Lab Analisis Data 1 - Modul 3

MATH2031 Aljabar Linear

Calvin Institute of Technology

Capaian Pembelajaran:

Setelah mengikuti modul ini, mahasiswa akan dapat melakukan manipulasi matriks dasar dengan Python dan NumPy

1 Membaca dan menulis matriks dari/ke file .csv

Dalam prakteknya, seringkali data numerik yang berukuran besar disimpan dalam bentuk .csv (*comma separated value*). File dengan bentuk .csv dapat dipahami sebagai versi primitif dari bentuk spreadsheet seperti Microsoft Excel atau Google Sheet. Anda dapat membuka file .csv dengan aplikasi Text Editor seperti Notepad atau TextEdit.

1.1 Membaca file .csv

Misalkan kita hendak memuat file .csv dari halaman https://docs.google.com/spreadsheets/d/e/2PACX-1vTsO1o2fddVbz8ZwdIZO0YYhie4trT30AMtg7Uyv8SsyUkGflHaONRVaIDlS-4KOkHhwwKJ2PaUWKya/pub?gid=0&single=true&output=csv. Berikut merupakan perintah-perintah yang dapat dilakukan dengan Python untuk memuat informasi dalam file tersebut ke dalam lingkungan Python.

[1]:
```
import numpy as np

A = np.loadtxt("https://docs.google.com/spreadsheets/d/e/
 ↪2PACX-1vTsO1o2fddVbz8ZwdIZO0YYhie4trT30AMtg7Uyv8SsyUkGflHaONRVaIDlS-4KOkHhwwKJ2PaUWKya/
 ↪pub?gid=0&single=true&output=csv", delimiter=",")
print(A)
```

```
[[10. 34. 63. … 50. 34. 27.]
 [ 1. 87. 64. … 24. 52. 54.]
 [73. 41. 53. … 37. 60. 46.]
 …
 [88.  4. 51. … 40.  5.  8.]
 [64.  6. 80. … 16. 57. 39.]
 [45. 91. 41. …  3. 71. 31.]]
```

Keterangan:

Python hanya menampilkan sebagian isi matriks dari file .csv di atas saja, terutama di bagian awal dan akhir baris dan kolom; demi kenyamanan pengguna. Untuk dapat menampilkan semua isi matriks besar, kita dapat menuliskan perintah: `np.set_printoptions(threshold=np.inf)` untuk membuat pengaturan pencetakan matriks dengan batasan ukuran cetak mencapai tak berhingga. (tidak disarankan)

1.2 Menulis file .csv

Sebaliknya, kita juga dapat menyimpan hasil perhitungan dengan Python ke dalam file .csv pula. Perhatikan contoh berikut. Misalkan kita ingin membuat matriks diagonal berukuran 100 x 100 yang setiap diagonalnya adalah angka 2 dan menyimpannya dalam bentuk .csv dengan judul `"BigTwiceIdentity.csv"`. Maka, kita dapat menulis perintah Python sebagai berikut.

[2]:
```
B = np.diag(2 * np.ones(100))
print(B)

np.savetxt("BigTwiceIdentity.csv", B, delimiter=",")
# Cek direktori Anda, seharusnya file BigTwiceIdentity.csv
# akan muncul di sana setelah perintah ini dijalankan
```

```
[[2. 0. 0. … 0. 0. 0.]
 [0. 2. 0. … 0. 0. 0.]
 [0. 0. 2. … 0. 0. 0.]
 …
 [0. 0. 0. … 2. 0. 0.]
 [0. 0. 0. … 0. 2. 0.]
 [0. 0. 0. … 0. 0. 2.]]
```

Keterangan:

Perintah `np.savetxt(a, b, delimiter=",")` berarti bahwa Python akan menyimpan matriks `b` ke dalam bentuk csv dengan judul seperti string `a`.

Setelah perintah tersebut dijalankan, Python akan menghasilkan file .csv tersebut di direktori yang sama di mana Anda menyimpan file Python Anda.

2 Ukuran, Indeks & Slicing Matriks

2.1 Mengecek ukuran suatu matriks

Misalkan `A` merupakan suatu variabel matriks. Kita dapat meminta Python mengembalikan nilai ukuran matriks `A` tersebut dengan menulis perintah `np.shape(A)`. Output yang dikembalikan oleh fungsi tersebut biasanya berbentuk tuplet `(m,n)`, yang berarti bahwa matriks `A` berdimensi $m \times n$.

Perhatikan contoh di bawah ini.

