



# Python Meets Mathematics:

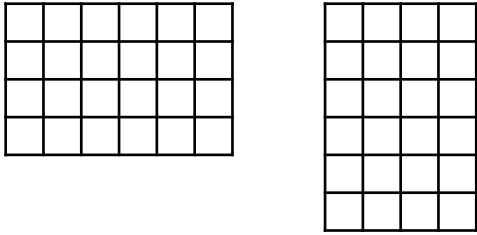
## Calculus

Sunglok Choi, Assistant Professor, Ph.D.  
Computer Science and Engineering Department, SEOULTECH  
[sunglok@seoultech.ac.kr](mailto:sunglok@seoultech.ac.kr) | <https://mint-lab.github.io/>

# Building Mathematical Intuition

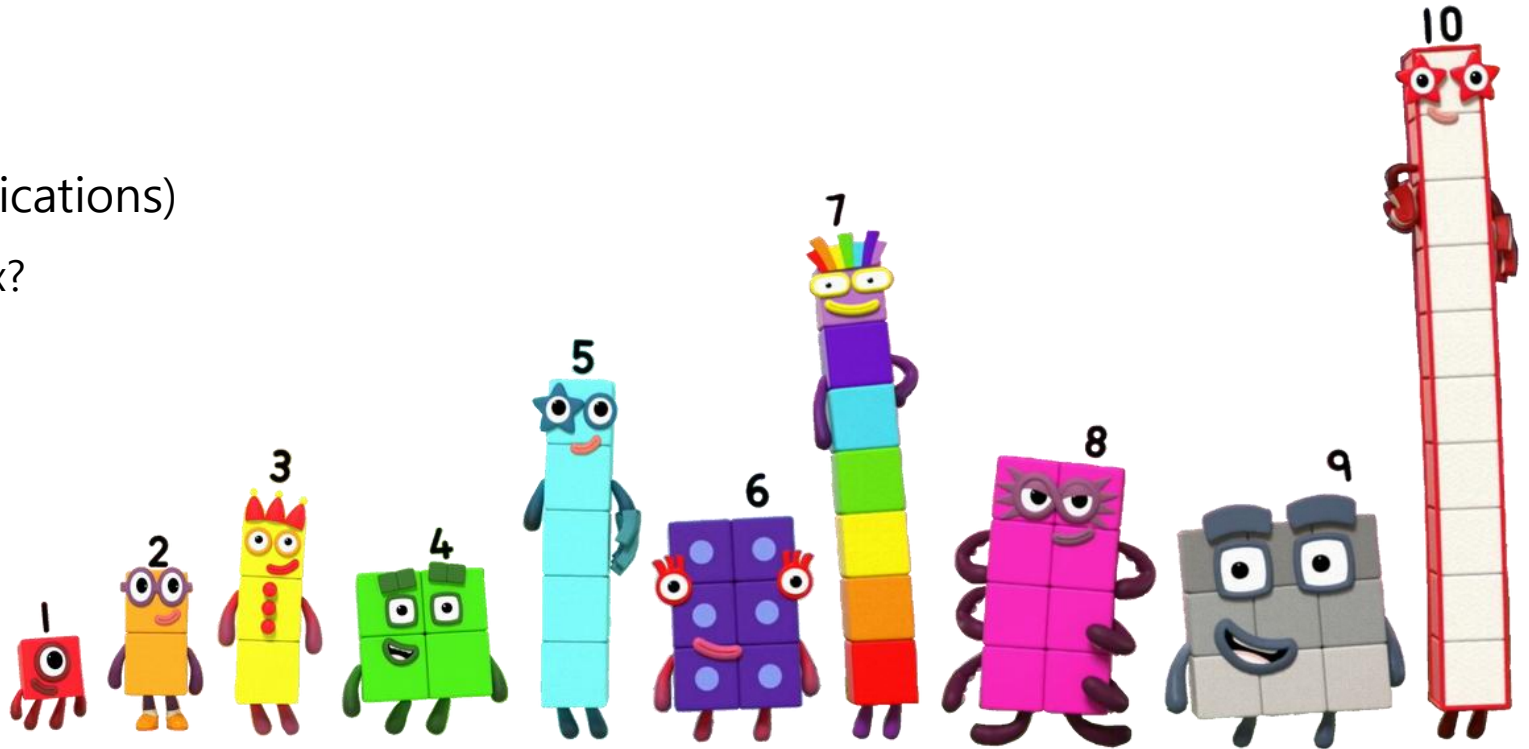
## 1. Imagine (or visualize)

- e.g. *Numberblocks* (by BBC) [\[YouTube\]](#)
- e.g. Why  $4 \times 6 == 6 \times 4$ ? (for kids without knowledge of a [multiplication table](#) and [commutativity](#))



## 2. Ask why (or think about their applications)

- e.g. Why did I learn about a matrix?



