LAPORAN PROYEK

METODE NUMERIK

"Gauss-Seidel Method & Gauss-Seidel Method with Relaxation"

Dosen Pengampu

Dr. Ir. Linggo Sumarno

DIBUAT OLEH KELOMPOK 5:

Julius Rakha Bowo Laksono / 205314057

Petrus Krisna Priya Nanda / 205314060

Yohanes Rio Septian / 205314061

Fransiskus Jremiegi Saputra / 205314062

PROGRAM STUDI INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS SANATA DHARMA

YOGYAKARTA

2023

A. Dasar Teori

Metode Gauss-Seidel adalah salah satu metode iteratif yang digunakan untuk menyelesaikan sistem persamaan linear. Metode ini ditemukan oleh Carl Friedrich Gauss dan Philipp Ludwig von Seidel pada abad ke-19. Metode ini sering digunakan dalam pemodelan matematika dan dalam pemecahan persamaan diferensial parsial.

Metode Gauss-Seidel menjadi salah satu metode penting dalam pemodelan matematika dan pemecahan persamaan diferensial parsial karena kepraktisannya dan kemampuannya untuk menyelesaikan sistem persamaan linear dengan tingkat akurasi yang diinginkan.

Metode Gauss-Seidel tanpa relaksasi melibatkan penggunaan rumus iterasi Gauss-Seidel standar yang telah dijelaskan sebelumnya. Gunakan rumus iterasi Gauss-Seidel tanpa memperkenalkan faktor relaksasi.

$$x_1^j = \frac{b_1 - a_{12}x_2^{j-1} - a_{13}x_3^{j-1}}{a_{11}}$$

$$x_2^j = \frac{b_2 - a_{21}x_1^{j} - a_{23}x_3^{j-1}}{a_{22}}$$

$$x_3^j = \frac{b_3 - a_{31}x_1^{j} - a_{32}x_2^{j}}{a_{33}}$$

Iterasi dilanjutkan hingga mencapai tingkat akurasi yang diinginkan atau konvergensi. berikut adalah rumusnya

$$\varepsilon_{a,i} = \left| \frac{x_i^j - x_i^{j-1}}{x_i^j} \right| \times 100\% \le \varepsilon_s$$

