線性代數 + Julia + $\LaTeX$ 的學習筆記

Date: 2020-10-01

整個學習過程將以如下「線性代數」課程為主軸學習：

線性代數 台灣大學電機系 蘇柏青

本課程是線性代數的入門課程。線性代數係以「向量空間」(Vector Space)為核心概念之數學工具，擁有極廣泛之應用，非常值得理工商管等科系大學部同學深入修習，作為日後專業應用之基礎。

課程來源：**http://ocw.aca.ntu.edu.tw/ntu-ocw/index.php/ocw/cou/102S207**

學習目標

如下為幾個學習的子目標：

學科

- 線性代數 - 重新學習線性代數，了解重要概念的中文及英文詞彙及應用。

工具

- Julia - 深入學習，了解重要套件的應用及使用。
- Pluto - 隨之成長，作為撰寫學習記錄的工具。
- LaTeX - 隨緣學習，作為撰寫學習記錄的工具。
- Markdown - 隨緣學習，作為撰寫學習記錄的工具。

服務

- GitHub - 學習使用 GitHub 服務，並記錄學習歷程及分享學習內容。

```
md"""
# 線性代數 + Julia + $$\LaTeX$$ 的學習筆記
Date: $_date_

整個學習過程將以如下「線性代數」課程為主軸學習：

### 線性代數 台灣大學電機系 蘇柏青

本課程是線性代數的入門課程。線性代數係以「向量空間」(Vector Space)為核心概念之數學工具，擁有極廣泛之應用，非常值得理工商管等科系大學部同學深入修習，作為日後專業應用之基礎。

課程來源：[http://ocw.aca.ntu.edu.tw/ntu-ocw/index.php/ocw/cou/102S207](http://ocw.aca.ntu.edu.tw/ntu-ocw/index.php/ocw/cou/102S207)

## 學習目標
```

```
如下為幾個學習的子目標：

### 學科
- 線性代數 - 重新學習線性代數，了解重要概念的中文及英文詞彙及應用。

### 工具

- Julia - 深入學習，了解重要套件的應用及使用。
- Pluto - 隨之成長，作為撰寫學習記錄的工具。
- LaTeX - 隨緣學習，作為撰寫學習記錄的工具。
- Markdown - 隨緣學習，作為撰寫學習記錄的工具。

### 服務
- GitHub - 學習使用 GitHub 服務，並記錄學習歷程及分享學習內容。
"""
```

目前進度： 單元 4 · The language of set theory <<<

Date: 2020-10-01

單元 1 · Basic Concepts on Matrices and Vectors

Matrix

$$A = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mn} \end{bmatrix} = [a_{ij}] = M_{mn}$$

Matrix Addition

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} + \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 2 & 3 \\ 4 & 5 \\ 6 & 8 \end{bmatrix}$$

```
3×2 Array{Int64,2}:
 2  3
 4  5
 6  8
```

```
[1 2; 3 4; 5 6]+[1 1; 1 1; 1 2]
```

Scalar Multiplication

$$cA$$

$$3 \cdot \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$$

```
3×2 Array{Int64,2}:
  3   6
  9  12
 15  18
```

```
3 * [1 2; 3 4; 5 6]
```

```
3×2 Array{Int64,2}:
  3   6
  9  12
 15  18
```

```
3 .* [1 2; 3 4; 5 6]
```

Transpose

$$C = \begin{bmatrix} 7 & 9 \\ 18 & 31 \\ 52 & 68 \end{bmatrix} \Rightarrow C^T = \begin{bmatrix} 7 & 18 & 52 \\ 9 & 31 & 68 \end{bmatrix}$$

```
2×3 LinearAlgebra.Adjoint{Int64,Array{Int64,2}}:
 7  18  52
 9  31  68
```

```
let
    C=[7 9; 18 31; 52 68]
    C'
end
```

Vectors

Row Vector:

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

Column Vector:

$$\begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\Downarrow$$

$$\begin{bmatrix} 1 & 2 & 3 & 4 \end{bmatrix}^T$$

The ith componet of $\mathbf{v}$

$$v_i$$

```
1×4 Array{Int64,2}:
 1  2  3  4
```

```
[ 1 2 3 4]
```

```
▶Int64[1, 2, 3, 4]
```

```
[1; 2; 3; 4;]
```

```
4×1 LinearAlgebra.Adjoint{Int64,Array{Int64,2}}:
 1
 2
 3
 4
```

```
[ 1 2 3 4]'
```

Linear Combination

A *linear combination of vectors* $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k$ is a vector of the form

$$c_1\mathbf{u}_1 + c_2\mathbf{u}_2 + \cdots + c_k\mathbf{u}_k$$

where $c_1, c_2, \ldots, c_k$ are scalars. These scalars are called the *coefficients* of the linear combination.

