Lab Analisis Data 1 - Modul 4

MATH2031 Aljabar Linear

Calvin Institute of Technology

Capaian Pembelajaran:

Setelah mengikuti modul ini, mahasiswa akan dapat menyelesaikan permasalahan-permasalahan dasar dalam topik SPL dan Matriks dengan menggunakan Python

1 Menghitung Determinan

Determinan dari suatu matriks persegi dapat dihitung dengan mudah oleh Python dengan menggunakan perintah `np.linalg.det()`. Perhatikan contoh di bawah ini.

[1]:
```
import numpy as np

A = np.array([
    [2, 3, -5],
    [-1, -4, 6],
    [0, 1, -7]
])
print("Matriks A adalah\n", A)

print("Determinan matriks A adalah\n", np.linalg.det(A))
```

```
Matriks A adalah
 [[ 2  3 -5]
 [-1 -4  6]
 [ 0  1 -7]]
Determinan matriks A adalah
 28.000000000000001
```

Keterangan:

Seharusnya nilai determinan di atas adalah tepat 28, namun Python mungkin mengoutputkan sedikit error, yaitu sebesar `0.00000000000001`. Hal ini disebabkan oleh kesalahan pembulatan yang seringkali tidak dapat dihindari oleh komputer ketika melakukan komputasi dengan data bertipe bilangan real. (Dalam arsitektur komputer, aritmatika bilangan real sering direpresentasikan oleh *floating-point arithmetics*. Penjelasan informal lebih lanjut dapat dilihat di https://en.wikipedia.org/wiki/Floating-point_arithmetic)

2 Menghitung Invers dan Menyelesaikan SPL

2.1 Mencari Invers Matriks

Melalui Python, invers matriks persegi juga dapat dihitung dengan mudah dengan perintah `np.linalg.inv()`. Perhatikan contoh di bawah ini.

[2]:
```
A = np.array([
    [2, 3, -5],
    [-1, -4, 6],
    [0, 1, -7]
])
print("Matriks A adalah\n", A)

# Sebelum menghitung invers, cek terlebih dahulu apakah
# matriks yang akan dihitung inversnya memiliki determinan 0?

print("Cek bahwa determinan matriks A adalah\n", np.linalg.det(A))

# Jika tidak nol maka matriks memiliki invers dan perintah
# selanjutnya dapat dijalankan.

A_invers = np.linalg.inv(A)

print("Invers matriks A adalah\n", A_invers)

# Menunjukkan bahwa A_invers adalah betul-betul invers dari A
# dengan cara mengalikan matriks tersebut dengan matriks A
# untuk dicek apakah hasilnya adalah matriks Identitas.

print("A @ A_invers adalah\n", A @ A_invers)

print("A_invers @ A adalah\n", A_invers @ A)
```

```
Matriks A adalah
 [[ 2  3 -5]
 [-1 -4  6]
 [ 0  1 -7]]
Cek bahwa determinan matriks A adalah
 28.000000000000001
Invers matriks A adalah
 [[ 0.78571429  0.57142857 -0.07142857]
 [-0.25       -0.5        -0.25      ]
 [-0.03571429 -0.07142857 -0.17857143]]
A @ A_invers adalah
 [[ 1.00000000e+00 -1.38777878e-17 -2.77555756e-17]
 [-1.38777878e-17  1.00000000e+00 -5.55111512e-17]
 [ 3.46944695e-17  6.93889390e-17  1.00000000e+00]]
A_invers @ A adalah
 [[1.00000000e+00 1.66533454e-16 1.66533454e-16]
 [0.00000000e+00 1.00000000e+00 0.00000000e+00]
 [0.00000000e+00 2.77555756e-17 1.00000000e+00]]
```

Remark:

Hasil perkalian `A` dab `A_invers` di atas adalah sebetulnya matriks identitas, namun lagi-lagi terdapat kesalahan pembulatan floating point. Notasi `6.93889390e-17`, misalnya, artinya adalah bilangan real $6.9388939 \times 10^{-17}$, suatu bilangan real yang sangat kecil, yang sangat dekat dengan 0.

2.2 Menyelesaikan SPL Konsisten bersolusi Tunggal

Selanjutnya, terdapat beberapa cara untuk menyelesaikan SPL dengan Python. Misalkan SPL yang hendak diselesaikan adalah

$$A\mathbf{x} = \mathbf{b}.$$

Pertama, kita **perlu mensyaratkan bahwa A matriks persegi dan** $\det(A) \neq 0$. Perhatikan bahwa dalam kasus ini, solusi SPL bersifat tunggal.

Metode 1: Invers Matriks.

