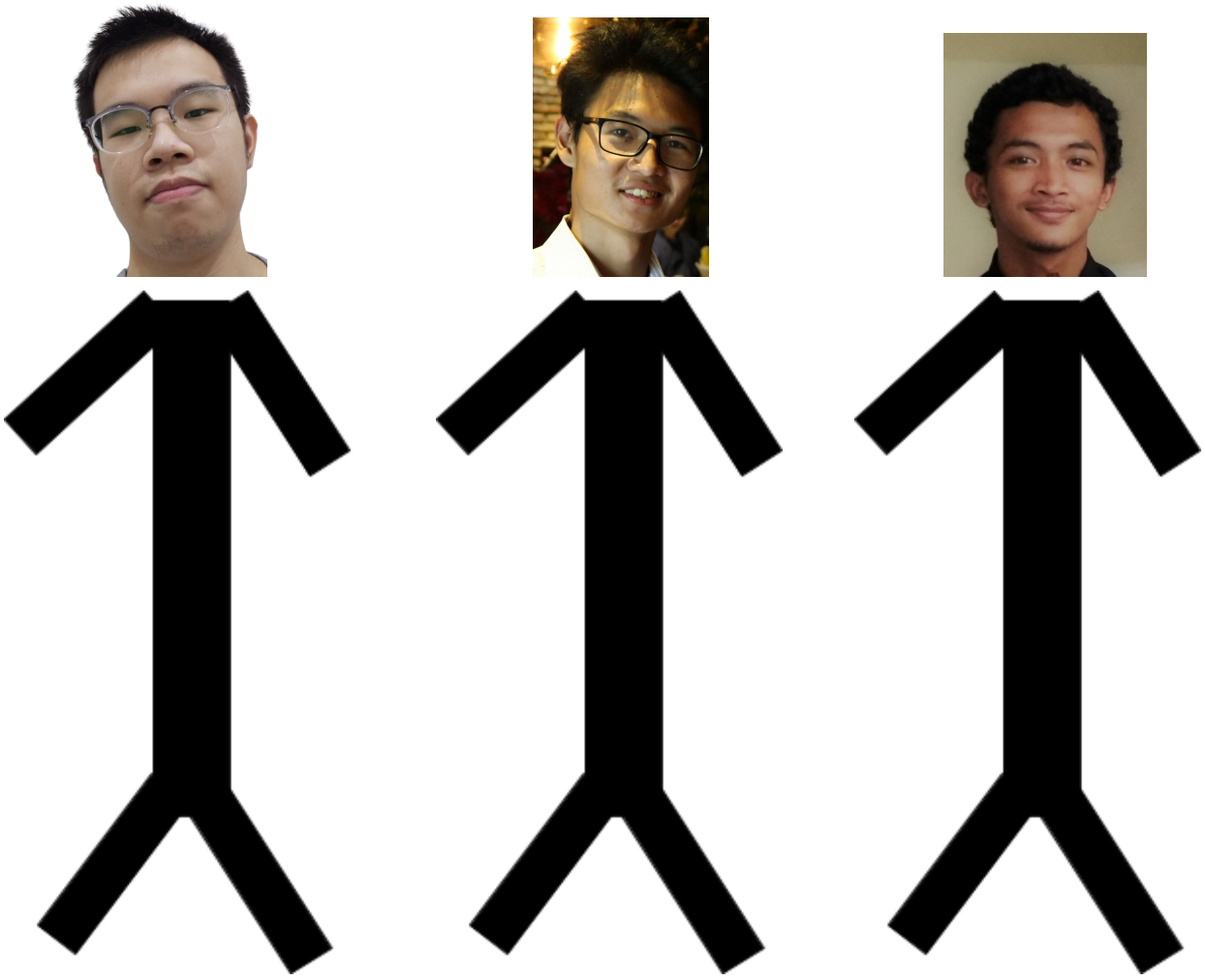
Laporan Tugas Besar 1 IF 2123 Aljabar Linier dan Geometri
Sistem Persamaan Linier, Determinan, dan Aplikasinya
Kelompok 54



(Nama - NIM Anggota)
Girvin Junod - 13519096
Jonathan Christhoper Jahja - 13519144
Jauhar Wibisono - 13519160

Semester 1 Tahun 2020/2021

BAB 1: DESKRIPSI MASALAH

Berikut adalah spesifikasi tugas yang dijelaskan dalam laporan ini.

SPESIFIKASI TUGAS

Buatlah program dalam **Bahasa Java** untuk

1. Menghitung solusi SPL dengan metode eliminasi Gauss, metode Eliminasi Gauss-Jordan, metode matriks balikan, dan kaidah Cramer (kaidah Cramer khusus untuk SPL dengan n peubah dan n persamaan).
2. Menyelesaikan persoalan interpolasi dan regresi linier.
3. Menghitung matriks balikan
4. Menghitung determinan matriks dengan berbagai metode (reduksi baris dan ekspansi kofaktor).

Dibuat program sesuai dengan spesifikasi tugas tersebut.

BAB 2: TEORI SINGKAT

Eliminasi Gauss

Eliminasi gauss ditemukan oleh Carl Friedrich Gauss**,** metode ini dapat dimanfaatkan untuk memecahkan sistem persamaan linear dengan merepresentasikan (mengubah) menjadi bentuk matriks, matriks tersebut lalu diubah menjadi Eselon Baris melalui Operasi Baris Elementer. Kemudian sistem diselesaikan dengan substitusi balik.

Matriks Eselon Baris adalah matriks yang memenuhi kriteria-kriteria berikut :

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

1. Jika ada elemen matriks bukan 0, maka pada setiap baris bilangan bukan 0 pertama harus 1. Elemen 1 ini biasanya disebut 1 utama.

2. Jika ada 2 baris yang memenuhi kriteria pertama, maka baris yang lebih rendah harus memiliki posisi 1 utama lebih kanan daripada baris atasnya.
3. Jika suatu baris memiliki elemen 0 semua, maka baris tersebut harus ditukar dengan baris paling bawah.

