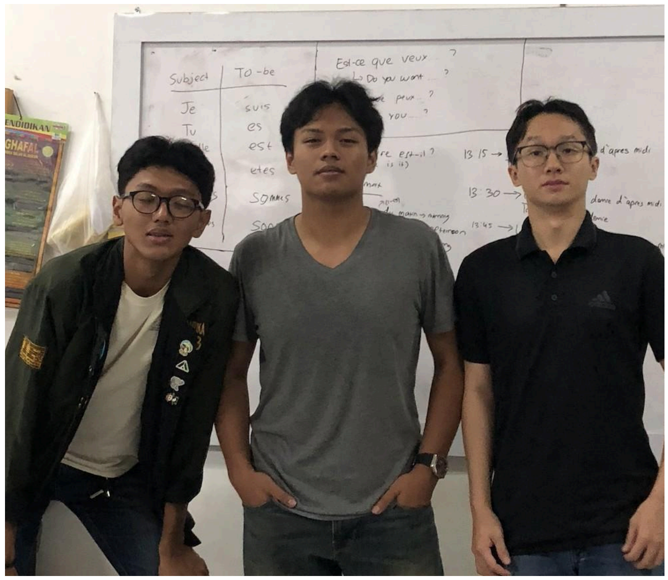
LAPORAN TUGAS BESAR 1



IF2110 Aljabar Linier dan Geometri

Anggota Kelompok:
Maheswara Bayu Kaindra (13523015)
Raka Daffa Iftikhaar (13523018)
Peter Wongsoredjo (13523039)

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

2024
Daftar Isi

BAB 1
DESKRIPSI MASALAH

1.1 Sistem Persamaan Linier (SPL)

Sistem persamaan linier (SPL) banyak ditemukan di dalam bidang sains dan rekayasa. Anda sudah mempelajari berbagai metode untuk menyelesaikan SPL, termasuk menghitung determinan matriks. Sembarang SPL dapat diselesaikan dengan beberapa metode, yaitu metode eliminasi Gauss, metode eliminasi Gauss-Jordan, metode matriks balikan (x = A -1b), dan kaidah Cramer (khusus untuk SPL dengan n peubah dan n persamaan). Solusi sebuah SPL mungkin tidak ada, banyak (tidak berhingga), atau hanya satu (unik/tunggal).

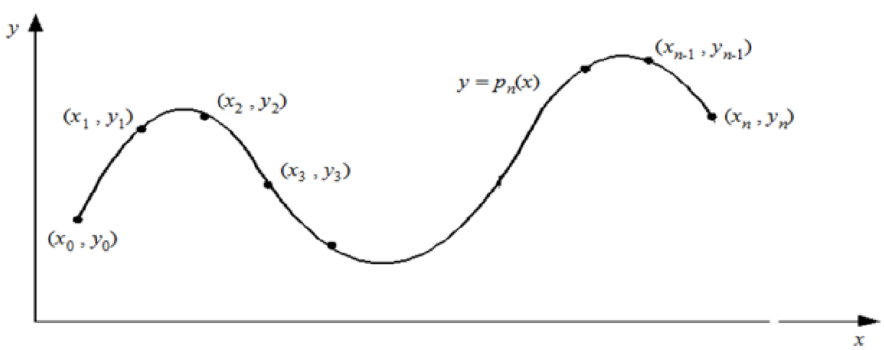
$$\begin{bmatrix} 0 & 2 & 1 & -1 \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \quad \begin{bmatrix} 0 & 1 & 0 & -\frac{2}{3} \\ 0 & 0 & 1 & \frac{1}{3} \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Gambar 1. Eliminasi Gauss dilakukan dengan matriks eselon baris dan eliminasi Gauss-Jordan dengan matriks eselon baris tereduksi.

Di dalam Tugas Besar 1 ini, Anda diminta membuat satu atau lebih library aljabar linier dalam Bahasa Java. Library tersebut berisi fungsi-fungsi seperti eliminasi Gauss, eliminasi Gauss-Jordan, menentukan balikan matriks, menghitung determinan, kaidah Cramer (kaidah Cramer khusus untuk SPL dengan n peubah dan n persamaan). Selanjutnya, gunakan library tersebut di dalam program Java untuk menyelesaikan berbagai persoalan yang dimodelkan dalam bentuk SPL, menyelesaikan persoalan interpolasi, dan persoalan regresi.

1.2 Interpolasi Polinomial

Persoalan interpolasi polinom adalah sebagai berikut: Diberikan n+1 buah titik berbeda, (x_0, y_0), (x_1, y_1), ..., (x_n, y_n). Tentukan polinom $p_n(x)$ yang menginterpolasi (melewati) semua titik-titik tersebut sedemikian rupa sehingga $y_i = p_n(x_i)$ untuk i = 0, 1, 2, …, n.



Gambar 2. Ilustrasi beberapa titik yang diinterpolasi secara polinomial.

Setelah polinom interpolasi $p_n(x)$ ditemukan, $p_n(x)$ dapat digunakan untuk menghitung perkiraan nilai y di sembarang titik di dalam selang $[x_0, x_n]$.

Polinom interpolasi derajat n yang menginterpolasi titik-titik $(x_0, y_0), (x_1, y_1), ..., (x_n, y_n)$ adalah berbentuk $p_n(x) = a_0 + a_1x + a_2x^2 + ... + a_nx^n$. Jika hanya ada dua titik, (x_0, y_0) dan (x_1, y_1), maka polinom yang menginterpolasi kedua titik tersebut adalah yaitu berupa persamaan garis lurus. Jika tersedia tiga titik (x_0, y_0), (x_1, y_1), dan (x_2, y_2), maka polinom yang menginterpolasi ketiga titik tersebut adalah $p_2(x) = a_0 + a_1x + a_2x^2$ atau persaman kuadrat dan kurvanya berupa parabola. Jika tersedia empat titik, (x_0, y_0), (x_1, y_1), (x_2, y_2), dan (x_3, y_3), polinom yang menginterpolasi keempat titik tersebut adalah $p_3(x) = a_0 + a_1x + a_2x^2 + a_3x^3$, demikian seterusnya. Dengan cara yang sama kita dapat membuat polinom interpolasi berderajat n untuk n yang lebih tinggi asalkan tersedia (n+1) buah titik data. Dengan menyulihkan (xi , yi) ke dalam persamaan polinom $p_n(x) = a_0 + a_1x + a_2x^2 +... + a_nx^n$ untuk i = 0, 1, 2, …, n, akan diperoleh n buah sistem persamaan lanjar dalam $a_0, a_1, a_2, ..., a_n$.

$$a_0 + a_1x_0 + a_2x_0^2 + ... + a_n x_0^n = y_0$$
$$a_0 + a_1x_1 + a_2x_1^2 + ... + a_n x_1^n = y_1$$
$$... \qquad ...$$
$$a_0 + a_1x_n + a_2x_n^2 + ... + a_n x_n^n = y_n$$

Solusi sistem persamaan lanjar ini, yaitu nilai $a_0, a_1, ..., a_n$, diperoleh dengan menggunakan metode eliminasi Gauss yang sudah anda pelajari. Sebagai contoh, misalkan diberikan tiga buah titik yaitu (8.0, 2.0794), (9.0, 2.1972), dan (9.5, 2.2513). Tentukan polinom interpolasi kuadratik lalu estimasi nilai fungsi pada x = 9.2. Polinom kuadratik berbentuk $p_2(x) = a_0 + a_1x + a_2x_2$. Dengan menyulihkan ketiga buah titik data ke dalam polinom tersebut, diperoleh sistem persamaan lanjar yang terbentuk adalah

$$a_0 + 8.0a_1 + 64.00a_2 = 2.0794$$
$$a_0 + 9.0a_1 + 81.00a_2 = 2.1972$$
$$a_0 + 9.5a_1 + 90.25a_2 = 2.2513$$

Penyelesaian sistem persamaan dengan metode eliminasi Gauss menghasilkan a_0 = 0.6762, a_1 = 0.2266, dan a_2 = -0.0064. Polinom interpolasi yang melalui ketiga buah titik tersebut adalah $p_2(x)$ = 0.6762 + 0.2266x - 0.0064x^2. Dengan menggunakan polinom ini, maka nilai fungsi pada x = 9.2 dapat ditaksir sebagai berikut: p_2(9.2) = 0.6762 + 0.2266(9.2) - 0.0064(9.2) 2 = 2.2192.

1.3 Regresi Berganda

1.3.1 Regresi Linier Berganda

Meskipun sudah ada persamaan jadi untuk menghitung regresi linear sederhana, terdapat persamaan umum dari regresi linear yang bisa digunakan untuk regresi linear berganda, yaitu.

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \beta_k x_{ki} + \epsilon_i$$

Untuk mendapatkan nilai dari setiap β_i dapat digunakan Normal Estimation Equation for Multiple Linear Regression sebagai berikut:

$$
\begin{array}{lllll}
nb_0 + b_1 \sum_{i=1}^{n} x_{1i} & + b_2 \sum_{i=1}^{n} x_{2i} & + \cdots + b_k \sum_{i=1}^{n} x_{ki} & = \sum_{i=1}^{n} y_i \\
b_0 \sum_{i=1}^{n} x_{1i} + b_1 \sum_{i=1}^{n} x_{1i}^2 & + b_2 \sum_{i=1}^{n} x_{1i}x_{2i} + & \cdots + b_k \sum_{i=1}^{n} x_{1i}x_{ki} & = \sum_{i=1}^{n} x_{1i}y_i \\
\vdots \qquad \vdots & \vdots & \vdots & \vdots \\
b_0 \sum_{i=1}^{n} x_{ki} + b_1 \sum_{i=1}^{n} x_{ki}x_{1i} & + b_2 \sum_{i=1}^{n} x_{ki}x_{2i} + & \cdots + b_k \sum_{i=1}^{n} x_{ki}^2 & = \sum_{i=1}^{n} x_{ki}y_i
\end{array}
$$

1.3.2 Regresi Kuadratik Berganda

Dalam kasus ini, proses mengubah data-data dalam regresi kuadratik berganda cukup berbeda dengan Regresi Linier Berganda. Bentuk persamaan dari regresi kuadratik ada 3, yaitu:

a. Variabel Linier: Variabel dengan derajat satu seperti X, Y, dan Z
b. Variabel Kuadrat: Variabel dengan derajat dua seperti X^2
c. Variabel Interaksi: 2 Variabel dengan derajat satu yang dikalikan dengan satu sama lain seperti XY, YZ, dan XZ

Setiap n-peubah, jumlah variabel linier, kuadrat, dan interaksi akan berbeda-beda.

$$
\begin{pmatrix}
N & \sum u_i & \sum v_i & \sum u_i^2 & \sum u_iv_i & \sum v_i^2 \\
\sum u_i & \sum u_i^2 & \sum u_iv_i & \sum u_i^3 & \sum u_i^2v_i & \sum u_iv_i^2 \\
\sum v_i & \sum u_iv_i & \sum v_i^2 & \sum u_i^2v_i & \sum u_iv_i^2 & \sum v_i^3 \\
\sum u_i^2 & \sum u_i^3 & \sum u_i^2v_i & \sum u_i^4 & \sum u_i^3v_i & \sum u_i^2v_i^2 \\
\sum u_iv_i & \sum u_i^2v_i & \sum u_iv_i^2 & \sum u_i^3v_i & \sum u_i^2v_i^2 & \sum u_iv_i^3 \\
\sum v_i^2 & \sum u_iv_i^2 & \sum v_i^3 & \sum u_i^2v_i^2 & \sum u_iv_i^3 & \sum v_i^4
\end{pmatrix}
\begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix}
=
\begin{pmatrix} \sum y_i \\ \sum y_iu_i \\ \sum y_iv_i \\ \sum y_iu_i^2 \\ \sum y_iu_iv_i \\ \sum y_iv_i^2 \end{pmatrix}
$$

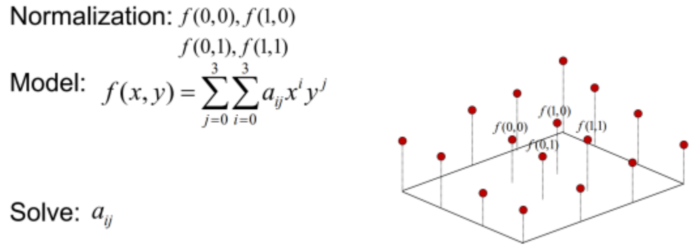
N menandakan jumlah peubah, terdapat 2 variabel linier yaitu u_i dan v_i, 2 variabel kuadrat yaitu u_i^2 dan v_i^2 dan 1 variabel interaksi yaitu *uv*. Untuk setiap n-peubah, akan terdapat 1 konstan N (Terlihat di bagian atas kiri gambar), n variabel linier, n variabel kuadrat, dan C_2^n variabel linier (dengan syarat n > 1). Tentu dengan bertambahnya peubah n, ukuran matriks akan bertumbuh lebih besar dibandingkan regresi linier berganda tetapi solusi tetap bisa didapat dengan menggunakan SPL.

Kedua model regresi yang dijadikan sistem persamaan linier tersebut diselesaikan dengan menggunakan metode eliminasi Gauss

1.4 Bicubic Spline Interpolation

Bicubic spline interpolation adalah metode interpolasi yang digunakan untuk mengaproksimasi fungsi di antara titik-titik data yang diketahui. Bicubic spline interpolation melibatkan konsep spline dan konstruksi serangkaian polinomial kubik di dalam setiap sel segi empat dari data yang diberikan. Pendekatan ini menciptakan permukaan yang halus dan kontinu, memungkinkan untuk perluasan data secara visual yang lebih akurat daripada metode interpolasi linear.

Dalam pemrosesan menggunakan interpolasi bicubic spline digunakan 16 buah titik, 4 titik referensi utama di bagian pusat, dan 12 titik di sekitarnya sebagai aproksimasi turunan dari keempat titik referensi untuk membagun permukaan bikubik. Bentuk pemodelannya adalah sebagai berikut.



Gambar 3. Pemodelan interpolasi *bicubic spline*.

Selain melibatkan model dasar, juga digunakan model turunan berarah dari kedua sumbu, baik terhadap sumbu x, sumbu y, mapun keduanya. Persamaan polinomial yang digunakan adalah sebagai berikut

$$f(x,y)=\sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}x^i y^j$$

$$f_x(x,y)=\sum_{j=0}^{3}\sum_{i=1}^{3} a_{ij}ix^{i-1} y^j$$

$$f_y(x,y)=\sum_{j=1}^{3}\sum_{i=0}^{3} a_{ij}jx^{i} y^{j-1}$$

$$f_{xy}(x,y)=\sum_{j=0}^{3}\sum_{i=0}^{3} a_{ij}ijx^{i-1} y^{j-1}$$

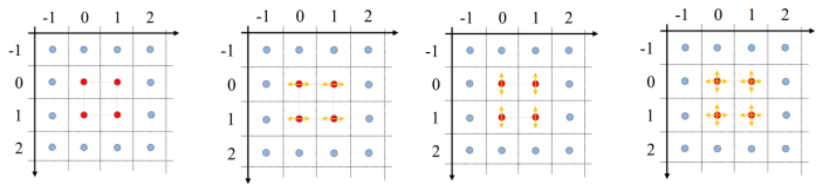
Dengan menggunakan nilai fungsi dan turunan berarah tersebut, dapat terbentuk sebuah matriks solusi X yang membentuk persamaan penyelesaian sebagai berikut

$$y = Xa$$

$$\begin{bmatrix} f(0,0) \\ f(1,0) \\ f(0,1) \\ f(1,1) \\ f_x(0,0) \\ f_x(1,0) \\ f_x(0,1) \\ f_x(1,1) \\ f_y(0,0) \\ f_y(1,0) \\ f_y(0,1) \\ f_y(1,1) \\ f_{xy}(0,0) \\ f_{xy}(1,0) \\ f_{xy}(0,1) \\ f_{xy}(1,1) \end{bmatrix} = \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&2&3&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0 \\ 0&1&2&3&0&1&2&3&0&1&2&3&0&1&2&3 \\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&2&0&0&0&3&0&0&0 \\ 0&0&0&0&1&1&1&1&2&2&2&2&3&3&3&3 \\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&2&3&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&2&0&0&0&3&0&0 \\ 0&0&0&0&0&1&2&3&0&2&4&6&0&3&6&9 \end{bmatrix} \begin{bmatrix} a_{00} \\ a_{10} \\ a_{20} \\ a_{30} \\ a_{01} \\ a_{11} \\ a_{21} \\ a_{31} \\ a_{02} \\ a_{12} \\ a_{22} \\ a_{32} \\ a_{03} \\ a_{13} \\ a_{23} \\ a_{33} \end{bmatrix}$$

Perlu diketahui bahwa elemen pada matriks X adalah nilai dari setiap komponen koefisien a_{ij} yang diperoleh dari persamaan fungsi maupun persamaan turunan yang telah dijelaskan sebelumnya. Sebagai contoh, elemen matriks X pada baris 8 kolom ke 2 adalah koefisien dari a_{10} pada ekspansi sigma untuk $f_x(1, 1)$ sehingga diperoleh nilai konstanta 1 x 1^{1-1}x 1^0 = 1, sesuai dengan isi matriks *X*.

Nilai dari vektor a dapat dicari dari persamaan $y = Xa$, lalu vektor a tersebut digunakan sebagai nilai variabel dalam *f(x, y)*, sehingga terbentuk fungsi interpolasi bicubic sesuai model. Tugas Anda pada studi kasus ini adalah membangun persamaan *f(x, y)* yang akan digunakan untuk melakukan interpolasi berdasarkan nilai *f(a, b)* dari masukan matriks 4 x 4. Nilai masukan *a* dan *b* berada dalam rentang [0, 1]. Nilai yang akan diinterpolasi dan turunan berarah disekitarnya dapat diilustrasikan pada titik berwarna merah pada gambar di bawah.



Gambar 4. Nilai fungsi yang akan di interpolasi pada titik merah, turunan berarah terhadap sumbu *x*, terhadap sumbu *y*, dan keduanya (kiri ke kanan).