Untuk menyelesaikan SPL, kita dapat menghitung nilai invers matriks koefisien untuk menyelesaikan SPL menjadi

$$\mathbf{x} = A^{-1}\mathbf{b}.$$

Misalkan SPL yang hendak diselesaikan adalah sebagai berikut:

$$\begin{aligned} 2x_1 + 3x_2 - 5x_3 &= 10 \\ -x_1 - 4x_2 + 6x_3 &= -11 \\ x_2 - 7x_3 &= 8 \end{aligned}$$

Maka SPL di atas dapat diselesaikan dengan Python sebagai berikut.

[3]:
```
# Definisikan matriks koefisien A
A = np.array([
    [2, 3, -5],
    [-1, -4, 6],
    [0, 1, -7]
])
print("Matriks A adalah\n", A)

# Definisikan vektor konstanta b
b = np.array([10, -11, 8])
b = np.reshape(b, (3,1)) # direshape agar menjadi vektor kolom
print("Vektor b adalah\n", b)

# Kemudian hitung invers matriks A
A_invers = np.linalg.inv(A)
print("Invers matriks A adalah\n", A_invers)

# Langkah terakhir hitung x = A_invers @ b
x = A_invers @ b
print("Vektor solusi x adalah\n", x)
```

```
Matriks A adalah
 [[ 2  3 -5]
 [-1 -4  6]
 [ 0  1 -7]]
Vektor b adalah
 [[ 10]
 [-11]
 [  8]]
Invers matriks A adalah
 [[ 0.78571429  0.57142857 -0.07142857]
 [-0.25       -0.5        -0.25      ]
 [-0.03571429 -0.07142857 -0.17857143]]
Vektor solusi x adalah
 [[ 1.]
 [ 1.]
 [-1.]]
```

Berdasarkan hasil di atas, dapat disimpulkan bahwa solusi SPL adalah

$$x_1 = 1, \qquad x_2 = 1, \qquad x_3 = -1$$

Metode 2: Perintah `np.linalg.solve()`.

Untuk menyelesaikan SPL $A\mathbf{x} = \mathbf{b}$, vektor variabel $\mathbf{x}$ dapat dicari dengan perintah `np.linalg.solve(A, b)`. Perhatikan contoh di bawah.

[4]:
```
# Definisikan matriks koefisien A
A = np.array([
    [2, 3, -5],
    [-1, -4, 6],
    [0, 1, -7]
])
print("Matriks A adalah\n", A)

# Definisikan vektor konstanta b
b = np.array([10, -11, 8])
b = np.reshape(b, (3,1)) # direshape agar menjadi vektor kolom
print("Vektor b adalah\n", b)

# Kemudian hitung vektor solusi
x = np.linalg.solve(A,b)
print("Vektor solusi x adalah\n", x)
```

```
Matriks A adalah
 [[ 2  3 -5]
 [-1 -4  6]
 [ 0  1 -7]]
Vektor b adalah
 [[ 10]
```

```
 [-11]
 [  8]]
Vektor solusi x adalah
 [[ 1.]
 [ 1.]
 [-1.]]
```

Latihan 1.
Ambil matriks A dalam file CSV dari https://docs.google.com/spreadsheets/d/e/2PACX-1vTsO1o2fddVbz8ZwdIZO0YYhie4trT30AMtg7Uyv8SsyUkGflHaONRVaIDlS-4KOkHhwwKJ2PaUWKya/pub?gid=445918484&single=true&output=csv

Juga, ambil vektor $\mathbf{b}$ dari https://docs.google.com/spreadsheets/d/e/2PACX-1vTsO1o2fddVbz8ZwdIZO0YYhie4trT30AMtg7Uyv8SsyUkGflHaONRVaIDlS-4KOkHhwwKJ2PaUWKya/pub?gid=200211429&single=true&output=csv

Selesaikan SPL

$$A\mathbf{x} = \mathbf{b}.$$

dan simpanlah vektor solusi $\mathbf{x}$ ke dalam bentuk CSV (judul file bebas).

3 Menghitung Statistik Vektor/Matriks Sederhana

Ketika menghadapi data berukuran besar, seringkali sangat membantu untuk kita dapat melihat rangkuman/intisari dari perilaku data tersebut secara gambar besar. Sebagai ilmuwan data, tentu kita berharap dapat memahami data tersebut secara keseluruhan tanpa perlu melihat satu demi satu nilai numerik dari setiap data secara manual. Berikut beberapa besaran statistik yang sering dipakai untuk dapat mendemonstasikan hal di atas.