Untuk membentuk matriks Eselon Baris dapat menggunakan metode OBE. OBE adalah metode pengoperasian setiap baris matriks dengan pembagian satu baris dan pengurangan baris lainnya. Kedua operasi ini dilakukan secara terus menerus pada setiap barisnya sampai mendapatkan matriks Eselon Baris.

Eliminasi Gauss dapat dimanfaatkan untuk mendapatkan solusi dari Sistem Persamaan Linear. Ketika mendapatkan matriks Eselon Baris perlu dilakukan substitusi mundur untuk mendapatkan solusi tiap variabelnya

Eliminasi Gauss-Jordan

Eliminasi Gauss-Jordan adalah pengembangan dari metode Eliminasi Gauss. Jika matriks yang dihasilkan Eliminasi Gauss adalah matriks Eselon Baris, Eliminasi Gauss-Jordan menghasilkan matriks Eselon Baris Tereduksi. Matriks Eselon Baris Tereduksi adalah matriks Eselon Baris tapi setiap elemen yang berada pada kolom 1 utama harus bernilai 0 semua kecuali 1 utama tersebut. Sehingga jika Eliminasi Gauss-Jordan diterapkan pada matriks yang jumlah baris dan kolomnya sama, akan menghasilkan matriks identitas.

$$I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad I_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Matriks Identitas

Eliminasi Gauss-Jordan melalui proses OBE yang sama pada Eliminasi Gauss, perbedaannya ketika mendapatkan matriks Eselon Baris, dilakukan proses OBE kembali untuk setiap elemen diatas 1 utama.

Metode eliminasi ini juga dapat dimanfaatkan untuk mendapatkan solusi SPL. Metode ini tidak memerlukan substitusi mundur.

Determinan Matriks

Determinan matriks adalah nilai skalar dari suatu matriks persegi. Menurut definisi ini, maka matriks yang ukurannya bukan n x n tidak memiliki determinan. Dalam laporan ini, akan dibahas 2 cara menghitung determinan matriks yaitu dengan ekspansi laplace atau ekspansi kofaktor dan dengan OBE. Dengan Metode OBE, determinan dapat dihitung dengan mengubah elemen2 dibawah diagonal utama menjadi 0 dan menghitung perkalian semua elemen diagonal utama. Dengan metode ekspansi kofaktor, determinan dapat dihitung dengan menghitung jumlah perkalian tiap elemen dalam suatu baris atau kolom dengan determinan matriks kofaktornya. Determinan memiliki banyak fungsi, contohnya adalah untuk menghitung matriks balikan dan menghitung solusi SPL dengan Kaidah Cramer.

Matriks Balikan

Matriks balikan atau invers matriks adalah suatu matriks yang jika dikalikan dengan matriks asalnya akan menghasilkan matriks identitas sama seperti pada aljabar biasa di mana bilangan X memiliki invers X^-1 = 1/X sehingga jika X . X^-1 = 1. Untuk mendapatkan matriks invers ini, dalam laporan ini dibahas dua cara yaitu dengan rumus M^-1 = 1/determinan(M) * adjoin(M) dengan adjoin merupakan transpose dari matriks Kofaktor M atau dengan eliminasi Gauss. Dapat dilihat dari rumusnya yang dibagi dengan determinan matriks maka matriks hanya memiliki matriks balikan jika determinan dari matriks tersebut tidak sama dengan nol dan

merupakan matriks persegi. Metode eliminasi Gauss digunakan untuk menemukan matriks balikan dengan mengubah matriks tersebut menjadi matriks identitas dan melakukan perubahan yang sama terhadap matriks identitas sehingga matriks identitas tersebut berubah menjadi matriks balikan.

Matriks balikan ini dapat digunakan untuk menyelesaikan suatu sistem persamaan linier. Namun, metode matriks balikan ini hanya dapat digunakan jika SPL dapat diubah menjadi matriks persegi dan memiliki determinan yang tidak 0. Misalkan dimiliki SPL yang diubah menjadi bentuk matriks AX = B. Maka jika kedua ruas dikalikan dengan matriks balikan A didapat A^-1.A.X = A^-1.B yang menjadi X = A^-1.B karena matriks A dikali dengan matriks balikannya akan menjadi matriks identitas yang jika dikali dengan matriks X hasilnya akan sama dengan matriks X. Dari ini didapat solusi SPL dengan menggunakan matriks balikan.

Matriks Kofaktor

Matriks kofaktor adalah suatu matriks yang terdiri dari kofaktor matriks. Nilai kofaktor sebuah elemen adalah nilai determinan dari elemen2 matriks yang tidak sebaris dan tidak sekolom dengan elemen tersebut. Nilai kofaktor ini digabungkan dengan + atau - tergantung dengan letak elemennya pada matriks yang mengikuti gambar berikut.

$$\begin{vmatrix} + & - & + & \dots \\ - & + & - & \dots \\ + & - & + & \dots \\ \dots & \dots & \dots & \dots \end{vmatrix}$$

Untuk contoh diberikan matriks M sebagai berikut $\begin{bmatrix} 1 & 4 & 7 \\ 3 & 0 & 5 \\ -1 & 9 & 11 \end{bmatrix}$.

Maka nilai kofaktor M2,3 adalah $\det \begin{bmatrix} 1 & 4 & \square \\ \square & \square & \square \\ -1 & 9 & \square \end{bmatrix}$ dikali dengan -1 karena M2,3 ada di tanda (-). Maka didapat kofaktor M2,3 = (-1) x (36 +4) = (-40). Matriks kofaktor M adalah matriks yang terdiri dari kofaktor tiap elemen matriks M.

Matriks kofaktor ini dipakai dalam perhitungan determinan matriks melalui ekspansi Laplace atau ekspansi kofaktor. Ekspansi kofaktor ini adalah metode perhitungan determinan matriks dengan memilih suatu baris atau kolom dalam matriks, mengalikan tiap elemen dalam baris atau kolom tersebut dengan determinan kofaktornya, dan menjumlahkan setiap nilai yang didapat dari perkalian elemen dengan determinan kofaktornya.

