

CS381: Numerical Computation & Softwares

Introduction to Julia programming

Jakramate Bootkrajang

Department of Computer Science

Chiang Mai University

Rev.2 – 2019

The logo for the Julia programming language. It features the word "julia" in a bold, black, lowercase sans-serif font. The letter 'j' has a blue circle above it. The letter 'i' has a red circle above it. The letter 'l' has a purple circle above it. The letter 'a' has a green circle above it. The circles are arranged in a slightly curved line above the letters.

julia

- เป็นภาษาโปรแกรมมิ่งที่ถูกพัฒนาขึ้นมาใหม่ เพื่อรองรับการคำนวณทางวิทยาศาสตร์
- การคำนวณที่ต้องเกี่ยวข้องกับ vector และ matrix สามารถทำได้โดยสะดวก
- เริ่มพัฒนาโดยกลุ่มของนักพัฒนาจากมหาวิทยาลัย MIT
- super fast.
- รองรับภาษาอื่นนอกจากภาษาอังกฤษ (Support unicode characters)
 - ▶ แปลว่าเราสามารถตั้งชื่อตัวแปรเป็นภาษาไทยได้

ถูกนำไปใช้งานในสาย Data science มากมาย

- Finance and Economics <https://lectures.quantecon.org/jl/>
- Machine learning <https://github.com/JuliaML>
- Statistics <https://github.com/JuliaStats>
- Bioinformatics <https://github.com/BioJulia>
- Astronomy <https://github.com/JuliaAstro>

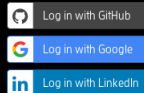
การติดตั้ง Julia

สามารถทำได้สองวิธีคือ

- การติดตั้งบนเครื่องของตนเอง
 - ▶ สามารถ download โปรแกรมแปลภาษา Julia ได้จาก <http://www.julialang.org>
 - ▶ การใช้งานแบบนี้มีความยืดหยุ่นสูงและตอบสนองการใช้งานเร็วกว่า
- การใช้งาน Julia ผ่านเว็บ
 - ▶ www.juliabox.com
 - ▶ จำเป็นต้องมี account ของ google หรือ LinkedIn หรือ Github
 - ▶ ใช้งานได้ทันทีโดยไม่ต้องติดตั้งโปรแกรมบนเครื่องตนเอง

Julia BOX

Run Julia in your
Browser



Jupyter Notebook
Interface



75,000+ users served
since 2015



Free registration,
free to use



Perfect for classes,
students, professors and
new Julia users



Includes 275+ carefully
curated popular Julia
packages



Multi-node deployment
capability



Parallel computing
capability



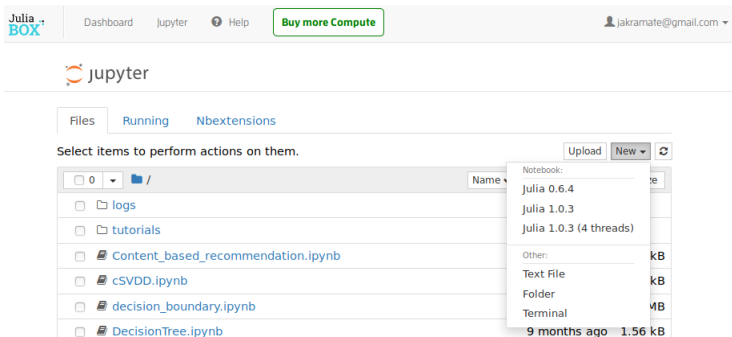
Buy added memory,
storage, nodes and
enterprise support.

- กดปุ่ม Launch เพื่อเริ่มใช้งาน

The screenshot shows the JuliaBox dashboard. At the top left, there is a navigation bar with the Julia logo and the text 'BOX'. To the right of the logo are links for 'Dashboard', 'Jupyter', and 'Help'. Further right is a green button labeled 'Buy more Compute'. On the far right of the navigation bar is a user profile icon and the email address 'jakramate@gmail.com'. Below the navigation bar is a large central area containing the Julia logo (two orange semi-circles with four black dots) and three buttons: a blue 'Launch' button, a white 'Launch with GPU' button, and a white 'Customize' button. Below this central area is a horizontal line, and underneath it are four smaller boxes, each with an icon and a label: 'Packages (0.6)' with a cardboard box icon, 'Git' with a red diamond icon, 'Google Drive' with the Google Drive logo, and 'My apps' with a colorful icon.

