelumnya.
8. Akan terjadi perulang selama toleransi lebih dari 0.00001. iterasi yang diinisaiisasi dimai dari 0 akan berubah nilai secara ascending pada setiap perulangan.
9. Terdapat seleksi didalam perulangan jika iterasi yang dilakukan lebih dari batas toleransi maka perulangan akan berhenti, dan menampilkan opsi apakah ingin menggunakan metode ini kembali, jika ya maka tekan huruf “Y” maka program akan dijalankan dari awal lagi. Jika tidak maka program akan berhenti.
10. Menghitung jumlah total sigma digunakan untuk menghitung nilai pada setiap iterasi.
11. Kemudian menghitung toleransi pada setiap iterasi, jika hasil dari toleransi kurang dari 0 maka nilai tersebut dikalikan dengan -1 ini berfungsi untuk mengubah nilai menjadi negatif.
12. Kemudian dilakukan perulangan untuk penyeteraan nilai x indeks yang ke-i sama dengan nilai f indeks yang ke-i.
13. Selesai.
14. **UJI COBA**

**Soal 1:** selesaikan SPL berikut dengan toleransi 0.1 dan batas iterasi 5 iterasi

x1 + 7x2  – 4x3  = -51

4x1 – 4x2  + 9x3  = 62

12x1 – x2 + 3x3  = 8

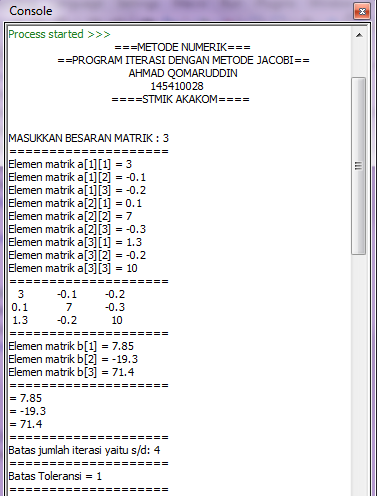
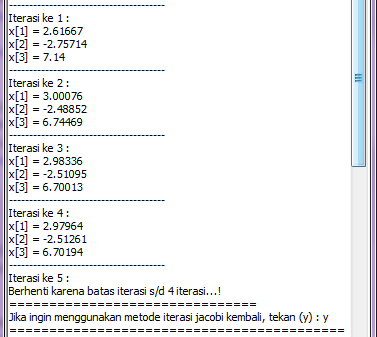
 

**Soal 2:** selesaikan SPL berikut dengan toleransi 1 dan batas iterasi 4 iterasi

3x1 – 0.1x2 – 0.2x3 = 7.85

0.1x1 + 7x2  – 0.3x3  = -19.3

1.3x1 – 0.2x2 + 10x3  = 71.4

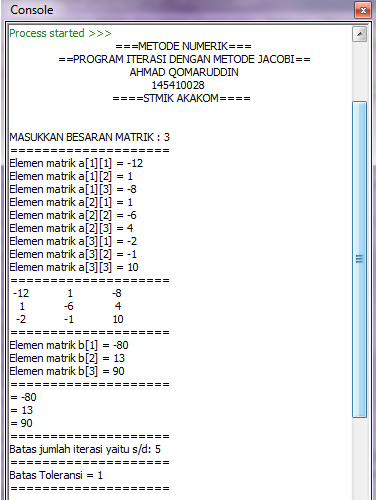
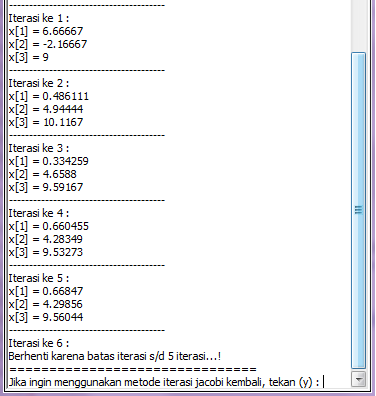
 

**Soal 3:** selesaikan SPL berikut dengan toleransi 1 dan batas iterasi 5 iterasi

-12x1 + x2 – 8x3 = -80

x1 – 6x2 + 4x3 = 13

-2x1 – x2 + 10x3 = 90

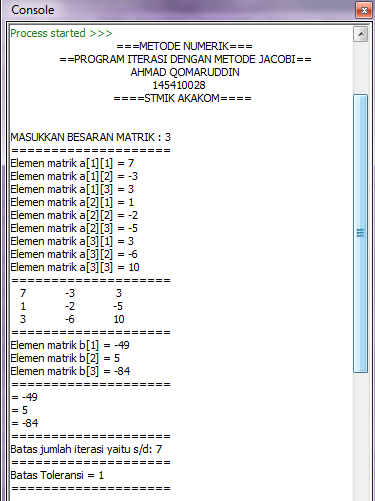
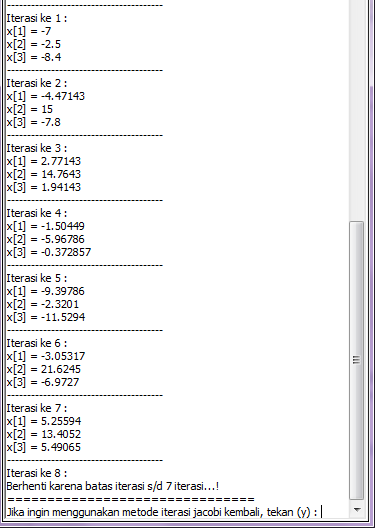
 