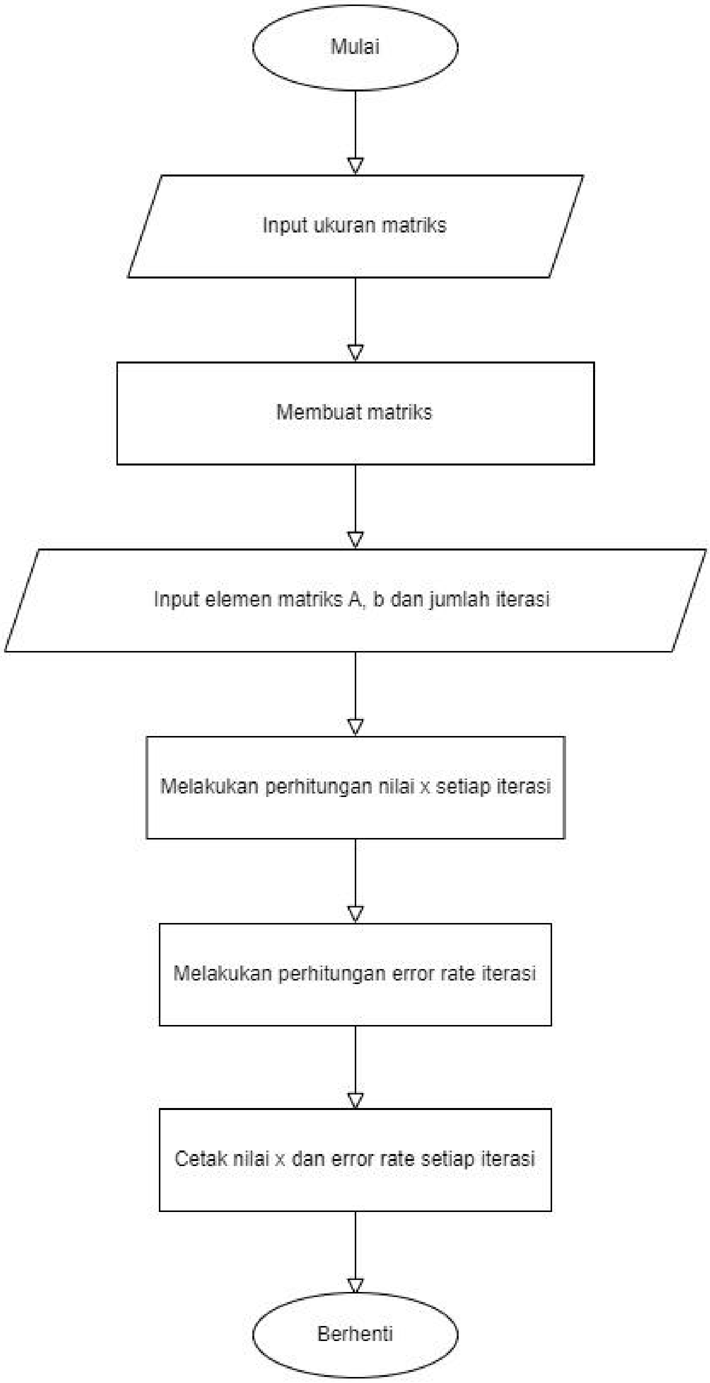
Keuntungan dari metode Gauss-Seidel tanpa relaksasi adalah kesederhanaannya. Namun, dalam beberapa kasus, metode ini mungkin mengalami konvergensi yang lambat atau bahkan divergensi.

Metode Gauss-Seidel dengan relaksasi memperkenalkan faktor relaksasi (relaxation factor) dalam rumus iterasi Gauss-Seidel. Faktor relaksasi (biasanya dilambangkan dengan simbol ω) adalah angka antara 0 dan 2. Faktor relaksasi dapat disesuaikan untuk mencapai konvergensi yang lebih cepat atau untuk menghindari divergensi. Dalam beberapa kasus, penggunaan faktor relaksasi dapat meningkatkan kecepatan konvergensi hingga beberapa kali lipat.