Matriks Adjoin

Matriks adjoin adalah transpose dari sebuah matriks kofaktor. Matriks adjoin ini dipakai dalam perhitungan matriks balikan suatu matriks dengan rumus M^-1 = 1/determinan(M) * adjoin(M).

Kaidah Cramer

Kaidah Cramer adalah suatu formula penyelesaian SPL yang dinamakan dari nama Gabriel Cramer yaitu orang yang mempublikasikan metode ini. Kaidah Cramer ini dapat menyelesaikan suatu SPL selama SPL itu memiliki solusi unik. Misal dimiliki SPL dalam bentuk matriks AX = B dengan ukuran matriks A n x n, matriks B,X n x 1. Maka menurut kaidah cramer nilai x1 sama dengan determinan matriks A yang kolom 1-nya diganti dengan matriks B dibagi dengan determinan matriks A. Ini berlaku hingga Xn yang nilainya sama dengan determinan matriks A dengan kolom ke n-nya diganti dengan matriks B dibagi dengan determinan matriks A. Jika dituliskan dalam bentuk formula maka akan berbentuk Xn = det(An)/det(A). Kaidah Cramer sebenarnya dapat diterapkan untuk matriks B,X ukuran n x m namun pada program ini diasumsikan input B selalu 1 kolom.

Interpolasi Polinom

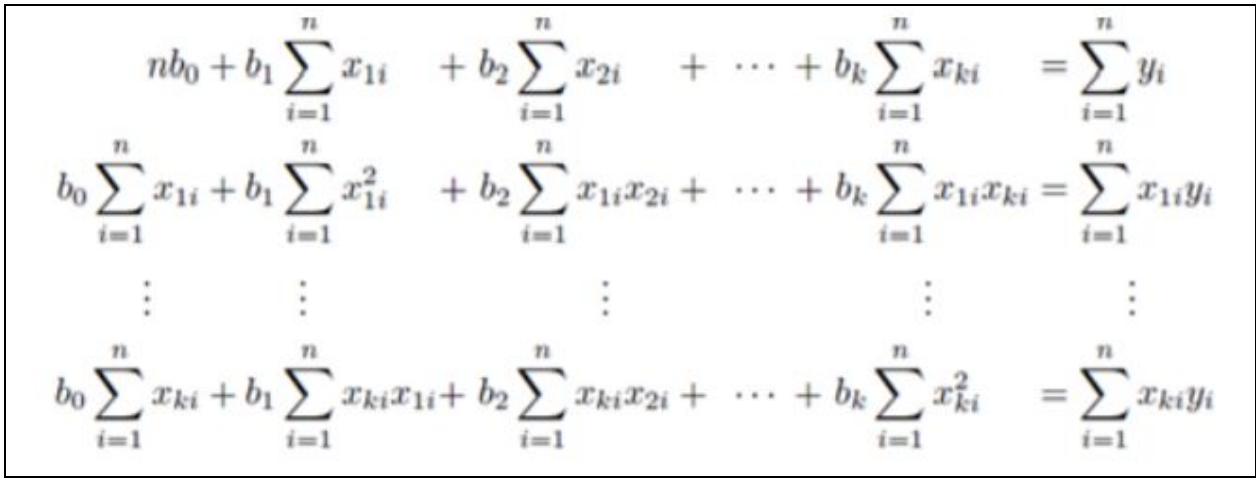
Interpolasi polinom adalah teknik untuk mengira-ngira suatu fungsi dengan satu peubah apabila diketahui titik-titik yang dilewatinya. Secara lebih formal, apabila diketahui n+1 buah titik berbeda yang dilewati oleh fungsi f(x): (x0,y0), (x1,y1) … (xn,yn); interpolasi polinom dapat digunakan untuk membangkitkan polinom berderajat n p(x) sedemikian sehingga p(xi)=yi untuk i anggota [0..n+1]. Dengan polinom yang dibangkitkan ini, nilai f(x) untuk x anggota [x0..xn] dapat diperkirakan.

Cara membangkitkan polinom p(x) adalah sebagai berikut. Pertama, misalkan p(x)=a0+a1x+...+an+1x^(n). Lalu, untuk setiap titik (xi,yi) yang ada, sulihkan x dengan xi dan p(x) dengan yi. Dengan demikian, terbentuk n+1 persamaan dengan n+1 variabel yang tidak diketahui nilainya: a0..an+1. Nilai a0..an+1 dapat dicari dengan menyelesaikan sistem persamaan tersebut. Terakhir, sulihkan nilai a0..an+1 ke p(x). Dengan demikian, polinom p(x) berhasil dibangkitkan. Untuk memperkirakan nilai f(x) untuk suatu x anggota [x0..xn], sulihkan x ke p(x), nilai yang didapatkan adalah perkiraan nilai f(x).

Regresi Linier Berganda

Regresi linier berganda adalah teknik untuk membangkitkan suatu persamaan linier yang dapat digunakan untuk menaksir nilai fungsi dengan banyak peubah. Untuk membangkitkan persamaan linier tersebut, diperlukan setidaknya satu tuple (x1,x2,...,xm,y).

Cara membangkitkan persamaan linier tersebut adalah sebagai berikut. Pertama, buat sistem persamaan berikut dengan memanfaatkan tuple-tuple yang dimiliki.



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

Pada sistem persamaan di atas, k menyatakan banyak nilai x pada tuple, n menyatakan banyak tuple, xij menyatakan nilai x ke-i tuple ke-j, yi menyatakan nilai y tuple ke-i, dan b0..bk menyatakan koefisien yang akan dicari nilainya. Lalu, dapatkan nilai b0..bk dengan menyelesaikan sistem persamaan tersebut. Terakhir, bangkitkan sebuah persamaan linier y=b0+b1x1+...+bkxk dengan nilai b0..bk yang telah didapatkan. Untuk memperkirakan nilai fungsi untuk suatu tuple (x01,x02,...,x0k,y0), sulihkan tuple tersebut ke persamaan yang telah dibangkitkan, nilai yang didapatkan adalah perkiraan nilai fungsi.