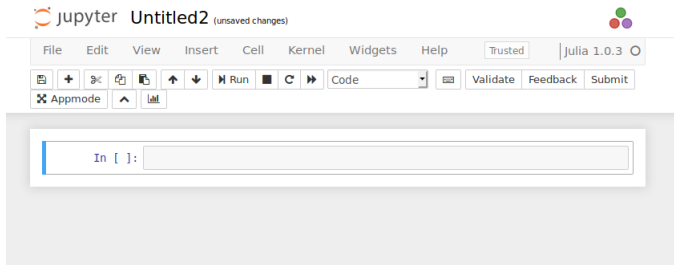
เริ่มสร้างไฟล์ใหม่

- สามารถทำได้โดยกด New -> แล้วเลือกสร้างไฟล์ Julia เวอร์ชัน 1.0.3

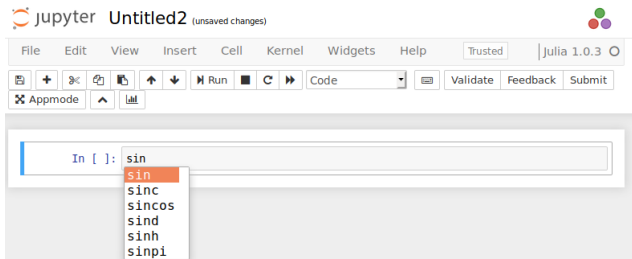


The screenshot shows the JupyterLab interface. At the top, there is a navigation bar with 'Julia BOX', 'Dashboard', 'Jupyter', 'Help', and a 'Buy more Compute' button. The user's email 'jakramate@gmail.com' is visible on the right. Below the navigation bar is the 'jupyter' logo. The main area has tabs for 'Files', 'Running', and 'Nbextensions'. A message says 'Select items to perform actions on them.' Below this is a file browser showing a directory structure with folders 'logs' and 'tutorials', and files 'Content_based_recommendation.ipynb', 'cSVDD.ipynb', 'decision_boundary.ipynb', and 'DecisionTree.ipynb'. A 'New' dropdown menu is open, showing options for 'Notebook' (Julia 0.6.4, Julia 1.0.3, Julia 1.0.3 (4 threads)), 'Other' (Text File, Folder, Terminal), and a file named '9 months ago' with a size of 1.56 kB.

การใช้งานทั่วไป



- พิมพ์คำสั่งที่ต้องการลงในช่อง Input cell
- หากมีหลายคำสั่งสามารถกด Enter เพื่อใส่คำสั่งเพิ่มเติมในบรรทัดใหม่
- หากต้องการประมวลผลคำสั่งทั้งหมดที่ใส่ลงไป ให้กด Ctrl+Enter ผลลัพธ์จะแสดงด้านล่าง



- ระหว่างพิมพ์คำสั่งสามารถกด Tab เพื่อให้ Julia แสดงคำสั่งที่ขึ้นต้นด้วยสิ่งที่พิมพ์ค้างไว้

Primitive data types

- Number:
 - ▶ Integers, Floating points
 - ▶ Special numbers: **NaN** (Not a number), **Inf** (Infinity)
- Array:
- String
- Note: Julia is a case-sensitive language.