Standard Vectors

The standard vectors of R^n are defined as

$$e_1 = \begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix}, e_2 = \begin{bmatrix}0\\1\\\vdots\\0\end{bmatrix}, \ldots, e_n = \begin{bmatrix}0\\0\\\vdots\\1\end{bmatrix}$$

Matrix-Vector Product

$$Av = v_1a_1 + v_2a_2 + \cdots + v_na_n$$

```
▶Int64[23,  53,  83]
```

```
let
    A=[1 2; 3 4; 5 6]
    v=[7;8]
    A*v
end
```

Identity Matrix

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

Stochastic Matrix

$$A = \begin{bmatrix}0.85 & 0.03\\0.15 & 0.97\end{bmatrix}$$

Slide to set number of **years**: 40

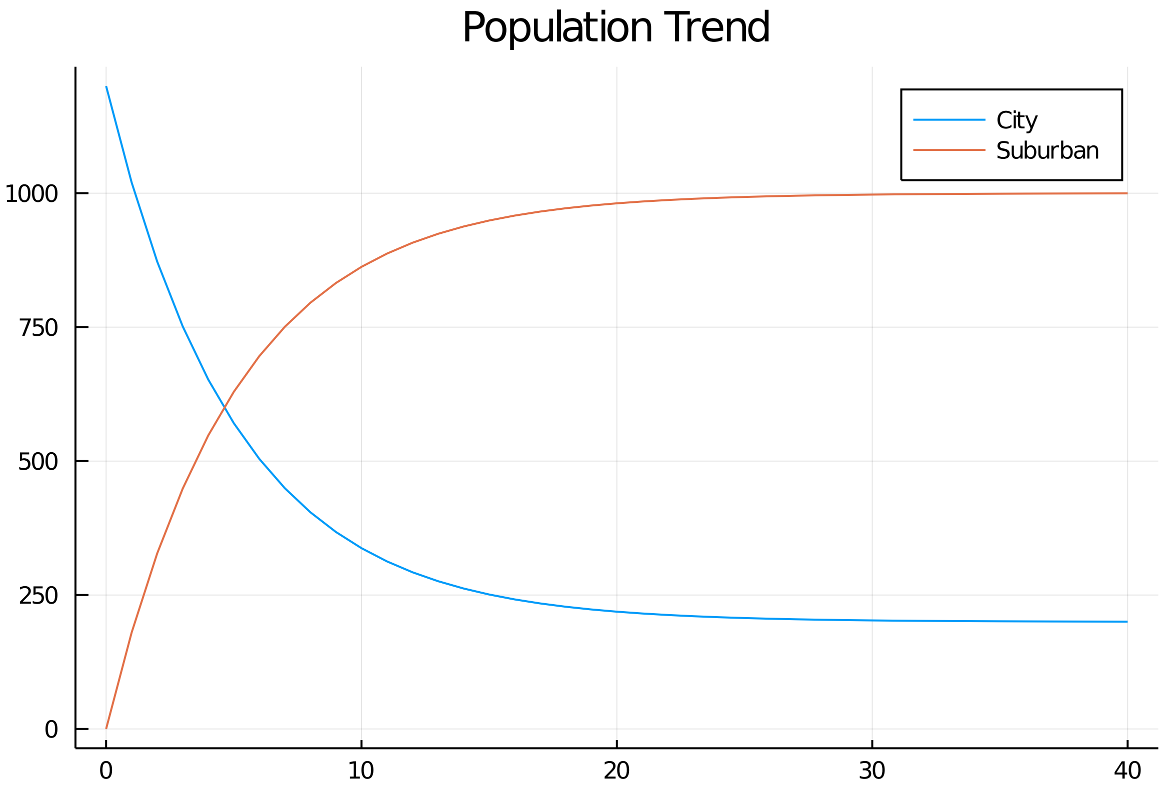
Slide to set population of **city**: 1200