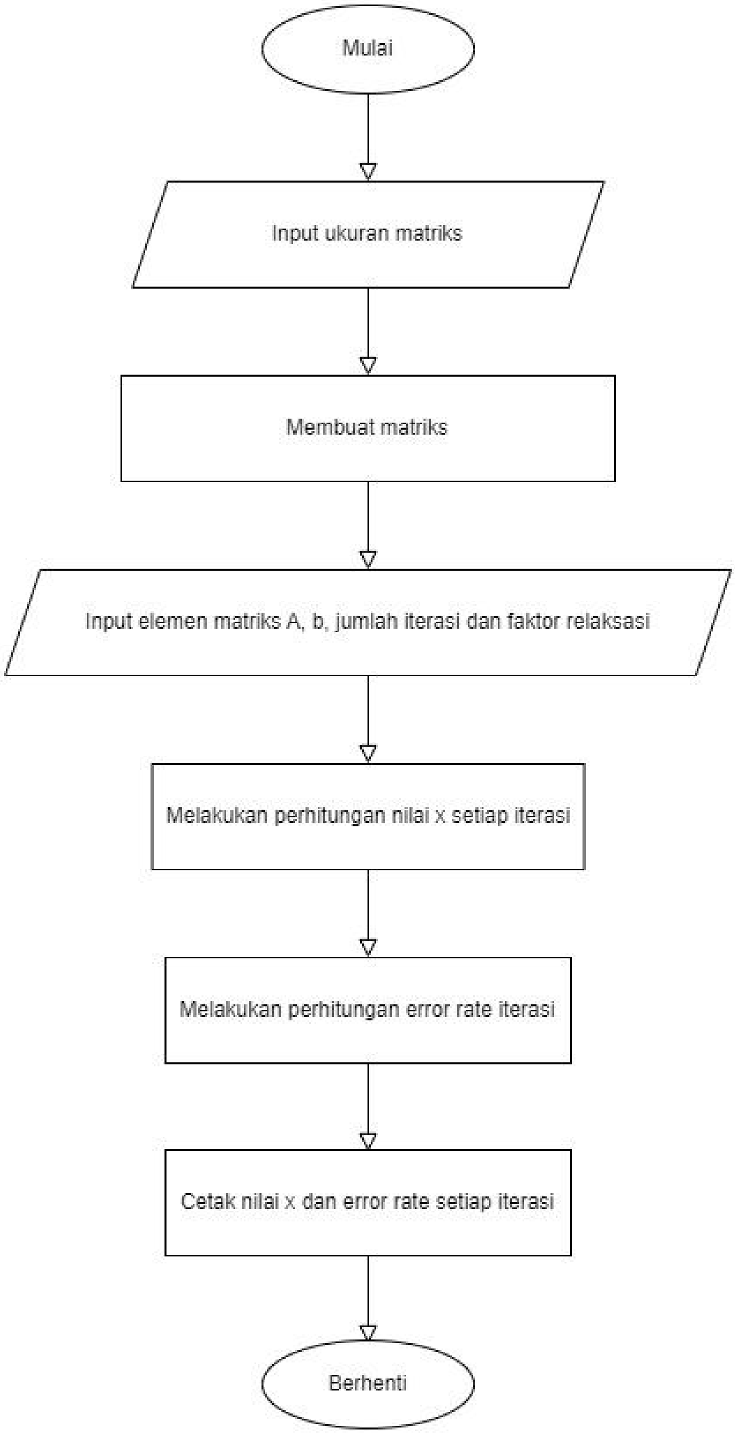
Pemilihan metode Gauss-Seidel dengan atau tanpa relaksasi tergantung pada sistem persamaan linear yang sedang dipecahkan dan tujuan konvergensi yang diinginkan. Percobaan dan analisis yang cermat sering diperlukan untuk menentukan metode yang paling efektif dalam kasus tertentu. Selain itu, metode Gauss-Seidel juga dapat dikombinasikan dengan metode lain seperti metode Jacobi atau metode SOR untuk mencapai hasil yang lebih baik dan efisien.