BAB 2
TEORI SINGKAT

2.1 Metode Eliminasi Gauss

Metode Eliminasi Gauss adalah prosedur sistematis yang digunakan untuk menyelesaikan sistem persamaan linear. Dalam metode ini, matriks koefisien persamaan linear diubah menjadi bentuk segitiga atas melalui operasi baris elementer (penukaran baris, penggandaan baris dengan skalar, dan penjumlahan atau pengurangan baris). Tujuannya adalah untuk membuat semua elemen dibawah diagonal utama menjadi nol. Setelah matriks berada dalam bentuk segitiga atas, solusi untuk variabel dapat diperoleh dengan proses substitusi mundur. Metode ini sangat efisien untuk menyelesaikan sistem dengan jumlah persamaan yang besar. Berikut langkah-langkah melakukan eliminasi Gauss secara jelas.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} & b_n \end{bmatrix} \sim\text{OBE}\sim \begin{bmatrix} 1 & * & * & \dots & * & * \\ 0 & 1 & * & \dots & * & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 & * \end{bmatrix}$$

2.2 Metode Eliminasi Gauss-Jordan

Metode Eliminasi Gauss-Jordan adalah perpanjangan dari eliminasi Gauss, di mana matriks diubah menjadi bentuk eselon baris tereduksi (reduced row echelon form), yaitu suatu bentuk di mana elemen-elemen di atas dan di bawah diagonal utama juga dibuat menjadi nol, menghasilkan matriks identitas. Dengan demikian, solusi sistem persamaan linear dapat langsung dibaca dari matriks yang telah diubah. Metode ini tidak hanya menyederhanakan penyelesaian sistem persamaan linear tetapi juga dapat digunakan untuk menghitung invers dari suatu matriks.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} & b_m \end{bmatrix} \sim\text{OBE}\sim \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & * \\ 0 & 1 & 0 & \dots & 0 & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 & * \end{bmatrix}$$

2.3 Determinan

Determinan adalah nilai skalar yang terkait dengan suatu matriks persegi. Determinan matriks dapat dihitung untuk matriks 2x2 secara langsung, sedangkan untuk matriks yang lebih besar, determinan dapat dihitung menggunakan ekspansi kofaktor atau eliminasi Gauss. Determinan penting karena dapat memberikan informasi tentang sifat matriks, seperti apakah matriks tersebut dapat dibalik (inversible) atau tidak. Jika determinan dari suatu matriks adalah nol, maka matriks tersebut tidak memiliki balikan, dan sistem persamaan linear yang diwakilinya tidak memiliki solusi tunggal (solusinya mungkin tidak ada atau tak terhingga banyaknya).

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

Untuk matriks berukuran 2x2, gunakan rumus $det(A) = a_{11}a_{22} - a_{12}a_{21}$ dan akan didapatkan hasil determinan dari matriks tersebut.

$$\left[\begin{matrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{matrix}\right] \begin{matrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{matrix}$$

Untuk matriks berukuran 3x3, gunakan rumus
$det(B) = (a_{11}a_{22}a_{33} + a_{12}a_{21}a_{31} + a_{13}a_{21}a_{32}) - (a_{13}a_{22}a_{31} + a_{11}a_{23}a_{32} + a_{12}a_{21}a_{33})$. Untuk matriks lebih besar dari 3x3 bisa menggunakan metode lain seperti kofaktor ataupun kaidah cramer.

2.4 Matriks Balikan (*Invers*)

Matriks balikan dari suatu matriks persegi A adalah matriks A^{-1} yang memenuhi $A \times A^{-1} = I$ di mana I adalah matriks identitas. Matriks balikan hanya ada jika determinan dari matriks tersebut tidak sama dengan nol. Matriks invers dapat didapatkan dengan rumus sebagai berikut.

$$A^{-1} = \frac{1}{\det(A)} \operatorname{adj}(A)$$

Fungsi det(A) adalah determinan dari matriks yang dicari sementara adj(A) adalah adjoin dari matriks A. Adjoin adalah matriks transpose dari matriks kofaktor yang dicari inversnya. Penjelasan tentang matriks kofaktor dan adjoin akan dijelaskan di bagian selanjutnya.

2.5 Matriks Kofaktor

Matriks kofaktor adalah matriks yang elemen-elemennya merupakan kofaktor dari elemen-elemen matriks asalnya. Kofaktor dihitung dengan menghilangkan baris dan kolom yang bersesuaian dengan elemen yang dihitung, kemudian mengambil determinan dari submatriks yang tersisa, dan mengalikan dengan tanda positif atau negatif sesuai dengan pola checkerboard. Matriks kofaktor sering digunakan dalam perhitungan determinan dan matriks balikan. Sebagai contoh, bisa diperhatikan matriks A di bawah ini.

$$A = \begin{bmatrix} 6 & -3 & 1 \\ 2 & 2 & -4 \\ 1 & 5 & 3 \end{bmatrix}$$

$M_{11}=\begin{vmatrix} 2 & -4 \\ 5 & 3 \end{vmatrix}=(2)(3)-(-4)(5)=26$ $\qquad C_{11}=(-1)^{1+1}M_{11}=M_{11}=26$

$M_{12}=\begin{vmatrix} 2 & -4 \\ 1 & 3 \end{vmatrix}=(2)(3)-(-4)(1)=10$ $\qquad C_{12}=(-1)^{1+2}M_{12}=-M_{12}=-10$

$M_{13}=\begin{vmatrix} 2 & 2 \\ 1 & 5 \end{vmatrix}=(2)(5)-(2)(1)=8$ $\qquad C_{13}=(-1)^{1+3}M_{13}=M_{13}=8$

$M_{22}=\begin{vmatrix} 6 & 1 \\ 1 & 3 \end{vmatrix}=(6)(3)-(1)(1)=17$ $\qquad C_{22}=(-1)^{2+2}M_{22}=M_{22}=17$

dan seterusnya untuk M_{21}, M_{23}, M_{31}, M_{32}, M_{33} dihitung dengan cara yang sama

dan seterusnya untuk C_{21}, C_{23}, C_{31}, C_{32}, C_{33} dihitung dengan cara yang sama

Selain itu, kita juga harus memperhitungkan positif dan negatif dari tiap elemen.

$$\begin{bmatrix} + & - & + & - & \cdots \\ - & + & - & + & \cdots \\ + & - & + & - & \cdots \\ - & + & - & + & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix}$$

2.6 Matriks Adjoin

Matriks adjoin adalah transpose dari matriks kofaktor. Matriks ini digunakan dalam proses menghitung invers suatu matriks persegi. Matriks adjoin sangat membantu dalam proses menghitung invers dengan pendekatan manual, terutama untuk matriks berukuran kecil hingga sedang. Matriks adjoin dihitung dari transpose matriks kofaktor yang telah dibahas pada bagian sebelumnya.

$$adj(A) = (kofaktor(A))^T$$

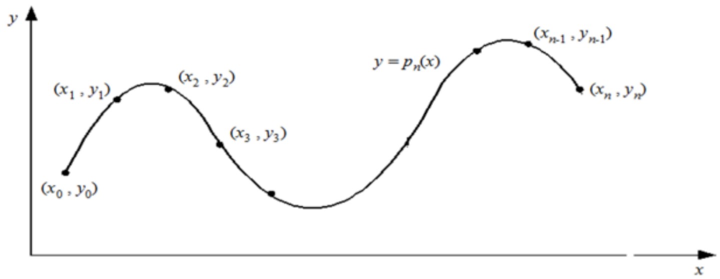
2.7 Kaidah *Cramer*

Kaidah Cramer adalah metode untuk menyelesaikan sistem persamaan linear menggunakan determinan. Metode ini berlaku untuk sistem persamaan linear yang memiliki jumlah persamaan sama dengan jumlah variabel (sistem yang persegi). Untuk menemukan nilai dari suatu variabel dalam sistem, Anda dapat mengganti kolom-koefisien yang bersesuaian dengan variabel tersebut dengan kolom konstanta (di sisi kanan persamaan) dan menghitung determinan dari matriks yang baru terbentuk. Nilai variabel tersebut kemudian dihitung sebagai rasio antara determinan dari matriks yang telah dimodifikasi dan determinan dari matriks koefisien asli.

$$x_1=\frac{\det(A_1)}{\det(A)}, \quad x_2=\frac{\det(A_2)}{\det(A)}, \quad \ldots, \quad x_n=\frac{\det(A_n)}{\det(A)}$$

2.8 Interpolasi Polinom

Interpolasi polinom adalah metode yang digunakan untuk menemukan polinom yang melewati sejumlah titik data yang diberikan. Dalam interpolasi ini, sebuah polinom derajat nnn dibentuk sehingga polinom tersebut melalui setiap titik data dengan sempurna. Interpolasi Lagrange dan interpolasi Newton adalah dua metode umum untuk membangun polinom interpolasi. Polinom interpolasi sering digunakan untuk memperkirakan nilai suatu fungsi di antara titik-titik data yang diberikan atau untuk menghasilkan kurva yang halus melalui titik-titik tersebut.



Interpolasi polinom adalah penggabungan sejumlah n titik menjadi persamaan polinom yang memiliki lintasan tertentu. Pola pada persamaan polinomial berupa,

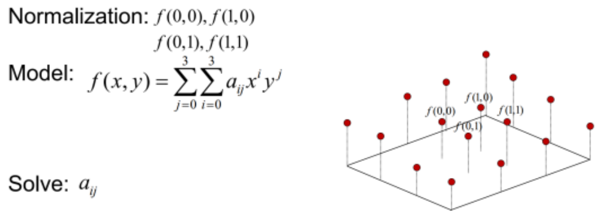
$$P_n(x) = a_0 + a_1x + a_1x^2 + ... + a_nx^n$$

Persamaan di atas dapat digunakan untuk mengakprosimasi sebuah nilai y berdasarkan nilai x pada range $[x_0, x_n]$. Untuk menghasilkan persamaan dengan pangkat tertinggi n, diperlukan titik sebanyak n+1. Untuk mendapatkan nilai $a_0, a_1, a_2, ...$ kita perlu penyelesaian dengan SPL. Pada tubes ini kita mengaplikasikan konsep penyelesaian SPL dengan matriks yang ter-augmentasi dan menggunakan berbagai metode penyelesaian SPL (Gauss, Gauss-Jordan, Cramer) sehingga didapatkan dengan nilai-nilai a fungsi $P_n(x) = a_0 + a_1x + a_2x^2 + ... + a_nx^n$ untuk mengaproksimasi nilai y dari sebuah x.

2.9 Interpolasi Bicubic Spline

Interpolasi bicubic spline adalah teknik interpolasi dua dimensi yang menggunakan polinomial kubik pada setiap arah (x dan y). Spline kubik bicubic memberikan transisi yang halus antara titik-titik data dengan memperhitungkan turunan pertama dan kedua dari fungsi pada setiap titik data. Teknik ini sering digunakan dalam grafik komputer dan pengolahan citra digital untuk menghasilkan gambar dengan kualitas lebih tinggi atau memperhalus tepi objek. Dibandingkan interpolasi bilinear, interpolasi bicubic memberikan hasil yang lebih halus, terutama pada gambar dengan perubahan intensitas yang cepat.

Dalam pemrosesan menggunakan interpolasi bicubic spline digunakan 16 buah titik, 4 titik referensi utama di bagian pusat, dan 12 titik di sekitarnya sebagai aproksimasi turunan dari keempat titik referensi untuk membagun permukaan bikubik. Bentuk pemodelannya adalah sebagai berikut.



Selain melibatkan model dasar, juga digunakan model turunan berarah dari kedua sumbu, baik terhadap sumbu x, sumbu y, mapun keduanya. Persamaan polinomial yang digunakan adalah sebagai berikut

$$f(x,y)=\sum_{i=0}^{3}\sum_{j=0}^{3}a_{ij}x^i y^j$$

$$f_x(x,y)=\sum_{j=0}^{3}\sum_{i=1}^{3}a_{ij}ix^{i-1}y^j$$

$$f_y(x,y)=\sum_{j=1}^{3}\sum_{i=0}^{3}a_{ij}jx^i y^{j-1}$$

$$f_{xy}(x,y)=\sum_{j=0}^{3}\sum_{i=0}^{3}a_{ij}ijx^{i-1}y^{j-1}$$

2.10 Regresi Linier

Regresi linier berganda adalah teknik statistik yang digunakan untuk memodelkan hubungan antara variabel terikat (dependen) dengan dua atau lebih variabel bebas (independen). Persamaan regresi linier berganda memiliki bentuk dasar:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + \epsilon$$

di mana Y adalah variabel terikat, $X_1, X_2, ..., X_n$ adalah variabel bebas, $\beta_0, \beta_1, ..., \beta_n$ adalah koefisien regresi, dan ϵ adalah kesalahan (error) yang tidak dijelaskan oleh model. Untuk mendapatkan nilai dari β_i, dapat digunakan *Normal Estimation Equation for Multiple Linear Regression sebagai berikut:*

$$\begin{array}{lllll}
nb_0 + b_1\sum_{i=1}^{n}x_{1i} & + b_2\sum_{i=1}^{n}x_{2i} & + \cdots + b_k\sum_{i=1}^{n}x_{ki} & = \sum_{i=1}^{n}y_i \\
b_0\sum_{i=1}^{n}x_{1i} + b_1\sum_{i=1}^{n}x_{1i}^2 & + b_2\sum_{i=1}^{n}x_{1i}x_{2i} + & \cdots + b_k\sum_{i=1}^{n}x_{1i}x_{ki} & = \sum_{i=1}^{n}x_{1i}y_i \\
\vdots \quad \vdots & \vdots & \vdots & \vdots \\
b_0\sum_{i=1}^{n}x_{ki} + b_1\sum_{i=1}^{n}x_{ki}x_{1i} & + b_2\sum_{i=1}^{n}x_{ki}x_{2i} + & \cdots + b_k\sum_{i=1}^{n}x_{ki}^2 & = \sum_{i=1}^{n}x_{ki}y_i
\end{array}$$

Teknik ini digunakan untuk memprediksi variabel terikat berdasarkan informasi dari beberapa variabel bebas dan sangat berguna dalam analisis data.

2.11 Kuadratik Berganda

Regresi kuadratik berganda adalah pengembangan dari regresi linier berganda di mana model tidak hanya mencakup variabel bebas secara linier, tetapi juga kuadrat dari variabel bebas tersebut. Model ini digunakan untuk menangkap hubungan non-linear antara variabel terikat dan variabel bebas. Bentuk dasar dari persamaan regresi kuadratik berganda adalah: $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + ... + \beta_n X_n^2 + \epsilon$ Dengan menambahkan kuadrat variabel, model dapat mencerminkan pola yang lebih kompleks, seperti kurva atau parabola, yang mungkin tidak dapat dijelaskan dengan model linier sederhana.

Dalam kasus ini, proses mengubah data-data dalam regresi kuadratik berganda cukup berbeda dengan Regresi Linier Berganda. Bentuk persamaan dari regresi kuadratik ada 3, yaitu:

d. Variabel Linier: Variabel dengan derajat satu seperti X, Y, dan Z

e. Variabel Kuadrat: Variabel dengan derajat dua seperti X^2

f. Variabel Interaksi: 2 Variabel dengan derajat satu yang dikalikan dengan satu sama lain seperti XY, YZ, dan XZ

Setiap n-peubah, jumlah variabel linier, kuadrat, dan interaksi akan berbeda-beda.

$$\begin{pmatrix} N & \sum u_i & \sum v_i & \sum u_i^2 & \sum u_i v_i & \sum v_i^2 \\ \sum u_i & \sum u_i^2 & \sum u_i v_i & \sum u_i^3 & \sum u_i^2 v_i & \sum u_i v_i^2 \\ \sum v_i & \sum u_i v_i & \sum v_i^2 & \sum u_i^2 v_i & \sum u_i v_i^2 & \sum v_i^3 \\ \sum u_i^2 & \sum u_i^3 & \sum u_i^2 v_i & \sum u_i^4 & \sum u_i^3 v_i & \sum u_i^2 v_i^2 \\ \sum u_i v_i & \sum u_i^2 v_i & \sum u_i v_i^2 & \sum u_i^3 v_i & \sum u_i^2 v_i^2 & \sum u_i v_i^3 \\ \sum v_i^2 & \sum u_i v_i^2 & \sum v_i^3 & \sum u_i^2 v_i^2 & \sum u_i v_i^3 & \sum v_i^4 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix} = \begin{pmatrix} \sum y_i \\ \sum y_i u_i \\ \sum y_i v_i \\ \sum y_i u_i^2 \\ \sum y_i u_i v_i \\ \sum y_i v_i^2 \end{pmatrix}$$

N menandakan jumlah peubah, terdapat 2 variabel linier yaitu u_i dan v_i, 2 variabel kuadrat yaitu u_i^2 dan v_i^2 dan 1 variabel interaksi yaitu uv. Untuk setiap n-peubah, akan terdapat 1 konstan N (Terlihat di bagian atas kiri gambar), n variabel linier, n variabel kuadrat, dan C_2^n variabel linier (dengan syarat n > 1). Tentu dengan bertambahnya peubah n, ukuran matriks akan bertumbuh lebih besar dibandingkan regresi linier berganda tetapi solusi tetap bisa didapat dengan menggunakan SPL.

BAB 3
IMPLEMENTASI DAN PROGRAM

3.1 ADT Matriks

File ADT Matrix yang dinamakan Matrix.java merupakan file yang menjadi ADT pusat dari pengerjaan tugas besar kali ini. File ini berisi beberapa hal sebagai berikut.

3.1.1 Atribut

Atribut	Penjelasan
`private double[][] ELMT`	Menyimpan elemen-elemen dalam array 2 dimensi.
`private int rows`	Menyimpan ukuran baris matrix.
`private int cols`	Menyimpan ukuran kolom matrix.

3.1.2 Konstruktor

Konstruktor	Penjelasan
`public void CreateMatrix(Matrix M, int y, int x)`	Melakukan inisialisasi matrix dengan y baris dan x kolom.

3.1.3 Metode

Metode	Penjelasan
`public double getElement(int baris, int kolom)`	Melakukan return nilai elemen ketika User memasukkan baris dan kolom Matrix.
`public int getRow(Matrix M)`	Melakukan return jumlah baris Matrix.
`public int getCol(Matrix M)`	Melakukan return jumlah kolom Matrix.
`public int getFirstIdx(Matrix M)`	Melakukan return index pertama Matrix.
`public int getLastRowIdx(Matrix M)`	Melakukan return index terakhir baris Matrix.

`public int getLastColIdx(Matrix M)`	Melakukan return index terakhir kolom Matrix.
`public boolean isRowValid(Matrix M, int i)`	Melakukan validasi index baris, mengembalikan true apabila 0 ≤ index ≤ index terakhir baris.
`public boolean isColValid(Matrix M, int i)`	Melakukan validasi index kolom, mengembalikan true apabila 0 ≤ index ≤ index terakhir kolom.
`public void setElement(Matrix M, int baris, int kolom, double value)`	Memasukkan suatu double sebagai nilai dari elemen Matrix.
`public void readMatrix(Matrix M)`	Memvalidasi ukuran Matrix M x N dan membaca setiap elemen Matrix.
`public void printMatrix(Matrix M)`	Mengeluarkan matrix dalam bentuk rows x cols. Setiap elemen Matrix dipisahkan oleh spasi dengan newline di setiap baris.
`public void copyMatrix(Matrix mIn, Matrix mOut)`	Membuat salinan Matrix dari mIn dan menyimpan semua elemen mIn sebagai mOut.
`public Matrix transposeMatrix(Matrix M)`	Melakukan return matrix yang sudah ditranspose.
`public Matrix multiplyMatrix(Matrix M1, Matrix M2)`	Melakukan return Matrix hasil perkalian dua buah Matrix.
`public Matrix addMatrix(Matrix M1, Matrix M2)`	Melakukan return Matrix hasil penjumlahan dua buah Matrix.
`public Matrix subMatrix(Matrix M1, Matrix M2)`	Melakukan return Matrix hasil pengurangan dua buah Matrix.