- สร้าง array โดยใช้ [] ครอบตัวเลข
 - ▶ $X = [1, 2, 3, 4]$ สร้าง array 1 มิติ
 - ▶ $Y = [1\ 2\ 3\ 4]$ สร้าง array 2 มิติ (1x4 matrix)
 - ▶ $Z = [1\ 2; 3\ 4]$ สร้าง array 2 มิติ (2x2 matrix)
- สามารถเรียกดูมิติของ array โดยใช้ฟังก์ชัน `size()`

Useful array creating functions

- `ones(n,m)`, `zeros(n,m)` สร้างเมทริกซ์ขนาด $n \times m$ ที่มีสมาชิกเป็น 1 หรือ 0 ทั้งหมด
- `fill(k, n, m)` สร้างเมทริกซ์ขนาด $n \times m$ ที่มีสมาชิกทุกตัวมีค่าเท่ากับ k
- `randn(n,m)` สร้างเมทริกซ์ขนาด $n \times m$ ที่มีสมาชิกทุกตัวสุ่มจาก normal distribution

Array indexing

- Index ใน Julia เริ่มจากเลข 1
- `a = randn(4, 4)`
- `a[1, 1]` คือการดึงสมาชิกแถวที่ 1 คอลัมน์ 1
- `a[1, :]` คือการดึงสมาชิกทุกตัวของแถวแรก
- `a[:, 1]` คือการดึงสมาชิกทุกตัวของหลักแรก

Functions on array

$x = [-1 \ 0 \ 1]$

ฟังก์ชัน	ค่าที่ให้	ตัวอย่าง
<code>sum(x)</code>	หาผลบวกของสมาชิกในอะเรย์	<code>y = sum(x)</code>
<code>mean(x)</code>	หาค่าเฉลี่ยของสมาชิกในอะเรย์	<code>m = mean(x)</code>
<code>std(x)</code>	หา s.d. ของสมาชิกในอะเรย์	<code>t = std(x)</code>
<code>var(x)</code>	หา variance ของสมาชิกในอะเรย์	<code>t = var(x)</code>
<code>maximum(x)</code>	หาค่าสูงสุดของอะเรย์	<code>t = maximum(x)</code>
<code>minimum(x)</code>	หาค่าต่ำสุดของอะเรย์	<code>z = minimum(x)</code>
<code>sort(x)</code>	จัดเรียงค่าของสมาชิกในอะเรย์ x	<code>z = sort(x)</code>

- สามารถสร้างลำดับของตัวเลขได้โดย start:stop
 - ▶ ตัวอย่างเช่น 1:10 จะทำการสร้างลำดับเริ่มต้นตั้งแต่ 1 ไปถึง 10
- สามารถกำหนดการเพิ่มของเลขโดยระบุ step เพิ่ม
 - ▶ ตัวอย่างเช่น 1:0.5:10 สร้างลำดับ 1,1.5,2,2.5,...,10
- สามารถเรียกดูลำดับได้โดยใช้ฟังก์ชัน collect()
 - ▶ `collect(1:10)`

Basic operations

สัญลักษณ์	หน้าที่	ตัวอย่าง
=	กำหนดค่าตัวแปร	$y = 5$
+	บวก	$y = x + 1$
-	ลบ	$z = m - 1$
*	คูณ	$k = 2 * 2$
/	หาร	$n = t / 10$
^	ยกกำลัง	$y = x^4$

หมายเหตุ จุดตามด้วยสัญลักษณ์ ใช้กระทำการบนเมทริกซ์แบบตัวต่อตัว

Element-wise operation

- ใช้ `.` นำหน้า operator ทางคณิตศาสตร์
- `ones(2, 2) * ones(2, 2)` คือ Matrix multiplication
- `ones(2, 2) .* ones(2, 2)` คือ Element by element multiplication

Basic maths function

ฟังก์ชัน	ค่าที่ให้	ตัวอย่าง
<code>sqrt(x)</code>	รากที่สองของ x	<code>y = sqrt(x+5)</code>
<code>abs(x)</code>	ค่าสัมบูรณ์ของ x	<code>d = abs(x) * y</code>
<code>sin(x)</code>	ค่าไซน์ของ x (เรเดียน)	<code>t = x + sin(x)</code>
<code>asin(x)</code>	ค่า arcsin ของ x (เรเดียน)	<code>t = x + asin(x)</code>
<code>sind(x)</code>	ค่าไซน์ของ x (degree)	<code>t = x + sind(x)</code>
<code>log(x)</code>	ค่า natural log ของ x	<code>z = log(1+x)</code>
<code>log10(x)</code>	ค่า log ฐาน 10 ของ x	<code>z = log10(1-2*x)</code>
<code>exp(x)</code>	ค่า exponential ของ x (e^x)	<code>p = .7 * exp(x)</code>