[3]:
```
import numpy as np

A = np.array([[1, 2], [-1, 9], [7, -2]])
print("Matriks A =\n", A)
[row, col] = np.shape(A)

print("Dimensi matriks A adalah", row, "x", col)
```

```
Matriks A =
 [[ 1  2]
 [-1  9]
```

```
 [ 7 -2]]
Dimensi matriks A adalah 3 x 2
```

[4]:
```
# Fungsi np.shape() juga dapat diterapkan pada vektor

x = np.array([1, 2, 5])
print("Vektor x =", x)

print("Dimensi vektor x adalah", np.shape(x))
```

```
Vektor x = [1 2 5]
Dimensi vektor x adalah (3,)
```

2.2 Pengindeksan vektor dan matriks dalam Python

Dalam Python, indeks vektor dan matriks dimulai dari 0 (bukan 1). Misalkan kita memiliki vektor `x` dengan panjang 4 sebagai berikut:

$$x = \begin{bmatrix} 2 & 4 & 1 & 5 \end{bmatrix}$$

Maka, Python akan 'menomori' vektor tersebut mulai dari indeks 0 sebagai berikut.

i	x[i]
0	2
1	4
2	1
3	5

Artinya, bagi Python, 2 merupakan komponen ke-0 dari `x`, atau `x[0] = 2`, atau dapat juga kita mengatakan bahwa nilai `x` pada indeks 0 adalah 2.

Contoh lain, komponen ke-2 dari vektor `x` adalah `x[2]=1` (dan bukan 5).

[5]:
```
x = np.array([2, 4, 1, 5])
print("Vektor x =", x)

print("Komponen ke-0 dari vektor x adalah", x[0])
print("Komponen ke-1 dari vektor x adalah", x[1])
print("Komponen ke-2 dari vektor x adalah", x[2])
print("Komponen ke-3 dari vektor x adalah", x[3])
```

```
Vektor x = [2 4 1 5]
Komponen ke-0 dari vektor x adalah 2
Komponen ke-1 dari vektor x adalah 4
Komponen ke-2 dari vektor x adalah 1
Komponen ke-3 dari vektor x adalah 5
```

Fitur pengindeksan di Python memberi keuntungan *best pratice* sebagai berikut. Untuk menampilkan elemen terakhir vektor x, kita dapat memanggil fungsi `x[-1]`. Secara umum, *elemen*

ke k dari belakang akan dapat diakses dengan memanggil `x[-k]`. Perhatikan bahwa ini menguntungkan, karena kita tidak perlu mengingat berapakah panjang vektor yang sedang dibicarakan untuk memperoleh elemen vektor dengan urutan indeks terbalik semacam ini. Perhatikan contoh di bawah ini.

[6]:
```
x = np.array([2, 4, 1, 5])
print("Vektor x =", x)

print("Komponen terakhir dari vektor x adalah", x[-1])
print("Komponen ke-3 dari vektor x dari belakang adalah", x[-3])
```

```
Vektor x = [2 4 1 5]
Komponen terakhir dari vektor x adalah 5
Komponen ke-3 dari vektor x dari belakang adalah 4
```

Pengindeksan matriks juga sama-sama dimulai dari indeks 0. Namun demikian, terdapat 2 indeks yang terlibat ketika kita perlu memanggil elemen dari sebuah matriks. Misalkan

$$A = \begin{bmatrix} 1 & 2 \\ -1 & 9 \\ 7 & -2 \end{bmatrix}.$$

Maka, Python akan 'menomori' vektor tersebut mulai dari indeks 0 sebagai berikut.

i	j	A[i][j]
0	0	1
0	1	2
1	0	-1
1	1	9
2	0	7
2	1	-2

Sebagai contoh, * Elemen matriks A pada baris 2 kolom 1 adalah `A[2][1]=-2`, yaitu yang dicetak merah berikut $A = \begin{bmatrix} 1 & 2 \\ -1 & 9 \\ 7 & -2 \end{bmatrix}$. * Elemen matriks A pada baris 0 kolom 1 adalah `A[0][1]=2`, yaitu yang dicetak merah berikut $A = \begin{bmatrix} 1 & 2 \\ -1 & 9 \\ 7 & -2 \end{bmatrix}$.

[7]:
```
A = np.array([[1, 2], [-1, 9], [7, -2]])
print("Matriks A =\n", A)

# elemen pada baris ke i dan kolom ke j dipanggil dengan perintah A[i][j]
print("Komponen baris ke-2 dan kolom ke-1 dari matriks A adalah", A[2][1])

# dapat juga dipanggil dengan perintah A[i, j]
```

```
print("Komponen baris ke-2 dan kolom ke-1 dari matriks A adalah", A[2, 1])
```

```
Matriks A =
 [[ 1  2]
 [-1  9]
 [ 7 -2]]
Komponen baris ke-2 dan kolom ke-1 dari matriks A adalah -2
Komponen baris ke-2 dan kolom ke-1 dari matriks A adalah -2
```

2.3 Slicing, Subvektor, dan Submatriks

Python menyediakan fitur untuk 'memotong-motong' vektor dan matriks dengan cepat. Hal ini dapat dilakukan dengan dengan menambahkan tanda : pada indeks vektor/matriks yang hendak dipotong. Perhatikan beberapa contoh berikut.