Slide to set population of **suburban**: 0

```
begin
    u01xslider = @bind u01x Slider(1:100; default=40, show_value=true)
    u01cslider = @bind u01c Slider(0:1200; default=1200, show_value=true)
    u01sslider = @bind u01s Slider(0:1200; default=0, show_value=true)
    md"""
    Slide to set number of **years**: $(u01xslider)

    Slide to set population of **city**: $(u01cslider)

    Slide to set population of **suburban**: $(u01sslider)
    """
end
```




```
let
    x=u01x # Number of Years (x)
    pc=u01c # Population of City
    ps=u01s # Population of Suburban
    A=[0.85 0.03; 0.15 0.97]
    #=
    # p0 Population in year 0
    p0=[500; 700]
    p1=A*p0
    p2=A*(p1)
    p3=A*(p2)
    p4=A*(p3)
    p5=A*(p4)
    x=0:5
    Y=hcat(p0, p1, p2, p3, p4, p5)
    plot(x, Y', title = "Population", label = ["City" "Suburban"])
    =#
    p=[pc; ps]
    Y=p
    for i in 1:x
        p=A*p
        Y=hcat(Y, p)
```

```
    end
    plot(0:x, Y', title = "Population Trend", label = ["City" "Suburban"])
end
```

單元 2．System of Linear Equations

System of Linear Equations

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

$$Ax = b$$

Solves Ax = b by (essentially) Gaussian elimination (Julia \ Operator):

$$x = A \setminus b$$

```
▶Float64[-4.0, -5.0, 3.0]
```

```
# Solve System of Linear Equations
let
    A=[1 -2 -1; 3 -6 -5; 2 -1 1]
    b=[3; 3; 0]
    A \ b
end
```

Row Echelon Form & Reduced Row Echelon Form

```
▶Float64[0.403743, -1.21123, 0.112299, 1.48128, 2.0]
```

```
# Solve System of Linear Equations
let
    A=[1 -3 0 2 0; 0 0 1 6 0; 0 0 0 0 1; 0 0 0 0 0]
    b=[7; 9; 2; 0]
    A \ b
end
```

單元 3．Gaussian Elimination

實作參考：

Gaussian-elimination.pdf

Numerical Analysis by Julia Series 1 — Gauss Elimination | by Treee July | Medium