3.2 Eliminasi Gauss

Eliminasi Gauss adalah suatu file yang mengubah suatu input Matrix menjadi bentuk Matriks OBE nya dengan menggunakan rumus Gauss.

3.4.1 Atribut

Atribut	Penjelasan
`private static double EPSILON = 1.0E-10;`	Konstanta EPSILON untuk memvalidasi angka 0 pada operasi.

3.4.2 Metode

Metode	Penjelasan
`public void gauss(Matrix var1)`	Melakukan eliminasi gauss pada Matrix input, sehingga menghasilkan Matrix yang memiliki satu utama.

3.3 Eleminasi Gauss - Jordan

Eliminasi Gauss adalah suatu file yang mengubah suatu input Matrix menjadi bentuk Matriks OBE nya dengan menggunakan rumus Gauss-Jordan.

3.3.1 Atribut

Atribut	Penjelasan
`private static double EPSILON = 1.0E-10;`	Konstanta EPSILON untuk memvalidasi angka 0 pada operasi.

3.3.2 Metode

Metode	Penjelasan
`public void divideByX(Matrix M, int row, double x)`	Membagikan suatu baris (row) pada matrix M dengan suatu double x, yang dalam pengoperasian gauss.
`public void switchRows(Matrix M, int row1, int row2)`	Menukar dua baris pada suatu Matrix M, dengan baris pada row1 bertukar posisi dengan baris pada row 2. Umum digunakan pada operasi Gauss dan Gauss-Jordan.
`public void plusMinRows(Matrix M, int`	Mengurangkan atau menambahkan setiap komponen pada baris row1 dengan setiap

`row1, int row2, double multiplier)`	komponen pada row2 yang dikalikan dengan multiplier.
`public double multiplier(Matrix M,int d, int rowx)`	Merupakan rumus multiplier yaitu berupa rasio dari kolom pada rowx dengan diagonalnya.
`public void gaussjordan(Matrix M)`	Melakukan eliminasi Gauss-Jordan pada Matrix input, sehingga menghasilkan Matrix yang memiliki satu utama.

3.4 Determinan (Reduksi Baris)

DeterminanBaris adalah suatu file java yang menerima suatu input Matrix M dan kemudian dicari determinan matrix nya menggunakan rumus reduksi baris.

3.4.1 Metode

Metode	Penjelasan
`public double determinanbaris(Matrix M)`	Mencari determinan suatu Matrix M dengan menggunakan eliminasi pada setiap baris menjadi matrix segitiga atas.

3.5 Hitung Kofaktor

Program ini dapat menerima matriks masukan untuk dihitung matriks kofaktornya dan mengeluarkan hasil perhitungan dalam bentuk matriks.

3.5.1 Metode

Metode	Penjelasan
`public Matrix keluaran(Matrix mIn)`	Metode ini bertugas untuk menerima input pertama dari matriks dan untuk keluaran matriks.
`public double hitung(Matrix M)`	Metode ini bertugas untuk menghitung determinan dari sebuah matriks dan mengeluarkan hasil determinan tersebut.
`public Matrix minor(Matrix M, int row, int col)`	Metode ini bertugas untuk membuat matriks minor dari matriks masukan ataupun matriks hasil rekursi metode hitung.

3.6 Determinan (Kofaktor)

Program ini dapat menerima masukan matriks MxM dan dapat menghitung determinan matriks MxM tersebut dengan menggunakan metode kofaktor.

3.6.1 Metode

Metode	Penjelasan
`public double detkof(Matrix M)`	Metode ini secara garis besar menggunakan metode hitung yang ada pada file kofaktor.java namun dibuat secara terpisah untuk menghitung determinan matriks menggunakan teori kofaktor.

3.7 Matriks Balikan (*Adjoin* - Determinan)

Program ini dapat menerima masukan matriks dan menghitung invers matriks dengan rumus $\frac{1}{det(M)} \times \ adj(M)$. Program ini mengeluarkan hasil dalam bentuk matriks juga.

3.7.1 Atribut

Atribut	Penjelasan
`double MARK = -999;`	MARK untuk matriks yang tidak dapat dikerjakan.

3.7.2 Metode

Metode	Penjelasan
`public Matrix balikan(Matrix M)`	Metode ini berfungsi untuk menghitung invers matriks menggunakan determinan dan adjoin matriks.

3.8 Matriks Balikan (Gauss - Jordan)

Program ini dapat menerima masukan matriks dan menghitung invers matriks dengan rumus Gauss-Jordan dan matriks identitas. Program ini mengeluarkan hasil dalam bentuk matriks juga.

3.8.1 Metode

Metode	Penjelasan
`public Matrix inversgauss(Matrix M)`	Mengembalikan suatu Matrix Invers, dengan metode berupa penambahan matrix identitas di kanan matrix awal, dan di gauss jordan, lalu diambil matrix yang di kanan.

3.9 SPL (Eleminasi Gauss)

Program ini dapat menerima masukan matriks dan melakukan operasi SPL dimana operasi SPL dilakukan dengan menggunakan fungsi Gauss, dan juga Gauss Jordan.

3.9.1 Metode

Metode	Penjelasan
`private boolean kosong(Matrix M, int i)`	Mengembalikan true jika suatu baris i dalam matriks M.
`public void gauss(Matrix M1, Matrix M2, Boolean kosong, Boolean no_solution, StringBuilder result)`	Mengembalikan suatu Matrix Solusi Gauss ke StringBuilder, yang merupakan gabungan dari Matrix x M1, dan Matrix solusi M2.
`public void gaussjordan(Matrix M1, Matrix M2, Boolean kosong, Boolean no_solution, StringBuilder result)`	Mengembalikan suatu Matrix Solusi GaussJordan ke StringBuilder, yang merupakan gabungan dari Matrix x M1, dan Matrix solusi M2.

3.10 SPL (Matriks Balikan)

Program ini dapat menghitung hasil SPL berupa x_1, x_2, ..., x_n menggunakan metode balikan (invers) dengan rumus $x = A^{-1}B$ dan mengeluarkannya dalam bentuk Matriks nx1.

3.10.1 Metode

Metode	Penjelasan
`public void cramer(Matrix M1, Matrix M2, Boolean kosong, Boolean no_solution, StringBuilder result)`	Metode ini berfungsi untuk menerima dua buah matriks, menghitung hasil SPL, dan menghasilkan SPL dalam bentuk x_1, x_2, ..., x_n yang dikeluarkan sebagai matriks nx1.

3.11 SPL (Metode Cramer)

Program ini dapat mengeluarkan hasil SPL berupa x_1, x_2, ..., x_n menggunakan metode/kaidah Cramer.

3.11.1 Atribut

Atribut	Penjelasan
`double MARK = -999;`	MARK untuk matriks yang tidak dapat dikerjakan.

3.11.2 Metode

Metode	Penjelasan
`public void SPLinv(Matrix M1, Matrix M2, Boolean kosong, Boolean no_solution, StringBuilder result)`	Metode ini berfungsi untuk menerima dua buah matriks, menghitung hasil SPL, dan menghasilkan SPL dalam bentuk $x_1, x_2, ..., x_n$ yang dikeluarkan sebagai matriks nx1.

3.12 Interpolasi Polinom

Program ini dapat mengeluarkan hasil taksiran Polinom melalui input x dan y dengan banyaknya test case, menggunakan rumus Interpolasi Polinomial.

3.12.1 Metode

Metode	Penjelasan
`public void polinomial()`	Menerima suatu input n yang berupa derajat polinom, serta n input x dan y, dan input x yang ingin ditaksir dan akan mengembalikan suatu

3.13 Regresi Linear Berganda

Program ini dapat mengeluarkan hasil taksiran dari suatu data x1, x2, … xn, dan hasil y, menggunakan rumus regresi linear berganda.

3.13.1 Metode

Metode	Penjelasan
`public Matrix processRegression(Matrix M, Matrix y)`	Menerima suatu input matrix M dan matrix y, dan mengembalikan suatu matriks Regresi Linear Berganda.

3.14 Regresi Kuadratik Berganda

Program ini dapat mengeluarkan hasil taksiran dari suatu data x1, x2, … xn, dan hasil y, menggunakan rumus regresi kuadratik berganda.

3.14.1 Metode

Metode	Penjelasan
`public void regresikuadratik(Matrix MX, Matrix MY, StringBuilder result)`	Menerima suatu input matrix x dan matrix y, dan mengembalikan suatu matriks Regresi Linear Berganda pada StringBuilder result.
`private boolean kosong(Matrix M, int i)`	Mengembalikan true jika suatu baris i dalam matriks M.
`public void regresikuadratiktaksir(Matrix MX, Matrix MY, double xi, double xj, StringBuilder result)`	Menerima suatu input matrix x dan matrix y, dan juga input x dan y, dan mengembalikan suatu hasil taksiran x pada StringBuilder result.

3.15 Bicubic Spline Interpolation

Program ini menerima suatu matriks 4x4, dan dengan menggunakan rumus B=Ax, pada matriks x, untuk dapat menaksir suatu nilai tengah yang diharapkan dari input x dan y.

3.15.1 Metode

Metode	Penjelasan
`public double Interpolation(Matrix M, double x, double y)`	Menerima suatu input Matrix 4x4 M, dan juga suatu input x dan y, dan akan mengembalikan hasil interpolasi menggunakan rumus B = Ax.

3.16 Main & GUI

File Main.java langsung terhubung ke GUI untuk interface pengguna. GUI yang digunakan adalah JavaFX.

3.16.1 Atribut

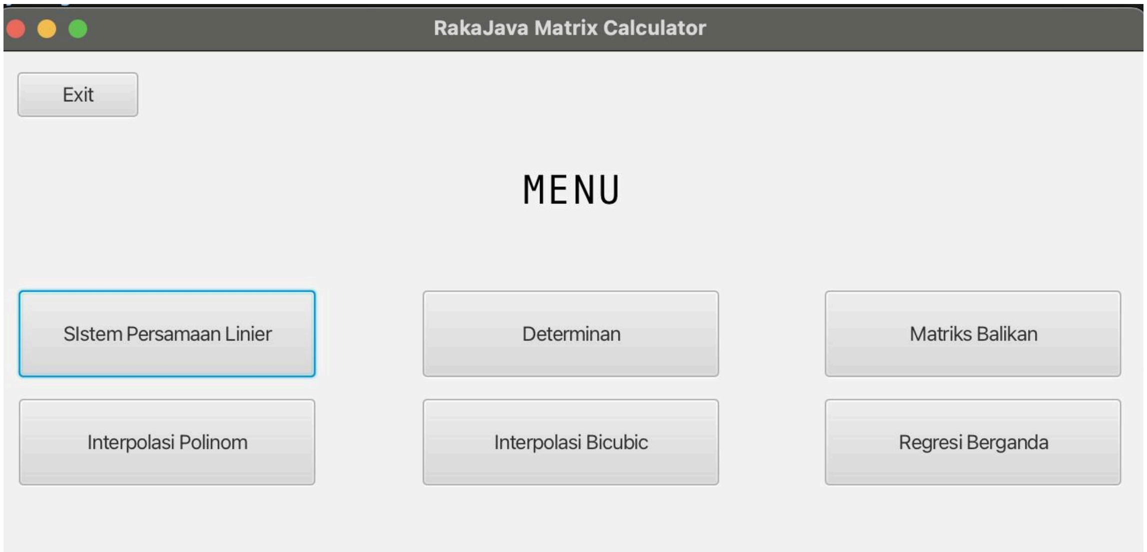
Atribut	Penjelasan
`private Scene startScene;`	Laman start diarahkan menjadi laman utama yang muncul saat program pertama kali dijalankan.

3.16.2 Metode

Metode	Penjelasan
`public void start(Stage primaryStage)`	Memulai primaryStage sehingga ketika code di run, stage utama (start page) akan muncul.

BAB 4
EKSPERIMEN

4.1 Inisiasi Program



Gambar 4.1.1 Tampilan Menu Utama

Gambar di atas adalah tampilan awal dari GUI kami. Pada tampilan tersebut, pengguna bisa langsung memilih menu untuk menyelesaikan persamaan matriks. Terdapat enam menu, yaitu penyelesaian sistem persamaan linier, menghitung determinan, menghitung invers matriks, menghitung interpolasi polinom, menghitung interpolasi bicubic, dan menghitung regresi berganda.

4.2 Testing

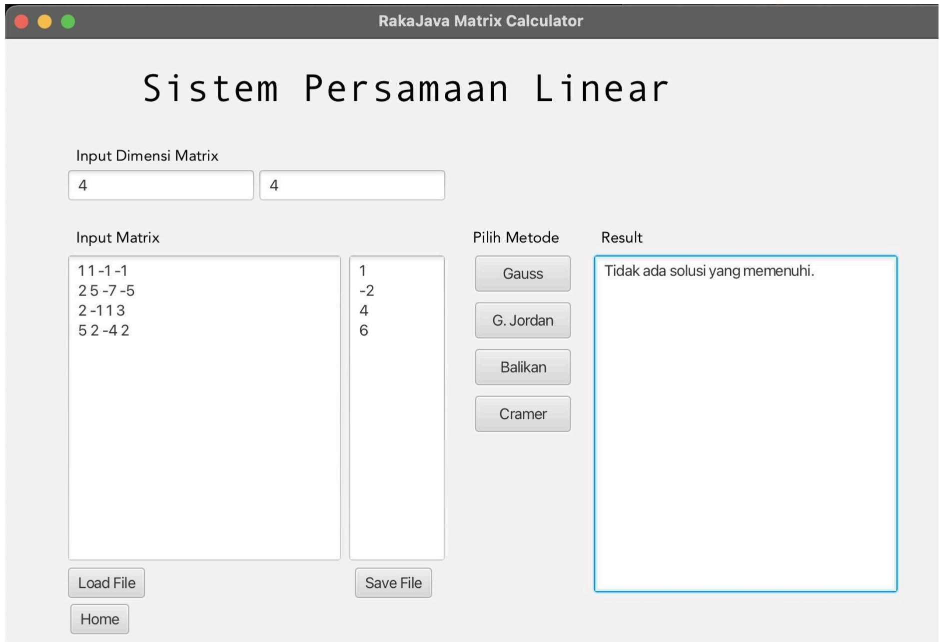
4.2.1 SPL Ax = B

a. Soal 1

$$A = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 2 & 5 & -7 & -5 \\ 2 & -1 & 1 & 3 \\ 5 & 2 & -4 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ -2 \\ 4 \\ 6 \end{bmatrix}$$

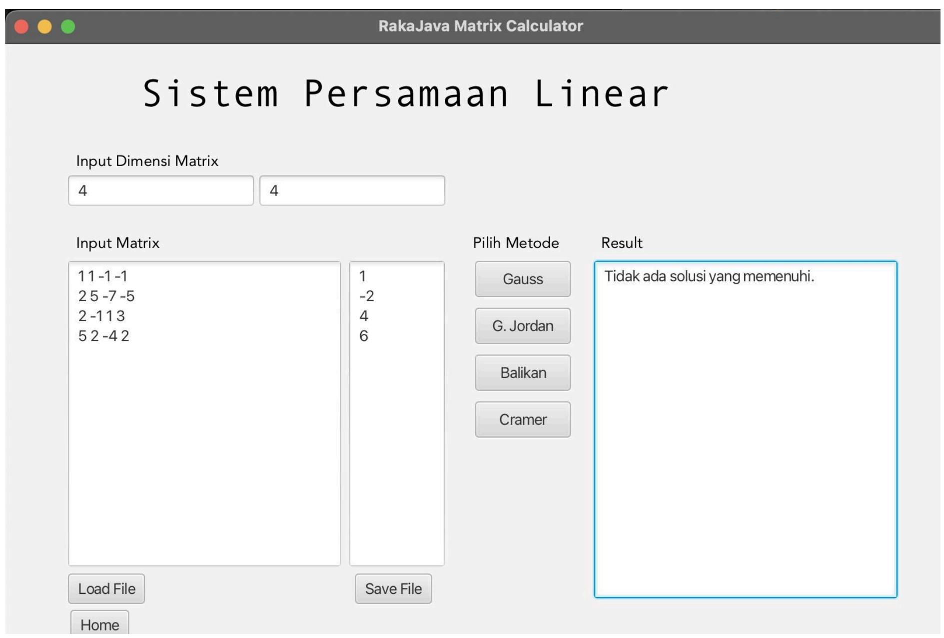
Gambar 4.2.1.1 Soal 1

- Metode Gauss



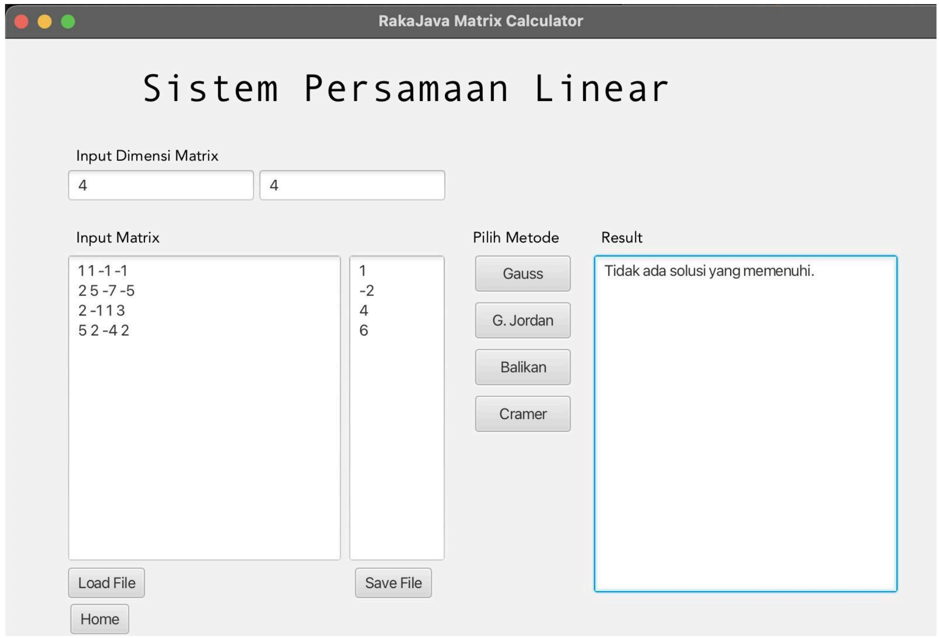
Gambar 4.2.1.2 Hasil Perhitungan Metode Gauss