B. Flowchart

1. Example 12.1



2. Example 12.2



C. Listing Program

1. Example 12.1

Gauss Seidel.py

D: > Semt 6 > Metode Numerik > UIAS > Gauss Seidel.py > ...

```python
def hitung_persamaan(A, b, x1, x2, x3):
    list_hasil = []
    hasil_1 = (b[0] + ((-1 * A[0][1]) * x2) + ((-1 * A[0][2]) * x3)) / A[0][0]
    hasil_2 = (b[1] + ((-1 * A[1][0]) * hasil_1) + ((-1 * A[1][2]) * x3)) / A[1][1]
    hasil_3 = (b[2] + ((-1 * A[2][0]) * hasil_1) + ((-1 * A[2][1]) * hasil_2)) / A[2][2]

    list_hasil.append(hasil_1)
    list_hasil.append(hasil_2)
    list_hasil.append(hasil_3)

    return list_hasil


def hitung_error(list):
    list_error = []

    error_1 = "{:.5f}".format(abs((list[len(list) - 1][0] - list[len(list) - 2][0]) / list[len(list) - 1][0]) * 100)
    error_2 = "{:.5f}".format(abs((list[len(list) - 1][1] - list[len(list) - 2][1]) / list[len(list) - 1][1]) * 100)
    error_3 = "{:.5f}".format(abs((list[len(list) - 1][2] - list[len(list) - 2][2]) / list[len(list) - 1][2]) * 100)

    list_error.append(error_1)
    list_error.append(error_2)
    list_error.append(error_3)

    return list_error


def gauss_seidel(A, b, jmlIterasi):
    list_x = []
    temp_x = []
    temp_error = []
    error_rate = []

    for i in range(0, jmlIterasi):
        if(i == 0):
            x_1 = 0
            x_2 = 0
```

```python
            x_3 = 0
            list_x = hitung_persamaan(A, b, x_1, x_2, x_3)
            temp_x.append(list_x)
        else:
            x_1 = list_x[0]
            x_2 = list_x[1]
            x_3 = list_x[2]
            list_x = hitung_persamaan(A, b, x_1, x_2, x_3)
            temp_x.append(list_x)

        error_rate = hitung_error(temp_x)
        temp_error.append(error_rate)

    return temp_x, temp_error

# Membaca ukuran matriks A dari input pengguna
n = int(input("Masukkan ukuran matriks A: "))

# Membaca elemen-elemen matriks A dari input pengguna
A = []
print("Masukkan elemen-elemen matriks A:")
for i in range(n):
    row = list(map(float, input().split()))
    A.append(row)

# Membaca elemen-elemen vektor b dari input pengguna
b = list(map(float, input("Masukkan elemen-elemen vektor b: ").split()))

# Membaca jumlah iterasi dari input
jumlahIterasi = int(input("Masukkan jumlah iterasi: "))

# Memanggil fungsi gasuss_seidel
list_x, list_error = gauss_seidel(A, b, jumlahIterasi)
```

```python
for i in range(0, jumlahIterasi):
    print('Iterasi', i+1)
    print('==========================')
    print('x1 = ', list_x[i][0])
    print('x2 = ', list_x[i][1])
    print('x3 = ', list_x[i][2])
    print('--------------------------')
    print('Error Rate')
    print('--------------------------')
    print('x1 = ', list_error[i][0])
    print('x2 = ', list_error[i][1])
    print('x3 = ', list_error[i][2])
    print('==========================\n')
```

2. Example 12.2

Guss Seidel Relax.py

D: > Semt 6 > Metode Numerik > UIAS > Guss Seidel Relax.py > ...

```python
def hitung_persamaan_relaksasi(A, b, x1, x2, omega):
    list_hasil = []
    hasil_1 = (b[0] + ((-1 * A[0][1]) * x2)) / A[0][0]
    hasil_1_relaksasi = omega * hasil_1 + (1 - omega) * x1

    hasil_2 = (b[1] + ((-1 * A[1][0]) * hasil_1_relaksasi)) / A[1][1]
    hasil_2_relaksasi = omega * hasil_2 + (1 - omega) * x2

    list_hasil.append(hasil_1_relaksasi)
    list_hasil.append(hasil_2_relaksasi)

    return list_hasil

def hitung_error(list):
    list_error = []

    error_1 = "{:.5f}".format(abs((list[len(list) - 1][0] - list[len(list) - 2][0]) / list[len(list) - 1][0]) * 100)
    error_2 = "{:.5f}".format(abs((list[len(list) - 1][1] - list[len(list) - 2][1]) / list[len(list) - 1][1]) * 100)

    list_error.append(error_1)
    list_error.append(error_2)
    return list_error

def gauss_seidel_relaksasi(A, b, jmlIterasi, omega):
    list_x = []
    temp_x = []
    temp_error = []

    for i in range(jmlIterasi):
        if i == 0:
            x_1 = 0
            x_2 = 0
            list_x = hitung_persamaan_relaksasi(A, b, x_1, x_2, omega)
            temp_x.append(list_x)
        else:
            x_1 = list_x[0]
            x_2 = list_x[1]
```

```python
            list_x = hitung_persamaan_relaksasi(A, b, x_1, x_2, omega)
            temp_x.append(list_x)

        error_rate = hitung_error(temp_x)
        temp_error.append(error_rate)

    return temp_x, temp_error

# Membaca ukuran matriks A dari input pengguna
n = int(input("Masukkan ukuran matriks A: "))

# Membaca elemen-elemen matriks A dari input pengguna
A = []
print("Inputkan Terbalik")
print("Masukkan elemen-elemen matriks A:")
for i in range(n):
    row = list(map(float, input().split()))
    A.append(row)

# Membaca elemen-elemen vektor b dari input pengguna
b = list(map(float, input("Masukkan elemen-elemen vektor b: ").split()))

# Membaca jumlah iterasi dari input
jumlahIterasi = int(input("Masukkan jumlah iterasi: "))

# Membaca faktor relaksasi (omega) dari input
omega = float(input("Masukkan faktor relaksasi (omega): "))

# Memanggil fungsi gauss_seidel_relaksasi
list_x, list_error = gauss_seidel_relaksasi(A, b, jumlahIterasi, omega)

for i in range(jumlahIterasi):
    print('Iterasi', i + 1)
    print('==========================')
    print('x1 =', list_x[i][0])
    print('x2 =', list_x[i][1])
    print('--------------------------')
```

```python
    print('Error Rate')
    print('---------------------------')
    print('x1 =', list_error[i][0])
    print('x2 =', list_error[i][1])
    print('==========================\n')
```