對列及行的參照：

```
3×3 Array{Int64,2}:
 1  2  3
 4  5  6
 7  8  9
```

```
let
    A=[ 1 2 3; 4 5 6; 7 8 9]
end
```

```
Array{Any}((7,))
  1: String "A: [1 2 3; 4 5 6; 7 8 9]"
  2: String "A[1, 1]: 1"
  3: String "A[end, end]: 9"
  4: String "r1: [1, 2, 3]"
  5: String "∑Ai: [12, 15, 18]"
  6: String "c1: [1, 4, 7]"
  7: String "∑Aj: [6, 15, 24]"
```

```
let
    o=[]
    # Matrix
    A=[ 1 2 3; 4 5 6; 7 8 9]
    push!(o, @sprintf("A: %s", A))
    # Elements
    push!(o, @sprintf("A[1, 1]: %s", A[1, 1]))
    push!(o, @sprintf("A[end, end]: %s", A[end, end]))
    # Row
    r1=A[1,:]
    push!(o, @sprintf("r1: %s", r1))
    ∑Ai=A[1,:]+A[2,:]+A[3,:]
    push!(o, @sprintf("∑Ai: %s", ∑Ai))
    # Column
    c1=A[:,1]
    push!(o, @sprintf("c1: %s", c1))
    ∑Aj=A[:,1]+A[:,2]+A[:,3]
    push!(o, @sprintf("∑Aj: %s", ∑Aj))
    with_terminal(dump, o)
end
```

```
👍 選定之輸出方案： 1) 容易以 do ... end 區塊包裝 3) 轉貼程式碼到他處不用修改

2020年10月 1日 週四 16時53分56秒 CST
A: Array{Int64}((3, 3)) [1 2 3; 4 5 6; 7 8 9]
A[1, 1]: Int64 1
A[end, end]: Int64 9
r1: Array{Int64}((3,)) [1, 2, 3]
∑Ai: Array{Int64}((3,)) [12, 15, 18]
c1: Array{Int64}((3,)) [1, 4, 7]
∑Aj: Array{Int64}((3,)) [6, 15, 24]

      10月 2020
日 一 二 三 四 五 六
             1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
```

```
let
    with_terminal() do
        println("👍 選定之輸出方案： 1) 容易以 do ... end 區塊包裝 3) 轉貼程式碼到他處不用修改\n")
        # Get Current Time
        command=`date`
        run(command)
        # Matrix
        A=[ 1 2 3; 4 5 6; 7 8 9]
        print("A: "); dump(A)
        # Elements
        print("A[1, 1]: "); dump(A[1, 1])
        print("A[end, end]: "); dump(A[end, end])
        # Row
        r1=A[1,:]
        print("r1: ");  dump(r1)
        ∑Ai=A[1,:]+A[2,:]+A[3,:]
        print("∑Ai: "); dump(∑Ai)
        # Column
        c1=A[:,1]
        print("c1: ");  dump(c1)
        ∑Aj=A[:,1]+A[:,2]+A[:,3]
        print("∑Aj: "); dump(∑Aj)
        println()
        run(`cal -h`)
    end
end
```

```
A: Array{Int64}((3, 3)) [1 2 3; 4 5 6; 7 8 9]
A[1, 1]: Int64 1
A[end, end]: Int64 9
r1: Array{Int64}((3,)) [1, 2, 3]
∑Ai: Array{Int64}((3,)) [12, 15, 18]
c1: Array{Int64}((3,)) [1, 4, 7]
∑Aj: Array{Int64}((3,)) [6, 15, 24]
```

```
let
    Text() do io
        # Matrix
        A=[ 1 2 3; 4 5 6; 7 8 9]
        print(io, "A: "); dump(io, A)
        # Elements
        print(io, "A[1, 1]: "); dump(io, A[1, 1])
        print(io, "A[end, end]: "); dump(io, A[end, end])
        # Row
        r1=A[1,:]
        print(io, "r1: ");  dump(io, r1)
        ∑Ai=A[1,:]+A[2,:]+A[3,:]
        print(io, "∑Ai: "); dump(io, ∑Ai)
        # Column
        c1=A[:,1]
        print(io, "c1: ");  dump(io, c1)
        ∑Aj=A[:,1]+A[:,2]+A[:,3]
        print(io, "∑Aj: "); dump(io, ∑Aj)
    end
end
```

A: [1 2 3; 4 5 6; 7 8 9]

A[1, 1]: 1

A[end, end]: 9

r1: [1, 2, 3]

ΣAi: [12, 15, 18]

c1: [1, 4, 7]

ΣAj: [6, 15, 24]

```
let
        # Matrix
        A=[ 1 2 3; 4 5 6; 7 8 9]
        # Row
        r1=A[1,:]
        ∑Ai=A[1,:]+A[2,:]+A[3,:]
        # Column
        c1=A[:,1]
        ∑Aj=A[:,1]+A[:,2]+A[:,3]
    md"""
    A: $(Text(A))

    A[1, 1]: $(Text(A[1, 1]))

    A[end, end]: $(Text(A[end, end]))

    r1: $(Text(r1))

    ∑Ai: $(Text(∑Ai))

    c1: $(Text(c1))

    ∑Aj: $(Text(∑Aj))
    """
end
```

```
Enter cell code...
```

```
Enter cell code...
```

```
Enter cell code...
```

```
Enter cell code...
```

```
Enter cell code...
```

```
Enter cell code...
```