- Metode Gauss - Jordan



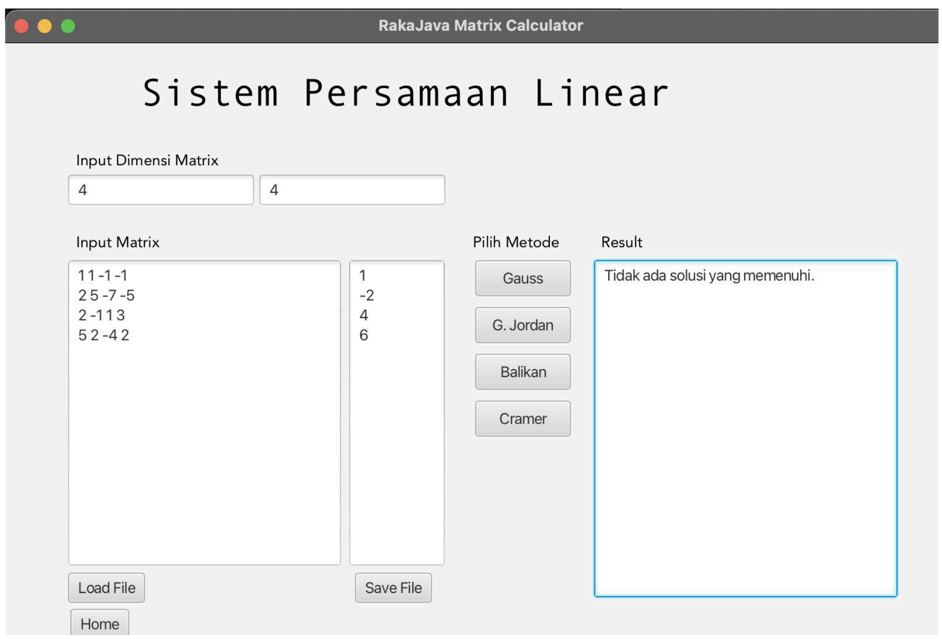
Gambar 4.2.1.3 Hasil Perhitungan Metode Gauss - Jordan

- Metode Invers



Gambar 4.2.1.4 Hasil Perhitungan Metode Balikan

- Metode Cramer



Gambar 4.2.1.5 Hasil Perhitungan Metode Balikan

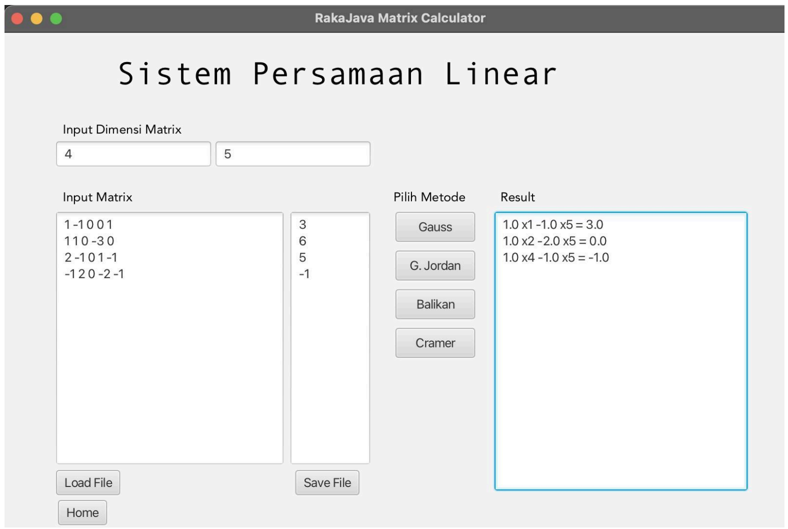
- Analisis
  Hasil penghitungan SPL dari empat metode tersebut menghasilkan jawaban yang sama.

b. Soal 2

$$A = \begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 1 & 1 & 0 & -3 & 0 \\ 2 & -1 & 0 & 1 & -1 \\ -1 & 2 & 0 & -2 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 6 \\ 5 \\ -1 \end{bmatrix}$$

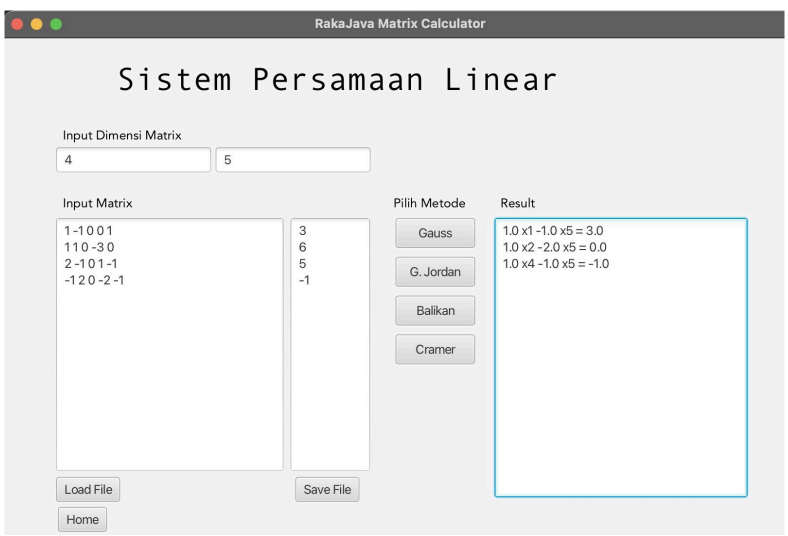
Gambar 4.2.1.6 Soal 2

- Metode Gauss



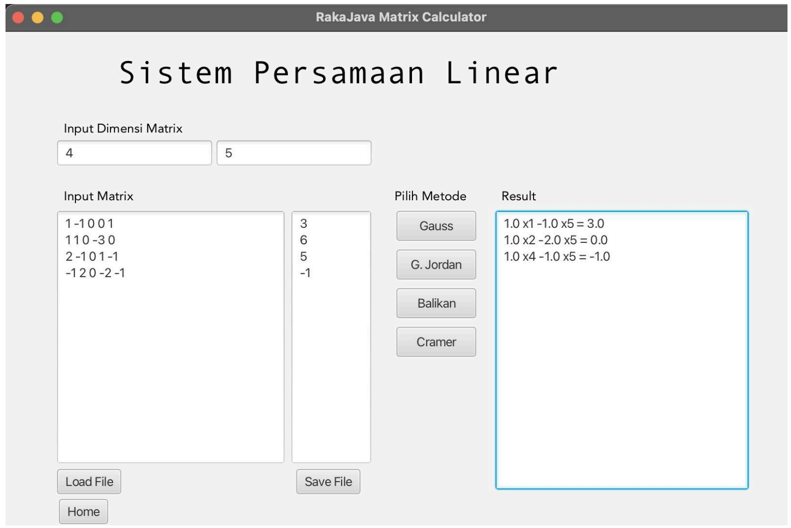
Gambar 4.2.1.7 Hasil Perhitungan Metode Gauss

- Metode Gauss - Jordan



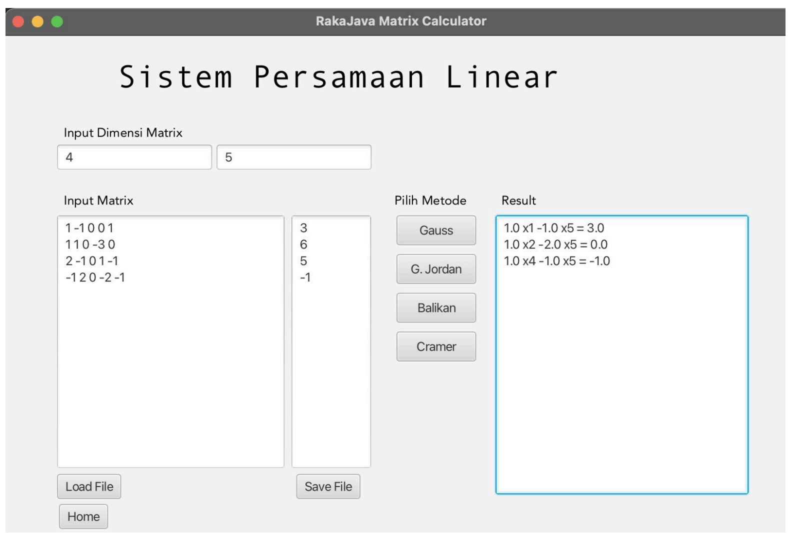
Gambar 4.2.1.8 Hasil Perhitungan Metode Gauss - Jordan

- Metode Invers



Gambar 4.2.1.9 Hasil Perhitungan Metode Invers

- Metode Cramer



Gambar 4.2.1.10 Hasil Perhitungan Metode Cramer

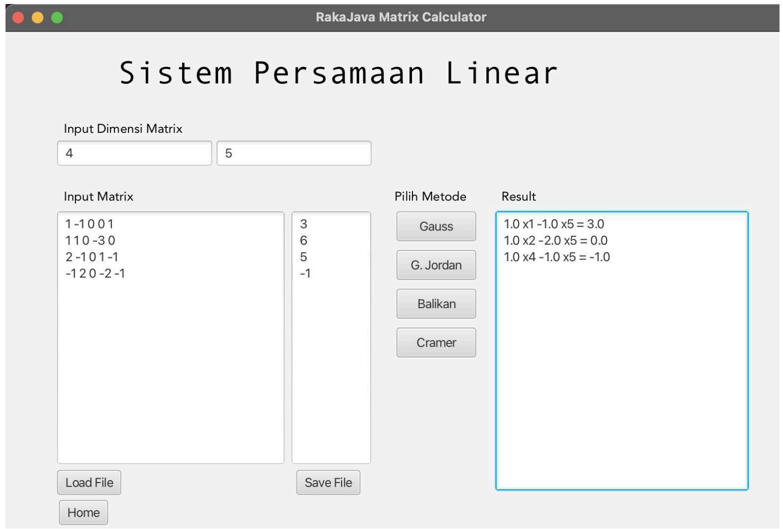
- Analisis
  Hasil penghitungan SPL dari empat metode tersebut menghasilkan jawaban yang sama.

c. Soal 3

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}$$

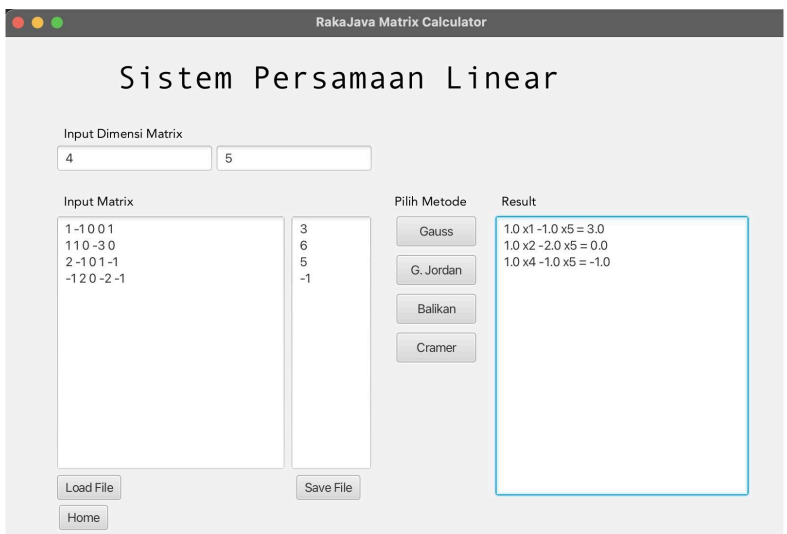
Gambar 4.2.1.11 Soal 3

- Metode Gauss



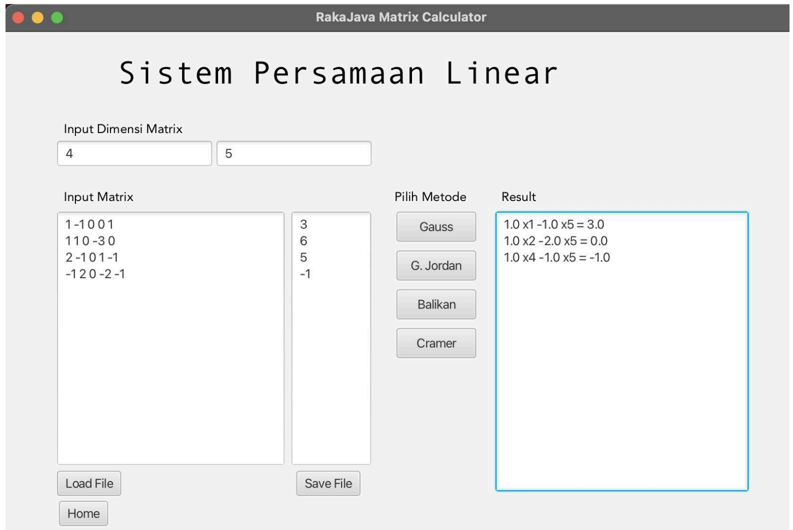
Gambar 4.2.1.12 Hasil Perhitungan Metode Gauss

- Metode Gauss - Jordan



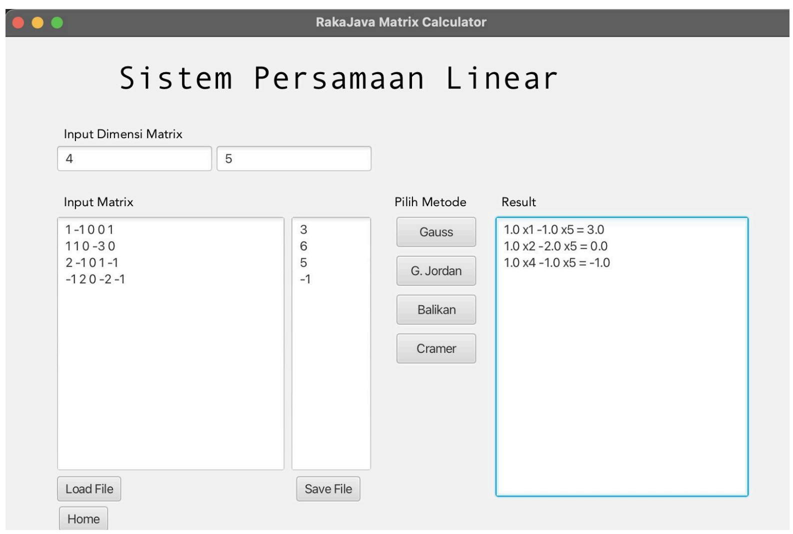
Gambar 4.2.1.13 Hasil Perhitungan Metode Gauss - Jordan

- Metode Invers



Gambar 4.2.1.14 Hasil Perhitungan Metode Invers

- Metode Cramer



Gambar 4.2.1.15 Hasil Perhitungan Metode Cramer

- Analisis
  Hasil penghitungan SPL dari empat metode tersebut menghasilkan jawaban yang sama.

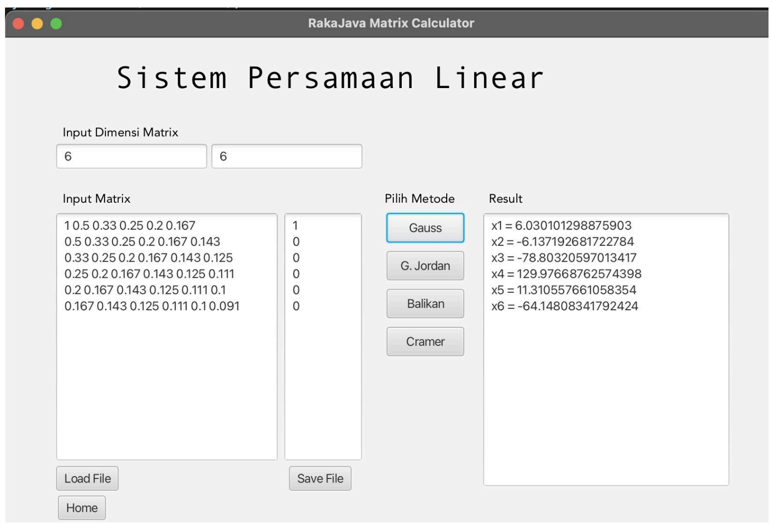
d. Soal 4

$$H = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \cdots & \frac{1}{n+2} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \cdots & \frac{1}{2n+1} \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

H adalah matriks *Hilbert*. Cobakan untuk $n = 6$ dan $n = 10$.