Try-out: `result = exp(-2.33) * cos(22 * 180 / pi)`

Boolean expressions

- Expression ที่ประมวลผลเป็น จริง หรือ เท็จ (true / false)

ฟังก์ชัน	ค่าที่ให้	ตัวอย่าง
==	เป็นจริงเมื่อค่าสองค่าเท่ากัน x	y == 5
!=	เป็นจริงเมื่อค่าสองค่าไม่เท่ากัน	7 != 7
<=	เป็นจริงเมื่อค่าด้านซ้ายน้อยกว่าหรือเท่ากับค่าด้านขวา	p <= 40
>=	เป็นจริงเมื่อค่าด้านซ้ายมากกว่าหรือเท่ากับค่าด้านขวา	p >= 2
<	เป็นจริงเมื่อค่าด้านซ้ายน้อยกว่าค่าด้านขวา	x < 5
>	เป็นจริงเมื่อค่าด้านซ้ายมากกว่าค่าด้านขวา	x+y+z > 16
&&	คำเชื่อม และ	x >= 5 && x <= 10
	คำเชื่อม หรือ	x <= 5 x >= 10
!	นิเสธ	!(5 == 5)

Control structure: if

การเลือกทำโดยใช้ if...else

```
if (age >= 18)
    print("You can drive")
else
    print("Sorry you are too young")
end
```

สังเกตว่าจะต้องปิดด้วย end

Control structure: if...elseif

การเลือกทำโดยใช้ if...elseif

```
if (age >= 18)
    print("You can drive")
elseif (age >= 15)
    print("You can ride a motorcycle")
else
    println("Sorry you are too young")
end
```

สังเกตว่าจะต้องปิดด้วย end

Control structure: for

การทำซ้ำโดยใช้ for loop รู้จำนวนรอบในการทำซ้ำที่แน่นอน

```
for i in 1:10      # loop over some range
    println(e^i)   # print with newline
end
```

```
A = [1 3 5 7 9]   # loop over an array
for j in A
    print(j%2)     # print without newline
end
```

สังเกตว่าจะต้องปิดด้วย end

Control structure: while

การทำซ้ำแบบ while ไม่รู้จำนวนรอบที่แน่นอน ทำจนกว่า condition จะเป็นเท็จ

```
while (condition)
    some statements
end
```

```
p = 10 # initialise condition variable
while (p > 0)
    println(p/10)
    p = p - 1 # update condition variable
end
```


- กลุ่มของคำสั่งที่ถูกผูกกันไว้ พร้อมกับถูกตั้งชื่อ
- Useful functions
 - ▶ `print("put this on the screen")`
 - ▶ `println("with new line")`
 - ▶ `readline()` รับข้อมูลจาก keyboard
 - ▶ `typeof()` เรียกดูชนิดของตัวแปร

```
# inline function definition
```

```
f(x) = x3
```

```
df(x) = 3*x2
```

```
# more common function definition
```

```
function cube(x)
```

```
    y = x3           # compute the cube of x
```

```
    return y         # return the result
```

```
end
```

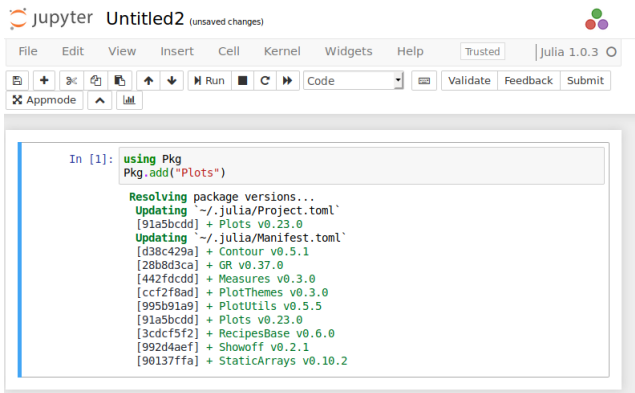
Packages

- กลุ่มของฟังก์ชันที่ใช้ในการประมวลข้อมูลเฉพาะทาง
- Julia มี package มากมายสามารถดูข้อมูลได้จาก
 - ▶ Julia package listing — <https://pkg.julialang.org/>
- การติดตั้ง Package (สำหรับ Julia version 1.0 ขึ้นไป) ใช้คำสั่ง

```
using Pkg

Pkg.add("packageName")
```
- ก่อนใช้งาน Package ต้องเรียกคำสั่ง `using packageName`