3.1 Menghitung nilai minimum dan maksimum

Misalkan vektor $\mathbf{x}$ berisi 10 data sebagai berikut.

$$\mathbf{x} = \begin{bmatrix} 2 & -1 & 5 & 12 & 4 & -1 & 0 & 8 & 12 & 6 \end{bmatrix}$$

Kita dapat menggunakan Python untuk mencari nilai terkecil (min) dan nilai terbesar (max) dari vektor di atas dengan perintah `np.min(x)` dan `np.max(x)`. Selain itu, dapat pula kita menggunakan perintah `np.argin(x)` dan `np.argmax(x)` untuk mencari tahu argumen/indeks di mana elemen terkecil dan terbesar terletak di vektor $\mathbf{x}$. Perhatikan kode berikut.

[5]:
```
x = np.array([2, -1, 5, 12, 4, -1, 0, 8, 12, 6])
print("Vektor x adalah\n", x)

# Mencari nilai minimum dan argumen min dari vektor x
print("\n")
x_min = np.min(x)
x_min_arg = np.argmin(x)
print("Nilai terkecil di x adalah elemen ke", x_min_arg
```

```
      , "yaitu sebesar", x_min)

# Mencari nilai maksimum dan argumen max dari vektor x
print("\n")
x_max = np.max(x)
x_max_arg = np.argmax(x)
print("Nilai terbesar di x adalah elemen ke", x_max_arg
      , "yaitu sebesar", x_max)
```

```
Vektor x adalah
 [ 2 -1  5 12  4 -1  0  8 12  6]


Nilai terkecil di x adalah elemen ke 1 yaitu sebesar -1


Nilai terbesar di x adalah elemen ke 3 yaitu sebesar 12
```

Keterangan:

Perhatikan bahwa nilai terkecil pada vektor x adalah -1, yang sebetulnya dapat dijumpai pada elemen ke 1 dan ke 5 dari vektor x. Namun demikian, fungsi `np.argmin()` hanya mengembalikan indeks terkecil dari semua elemen terkecil yang mungkin muncul di vektor x. Hal ini juga berlaku pada fungsi `np.argmax()`.

Dalam konteks matriks, kita dapat pula mencari nilai min & max elemen-elemennya, namun kita dapat memahaminya dalam tiga pendekatan sebagai berikut.

1. Nilai min/max dari masing-masing kolom (*axis = 0*)
2. Nilai min/max dari masing-masing baris (*axis = 1*)
3. Nilai min/max dari keseluruhan matriks

Perhatikan contoh berikut.

[6]:
```
A = np.reshape(np.array([2, -1, 5, 12, 6, -1, 8, 8, 2, 6]), (2,5))
print("Matriks A adalah\n", A)

# Mencari nilai min dan max dari masing-masing kolom
print("\n")
A_min0 = np.min(A, axis=0)
A_min_arg0 = np.argmin(A, axis=0)
A_max0 = np.max(A, axis=0)
A_max_arg0 = np.argmax(A, axis=0)
print("Nilai min per kolom          :", A_min0)
print("Masing-masing pada elemen ke :", A_min_arg0)
print("Nilai max per kolom          :", A_max0)
print("Masing-masing pada elemen ke :", A_max_arg0)

# Mencari nilai min dan max dari masing-masing baris
```

```
print("\n")
A_min1 = np.min(A, axis=1)
A_min_arg1 = np.argmin(A, axis=1)
A_max1 = np.max(A, axis=1)
A_max_arg1 = np.argmax(A, axis=1)
print("Nilai min per baris          :", A_min1)
print("Masing-masing pada elemen ke :", A_min_arg1)
print("Nilai max per baris          :", A_max1)
print("Masing-masing pada elemen ke :", A_max_arg1)


# Mencari nilai min dan max dari keseluruhan matriks
print("\n")
A_min = np.min(A)
A_min_arg = np.argmin(A)
A_max = np.max(A)
A_max_arg = np.argmax(A)
print("Nilai min seluruh matriks :", A_min)
print("yaitu pada elemen ke      :", A_min_arg)
print("Nilai max seluruh matriks :", A_max)
print("yaitu pada elemen ke      :", A_max_arg)
```

```
Matriks A adalah
 [[ 2 -1  5 12  6]
 [-1  8  8  2  6]]


Nilai min per kolom          : [-1 -1  5  2  6]
Masing-masing pada elemen ke : [1 0 0 1 0]
Nilai max per kolom          : [ 2  8  8 12  6]
Masing-masing pada elemen ke : [0 1 1 0 0]


Nilai min per baris          : [-1 -1]
Masing-masing pada elemen ke : [1 0]
Nilai max per baris          : [12  8]
Masing-masing pada elemen ke : [3 1]


Nilai min seluruh matriks : -1
yaitu pada elemen ke      : 1
Nilai max seluruh matriks : 12
yaitu pada elemen ke      : 3
```

3.2 Menghitung jumlah dan rata-rata nilai vektor/matriks

Prinsip perhitungan dengan Python untuk jumlah dan rata-rata elemen-elemen dari suatu matriks juga mirip dengan prinsip pada mencari nilai minimum dan maksimum. Perhatikan contoh di bawah ini.