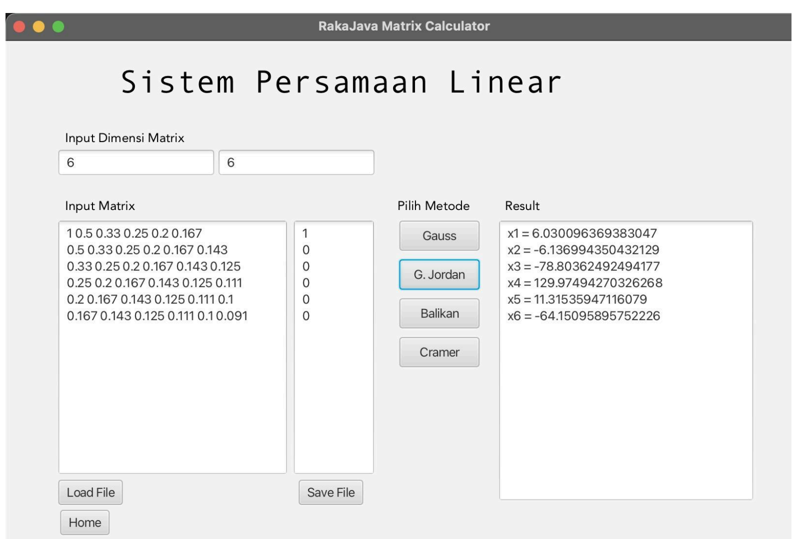
Gambar 4.2.1.16 Soal 4

- Metode Gauss 6x6



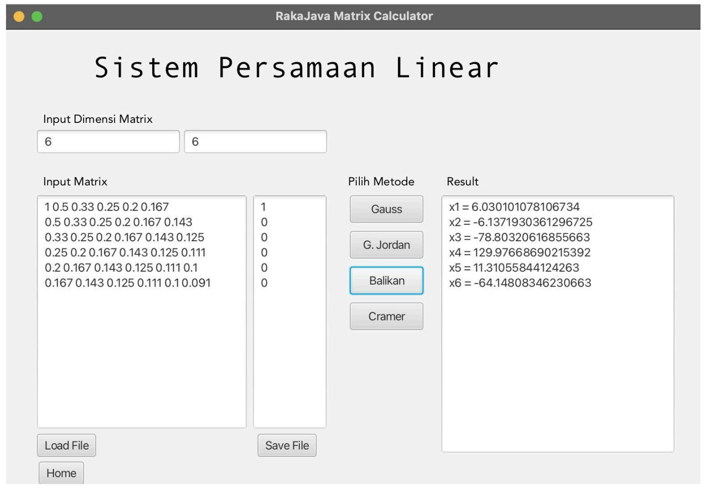
Gambar 4.2.1.17 Hasil Perhitungan Metode Gauss

- Metode Gauss - Jordan 6x6



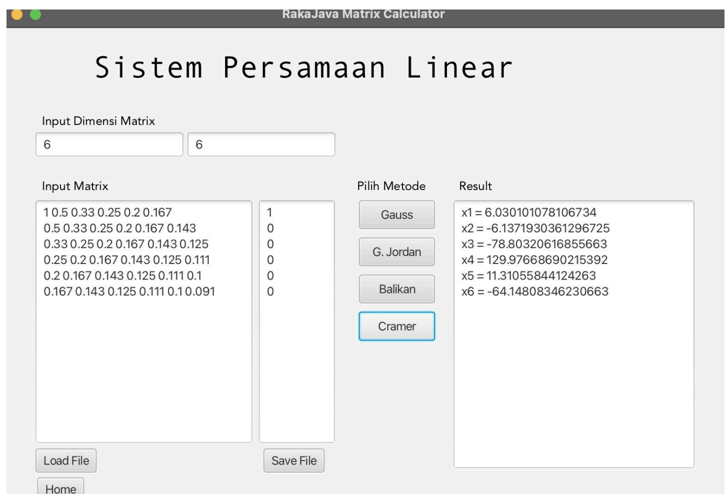
Gambar 4.2.1.13 Hasil Perhitungan Metode Gauss - Jordan

- Metode Invers 6x6



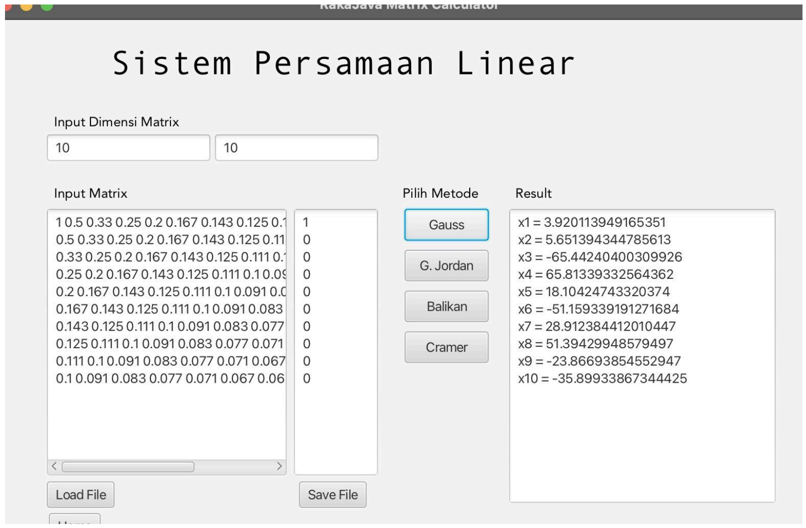
Gambar 4.2.1.14 Hasil Perhitungan Metode Invers

- Metode Cramer 6x6



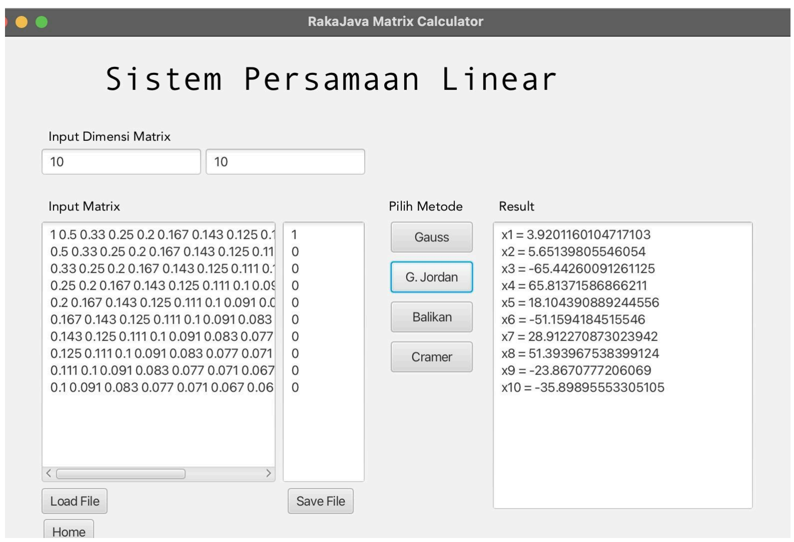
Gambar 4.2.1.15 Hasil Perhitungan Metode Cramer

- Metode Gauss 10x10



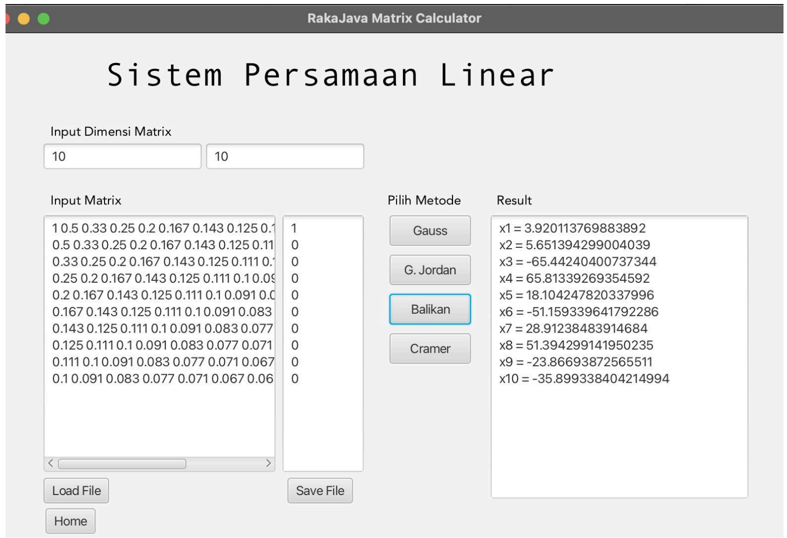
Gambar 4.2.1.16 Hasil Perhitungan Metode Gauss

- Metode Gauss - Jordan 10x10



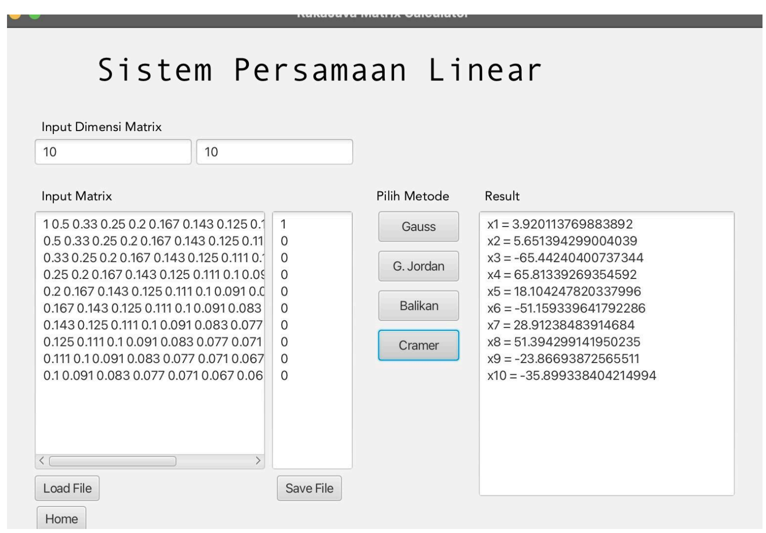
Gambar 4.2.1.17 Hasil Perhitungan Metode Gauss - Jordan

- Metode Invers 10x10



Gambar 4.2.1.18 Hasil Perhitungan Metode Invers

- Metode Cramer 10x10



Gambar 4.2.1.19 Hasil Perhitungan Metode Cramer

- Analisis
  Terdapat beberapa perbedaan desimal di akhir karena penggunaan *type* data double sangat presisi dan perubahan bit dapat mempengaruhi hasil akhir.

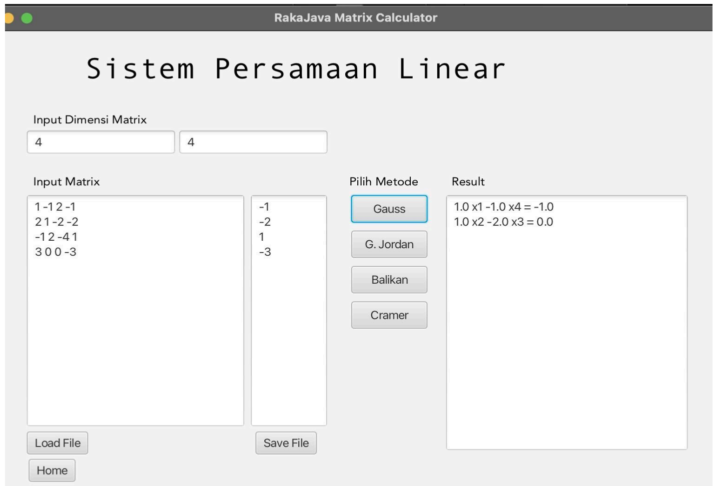
4.2.2 SPL Matriks Augmented

a. Soal 1

$$\begin{bmatrix} 1 & -1 & 2 & -1 & -1 \\ 2 & 1 & -2 & -2 & -2 \\ -1 & 2 & -4 & 1 & 1 \\ 3 & 0 & 0 & -3 & -3 \end{bmatrix}.$$

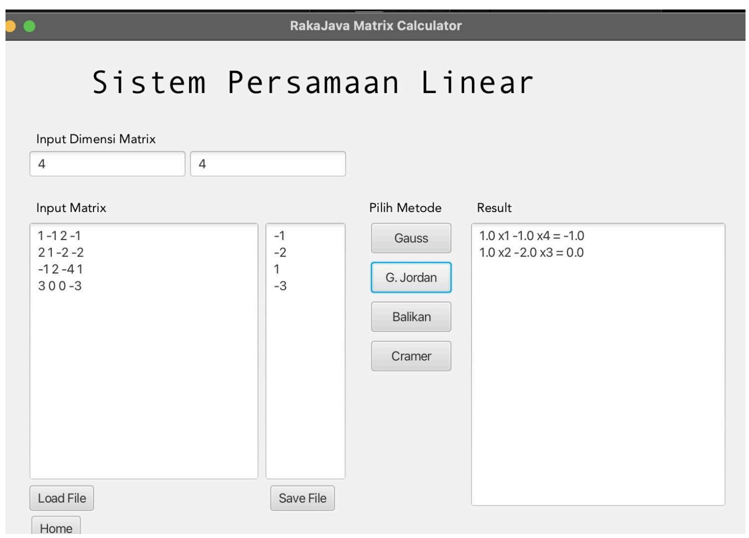
Gambar 4.2.2.1 Soal 1

- Metode Gauss



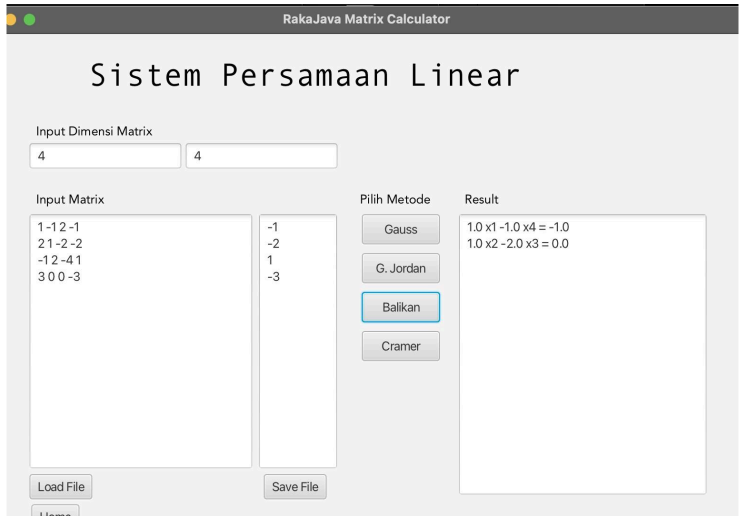
Gambar 4.2.2.2 Hasil Perhitungan Metode Gauss

- Metode Gauss - Jordan



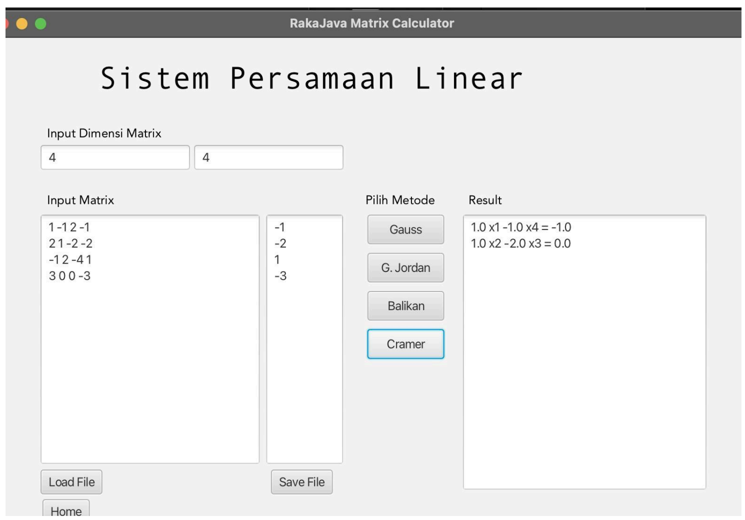
Gambar 4.2.2.3 Hasil Perhitungan Metode Gauss - Jordan

- Metode Invers



Gambar 4.2.2.4 Hasil Perhitungan Metode Invers

- Metode Cramer



Gambar 4.2.2.5 Hasil Perhitungan Metode Cramer

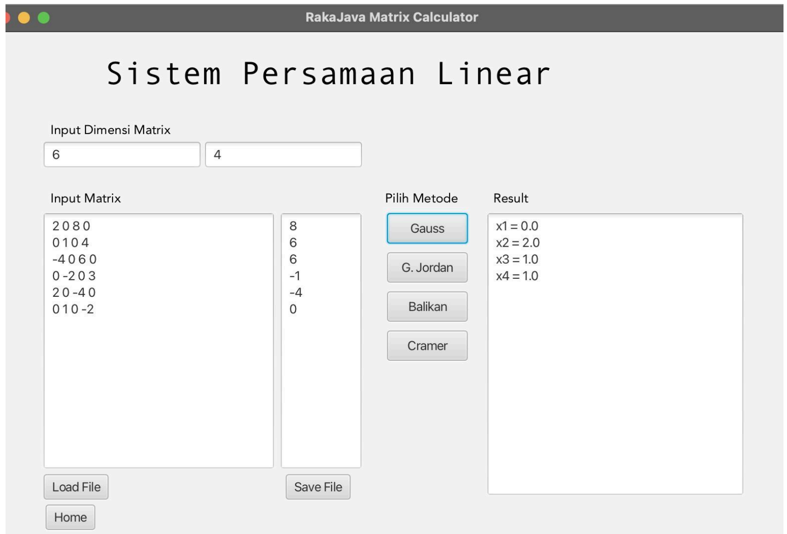
- Analisis
  Hasil penghitungan SPL dari empat metode tersebut menghasilkan jawaban yang sama.

b. Soal 2

$$\begin{bmatrix} 2 & 0 & 8 & 0 & 8 \\ 0 & 1 & 0 & 4 & 6 \\ -4 & 0 & 6 & 0 & 6 \\ 0 & -2 & 0 & 3 & -1 \\ 2 & 0 & -4 & 0 & -4 \\ 0 & 1 & 0 & -2 & 0 \end{bmatrix}.$$

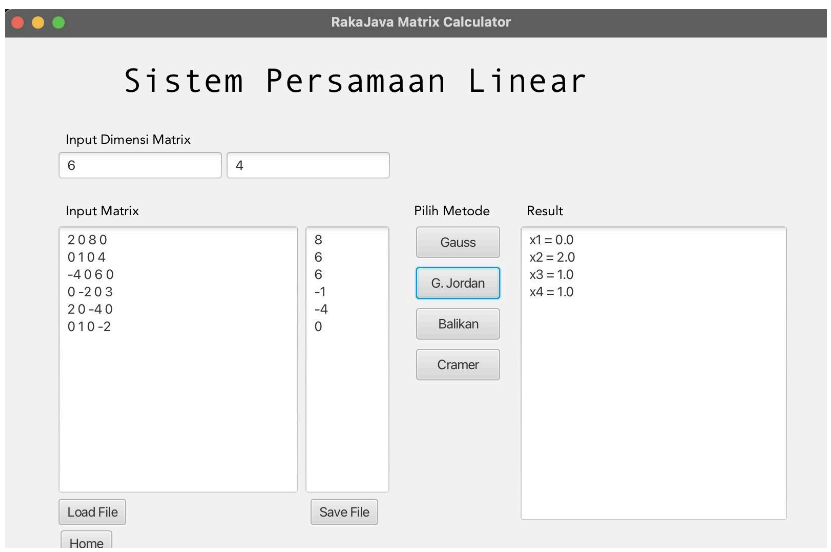
Gambar 4.2.2.6 Soal 2

- Metode Gauss



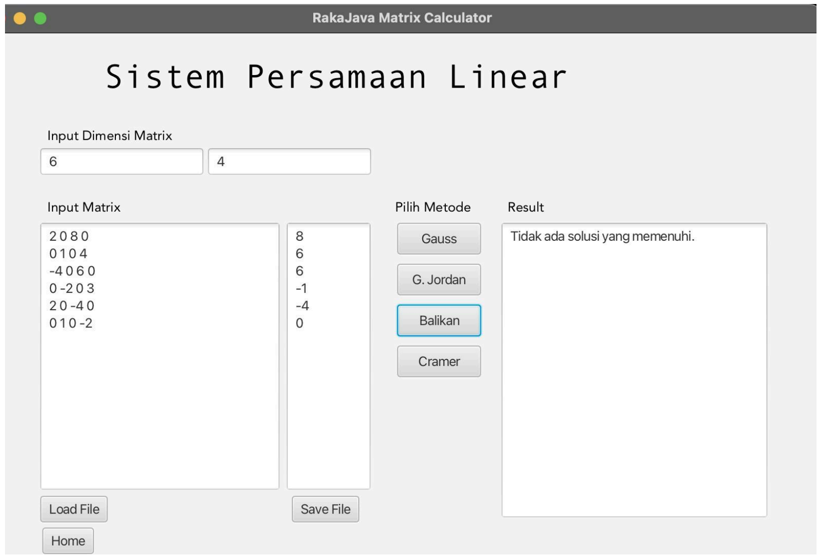
Gambar 4.2.2.7 Hasil Perhitungan Metode Gauss