Packages (ติดตั้งบน Juliabox)



The screenshot shows a Jupyter Notebook window titled "Untitled2 (unsaved changes)" with the Julia 1.0.3 kernel. The code cell contains the following commands and their output:

```
In [1]: using Pkg
        Pkg.add("Plots")

Resolving package versions...
Updating ~/julia/Project.toml`
 [91a5bcd] + Plots v0.23.0
Updating ~/julia/Manifest.toml`
 [d38c429a] + Contour v0.5.1
 [28b8d3ca] + GR v0.37.0
 [442fcd] + Measures v0.3.0
 [ccf2f8ad] + PlotThemes v0.3.0
 [995b91a9] + PlotUtils v0.5.5
 [91a5bcd] + Plots v0.23.0
 [3cdcf5f2] + RecipesBase v0.6.0
 [992d4aef] + Showoff v0.2.1
 [90137ffa] + StaticArrays v0.10.2
```

Useful package: Plots

- การสร้างกราฟใน Julia 1.0 ขึ้นไปจะผ่าน package ที่ชื่อว่า Plots
- Plots ใช้งานง่าย รองรับการวาดกราฟหลายประเภท
- หาข้อมูลเพิ่มเติมได้จาก <http://docs.juliaplots.org/latest/>

การเรียกใช้งาน Plots

In [2]: `using Plots`

```
In [4]: x = 1:10
        y1 = x.^2
        y2 = x.^3

        plot(x, y1, title="my graph", xlabel="x", ylabel="y")
        plot!(x, y2)
```

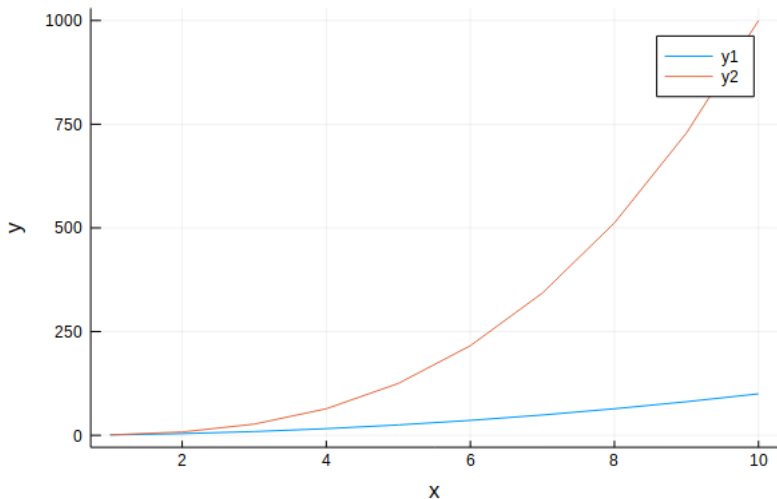
Plotting function

- `plot()` : ใช้พล็อตเส้น โดยมี parameter อย่างน้อย 2 ตัว คือ เวกเตอร์ x , เวกเตอร์ y
- `plot!` : ใช้พล็อตเส้นทับบนกราฟเดิม โดยมี parameter อย่างน้อย 2 ตัว คือ เวกเตอร์ x , เวกเตอร์ y
- สามารถกำหนด keyword argument ให้ `plot()` เพิ่มได้ เช่น
 - ▶ `plot(title="ชื่อกราฟ", xlabel="ชื่อแกน x", ylabel="ชื่อแกน y")`

The result

Out[4]:

my graph



Must read

- Learn X in Y minutes for Julia

`https://learnxinyminutes.com/docs/julia/`

- Plots in Julia

`http://docs.juliaplots.org/latest/`