# Tools

- [SciPy](#)

- A Python-based open-source ecosystem for mathematics, science, and engineering.
  - An open-source computing tool as an alternative to commercial software such as [MATLAB](#)
  - Included in [Anaconda](#) by default
- Major components



NumPy  
Base N-dimensional  
array package



SciPy library  
Fundamental library for  
scientific computing



Matplotlib  
Comprehensive 2-D  
plotting



IPython  
Enhanced interactive  
console



SymPy  
Symbolic mathematics



pandas  
Data structures &  
analysis

- Online references: [MINT Lab's Know-where](#)

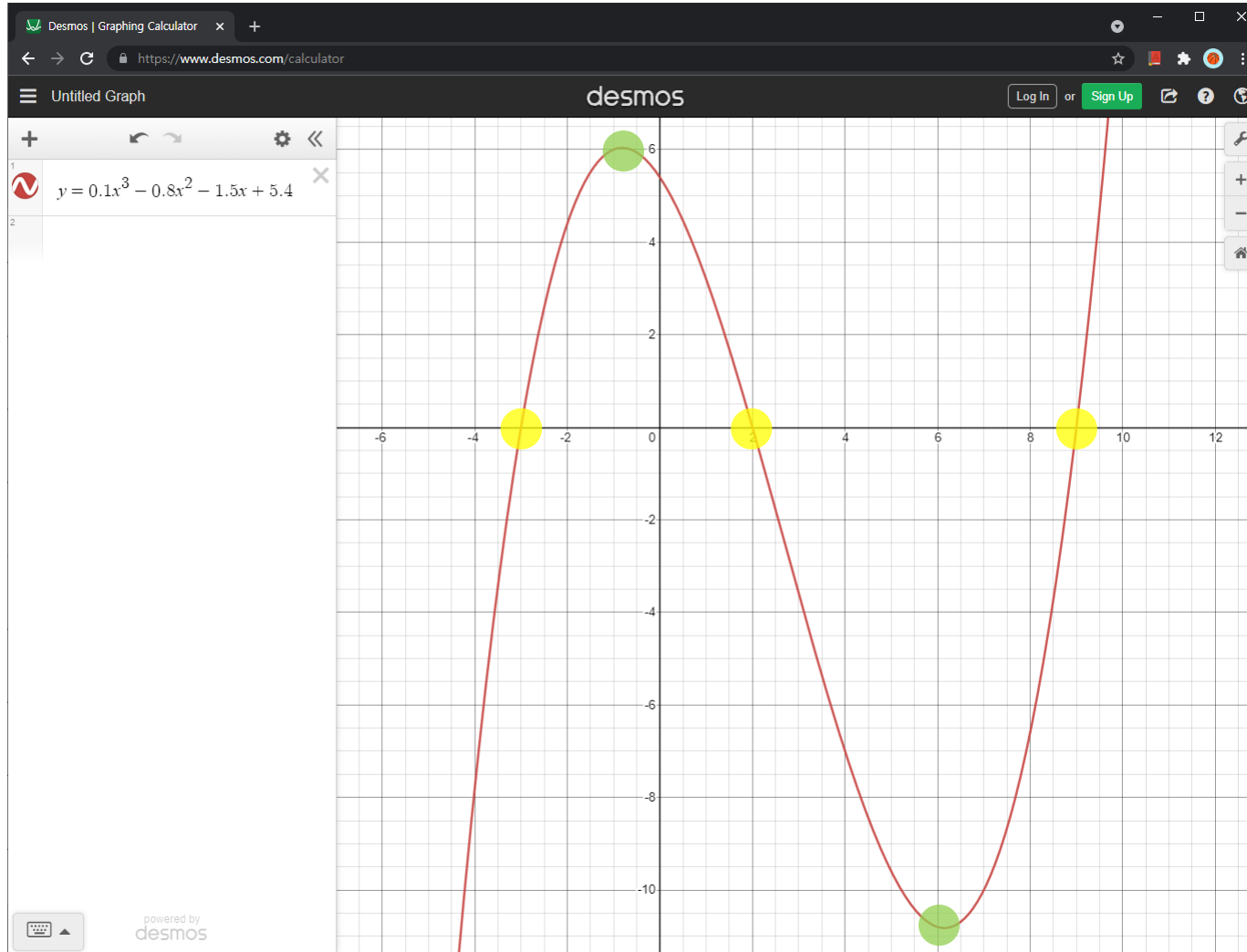
- Please refer *Programming (Python)* category, especially about *P3. NumPy, SciPy, and Matplotlib* section.