**Soal 4:** selesaikan SPL berikut dengan toleransi 1 dan batas iterasi 7 iterasi

7x1 – 3x2 + 3x3 = -49

x1 – 2x2 – 5x3 = 5

3x1 – 6x2 + 10x3 = -84

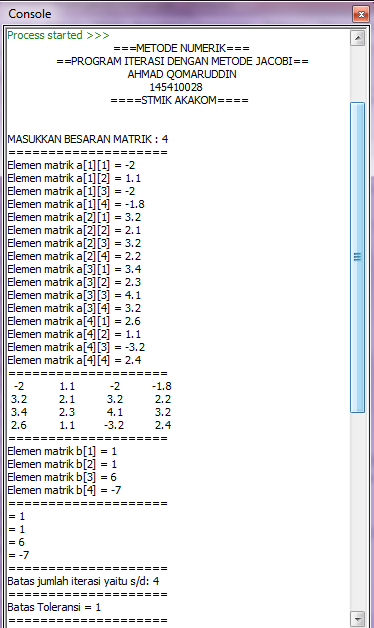
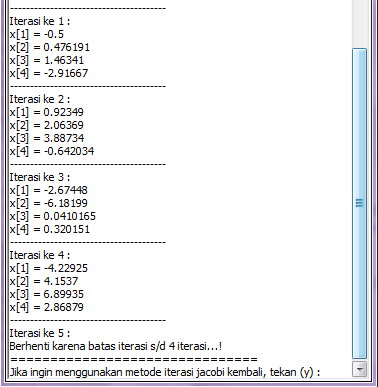
**Soal 5:** selesaikan SPL berikut dengan toleransi 1 dan batas iterasi 4 iterasi

-2x1 + 1.1x2 – 2x3 – 1.8 x4 = 1

3.2x1 + 2.1x2 + 3.2x3 + 2.2 x4 = 1

3.4x1 + 2.3x2 + 4.1x3 + 3.2 x4 = 6

2.6x1 + 1.1x2 – 3.2x3 + 2.4 x4= -7

1. **KODE PROGRAM**

#include<iostream>

#include<conio>

#include<iomanip>

using namespace std;

void main()

{

awal:

cout<<" ===METODE NUMERIK===\n";

cout<<" ==PROGRAM ITERASI DENGAN METODE JACOBI==\n";

cout<<" AHMAD QOMARUDDIN\n";

cout<<" 145410028\n";

cout<<" ====STMIK AKAKOM====\n";

cout<<endl<<endl;

int i, j, k, l, ukuran, iterasi, btsiterasi;

float f[10], a[10][10],b[10],x[10], toleransi, sigma[10];

char ulang;

//input ukuran matrik

cout<<"MASUKKAN BESARAN MATRIK : "; cin>>ukuran;

cout<<"===================="<<endl;

//masukkan matrik a

for(i=1;i<=ukuran;i++){

for(j=1;j<=ukuran;j++){

cout<<"Elemen matrik a[" <<i<<"]["<<j<<"] = ";

cin>>a[i][j];

}

}

cout<<"===================="<<endl;

for(i=1;i<=ukuran;i++){

for(j=1;j<=ukuran;j++){

cout<<setw(4)<<a[i][j]<<" ";

}

cout<<endl;

}

cout<<"===================="<<endl;

//masukkan matrik b

for(i=1;i<=ukuran;i++){

cout<<"Elemen matrik b["<<i<< "] = ";

cin>>b[i];

}

cout<<"===================="<<endl;

for(i=1;i<=ukuran;i++){

cout<<"= "<<b[i]<<endl;

}

cout<<"===================="<<endl;

cout<<"Batas jumlah iterasi yaitu s/d: "; cin>>btsiterasi;

cout<<"===================="<<endl;

cout<<"Batas Toleransi = ";cin>>toleransi;

cout<<"===================="<<endl;

//inisialisasi nilai awal

for(i=1;i<=ukuran;i++){

x[i]=0;

}

iterasi=0;

//ITERASI JACOBI

while(toleransi>0.00001){

iterasi++;

cout<<"---------------------------------------\n";

cout<<"Iterasi ke " <<iterasi<<" : "<<endl;

if(iterasi>btsiterasi){

cout<<"Berhenti karena batas iterasi s/d "<<btsiterasi<<" iterasi...!"<<endl;

cout<<"===============================\n";

cout<<"Jika ingin menggunakan metode iterasi jacobi kembali, tekan (y) : "; cin>>ulang;

cout<<"==========================================\n";

cout<<endl;

while(ulang=='y' || ulang=='Y')

goto awal;

break;

}

//menghitung jumlah total sigma

for(k=1;k<=ukuran;k++){

sigma[k]=0;

for(l=1;l<=ukuran;l++){

if(k!=l){

sigma[k]=sigma[k]+(a[k][l]\*x[l]);

}

}

f[k]=(b[k]-sigma[k])/a[k][k];

cout<<"x["<<k<< "] = " <<f[k]<<endl;

}

toleransi=(f[1]-x[1])+(f[2]-x[2])+(f[3]-x[3]);

if(toleransi<0){

toleransi\*=-1;

}

for(i=0;i<=ukuran;i++){

x[i]=f[i];

}

}

}

**DAFTAR REFERENSI**

Kosasih, P. Buyung. 2006. Komputasi Numerik Teori dan Aplikasi. Yogyakarta : Andi.

<http://arwincourse.tripod.com/publikasi/metkomp-code.pdf>

<https://id.wikipedia.org/wiki/Metode_Jacobi#Algoritma_Metode_Iterasi_Jacobi_dalam_bentuk_software_Matlab>

<https://ahliswiwite.files.wordpress.com/2007/11/iterasi-jacobi.doc>