D. Output

1. **Example 12.1**

```
PS C:\Users\M S I> & "C:/Users/M S I/AppData/Local/
Masukkan ukuran matriks A: 3
Masukkan elemen-elemen matriks A:
3 -0.1 -0.2
0.1 7 -0.3
0.3 -0.2 10
Masukkan elemen-elemen vektor b: 7.85 -19.3 71.4
Masukkan jumlah iterasi: 2
Iterasi 1
===========================
x1 =  2.6166666666666667
x2 =  -2.7945238095238096
x3 =  7.005609523809525
---------------------------
Error Rate
---------------------------
x1 =  0.00000
x2 =  0.00000
x3 =  0.00000
===========================

Iterasi 2
===========================
x1 =  2.990556507936508
x2 =  -2.499624684807256
x3 =  7.000290811065 76
---------------------------
Error Rate
---------------------------
x1 =  12.50235
x2 =  11.79774
x3 =  0.07598
===========================
```

2. **Example 12.2**

```
PS C:\Users\M S I> & "C:/Users/M S I/AppData/Lo
Masukkan ukuran matriks A: 2
Inputkan Terbalik
Masukkan elemen-elemen matriks A:
10 -2
-3 12
Masukkan elemen-elemen vektor b: 8 9
Masukkan faktor relaksasi (omega): 1.2
Iterasi 1
===========================
x1 = 0.96
x2 = 1.1879999999999997
---------------------------
Error Rate
---------------------------
x1 = 0.00000
x2 = 0.00000
===========================

Iterasi 2
===========================
x1 = 1.0531199999999998
x2 = 0.978336
---------------------------
Error Rate
---------------------------
x1 = 8.84230
x2 = 21.43067
===========================

Iterasi 3
===========================
x1 = 0.98417664
x2 = 0.999585792
---------------------------
Error Rate
---------------------------
x1 = 7.00518
x2 = 2.12586
===========================
```

E. Perbandingan Hasil

1. **Example 12.1**

   a. **Materi**

   - **Iterasi 1**

$$x_1 = \frac{7.85 + 0.1(0) + 0.2(0)}{3} = 2.616667$$

$$x_2 = \frac{-19.3 - 0.1(2.616667) + 0.3(0)}{7} = -2.794524$$

$$x_3 = \frac{71.4 - 0.3(2.616667) + 0.2(-2.794524)}{10} = 7.005610$$

   - **Iterasi 2**

For the second iteration, the same process is repeated to compute

$$x_1 = \frac{7.85 + 0.1(-2.794524) + 0.2(7.005610)}{3} = 2.990557$$

$$x_2 = \frac{-19.3 - 0.1(2.990557) + 0.3(7.005610)}{7} = -2.499625$$

$$x_3 = \frac{71.4 - 0.3(2.990557) + 0.2(-2.499625)}{10} = 7.000291$$

   - **Error rate**

$$\varepsilon_{a,1} = \left|\frac{2.990557 - 2.616667}{2.990557}\right| \times 100\% = 12.5\%$$