# ~~Programming~~ Python with SciPy Meets Mathematics

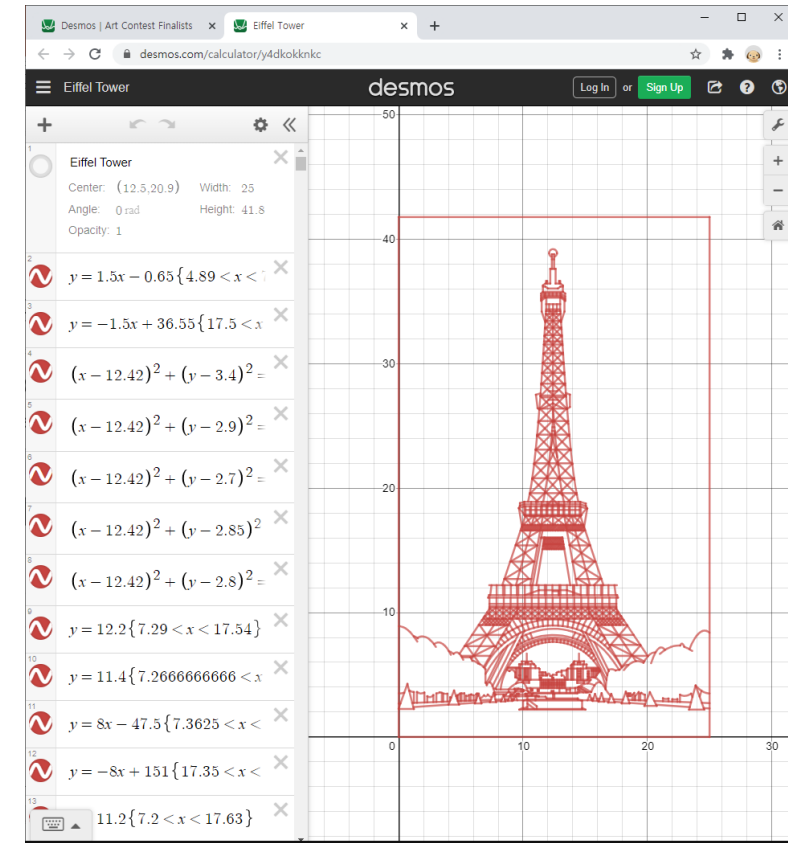
- Calculus
- Linear Algebra
- Optimization
- Probability
- Information Theory

# Getting Started with Drawing an Equation

- Example) Drawing  $y = 0.1x^3 - 0.8x^2 - 1.5x + 5.4$
- Visualization with [Desmos Graphing Calculator](https://www.desmos.com/calculator)