- Metode Gauss - Jordan



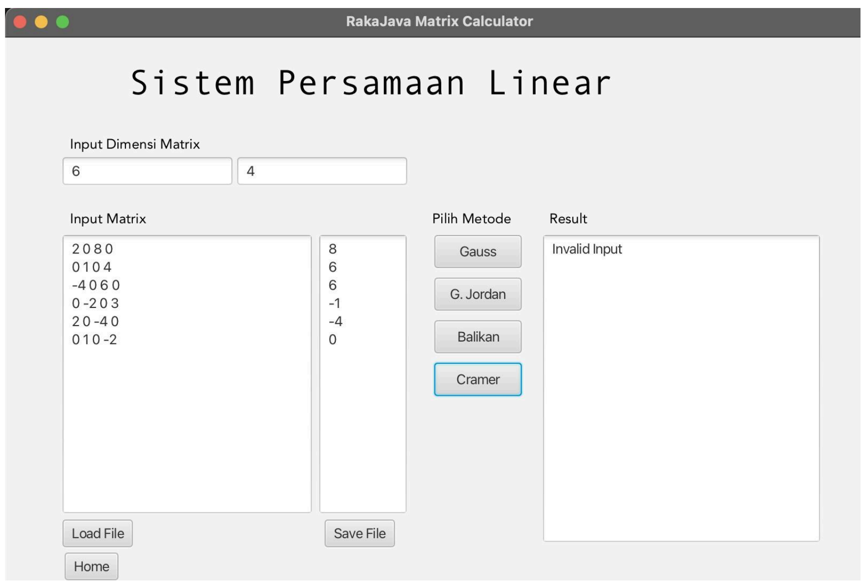
Gambar 4.2.2.8 Hasil Perhitungan Metode Gauss - Jordan

- Metode Invers



Gambar 4.2.2.9 Hasil Perhitungan Metode Invers

- Metode Cramer



Gambar 4.2.2.10 Hasil Perhitungan Metode Cramer

- Analisis

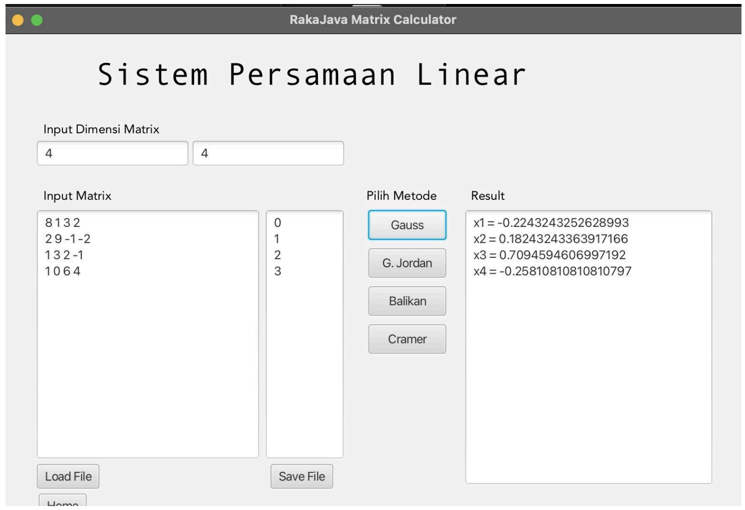
4.2.3 SPL Persamaan

a. Soal 1

$$
\begin{aligned}
8x_1 + x_2 + 3x_3 + 2x_4 &= 0 \\
2x_1 + 9x_2 - x_3 - 2x_4 &= 1 \\
x_1 + 3x_2 + 2x_3 - x_4 &= 2 \\
x_1 + 6x_3 + 4x_4 &= 3
\end{aligned}
$$

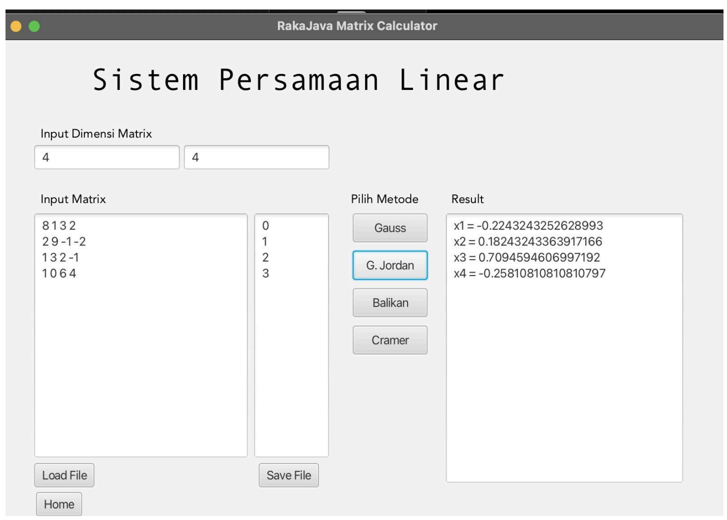
Gambar 4.2.3.1 Soal 1

- Metode Gauss



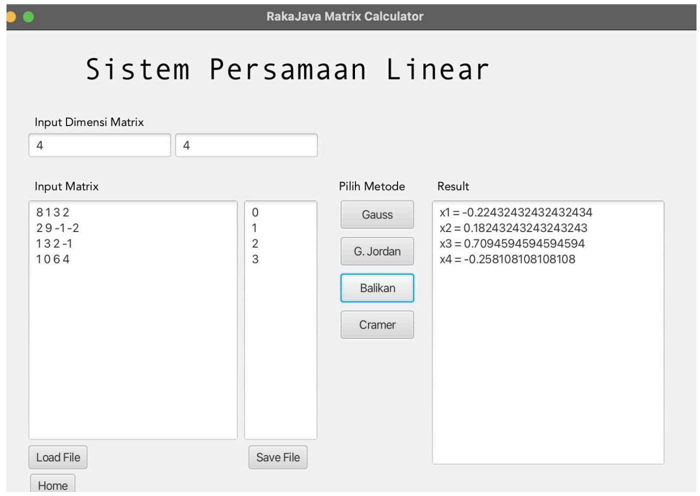
Gambar 4.2.3.2 Hasil Perhitungan Metode Gauss

- Metode Gauss - Jordan



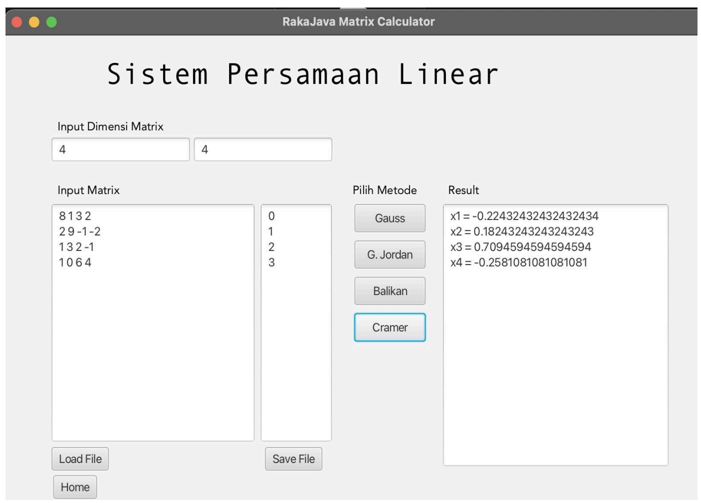
Gambar 4.2.3.3 Hasil Perhitungan Metode Gauss - Jordan

- Metode Invers



Gambar 4.2.3.4 Hasil Perhitungan Metode Invers

- Metode Cramer



Gambar 4.2.3.5 Hasil Perhitungan Metode Cramer

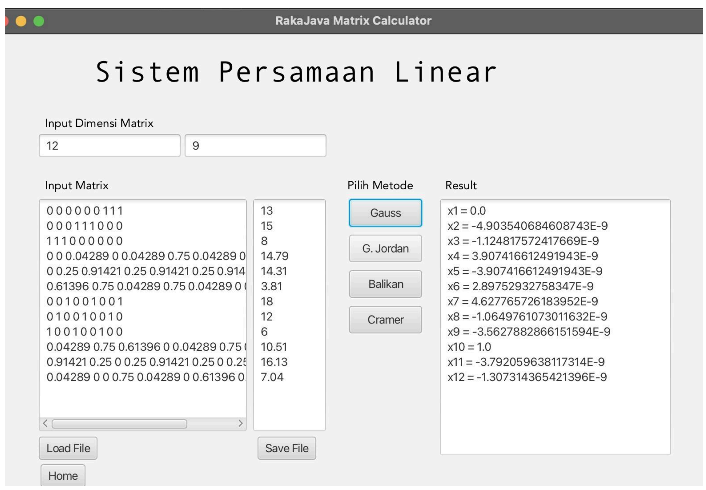
- Analisis
  Terdapat beberapa perbedaan desimal di akhir karena penggunaan *type* data double sangat presisi dan perubahan bit dapat mempengaruhi hasil akhir.

b. Soal 2

$$
\begin{aligned}
x_7 + x_8 + x_9 &= 13.00 \\
x_4 + x_5 + x_6 &= 15.00 \\
x_1 + x_2 + x_3 &= 8.00 \\
0.04289(x_3 + x_5 + x_7) + 0.75(x_6 + x_8) + 0.61396x_9 &= 14.79 \\
0.91421(x_3 + x_5 + x_7) + 0.25(x_2 + x_4 + x_6 + x_8) &= 14.31 \\
0.04289(x_3 + x_5 + x_7) + 0.75(x_2 + x_4) + 0.61396x_1 &= 3.81 \\
x_3 + x_6 + x_9 &= 18.00 \\
x_2 + x_5 + x_8 &= 12.00 \\
x_1 + x_4 + x_7 &= 6.00 \\
0.04289(x_1 + x_5 + x_9) + 0.75(x_2 + x_6) + 0.61396x_3 &= 10.51 \\
0.91421(x_1 + x_5 + x_9) + 0.25(x_2 + x_4 + x_6 + x_8) &= 16.13 \\
0.04289(x_1 + x_5 + x_9) + 0.75(x_4 + x_8) + 0.61396x_7 &= 7.04
\end{aligned}
$$

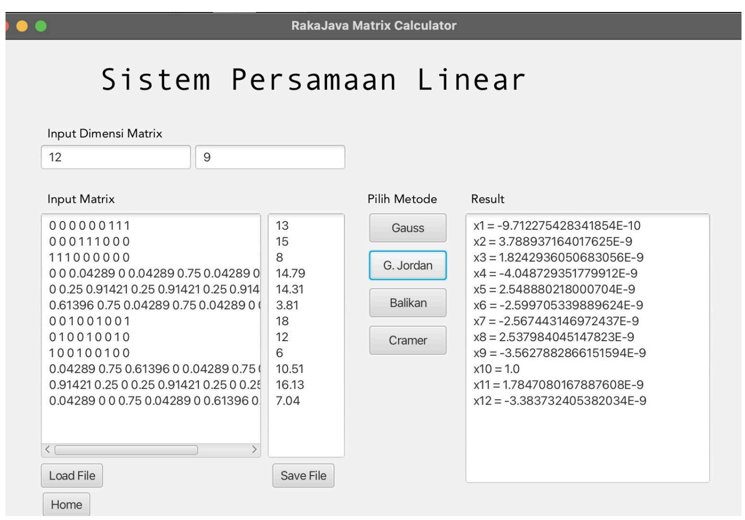
Gambar 4.2.3.6 Soal 2

- Metode Gauss



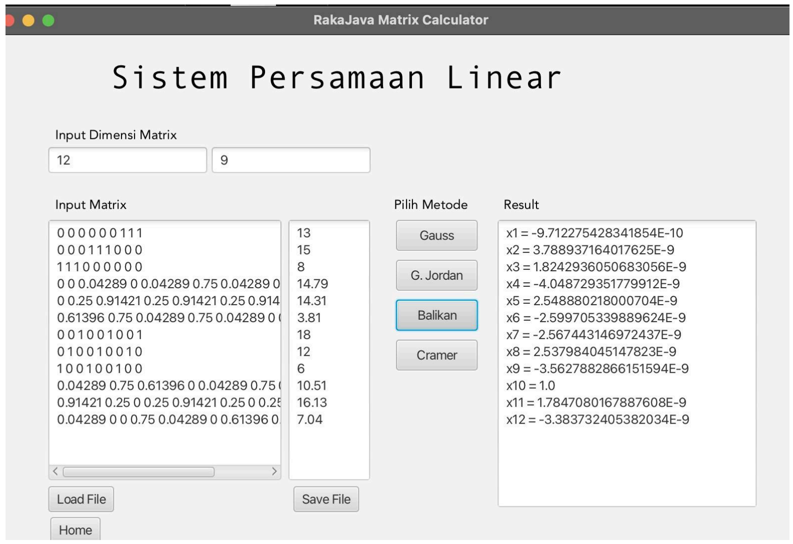
Gambar 4.2.3.7 Hasil Perhitungan Metode Gauss

- Metode Gauss - Jordan



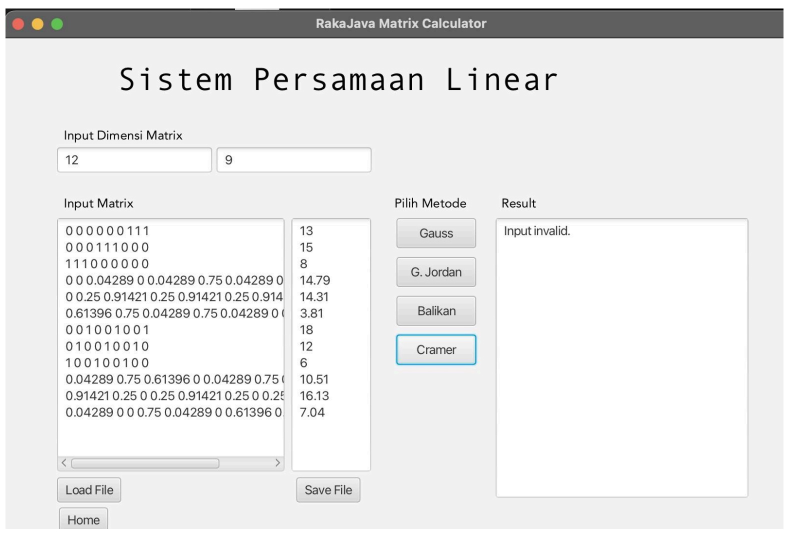
Gambar 4.2.3.8 Hasil Perhitungan Metode Gauss - Jordan

- Metode Invers



Gambar 4.2.3.9 Hasil Perhitungan Metode Invers

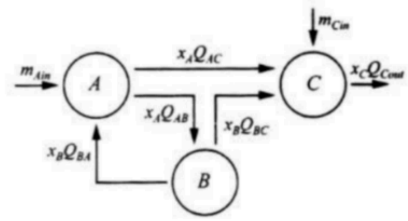
- Metode Cramer



Gambar 4.2.3.10 Hasil Perhitungan Metode Cramer

- Analisis
  Terdapat beberapa perbedaan desimal di akhir karena penggunaan *type* data double sangat presisi dan perubahan bit dapat mempengaruhi hasil akhir.

4.2.4 Sistem Reaktor



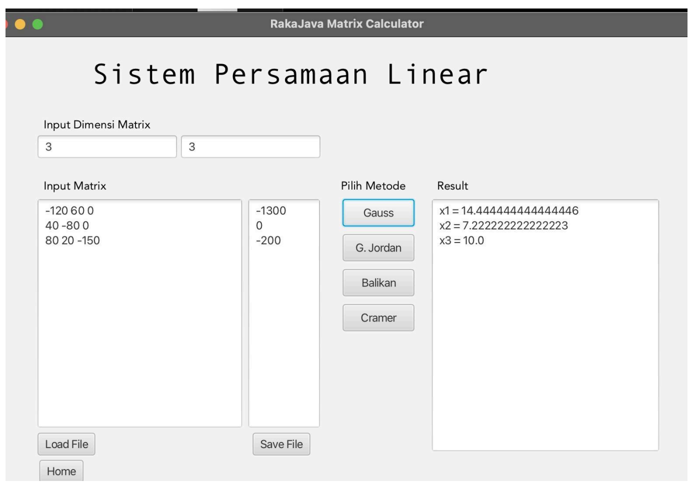
Dengan laju volume Q dalam m^3/s dan input massa min dalam mg/s. Konservasi massa pada tiap inti reaktor adalah sebagai berikut:

$$\begin{aligned}
&\text{A:} \quad m_{A_{in}} + Q_{BA}x_B - Q_{AB}x_A - Q_{AC}x_A = 0 \\
&\text{B:} \quad Q_{AB}x_A - Q_{BA}x_B - Q_{BC}x_B = 0 \\
&\text{C:} \quad m_{C_{in}} + Q_{AC}x_A + Q_{BC}x_B - Q_{C_{out}}x_C = 0
\end{aligned}$$

Tentukan solusi x_A, x_B, x_C dengan menggunakan parameter berikut : Q_{AB} = 40, Q_{AC} = 80, Q_{BA} = 60, Q_{BC} = 20 dan Q_{Cout} = 150 m^3/s dan m_{Ain} = 1300 dan m_{Cin} = 200 mg/s.