   b. **Program**

```
Iterasi 1
===========================
x1 =  2.6166666666666667
x2 =  -2.7945238095238096
x3 =  7.005609523809525
---------------------------
Error Rate
---------------------------
x1 =  0.00000
x2 =  0.00000
x3 =  0.00000
===========================

Iterasi 2
===========================
x1 =  2.990556507936508
x2 =  -2.499624684807256
x3 =  7.00029081106576
---------------------------
Error Rate
---------------------------
x1 =  12.50235
x2 =  11.79774
x3 =  0.07598
===========================
```

Untuk program jika dibandingkan sudah sama dengan materi bisa dilihat pada iterasi 1: x1=2.617, x2=-2.795, x3=7.006 begitu juga pada iterasi 2 dan juga untuk error rate pada materi dilakukan perhitungan pada hasil iterasi 1 dan 2 sehingga x1 hasilnya yaitu 12.5% dan jika dibandingkan hasilnya sama.

2. **Example 12.2**

a. **Materi**

- **Iterasi 1**

Before solving for x_2, we first apply relaxation to our result for x_1:

$$x_{1,r} = 1.2(0.8) - 0.2(0) = 0.96$$

We then apply relaxation to this result to give

$$x_{2,r} = 1.2(0.99) - 0.2(0) = 1.188$$

- **Iterasi 2 dan Error rate**

Second iteration: Using the same procedure as for the first iteration, the second iteration yields

$$x_1 = 0.8 + 0.2(1.188) = 1.0376$$
$$x_{1,r} = 1.2(1.0376) - 0.2(0.96) = 1.05312$$
$$\varepsilon_{a,1} = \left|\frac{1.05312 - 0.96}{1.05312}\right| \times 100\% = 8.84\%$$
$$x_2 = 0.75 + 0.25(1.05312) = 1.01328$$
$$x_{2,r} = 1.2(1.01328) - 0.2(1.188) = 0.978336$$
$$\varepsilon_{a,2} = \left|\frac{0.978336 - 1.188}{0.978336}\right| \times 100\% = 21.43\%$$

b. **Program**

```
Iterasi 1
===========================
x1 = 0.96
x2 = 1.1879999999999997
---------------------------
Error Rate
---------------------------
x1 = 0.00000
x2 = 0.00000
===========================

Iterasi 2
===========================
x1 = 1.0531199999999998
x2 = 0.978336
---------------------------
Error Rate
---------------------------
x1 = 8.84230
x2 = 21.43067
===========================
```

Untuk program jika dibandingkan sudah sama dengan materi bisa dilihat pada iterasi 1 setelah dilakukan relaksasi maka hasilnya x1=0.96 dan x2=1.187 pada materi dilakukan pembulatan sehingga hasilnya x2=1.188 begitu juga pada iterasi 2 dan untuk error rate memakai hasil perhitungan di iterasi 1 dan 2 sehingga hasilnya pada x1=8.84 dan x2=21.43 dan jika dibandingkan hasilnya sama.

F. Kesimpulan

Berdasarkan hasil running program menunjukan bahwa program dapat dikatakan berhasil karena hasil output program sama dengan hasil dari materi yang diberikan. Namun terdapat perbedaan pada bagian pembulatan nilai/bilangan, jikalau di program terdapat hasilnya ada yang belum dibulatkan sedangkan pada materi hasilnya ada yang sudah dibulatkan sehingga pada dasarnya sama.