## [Desmos Global Math Art Contest](#) (Ages 13-14)



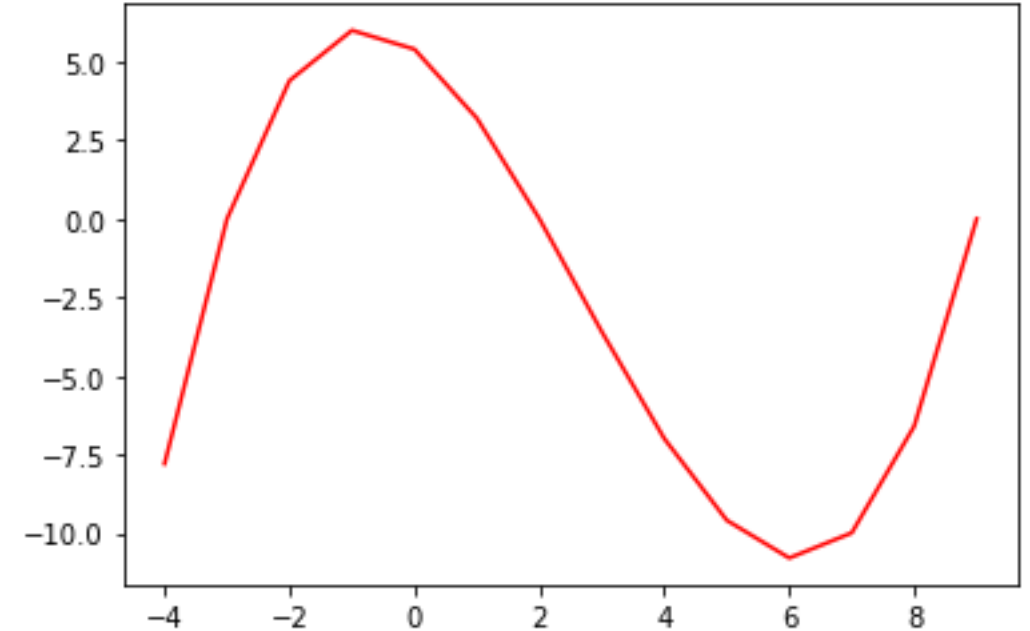
# Getting Started with Drawing an Equation

- Example) Drawing  $y = 0.1x^3 - 0.8x^2 - 1.5x + 5.4$
- Visualization with [Matplotlib](#)

```
import matplotlib.pyplot as plt
```

```
xs = [x for x in range(-4, 10)]  
ys = [0.1*x**3 - 0.8*x**2 - 1.5*x + 5.4 for x in xs]
```

```
plt.plot(xs, ys, 'r-')  
plt.show()
```

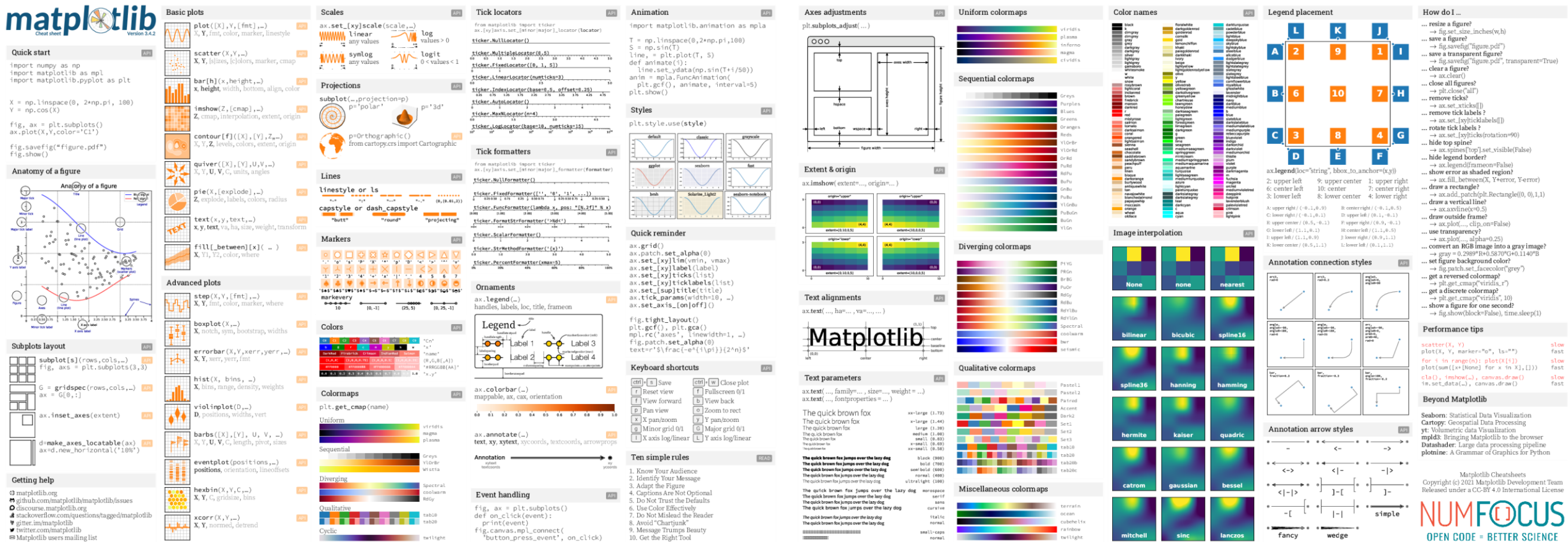


Further improvement)

- How to make the graph smooth?
- How to draw the grid?
- How to display labels on X and Y axes?
- How to make its aspect ratio equal?