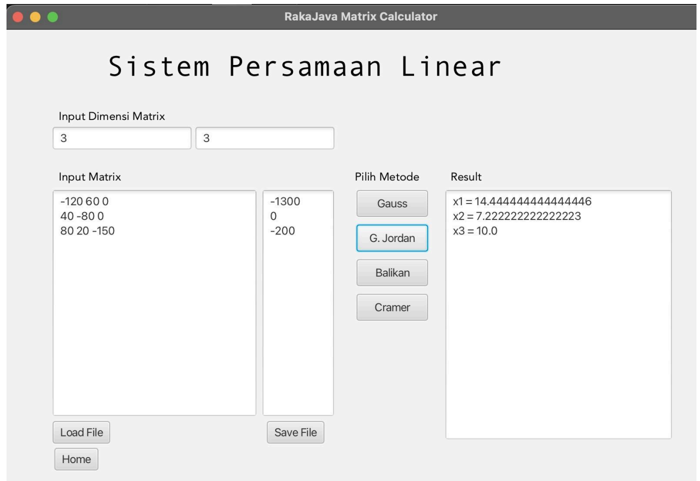
Gambar 4.2.4.1 Sistem Reaktor

- Metode Gauss



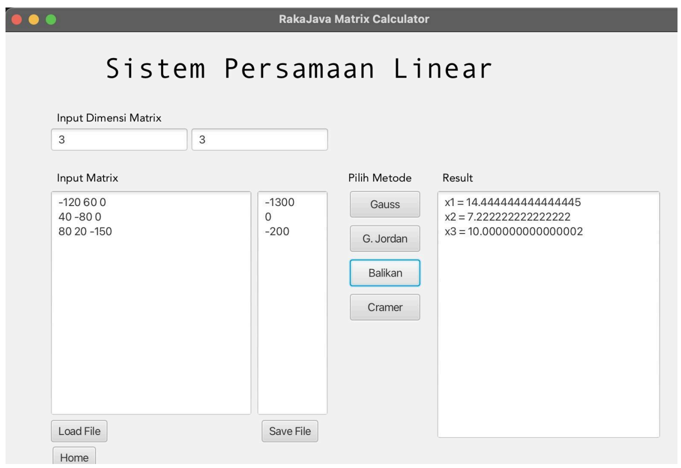
Gambar 4.2.4.2 Hasil Perhitungan Metode Gauss

- Metode Gauss - Jordan



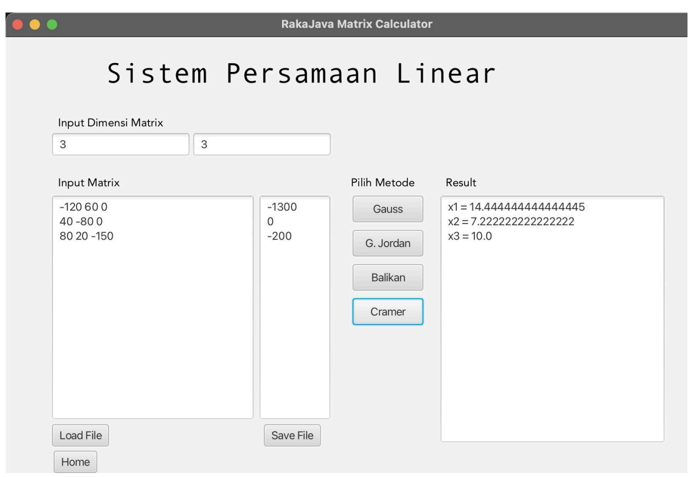
Gambar 4.2.4.3 Hasil Perhitungan Metode Gauss - Jordan

- Metode Invers



Gambar 4.2.4.4 Hasil Perhitungan Metode Invers

- Metode Cramer



Gambar 4.2.4.5 Hasil Perhitungan Metode Cramer

- Analisis
  Terdapat beberapa perbedaan desimal di akhir karena penggunaan *type* data double sangat presisi dan perubahan bit dapat mempengaruhi hasil akhir.

4.2.5 Studi Kasus Interpolasi

a. Soal 1

Gunakan tabel di bawah ini untuk mencari polinom interpolasi dari pasangan titik-titik yang terdapat dalam tabel. Program menerima masukan nilai *x* yang akan dicari nilai fungsi *f*(*x*).

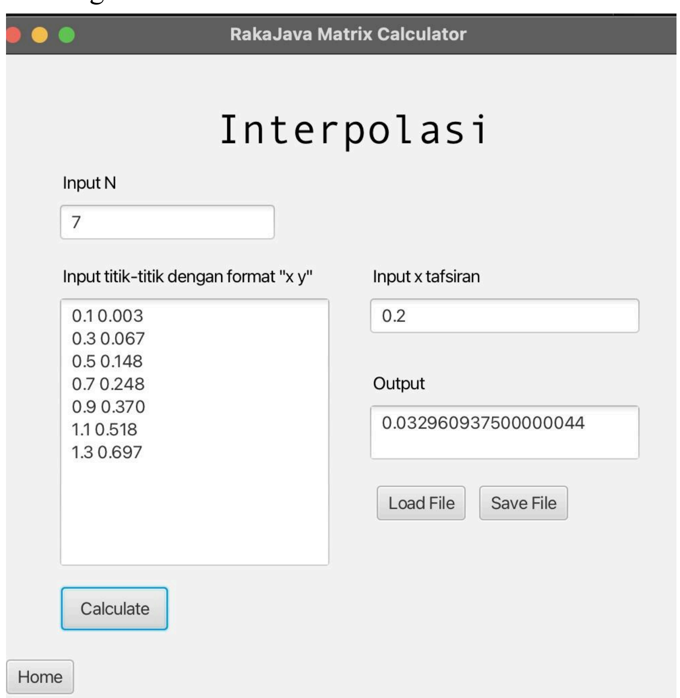
x	0.1	0.3	0.5	0.7	0.9	1.1	1.3
f(*x*)	0.003	0.067	0.148	0.248	0.370	0.518	0.697

Lakukan pengujian pada nilai-nilai berikut:

x = 0.2 *f*(*x*) = ?
x = 0.55 *f*(*x*) = ?
x = 0.85 *f*(*x*) = ?
x = 1.28 *f*(*x*) = ?

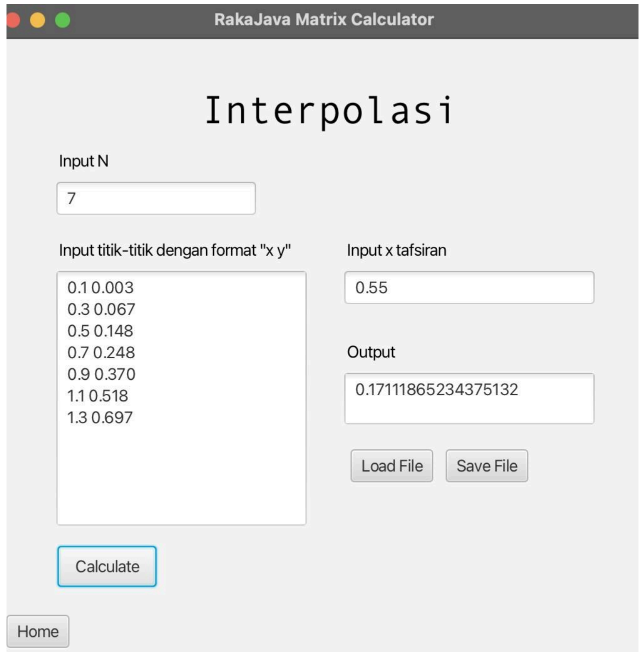
Gambar 4.2.5.1 Soal 1

- Hasil Perhitungan x = 0.2



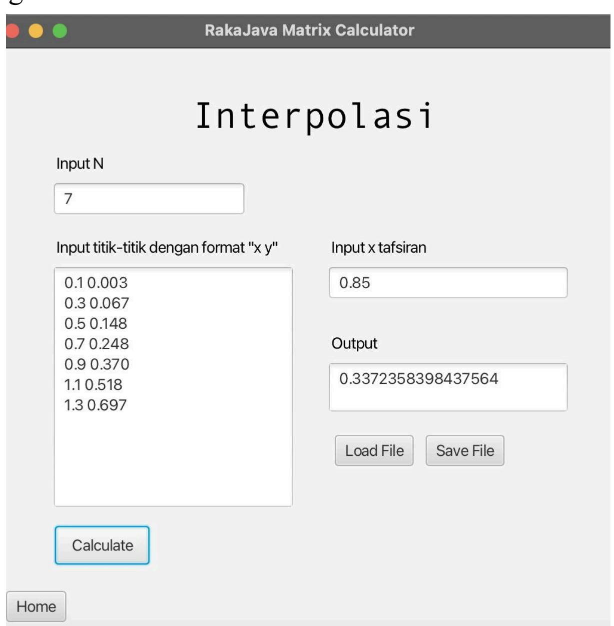
Gambar 4.2.5.2 Solusi Soal 1 x = 0.2

- Analisis
- Hasil Perhitungan x = 0.55



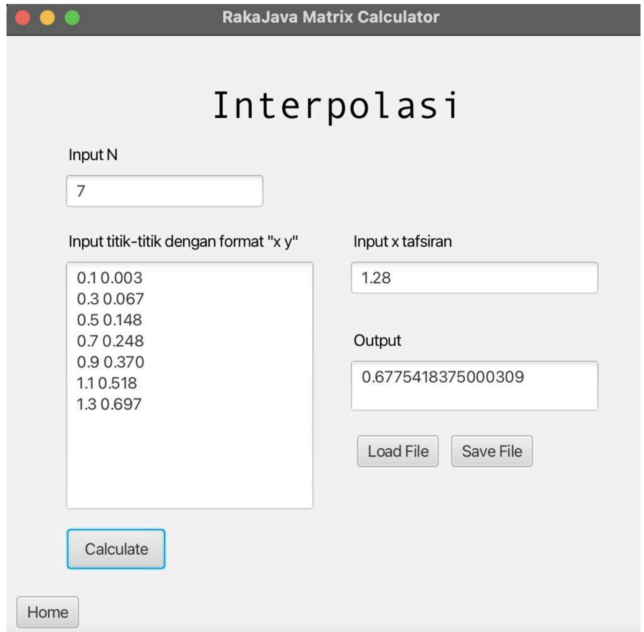
Gambar 4.2.5.4 Solusi Soal 1 x = 0.55

- Analisis
- Hasil Perhitungan x = 0.85



Gambar 4.2.5.6 Solusi Soal 1 x = 0.85

- Analisis
- Hasil Perhitungan x = 1.28



Gambar 4.2.5.8 Solusi Soal 1 x = 1.28

- Analisis

b. Soal 2

Jumlah kasus positif baru Covid-19 di Indonesia semakin fluktuatif dari hari ke hari. Di bawah ini diperlihatkan jumlah kasus baru Covid-19 di Indonesia mulai dari tanggal 17 Juni 2022 hingga 31 Agustus 2022:

Tanggal	Tanggal (desimal)	Jumlah Kasus Baru
17/06/2022	6,567	12.624
30/06/2022	7	21.807
08/07/2022	7,258	38.391
14/07/2022	7,451	54.517
17/07/2022	7,548	51.952
26/07/2022	7,839	28.228
05/08/2022	8,161	35.764
15/08/2022	8,484	20.813
22/08/2022	8,709	12.408
31/08/2022	9	10.534

Tanggal (desimal) adalah tanggal yang sudah diolah ke dalam bentuk desimal 3 angka di belakang koma dengan memanfaatkan perhitungan sebagai berikut:

Tanggal (desimal) = bulan + (tanggal / jumlah hari pada bulan tersebut)

Sebagai contoh, untuk tanggal 17/06/2022 (dibaca: 17 Juni 2022) diperoleh tanggal(desimal) sebagai berikut:

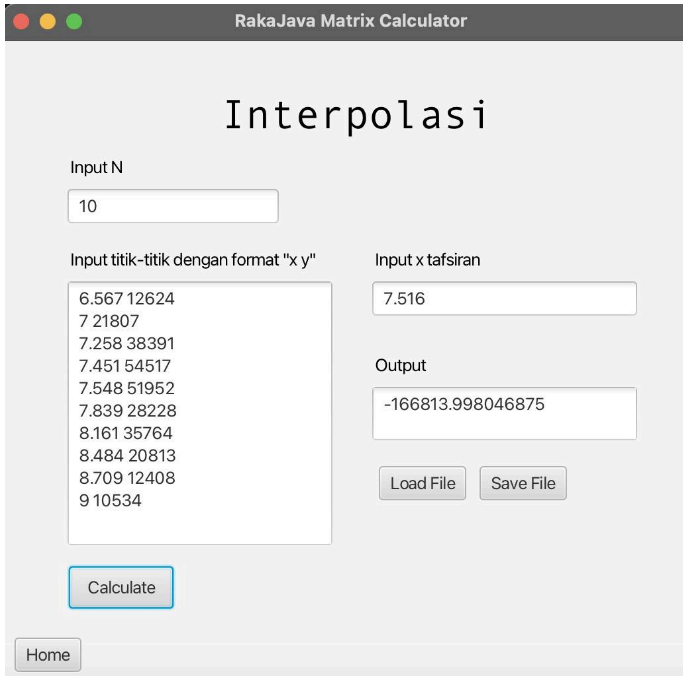
Tanggal (desimal) = 6 + (17/30) = 6,567

Gunakanlah data di atas dengan memanfaatkan **interpolasi polinomial** untuk melakukan prediksi jumlah kasus baru Covid-19 pada tanggal-tanggal berikut:

a. 16/07/2022
b. 10/08/2022
c. 05/09/2022
d. Masukan user lainnya berupa **tanggal (desimal) yang sudah diolah** dengan asumsi prediksi selalu dilakukan untuk tahun 2022.

Gambar 4.2.5.9 Soal 2

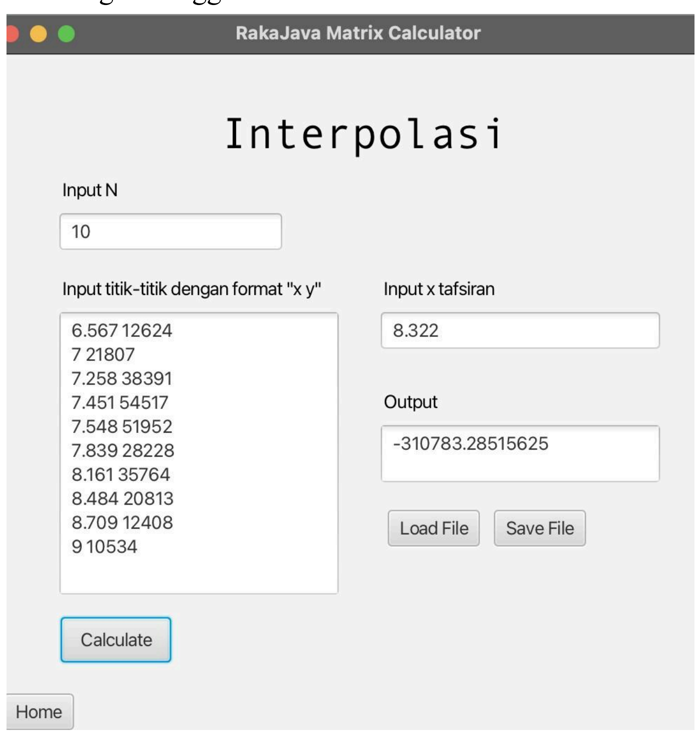
- Hasil Perhitungan Tanggal 16/07/2022



Gambar 4.2.5.10 Solusi Soal 2 untuk 16/07/2022

- Analisis

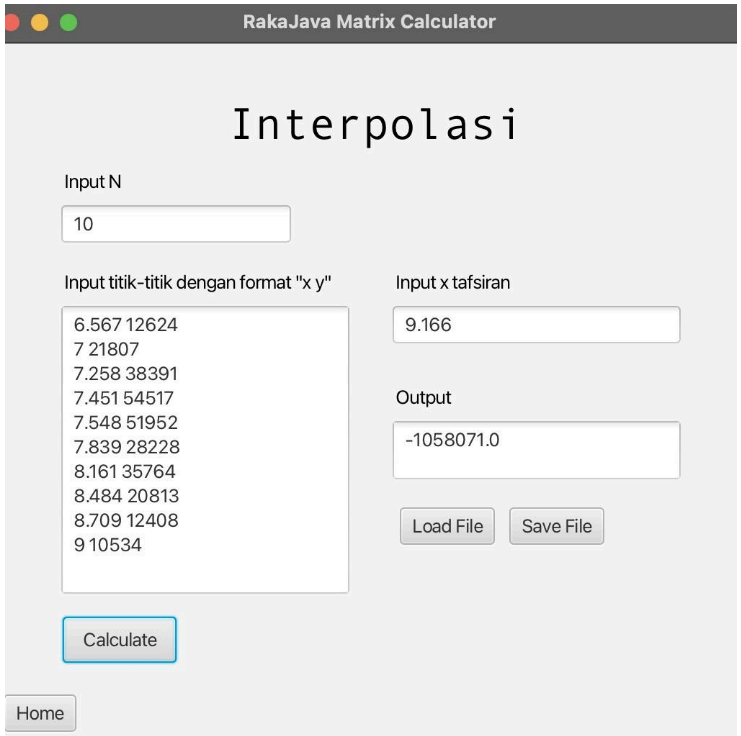
- Hasil Perhitungan Tanggal 10/08/2022



Gambar 4.2.5.11 Solusi Soal 2 untuk 10/08/2022

- Analisis

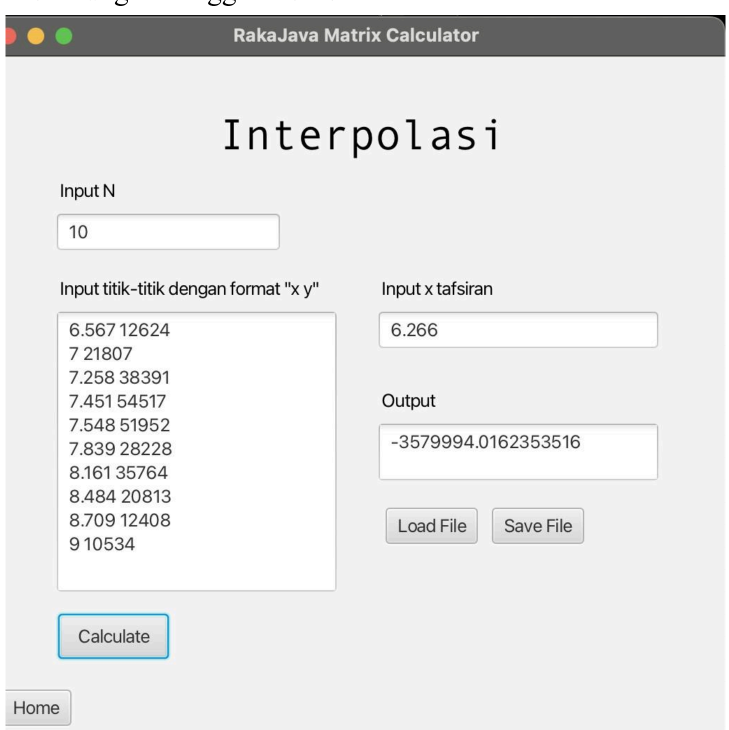
- Hasil Perhitungan Tanggal 05/09/2022



Gambar 4.2.5.12 Solusi Soal 2 untuk 05/09/2022

- Analisis

- Hasil Perhitungan Tanggal 08/06/2022



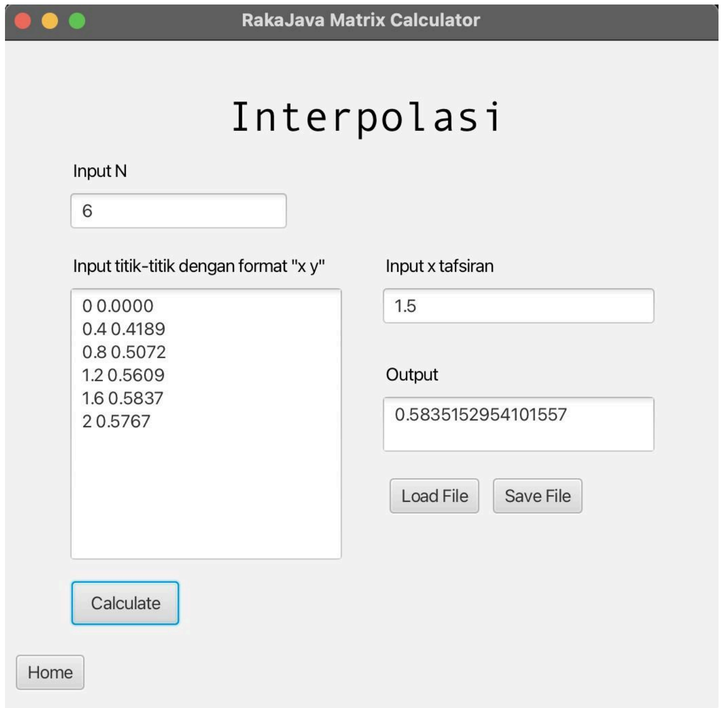
Gambar 4.2.5.13 Solusi soal 2 untuk 06/08/2022

- Analisis

c. Soal 3

Gambar 4.2.5.14 Soal 3

- Solusi



Gambar 4.2.5.14 Solusi Soal 3

- Analisis

4.2.6 Studi Kasus Regresi Linear dan Kuadratik Berganda

Diberikan sekumpulan data sesuai pada tabel berikut ini.