[7]:
```
A = np.reshape(np.array([2, -1, 5, 12, 6, -1, 8, 8, 2, 6]), (2,5))
print("Matriks A adalah\n", A)

# Mencari jumlah dan rata2 dari masing-masing kolom
print("\n")
Asum0 = np.sum(A, axis=0)
Amean0 = np.mean(A, axis=0)
print("Jumlah per kolom    :", Asum0)
print("Rata-rata per kolom :", Amean0)

# Mencari jumlah dan rata2 dari masing-masing baris
print("\n")
Asum1 = np.sum(A, axis=1)
Amean1 = np.mean(A, axis=1)
print("Jumlah per baris    :", Asum1)
print("Rata-rata per baris :", Amean1)

# Mencari jumlah dan rata2 dari keseluruhan matriks
print("\n")
Asum = np.sum(A)
Amean = np.mean(A)
print("Jumlah keseluruhan    :", Asum)
print("Rata-rata keseluruhan :", Amean)
```

```
Matriks A adalah
 [[ 2 -1  5 12  6]
 [-1  8  8  2  6]]


Jumlah per kolom    : [ 1  7 13 14 12]
Rata-rata per kolom : [0.5 3.5 6.5 7.  6. ]


Jumlah per baris    : [24 23]
Rata-rata per baris : [4.8 4.6]


Jumlah keseluruhan    : 47
Rata-rata keseluruhan : 4.7
```

3.3 Menyaring data dengan kondisi tertentu

Python memiliki fitur yang praktis dan nyaman untuk menyaring data dalam bentuk vektor. Misalkan vektor $\mathbf{x}$ berisi 10 data sebagai berikut.

$$\mathbf{x} = \begin{bmatrix} 2 & -1 & 5 & 12 & 4 & -1 & 0 & 8 & 12 & 6 \end{bmatrix}$$

Kemudian kita ingin membuang semua data **yang lebih kecil dari 5** dari vektor $\mathbf{x}$, sehingga kita mengharapkan dapat memperoleh vektor baru sebagai berikut:

$$\mathbf{x}' = \begin{bmatrix} 5 & 12 & 8 & 12 & 6 \end{bmatrix}$$

Hal di atas dapat dilakukan dengan mudah melalui Python dengan perintah sebagai berikut.

[8]:
```
x = np.array([2, -1, 5, 12, 4, -1, 0, 8, 12, 6])
print("Vektor x awalnya adalah\n", x)

x_baru = x[x >= 5]
# Perintah di atas berarti mengambil semua elemen di x yang
# memiliki sifat ">= 5"
print("Vektor x yang baru adalah\n", x_baru)

print("Banyaknya elemen di x yang >=5 adalah", np.size(x_baru))
```

```
Vektor x awalnya adalah
 [ 2 -1  5 12  4 -1  0  8 12  6]
Vektor x yang baru adalah
 [ 5 12  8 12  6]
Banyaknya elemen di x yang >=5 adalah 5
```

Latihan 2.
Ambil matriks A dalam file CSV dari
https://raw.githubusercontent.com/yozeftjandra/MATH2031/main/BigMat%20-%20A.csv

Juga, ambil vektor $\mathbf{b}$ dari
https://raw.githubusercontent.com/yozeftjandra/MATH2031/main/BigMat%20-%20b.csv

1. Hitunglah jumlah elemen dari matriks A pada baris ke 14, 71, dan 89 (cetaklah tiga nilai tersebut)
2. Hitunglah rata-rata elemen dari matriks A pada kolom ke 25, 37, 61 (cetaklah tiga nilai tersebut)
3. Cetaklah elemen-elemen pada vektor b yang positif saja dan simpanlah dalam bentuk CSV.
4. Ada berapa elemen pada vektor b yang nilainya di bawah rata-rata keseluruhan nilai vektor b?
5. Dari semua nilai elemen pada vektor b yang lebih kecil dari 5, berapakah yang nilainya paling besar? Pada indeks ke berapa dari vektor x kah nilai tersebut dapat kita temukan?
6. Berapakah jumlahan dari nilai terbesar dari masing-masing baris di matriks A?