Contoh 1. Mengambil beberapa elemen dari suatu vektor dengan aturan tertentu (Membuat Subvektor).

Misalkan

$$x = \begin{bmatrix} 4 & 2 & 0 & -2 & 3 & 3 & 9 & 12 & 5 \end{bmatrix}$$

dan kita hendak membuat beberapa subvektor baru dari x sebagai berikut.

Nama vektor	Isi vektor	Penjelasan	Perintah Python
y1	[4, 2, 0, -2]	4 elemen pertama dari x	y1 = x[0:4]
y2	[0, -2, 3, 3, 9]	elemen x dari indeks 2 s.d indeks 6	y1 = x[2:7]
y3	[2, -2, 3, 12]	elemen x dari indeks 1, 3, 5, 7	y1 = x[1:8:2]
y4	[5, 12, 9, 3, 3, -2]	elemen x dari indeks terakhir hingga elemen indeks 3	y1 = x[-1:2:-1]

Maka, kita dapat menulis dalam Python sebagai berikut

[8]:
```
x = np.array([4, 2, 0, -2, 3, 3, 9, 12, 5])
print("Vektor x =", x)

y1 = x[0:4]
print("Vektor y1 =", y1, "\n")

y2 = x[2:7]
print("Vektor y2 =", y2, "\n")

y3 = x[1:8:2]
print("Vektor y3 =", y3, "\n")

y4 = x[-1:2:-1]
print("Vektor y4 =", y4, "\n")
```

```
Vektor x = [ 4  2  0 -2  3  3  9 12  5]
```

```
Vektor y1 = [ 4  2  0 -2]

Vektor y2 = [ 0 -2  3  3  9]

Vektor y3 = [ 2 -2  3 12]

Vektor y4 = [ 5 12  9  3  3 -2]
```

Penjelasan:

Perintah `y = x[0:4]` berarti bahwa variabel `y` adalah vektor yang diambil dari `x` yang dimulai dari indeks 0 dan berakhir di indeks 4, namun indeks 4 sendiri tidak termasuk. Dengan kata lain, `y` berisi elemen-elemen ke `0, 1, 2, 3` dari vektor `x` (elemen ke-4 tidak termasuk).

Contoh 2. Membuat submatriks.

Misalkan

$$A = \begin{bmatrix} 4 & 2 & 0 & 7 \\ -2 & 3 & 3 & 0 \\ 9 & 12 & 5 & 6 \end{bmatrix}$$

dan kita hendak membuat beberapa submatriks baru dari `A` sebagai berikut.

$$A_1 = \begin{bmatrix} 4 & 2 & 0 & 7 \\ -2 & 3 & 3 & 0 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 2 & 0 & 7 \\ 3 & 3 & 0 \\ 12 & 5 & 6 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 3 & 3 \\ 12 & 5 \end{bmatrix}$$

Maka, kita dapat menulis dalam Python sebagai berikut

[9]:
```
A = np.array([
    [4, 2, 0, 7],
    [-2, 3, 3, 0],
    [9, 12, 5, 6]
])
print("Matriks A adalah\n", A)

A1 = A[0:2, :]
print("Matriks A1 adalah\n", A1)

A2 = A[:, 1:]
print("Matriks A2 adalah\n", A2)

A3 = A[1:, 1:3]
print("Matriks A3 adalah\n", A3)
```

```
Matriks A adalah
 [[ 4  2  0  7]
 [-2  3  3  0]
 [ 9 12  5  6]]
Matriks A1 adalah
 [[ 4  2  0  7]
 [-2  3  3  0]]
Matriks A2 adalah
 [[ 2  0  7]
 [ 3  3  0]
 [12  5  6]]
Matriks A3 adalah
 [[ 3  3]
 [12  5]]
```

Berikut penjelasan dari perintah-perintah di atas.

Perintah	Penjelasan
`A[0:2, :]`	Mengambil elemen A dari baris 0 s.d 1; dan *semua* kolom
`A[:, 1:]`	Mengambil elemen A dari *semua* baris; dan kolom 1 sampai terakhir
`A[1:, 1:3]`	Mengambil elemen A dari baris 1 sampai terakhir; dan kolom 1 s.d 2

Gunakan matriks dalam file CSV berikut untuk latihan 1:
https://docs.google.com/spreadsheets/d/e/2PACX-1vTsO1o2fddVbz8ZwdIZO0YYhie4trT30AMtg7Uyv8SsyUkG
4KOkHhwwKJ2PaUWKya/pub?gid=0&single=true&output=csv

Latihan 1.