BAB 3: IMPLEMENTASI PROGRAM

Tabel Atribut tiap Class

Class	Atribut	Tipe Data	Deskripsi
InputMatrix	nBrs	int	Jumlah baris matriks yang di-input
	nKol	int	Jumlah kolom matriks yang di-input
SPL	nBrs	int	Jumlah baris Maug
	nKol	int	Jumlah kolom Maug
	Maug	double[][]	Matriks augmented SPL
	variable	String[]	Array variabel SPL
splCramer	nBrs	int	Jumlah baris Maug
	nKol	int	Jumlah kolom Maug
	N	int	Jumlah baris Maug
	Maug	double[][]	Matriks augmented SPL, selalu memiliki N baris dan N+1 kolom
	A	double[][]	Matriks koefisien variabel
	B	double[][]	Matriks hasil persamaan
	variable	String[]	Array variabel SPL
splinverse	nBrs	int	Jumlah baris Maug
	nKol	int	Jumlah kolom Maug
	N	int	Jumlah baris Maug
	Maug	double[][]	Matriks augmented SPL, selalu memiliki N baris dan N+1 kolom
	A	double[][]	Matriks koefisien variabel

	B	double[][]	Matriks hasil persamaan
	variable	String[]	Array variabel SPL

Tabel Method tiap class

Class	Method	Return Type	Parameter	Deskripsi
menu	main	void	String args[]	Main method atau method utama dari program
SPL	input	void	void	Input khusus matriks SPL dalam bentuk file maupun keyboard dari command line. Jumlah persamaan dan variabel dapat berbeda.
	output	void	String Var[]	Output khusus SPL, yang akan mencetak di command line atau file “Var = solusi” untuk setiap variabel dan solusi yang ada. Solusi yang dapat di output adalah solusi unik, solusi tidak unik atau banyak, dan tidak ada solusi.
			String solusi[]	
			integer Nvar	
			boolean isSolve	
	SPLGauss	void	double M[][]	Mencari solusi SPL menggunakan metode Eliminasi Gauss dengan class Gauss. Menggunakan method input untuk input matriks dan method output untuk mencetak solusinya.
			int Nbar	
			int Ncol	
	SPLGaussJordan	void	double M	Mencari solusi SPL menggunakan metode Eliminasi Gauss-Jordan dengan class GaussJordan. Menggunakan method input untuk input matriks dan method output untuk mencetak solusinya.
			int Nbar	
			int Ncol	
	checkBar	boolean	double M[][]	Memeriksa elemen pada baris i mulai dari indeks kolom 0 hingga Ncol-1 adalah 0 semua atau tidak
			int Ncol	
			int i	

	checkCol	boolean	double M[][]	Memeriksa elemen pada kolom j mulai dari indeks baris 0 hingga Nbar-1 adalah 0 semua atau tidak
			int Nbar	
			int j	
detKofaktor	determinan	double	double M[][]	Menghitung determinan input dengan ekspansi kofaktor.
			int N	
	driverdetkofaktor	void	void	Menerima input matriks, menjalankan method determinan, dan mengeluarkan output
DetReduksi	driverDetReduksi	void	void	Menghitung determinan matriks menggunakan OBE dari class Gauss, sehingga determinan dapat dihitung dari perkalian diagonalnya. Input menggunakan class InputMatrix, sedangkan output langsung pada method
inversekofaktor	inverskofaktor	double[][]	double M[][]	Menghitung invers matriks melalui ekspansi kofaktor
			int N	
	driverinverskofaktor	void	void	Menerima input matriks, menjalankan method inverskofaktor, dan mengeluarkan output
splCramer	input	void	void	Input matriks
	output	void	void	Output solusi SPL
	cramerSPL	void	double M[][]	Menghitung solusi SPL menggunakan Kaidah Cramer
			int N	
splinverse	input	void	void	Input matriks
	output	void	void	Output solusi SPL
	inversSPL	void	double M[][]	Menghitung solusi SPL dengan metode matriks balikan
			int N	
InverseOBE	inverse	double[][]	double M[][]	Mencari inverse matriks dari matriks [MI] ketika matriks M diterapkan eliminasi Gauss
			int N	

				Jordan akan menjadi IM^{-1}
	driverInverse	void	void	Mengatur input dan output matriks yang akan di inverse
Gauss	OBEGauss	double[][]	int nBrs	Melakukan proses OBE pada matriks M. Jika iden = True maka, diagonal pada hasil OBE akan menjadi matriks eselon baris, sedangkan jika iden = False, diagonal tidak perlu dijadikan 1. nBrs tidak harus sama dengan nKol
			int nKol	
			double M[][]	
			boolean iden	
GaussJordan	OBEGaussJ	double[][]	int nBrs	Melakukan proses OBE pada matriks M hingga matriks berbentuk matriks eselon baris tereduksi. nBrs tidak harus sama dengan nKol
			int nKol	
			double M[][]	
InputMatrix	input	void	void	Input matriks dari file atau command line yang akan memvalidasi kesamaan jumlah baris dan kolom. Sehingga matriks yang diterima hanya yang jumlah baris dan kolomnya sama
interpolasi	driver_interpolasi	void	void	Menerima input titik-titik interpolasi dan nilai yang akan ditaksir nilai fungsinya, mendapatkan polinom interpolasi dengan memanggil method interpolate, menampilkan polinom interpolasi dan nilai taksiran, dan memberi pilihan menyimpan polinom dalam file
	interpolate	double[]	int n	Mengembalikan koefisien polinom interpolasi hasil interpolasi titik-titik (x,y) dalam bentuk array double
			double x[]	
			double y[]	
	GaussJordan	double[]	int n	Mengembalikan solusi SPL masukan (a[][]) berorde nxn+1 dalam bentuk array double
			double a[][]	
regresi	driver_regresi	void	void	Menerima input tuple-tuple regresi dan tuple yang akan

				dicari nilai regresinya, mendapatkan koefisien persamaan regresi dengan memanggil method regresi, menampilkan persamaan regresi dan nilai taksiran, dan memberi pilihan menyimpan persamaan regresi dalam file
	regresi	double[]	int n	Mengembalikan koefisien persamaan regresi hasil regresi tuple-tuple (x1,x2,...,xn,y) dalam bentuk array double
			int m	
			double x[][]	
			double y[]	
	GaussJordan	double[]	Int n	Mengembalikan solusi SPL masukan (a[][]) berorde nxn+1 dalam bentuk array double
			double a[][]	

Garis Besar Program

Program dijalankan dengan menjalankan binary executable menu.class. Setelah menu.class. Setelah menu.class dijalankan, muncul tampilan menu. Pengguna dapat mengikuti panduan menu untuk menggunakan seluruh fitur program, antara lain: menyelesaikan SPL, mencari determinan matriks, mencari matriks balikan sebuah matriks, membuat polinom interpolasi, dan membuat persamaan regresi linier berganda.