Table 12.1: Data for Example 12.1

Nitrous Oxide, y	Humidity, x_1	Temp., x_2	Pressure, x_3	Nitrous Oxide, y	Humidity, x_1	Temp., x_2	Pressure, x_3
0.90	72.4	76.3	29.18	1.07	23.2	76.8	29.38
0.91	41.6	70.3	29.35	0.94	47.4	86.6	29.35
0.96	34.3	77.1	29.24	1.10	31.5	76.9	29.63
0.89	35.1	68.0	29.27	1.10	10.6	86.3	29.56
1.00	10.7	79.0	29.78	1.10	11.2	86.0	29.48
1.10	12.9	67.4	29.39	0.91	73.3	76.3	29.40
1.15	8.3	66.8	29.69	0.87	75.4	77.9	29.28
1.03	20.1	76.9	29.48	0.78	96.6	78.7	29.29
0.77	72.2	77.7	29.09	0.82	107.4	86.8	29.03
1.07	24.0	67.7	29.60	0.95	54.9	70.9	29.37

Source: Charles T. Hare, "Light-Duty Diesel Emission Correction Factors for Ambient Conditions," EPA-600/2-77-116. U.S. Environmental Protection Agency.

Gunakan *Normal Estimation Equation for Multiple Linear Regression* untuk mendapatkan regresi linear berganda dari data pada tabel di atas, kemudian estimasi nilai Nitrous Oxide apabila Humidity bernilai 50%, temperatur 76°F, dan tekanan udara sebesar 29.30.

Dari data-data tersebut, apabila diterapkan *Normal Estimation Equation for Multiple Linear Regression*, maka diperoleh sistem persamaan linear sebagai berikut.

$20b_0 + 863.1b_1 + 1530.4b_2 + 587.84b_3 = 19.42$

$863.1b_0 + 54876.89b_1 + 67000.09b_2 + 25283.395b_3 = 779.477$

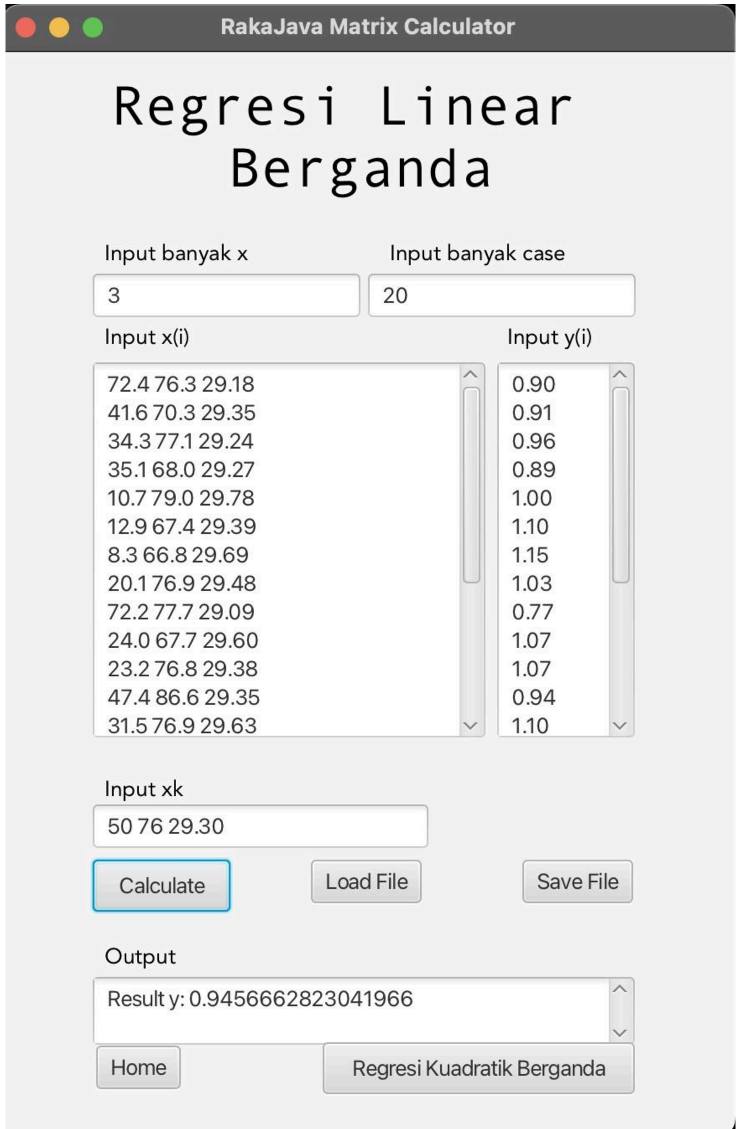
$1530.4b_0 + 67000.09b_1 + 117912.32b_2 + 44976.867b_3 = 1483.437$

$587.84b_0 + 25283.395b_1 + 44976.867b_2 + 17278.5086b_3 = 571.1219$

Silahkan terapkan model-model ini pada *Multiple Quadratic Equation* juga dan bandingkan hasilnya. Sistem persamaan linear tidak akan diberikan untuk kasus ini.

Gambar 4.2.6.1 Soal 1

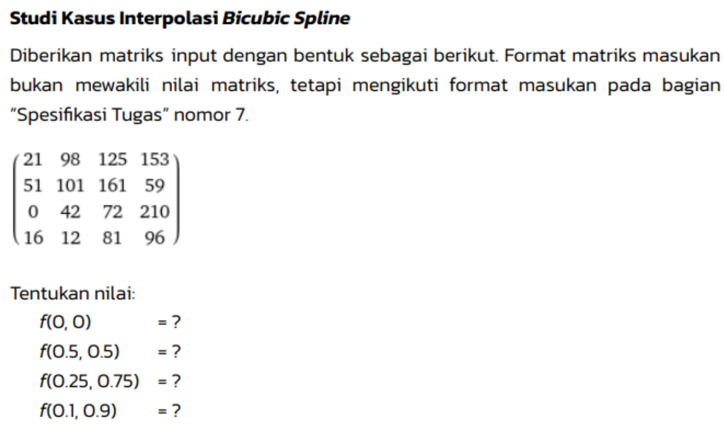
- Solusi



Gambar 4.2.6.2 Solusi Soal 1 Regresi Linear Berganda

- Analisis

4.2.7 Studi Kasus Interpolasi Bicubic Spline



Studi Kasus Interpolasi *Bicubic Spline*

Diberikan matriks input dengan bentuk sebagai berikut. Format matriks masukan bukan mewakili nilai matriks, tetapi mengikuti format masukan pada bagian "Spesifikasi Tugas" nomor 7.

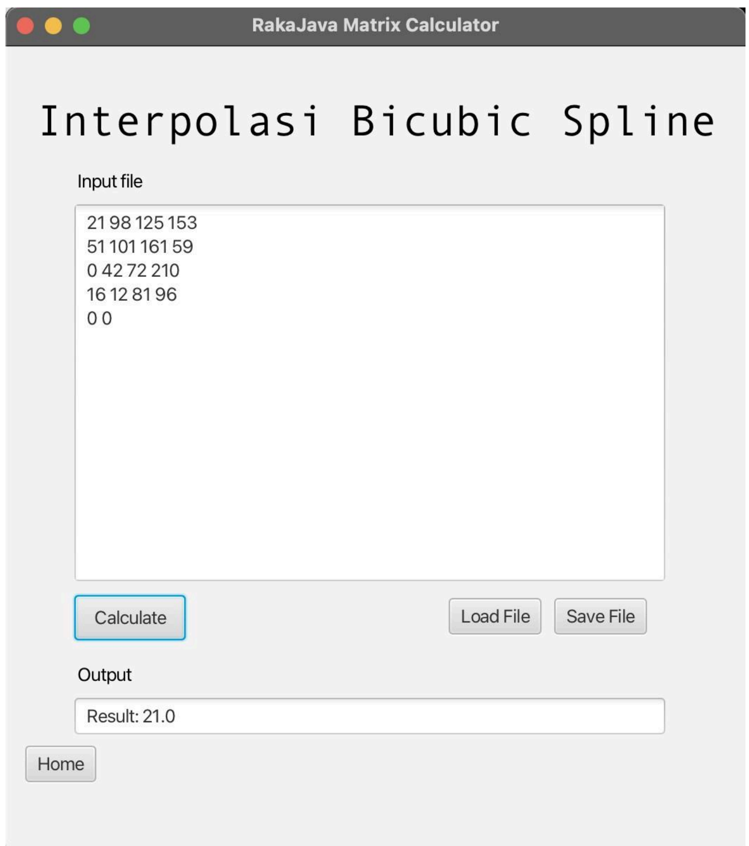
$$\begin{pmatrix} 21 & 98 & 125 & 153 \\ 51 & 101 & 161 & 59 \\ 0 & 42 & 72 & 210 \\ 16 & 12 & 81 & 96 \end{pmatrix}$$

Tentukan nilai:

f(0, 0) = ?
f(0.5, 0.5) = ?
f(0.25, 0.75) = ?
f(0.1, 0.9) = ?

Gambar 4.2.7.1 Soal 1

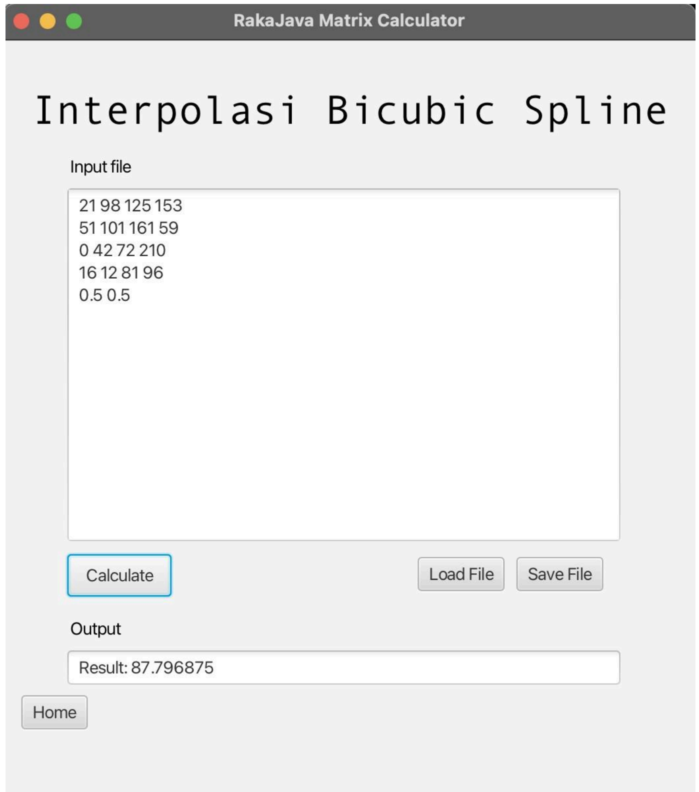
- Solusi f(0, 0)



Gambar 4.2.7.2 Hasil Perhitungan f(0, 0)

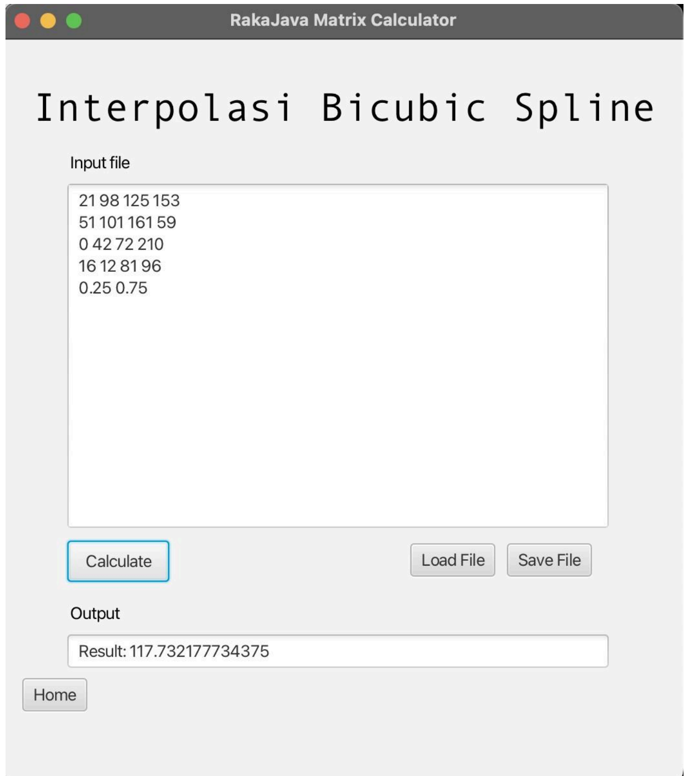
- Analisis

- Solusi f(0.5, 0.5)



Gambar 4.2.7.3 Hasil Perhitungan f(0.5, 0.5)

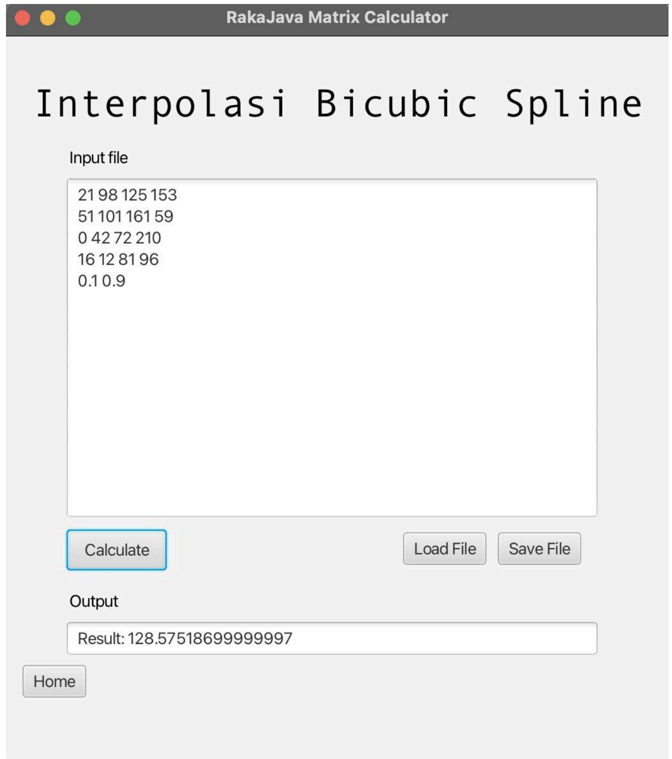
- Analisis
- Solusi f(0.25, 0.75)



Gambar 4.2.7.3 Hasil Perhitungan f(0.25, 0.75)

- Analisis

- Solusi f(0.1, 0.9)



Gambar 4.2.7.3 Hasil Perhitungan f(0.25, 0.75)

- Analisis

BAB 5
KESIMPULAN

5.1 Kesimpulan

Program yang telah kami perjuangkan selama beberapa minggu ke belakang sudah sangat memuaskan kami sebagai pembuat. Program kami telah berhasil mengimplementasikan berbagai jenis fungsi yang dibuat dengan baik. Berbagai jenis masukan matriks sudah kami coba dan mendapatkan hasil yang baik.

Kami harap, program ini dapat berguna baik untuk kami ataupun siapapun yang membaca dokumen ini nanti. Tolong manfaatkan dengan baik apapun yang ada pada dokumen ini.

5.2 Saran

Melihat bagaimana kelompok kami telah melewati pengerjaan tugas besar dengan penuh dinamika, salah satu hal yang membuat kelompok kami kesulitan di awal adalah tentang kurangnya batasan, dan juga arahan terkait pengerjaan. Sehingga melalui pengalaman, kami merasa bahwa spesifikasi dari tugas besar perlu dirincikan lebih lanjut, dengan struktur yang lebih jelas kedepannya, sehingga tidak menimbulkan misinformasi dan kebingungan lebih lanjut. Lebih baik apabila penyusunan laporan didasarkan pada kelas masing masing sehingga informasi dan spesifikasi tidak tersebar.

5.3 Komentar

Meski hati tersakiti, tubes juga sakit.
-Maheswara Bayu Kaindra-

Siapapun kalian, tubes saya masih sama.
-Peter Wongsoredjo-

Terima kasih tubes, sekarang saya tidak mempan minum kopi.
-Raka Daffa Iftikhaar-

5.4 Refleksi

Melalui adanya tugas besar ini, kami mendapatkan banyak sekali pengetahuan dan juga pemahaman. Dengan jadwal yang begitu terbagi bagi dengan tugas lainnya, kami menyadari bahwa penundaan tugas menjadi sesuatu yang seharusnya kami hindari. Melalui tugas besar ini juga, kamu juga memahami pentingnya suatu GUI untuk membantu dalam penyelarasan metode dan fungsi yang kami miliki. Dengan adanya tugas besar ini juga kami juga mendapatkan pemahaman materi yang lebih mendalam.

LAMPIRAN

Link Repository: https://github.com/PeterWongsoredjo/Algeo01-23015

Link Video: https://youtu.be/pGV5bbFkPnI