Misalkan matriks dalam file CSV di atas disebut matriks A.
Dengan menggunakan perintah Python, jawablah pertanyaan berikut:

1. Berapakah ukuran matriks A?
2. (Mengasumsikan indeks baris dan kolom dimulai dari 0) Tampilkan elemen pada baris ke 89 dan kolom 123 dari A.
3. Tampilkan elemen pada baris ke 121 dan kolom 75 dari A^2.
4. Tampilkan 5 elemen pertama dari baris terakhir matriks A.
5. Tampilkan 10 elemen terakhir dari kolom ke 163 dari matriks A.
6. Misalkan kita hanya ingin baris ke $0, 3, 6, 9, \ldots$, yaitu semua baris kelipatan 3 dari matriks A untuk disimpan menjadi file CSV baru berjudul "MatKelp3.csv". Tuliskan program untuk mengerjakan hal tersebut.

3 Matriks Blok

Berkebalikan dengan konsep slicing, dengan matriks blok, kita dapat menempel dan menggabungkan matriks-matriks yang lebih kecil untuk membangun matriks yang lebih besar.

Perhatikan contoh berikut. Misalkan

$$A = \begin{bmatrix} 1 & 0 & 2 & 2 \\ 0 & 1 & 2 & 2 \\ 3 & 3 & 1 & 0 \\ 3 & 3 & 0 & 1 \end{bmatrix}$$

Matriks A di atas dapat ditulis sebagai matriks blok sebagai berikut

$$A = \begin{bmatrix} I & 2J \\ 3J & I \end{bmatrix}$$

di mana I dan J secara berturut-turut merupakan matriks identitas dan matriks satu, masing-masing berukuran 2×2.

Matriks A seperti di atas dapat dibuat dengan Python sebagai berikut.

[10]:
```
I = np.eye(2)
print("Matriks I adalah\n", I)

J = np.ones((2,2))
print("Matriks J adalah\n", J)

A = np.block([
    [I, 2*J],
    [3*J, I]
])
print("Matriks A adalah\n", A)
```

```
Matriks I adalah
 [[1. 0.]
 [0. 1.]]
Matriks J adalah
 [[1. 1.]
 [1. 1.]]
Matriks A adalah
 [[1. 0. 2. 2.]
 [0. 1. 2. 2.]
 [3. 3. 1. 0.]
 [3. 3. 0. 1.]]
```

Dengan menggunakan konsep matriks blok seperti di atas, kita dapat memanipulasi matriks dengan lebih leluasa. Perhatikan contoh berikut. Misalkan matriks A seperti di bawah ini.

$$A = \begin{bmatrix} 1 & 0 & 2 & 2 \\ 0 & 1 & 2 & 2 \\ 3 & 3 & 1 & 0 \\ 3 & 3 & 0 & 1 \end{bmatrix}$$

Kemudian kita butuh untuk menyusun ulang matriks A sehingga kolom ke 0 dan 3 diletakkan di sebelah kiri matriks dan kolom ke 2 dan 4 diletakkan di bagian kanannya. Maka kita dapat melakukan perintah Python sebagai berikut.

[11]:
```
A = np.block([
    [I, 2*J],
    [3*J, I]
])

print("Matriks A sebelum manipulasi\n", A)

col0 = A[:, 0:1]
col1 = A[:, 1:2]
col2 = A[:, 2:3]
col3 = A[:, 3:4]

print(col1)

A = np.block([[col0, col2, col1, col3]])

print("Matriks A setelah manipulasi\n", A)
```

```
Matriks A sebelum manipulasi
 [[1. 0. 2. 2.]
 [0. 1. 2. 2.]
 [3. 3. 1. 0.]
 [3. 3. 0. 1.]]
[[0.]
 [1.]
 [3.]
 [3.]]
Matriks A setelah manipulasi
 [[1. 2. 0. 2.]
 [0. 2. 1. 2.]
 [3. 1. 3. 0.]
 [3. 0. 3. 1.]]
```

Latihan 2.

1. Buat matriks C berukuran 100×100 yang memiliki elemen 0 di pinggir dan elemen 1 di bagian tengahnya sebagai berikut:

$$C = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & 1 & \dots & 1 & 1 & 0 \\ 0 & 1 & 1 & \dots & 1 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & 1 & \dots & 1 & 1 & 0 \\ 0 & 1 & 1 & \dots & 1 & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 & 0 \end{bmatrix}$$

2. Pecahlah matriks pada file CSV seperti pada Latihan 1 menjadi 2 matriks. Pertama, matriks X yang hanya berisi baris-baris indeks genap saja. Kedua, matriks Y yang hanya berisi baris-baris indeks ganjil saja. Kemudian gabungkan kedua matriks X dan Y menjadi matriks Z, di mana matriks X berada di bagian atas dan matriks Y di bagian bawah matriks Z. Setelah itu, simpanlah ketiga matriks tersebut dalam bentuk CSV dengan penamaan judul bebas.