Apabila pengguna ingin memberi masukan dari file, pengguna perlu membuat file masukan dengan format .txt, lalu memasukkannya ke dalam folder test. Setelah itu, pengguna dapat mengikuti panduan menu untuk memberi masukan dari file.

Pada program terdapat fitur untuk menyimpan luaran program dalam file. Pengguna dapat mengikuti panduan menu untuk menggunakan fitur tersebut. File luaran program akan disimpan dalam folder test.

BAB 4: EKSPERIMEN

1. SPL berbentuk Ax=B
   a.

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

Dengan menggunakan metode eliminasi Gauss,

- Input

```
Masukan matriks koefisien (A[i][j])
1 1 -1 -1
2 5 -7 -5
2 -1 1 3
5 2 -4 2
```

```
Masukan matriks hasil persamaan (B[i])
1
-2
4
6
```

- Output

```
Apakah anda ingin masukan output kedalam file ?
1.Ya   2.Tidak
2
SPL tidak memiliki solusi
```

   b.

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

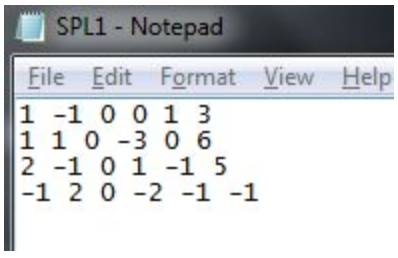
Dengan menggunakan metode Gauss-Jordan,

- Input

Input pada command line

```
1. Masukkan Matriks dari keyboard
2. Masukkan Matriks dari file
Input pilihan: 2
Masukan nama file: SPL1.txt
```

Isi File SPL1.txt



- Output

```
Apakah anda ingin masukan output kedalam file ?
1.Ya   2.Tidak
2
x1 = 3.00+q
x2 = 2.00q
x3 = p
x4 = -1.00+q
x5 = q
```

c.

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

Dengan metode eliminasi Gauss

-input

```
1. Masukkan Matriks dari keyboard
2. Masukkan Matriks dari file
Input pilihan: 1
Input Jumlah Persamaan: 3
Input Jumlah Variabel: 6
Masukan matriks koefisien (A[i][j])
0 1 0 0 1 0
0 0 0 1 1 0
0 1 0 0 0 1
Masukan matriks hasil persamaan (B[i])
2
-1
1
```

-output

```
Apakah anda ingin masukan output kedalam file ?
1.Ya   2.Tidak
2
x1 = p
x2 = 1.000-r
x3 = q
x4 = -2.000-r
x5 = 1.000+r
x6 = r
```

d.

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

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

Dengan menggunakan metode Kaidah Cramer untuk n=6

-input

```
1. Masukkan Matriks dari keyboard
2. Masukkan Matriks dari file
Input pilihan :2
Masukan nama file: Hilbert.txt
```

Hilbert.txt - Notepad

File Edit Format View Help

```
1.0 0.5 0.3333333333333333 0.25 0.2 0.16666666666666667 1.0
0.5 0.3333333333333333 0.25 0.2 0.16666666666666667 0.14285714285714286 0.0
0.3333333333333333 0.25 0.2 0.16666666666666667 0.14285714285714286 0.125 0.0
0.25 0.2 0.16666666666666667 0.14285714285714286 0.125 0.1111111111111111 0.0
0.2 0.16666666666666667 0.14285714285714286 0.125 0.1111111111111111 0.1 0.0
0.16666666666666667 0.14285714285714286 0.125 0.1111111111111111 0.1 0.09090909090909091 0.0
```

-output

```
Apakah Anda ingin masukan output kedalam file ?
1.Ya   2.Tidak
2
x1 = 36.000
x2 = -630.000
x3 = 3360.000
x4 = -7560.000
x5 = 7560.000
x6 = -2772.000
```

Menggunakan metode Matriks Balikan untuk n=10

-input

```
1. Masukkan Matriks dari keyboard
2. Masukkan Matriks dari file
Input pilihan :2
Masukan nama file: Hilbert2.txt
```

Hilbert2.txt - Notepad

File Edit Format View Help

```
1.0 0.5 0.3333333333333333 0.25 0.2 0.16666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 1.0
0.5 0.3333333333333333 0.25 0.2 0.16666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0.09090909090909091 0.0
0.3333333333333333 0.25 0.2 0.16666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.0
0.25 0.2 0.16666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.0
0.2 0.16666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142 0.0
0.16666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142 0.06666666666666667 0.0
0.14285714285714285 0.125 0.1111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142 0.06666666666666667 0.0625 0.0
0.125 0.1111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142 0.06666666666666667 0.0625 0.05882352941176705 0.0
0.1111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142 0.06666666666666667 0.0625 0.05882352941176705 0.05555555555555555 0.0
0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142 0.06666666666666667 0.0625 0.05882352941176705 0.05555555555555555 0.05263157894736842 0.0
```

-output

```
Apakah Anda ingin masukan output kedalam file ?
1.Ya   2.Tidak
2
x1 = 99.998
x2 = -4949.798
x3 = 79195.672
x4 = -600560.529
x5 = 2522331.254
x6 = -6305780.063
x7 = 9608745.465
x8 = -8750772.982
x9 = 4375365.278
x10 = -923684.294
```

2. SPL berbentuk matriks augmented

   a.