# Matplotlib: A Plotting Library in Python

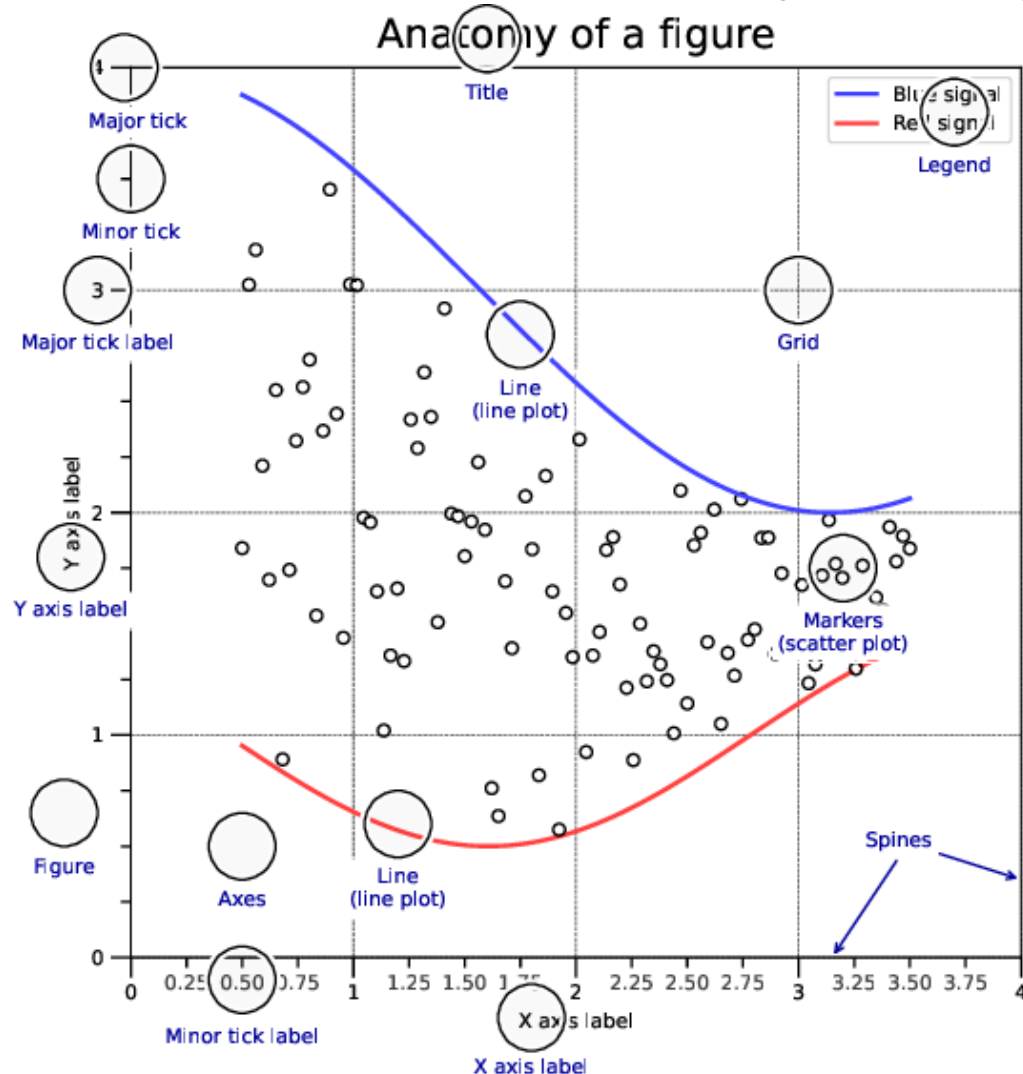
- [Matplotlib](#) is a plotting library for creating static, animated, and interactive visualization in Python.
- References: [Documentation](#), [Examples](#) (Gallery), [Tutorials](#), and [Cheatsheets](#) (made by Matplotlib)





# Matplotlib: A Plotting Library in Python

- [Matplotlib](#) is a plotting library for creating static, animated, and interactive visualization in Python.
- References: [Documentation](#), [Examples](#) (Gallery), [Tutorials](#), and [Cheatsheets](#) (created by Matplotlib)