單元 4 · The language of set theory

Subset

$$Let\ S_1 = \{a, b, c, d, e\},\ S_2 = a, b, e$$

$$S_2 \subset S_1\ means$$

$$\forall x \in S_2,\ x\ is\ also \in S_1.$$

```
🖐 Give Me Five, 原來集合在 Julia 裏頭是長這様子喔！☯
Set{String}
  dict: Dict{String,Nothing}
    slots: Array{UInt8}((16,)) UInt8[0x00, 0x00, 0x00, 0x01, 0x01, 0x00, 0x00, 0x00, 0x0
1, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x01]
    keys: Array{String}((16,))
      1: #undef
      2: #undef
      3: #undef
      4: String "c"
      5: String "e"
      ...
      12: #undef
      13: #undef
      14: String "a"
      15: #undef
      16: String "d"
    vals: Array{Nothing}((16,))
      1: Nothing nothing
      2: Nothing nothing
      3: Nothing nothing
      4: Nothing nothing
      5: Nothing nothing
      ...
      12: Nothing nothing
      13: Nothing nothing
      14: Nothing nothing
      15: Nothing nothing
      16: Nothing nothing
    ndel: Int64 0
    count: Int64 5
    age: UInt64 0x0000000000000005
    idxfloor: Int64 1
    maxprobe: Int64 0
Set(["c", "e", "b", "a", "d"])
Set(["e", "b", "a"])
true
Set(["c", "e", "b", "a", "d"])
Set(["e", "b", "a"])
Set(["c", "d"])
Set{String}()
```

```julia
let
    with_terminal() do
        s1=Set(["a", "b", "c", "d", "e"])
        println("🖐 Give Me Five, 原來集合在 Julia 裏頭是長這様子喔！☯")
        dump(s1)
        println(s1)
        s2=Set(["a", "b", "e"])
        println(s2)
        # subset
        println(⊆(s2, s1))
        # union set
        println(∪(s1, s2))
        # intersection set
        println(∩(s1, s2))
        # difference set
        println(setdiff(s1, s2))
        println(setdiff(s2, s1))
    end
end
```

<<<

```
md"""
### <<<
"""
```

參考資料

Linear Algebra

[ ] 線性代數 - 臺大開放式課程 (NTU OpenCourseWare)

Julia

[ ] Introduction to Julia

[ ] Advanced topics

[ ] Julia for Data Science

[ ] 18.S191 Introduction to Computational Thinking

Unicode Input · The Julia Language

Pluto

Docstrings · PlutoUI.jl

$\LaTeX$

LaTeX syntax · Documenter.jl

LaTeX - Mathematical Python

LaTeX help 1.1 - Table of Contents

List of mathematical symbols - Wikiwand

Markdown

Markdown Cheatsheet · adam-p/markdown-here Wiki

Markdown · The Julia Language

GitHub

[ ] Hello World · GitHub Guides

其他

三度辭典網 > 術語中英雙語詞典

```
md"""
## 參考資料
### Linear Algebra

[ ] [線性代數 - 臺大開放式課程 (NTU OpenCourseWare)](http://ocw.aca.ntu.edu.tw/ntu-ocw/index.php/ocw/cou/102S207/3)

### Julia

[ ] [Introduction to Julia](https://juliaacademy.com/courses/enrolled/375479)

[ ] Advanced topics

[ ] [Julia for Data Science](https://juliaacademy.com/courses/enrolled/937702)

[ ] [18.S191 Introduction to Computational Thinking](https://computationalthinking.mit.edu/Fall20/)

[Unicode Input · The Julia Language](https://docs.julialang.org/en/v1/manual/unicode-input/)

### Pluto
[Docstrings · PlutoUI.jl](https://juliahub.com/docs/PlutoUI/abXFp/0.6.3/autodocs/)

### $$\LaTeX$$

[LaTeX syntax · Documenter.jl](https://juliadocs.github.io/Documenter.jl/v0.7/man/latex.html)

[LaTeX - Mathematical Python](https://www.math.ubc.ca/~pwalls/math-python/jupyter/latex/)

[LaTeX help 1.1 - Table of Contents](http://www.emerson.emory.edu/services/latex/latex_toc.html)

[List of mathematical symbols - Wikiwand](https://www.wikiwand.com/en/List_of_mathematical_symbols)

### Markdown
[Markdown Cheatsheet · adam-p/markdown-here Wiki](https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet)

[Markdown · The Julia Language](https://docs.julialang.org/en/v1/stdlib/Markdown/)

### GitHub

[ ] [Hello World · GitHub Guides](https://guides.github.com/activities/hello-world/)

### 其他
[三度辭典網 > 術語中英雙語詞典](https://www.3du.tw/term/)
"""
```