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

Menggunakan metode Eliminasi Gauss-Jordan,

- Input

```
1. Masukkan Matriks dari keyboard
2. Masukkan Matriks dari file
Input pilihan: 1
Input Jumlah Persamaan: 4
Input Jumlah Variabel: 4
Masukan matriks koefisien (A[i][j])
1 -1 2 -1
2 1 -2 -2
-1 2 -4 1
3 0 0 -3
```

```
Masukan matriks hasil persamaan (B[i])
-1
-2
1
-3
```

- Output

```
Apakah anda ingin masukan output kedalam file ?
1.Ya   2.Tidak
2
x1 = -1.000+q
x2 = 2.000p
x3 = p
x4 = q
```

b.

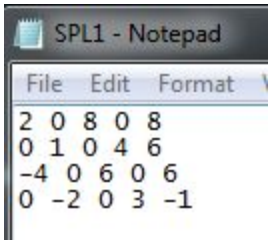
$$\begin{bmatrix} 2 & 0 & 8 & 0 & 8 \\ 0 & 1 & 0 & 4 & 6 \\ -4 & 0 & 6 & 0 & 6 \\ 0 & -2 & 0 & 3 & -1 \\ 2 & 0 & -4 & 0 & -4 \\ 0 & 1 & 0 & -2 & 0 \end{bmatrix}.$$

- Input

Input dari command line

```
1. Masukkan Matriks dari keyboard
2. Masukkan Matriks dari file
Input pilihan: 2
Masukan nama file: SPL1.txt
```

Isi File SPL1.txt

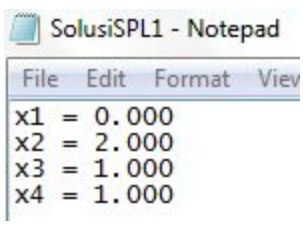


- Output

Output pada command line

```
Apakah anda ingin masukan output kedalam file ?
1.Ya   2.Tidak
1
Masukan nama file: SolusiSPL1.txt
x1 = 0.000
x2 = 2.000
x3 = 1.000
x4 = 1.000
```

Contoh Output pada file ../test/SolusiSPL1.txt



3. SPL berbentuk persamaan

a.

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

Menggunakan metode Kaidah Cramer,

- Input

```
1. Masukkan Matriks dari keyboard
2. Masukkan Matriks dari file
Input pilihan :1
Input ukuran matriks (NxN): 4
Masukan matriks koefisien (A[i][j])
8 1 3 2
2 9 -1 -2
1 3 2 -1
1 0 6 4
```

```
Masukan matriks hasil persamaan (B[i])
0
1
2
3
```

- Output

```
Apakah Anda ingin masukan output kedalam file ?
1.Ya   2.Tidak
2
x1 = -0.224
x2 = 0.182
x3 = 0.709
x4 = -0.258
```

b.

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

Menggunakan metode Eliminasi Gauss-Jordan,

- Input

```
1. Masukkan Matriks dari keyboard
2. Masukkan Matriks dari file
Input pilihan: 2
Masukan nama file: SPL2.txt
```

Isi File SPL2.txt

SPL2 - Notepad

File Edit Format View Help

```
0 0 0 0 0 0 1 1 1 13
0 0 0 1 1 1 0 0 0 15
1 1 1 0 0 0 0 0 0 8
0 0 0.04289 0 0.04289 0.75 0.04289 0.75 0.6139 14.79
0 0.25 0.91421 0.25 0.91421 0.25 0.91421 0.25 0 14.31
0.61396 0.75 0.04289 0.76 0.04289 0 0.04289 0 0 3.81
0 0 1 0 0 1 0 0 1 18
0 1 0 0 1 0 0 1 0 12
1 0 0 1 0 0 1 0 0 6
0.04289 0.75 0.61396 0 0.04289 0.75 0 0 0.04289 10.51
0.9142 0.25 0 0.25 0.9142 0.25 0 0.25 0.91421 16.13
0.04289 0 0 0.75 0.04289 0 0.61396 0.75 0.04289 7.04
```

- Output

Output pada command line

```
Apakah anda ingin masukan output kedalam file ?
1.Ya   2.Tidak
1
Masukan nama file: SolusiSPL2.txt
SPL tidak memiliki solusi
```

Contoh Output pada file ../test/SolusiSPL2.txt

SolusiSPL2 - Notepad

File Edit Format View Help

```
SPL tidak memiliki solusi
```

4. SPL dalam analisis rangkaian

$$
\begin{aligned}
i_{12} + i_{52} + i_{32} &= 0 \\
- i_{52} + i_{65} - i_{54} &= 0 \\
- i_{32} + i_{43} &= 0 \\
i_{54} - i_{43} &= 0 \\
i_{32} R_{32} - V_2 - V_3 &= 0 \\
i_{43} R_{43} + V_3 - V_4 &= 0 \\
i_{65} R_{65} + V_5 &= V_6 \\
i_{12} R_{12} + V_2 &= V_1 \\
i_{54} R_{54} + V_4 - V_5 &= 0 \\
i_{52} R_{52} + V_2 - V_5 &= 0
\end{aligned}
$$

Tentukan nilai dari:

$$i_{12}, \quad i_{52}, \quad i_{32}, \quad i_{65}, \quad i_{54}, \quad i_{13}, \quad V_2, \quad V_3, \quad V_4, \quad V_5$$

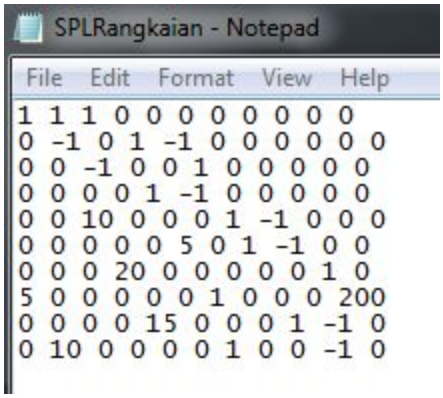
bila diketahui

$$
\begin{aligned}
R_{12} &= 5 \text{ ohm}, \quad R_{52} = 10 \text{ ohm}, \quad R_{32} = 10 \text{ ohm} \\
R_{65} &= 20 \text{ ohm}, \quad R_{54} = 15 \text{ ohm}, \quad R_{14} = 5 \text{ ohm}. \\
V_1 &= 200 \text{ volt}, \quad V_6 = 0 \text{ volt}.
\end{aligned}
$$