## Basic plots

	<code>plot([X], Y, [fmt], ...)</code> X, Y, fmt, color, marker, linestyle	API
	<code>scatter(X, Y, ...)</code> X, Y, [s]izes, [c]olors, marker, cmap	API
	<code>bar[h](x, height, ...)</code> x, height, width, bottom, align, color	API
	<code>imshow(Z, [cmap], ...)</code> Z, cmap, interpolation, extent, origin	API
	<code>contour[f]([X], [Y], Z, ...)</code> X, Y, Z, levels, colors, extent, origin	API
	<code>quiver([X], [Y], U, V, ...)</code> X, Y, U, V, C, units, angles	API
	<code>pie(X, [explode], ...)</code> Z, explode, labels, colors, radius	API
	<code>text(x, y, text, ...)</code> x, y, text, va, ha, size, weight, transform	API
	<code>fill[_between](x)( ... )</code> X, Y1, Y2, color, where	API

## Advanced plots

	<code>step(X, Y, [fmt], ...)</code> X, Y, fmt, color, marker, where	API
	<code>boxplot(X, ...)</code> X, notch, sym, bootstrap, widths	API
	<code>errorbar(X, Y, xerr, yerr, ...)</code> X, Y, xerr, yerr, fmt	API
	<code>hist(X, bins, ...)</code> X, bins, range, density, weights	API
	<code>violinplot(D, ...)</code> D, positions, widths, vert	API
	<code>barbs([X], [Y], U, V, ...)</code> X, Y, U, V, C, length, pivot, sizes	API
	<code>eventplot(positions, ...)</code> positions, orientation, lineoffsets	API
	<code>hexbin(X, Y, C, ...)</code> X, Y, C, gridsize, bins	API
	<code>xcorr(X, Y, ...)</code> X, Y, normed, detrend	API



# Matplotlib: A Plotting Library in Python

- [Matplotlib](#) is a plotting library for creating static, animated, and interactive visualization in Python.
- References: [Documentation](#), [Examples](#) (Gallery), [Tutorials](#), and [Cheatsheets](#) (made by Matplotlib)
- API examples @ [matplotlib.pyplot](#) **module** (a state-based interface; ~ [MATLAB](#))
  - `plot([x], y, [fmt], ...)`: Plot y versus x as lines and/or markers
  - `hist(x, bins=None, range=None, ...)`: Plot a [histogram](#)
  - `contour([X, Y,] Z, [levels], ...)`: Plot [contour lines](#)
  - `imshow(X, cmap=None, ...)`: Display data as an image (a 2D regular [raster](#))
  - `title(label, ...)`: Set a title for the Axes
  - `axis(...)`: Get or set axis properties (e.g. 'on'/'off', 'equal', and range of X/Y axes)
  - `legend(...)`: Place a legend on the Axes
  - `grid(...)`: Configure the grid lines
  - `xlabel(label, ...)`, `xlim(left, right)`, ...: Set the label and limit for the x-axis
  - `figure(num=None, ...)`: Create a new figure or activate an existing figure
  - `show(...)`: Display all open figures
  - `savefig(filename, ...)`: Save the current figure

# Matplotlib: A Plotting Library in Python

- Usage example) Drawing  $y = 0.1x^3 - 0.8x^2 - 1.5x + 5.4$

```
import matplotlib.pyplot as plt
```

```
scale = 10
```

```
xs = [x/scale for x in range(-4*scale, 10*scale)]
```

```
ys = [0.1*x**3 - 0.8*x**2 - 1.5*x + 5.4 for x in xs]
```

```
plt.title('$y = 0.1x^3 - 0.8x^2 - 1.5x + 5.4$') # LaTeX style
```

```
plt.plot(xs, ys, 'r-')
```

```
plt.xlabel('x')
```

```
plt.ylabel('y')
```

```
plt.grid()
```

```
plt.axis('equal')
```

```
plt.show()
```



Further improvement)

- ✓ How to make the graph smooth?
- ✓ How to draw the grid?
- ✓ How to display labels on X and Y axes?
- ✓ How to make its aspect ratio equal?