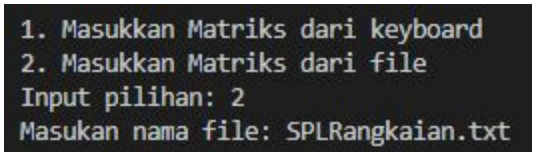
Dengan menggunakan metode Matriks Balikan,

- Input

Isi File SPLRangkaian.txt



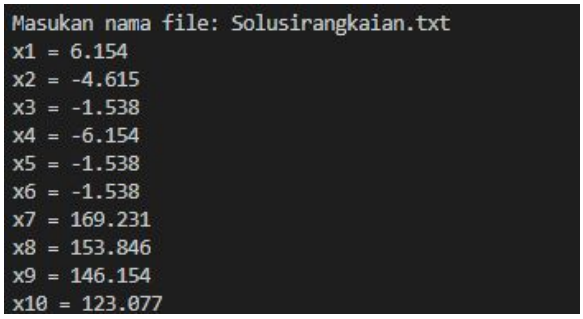
Input pada command line



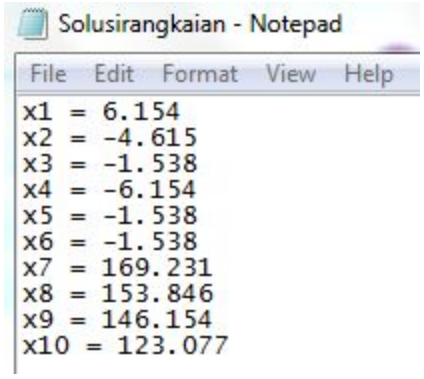
- Output

Misal,

i12 = x1
i52 = x2
i32 = x3
i65 = x4
i54 = x5
i43 = x6
V2 = x7
V3 = x8
V4 = x9
V5 = x10



Contoh output dalam bentuk file ../test/Solusirangkaian.txt



5. Interpolasi

(Interpolasi) 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 default berikut:

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

Polinom interpolasi yang dihasilkan adalah sebagai berikut.

```
polinom interpolasi:
0.000000 x^7
-0.000000 x^6
+0.000000 x^5
+0.026042 x^4
+0.000000 x^3
+0.197396 x^2
+0.240000 x^1
-0.022977 x^0
```

Perkiraan nilai f(x) untuk beberapa nilai x adalah sebagai berikut.

```
taksiran fungsi pada x=0.200000: 0.032961
taksiran fungsi pada x=0.550000: 0.171119
taksiran fungsi pada x=0.850000: 0.337236
taksiran fungsi pada x=1.280000: 0.677542
```

6. Interpolasi dalam prediksi jumlah kasus COVID-19

Jumlah kasus positif Covid-19 di Indonesia semakin bertambah dari hari ke hari. Di bawah ini diperlihatkan jumlah kasus Covid-19 di Indonesia mulai dari tanggal 24 April 2020 hingga 10 September 2020:

Tanggal	Tanggal (desimal)	Jumlah Kasus
24/04/20	4,800	8.211
30/04/20	5,000	10.118
16/05/20	5,516	17.025
22/05/20	5,710	20.796
15/06/20	6,500	39.294
06/07/20	7,194	64.958
03/08/20	8,097	113.134
08/08/20	8,258	123.503
01/09/20	9,033	177.571
10/09/20	9,333	145.510

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 24/04/20 (dibaca: 24 April 2020) diperoleh tanggal(desimal) sebagai berikut:

$$\text{Tanggal(desimal)} = 4 + (24/30) = 4{,}800$$

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

a. 25/05/20
b. 30/08/20
c. 15/09/20
d. beserta masukan user lainnya berupa **tanggal (desimal) yang sudah diolah** dengan asumsi prediksi selalu dilakukan untuk tahun 2020.

Polinom interpolasi yang dihasilkan adalah sebagai berikut.

```
polinom interpolasi:
0.000000 x^10
-36.470955 x^9
+2041.919610 x^8
-50061.951418 x^7
+704212.228636 x^6
-6249553.975040 x^5
+36176035.665877 x^4
-136003269.391158 x^3
+318150023.106310 x^2
-415842572.909669 x^1
+227096327.662611 x^0
```

a. Tanggal(desimal) untuk 25/05/20 adalah 5+(25/31)=5,80645.

```
taksiran fungsi pada x=5.806450: 22804.210727
```

Taksiran nilai fungsi yang didapat untuk tanggal(desimal) 5,80645 adalah 22.804 (dibulatkan). Jadi, jumlah kasus Covid-19 pada tanggal 25/05/20 kira-kira 22.804.

b. Tanggal(desimal) untuk 30/08/20 adalah 8+(30/31)=8,967741=9.

```
taksiran fungsi pada x=9.000000: 176872.858131
```

Taksiran nilai fungsi yang didapat untuk tanggal(desimal) 9 adalah 176.873 (dibulatkan). Jadi, jumlah kasus Covid-19 pada tanggal 30/08/20 kira-kira 176.873.

c. Tanggal(desimal) untuk 15/09/20 adalah 9+(15/30)=9,5.

```
taksiran fungsi pada x=9.500000: 68216.427071
```

Taksiran nilai fungsi yang didapat untuk tanggal(desimal) 9,5 adalah 68.216 (dibulatkan). Jadi, jumlah kasus Covid-19 pada tanggal 15/09/20 kira-kira 68.216. Perlu diperhatikan bahwa tanggal 15/09/20 melebihi tanggal maksimal pada data yang diberikan (10/09/20), sehingga taksiran yang didapat cenderung tidak akurat dan tidak mengikuti tren menaik.

d. Tanggal(desimal) untuk 17/08/20 adalah 8+(17/31)=8,54839.

```
taksiran fungsi pada x=8.548390: 145114.763111
```

Taksiran nilai fungsi yang didapat untuk tanggal(desimal) 8,54839 adalah 145.115 (dibulatkan). Jadi, jumlah kasus Covid-19 pada tanggal 17/08/20 kira-kira 145.115.

7. Interpolasi dalam penyederhanaan fungsi

7. Sederhanakan fungsi

$$f(x) = \frac{x^2 + \sqrt{x}}{e^x + x}$$

dengan polinom interpolasi derajat n di dalam selang [0, 2]. Sebagai contoh, jika $n = 5$, maka titik-titik x yang diambil di dalam selang [0, 2] berjarak $h = (2 - 0)/5 = 0.4$.

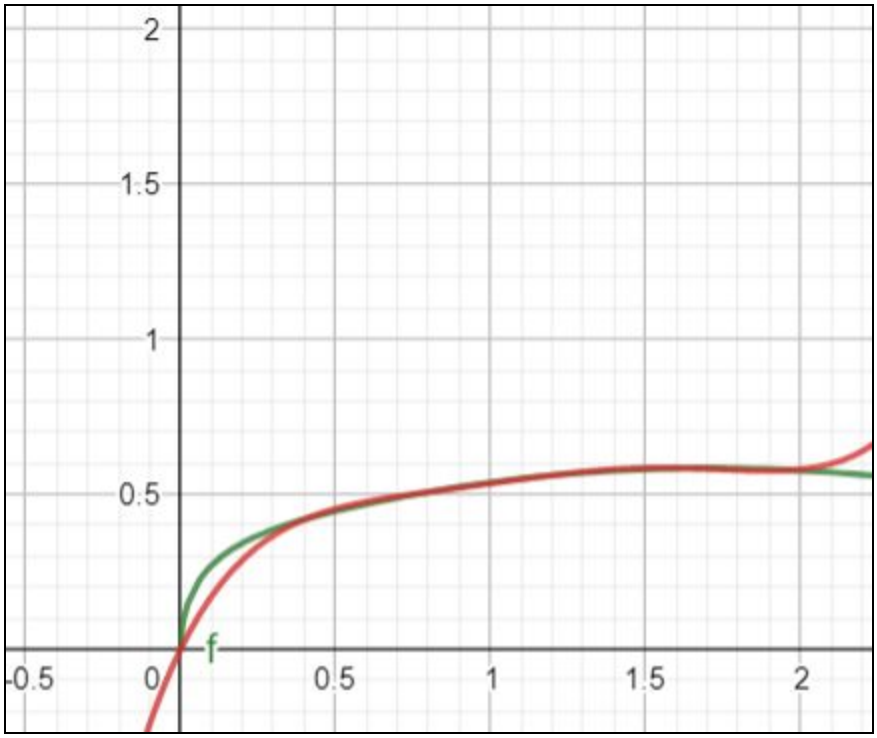
Untuk n=5, titik-titik yang diambil adalah sebagai berikut.

```
0.0 0.0
0.4 0.4188842
0.8 0.5071580
1.2 0.5609247
1.6 0.5836857
2.0 0.5766515
```

Catatan: tiap baris berisi x dan y dipisahkan spasi, didapatkan dengan kalkulator.
Polinom interpolasi yang dihasilkan adalah sebagai berikut.

```
polinom interpolasi:
0.000000 x^6
+0.236256 x^5
-1.421263 x^4
+3.237110 x^3
-3.552681 x^2
+2.035258 x^1
0.000000 x^0
```

Berikut grafik f(x) dan polinom interpolasi. Kurva hijau menyatakan f(x) dan kurva merah menyatakan polinom interpolasi.



8. Regresi linier 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.

Persamaan regresi linier berganda dari data pada tabel adalah sebagai berikut.

```
Persamaan Regresi:
y=
-3.507778
-0.002625 x1
+0.000799 x2
+0.154155 x3
```

Untuk x1=50, x2=76, dan x3=29.30, estimasi nilai Nitrous Oxide adalah sebagai berikut.

```
Nilai taksiran: 0.938434
```

BAB 5: KESIMPULAN, SARAN, DAN REFLEKSI

Kesimpulan

Program ini dapat:

1. menghitung solusi SPL dengan metode eliminasi Gauss, metode eliminasi Gauss-Jordan, metode matriks balikan, dan kaidah Cramer;
2. menyelesaikan persoalan interpolasi dan regresi linier;
3. menghitung matriks balikan;
4. menghitung determinan matriks dengan metode reduksi baris dan metode ekspansi kofaktor.

Saran

- Memeriksa setiap kasus khusus untuk penyelesaian SPL yang memiliki solusi banyak/tidak unik, karena matriks eselonnya dapat melalui kasus-kasus yang tidak dapat diselesaikan dengan OBE bertahap dari baris dan kolom 1, hingga baris dan kolom terakhir seperti biasanya. Tetapi memerlukan pertukaran dan pemeriksaan ketika ada koef variabel yang elemen 0 pada diagonal utamanya.
- Meningkatkan presisi perhitungan terutama dalam perhitungan determinan karena ada kemungkinan terbuat determinan 0 dari yang sebenarnya mempunyai nilai namun sangat kecil sehingga membuat perhitungan metode-metode yang memakai determinan menjadi tidak mungkin.

Refleksi

- Tugas ini mendorong penulis untuk belajar tentang aljabar linier, bahasa Java, dan berkolaborasi dengan platform Github.

Daftar Referensi

https://www.profematika.com/eliminasi-gauss-dan-contoh-penerapannya/
https://en.wikipedia.org/wiki/Laplace_expansion
https://en.wikipedia.org/wiki/Minor_(linear_algebra)
https://en.wikipedia.org/wiki/Invertible_matrix
https://www.uniksharianja.com/2015/03/minor-kofaktor-matrik-kofaktor-dan.html
https://en.wikipedia.org/wiki/Determinant
Laaksonen, Antti. 2017. *Competitive Programmer's Handbook*. diambil dari https://cses.fi/book/book.pdf.