# Matplotlib: A Plotting Library in Python

- Example) Plotting midterm and final scores
  - The given data (file: data/class\_score\_en.csv)  
# midterm (max 125), final (max 100)

113, 86

104, 83

110, 78

101, 79

101, 77

103, 76

71, 94

102, 71

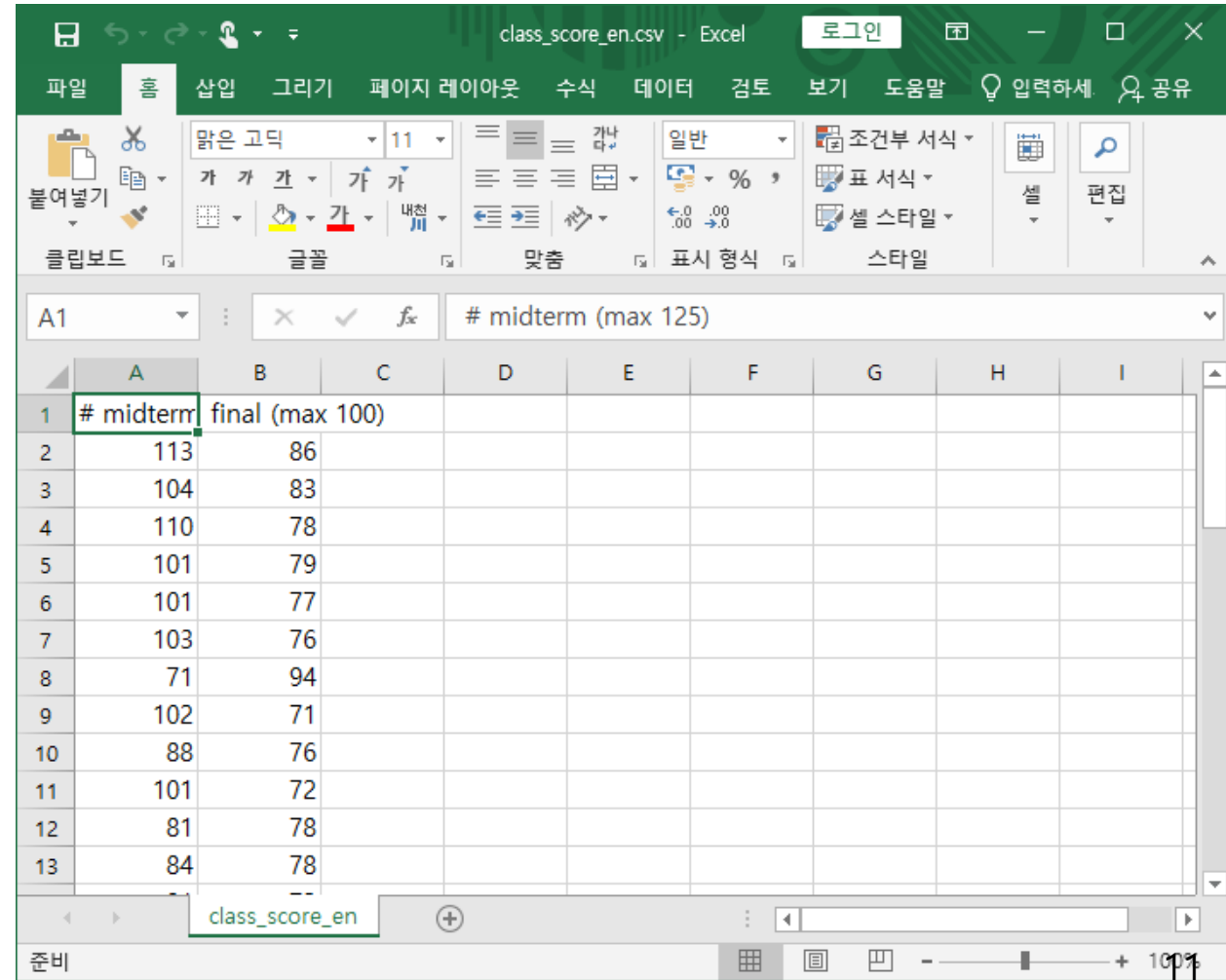
88, 76

101, 72

81, 78

84, 78

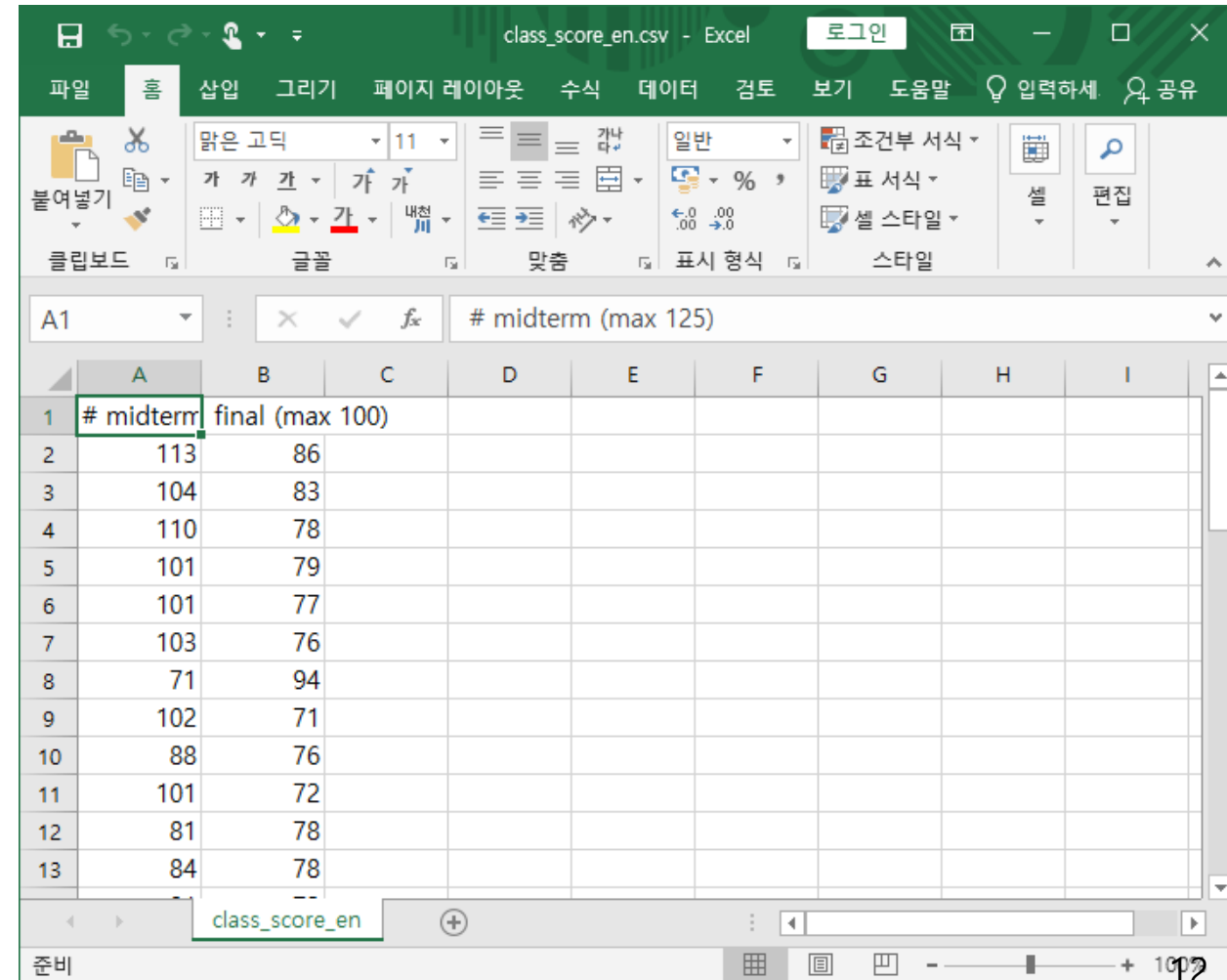
. . .



	# midterm (max 125)	final (max 100)
1	113	86
2	104	83
3	110	78
4	101	79
5	101	77
6	103	76
7	71	94
8	102	71
9	88	76
10	101	72
11	81	78
12	84	78
13		

# Matplotlib: A Plotting Library in Python

- Example) Plotting midterm and final scores
  - Analysis using representative values (e.g. mean, median, ...; [대표값](#) in Korean)
    - Midterm
      - Mean: **74.209**
      - Variance: 632.817
      - Median: **72.000**
      - Min/Max: (21.000, 117.000)
    - Final
      - Mean: **58.674**
      - Variance: 618.545
      - Median: **66.000**
      - Min/Max: (0.000, 94.000)
    - Total
      - Mean: **58.952**
      - Variance: 423.546
      - Median: **65.000**
      - Min/Max: (6.720, 87.760)

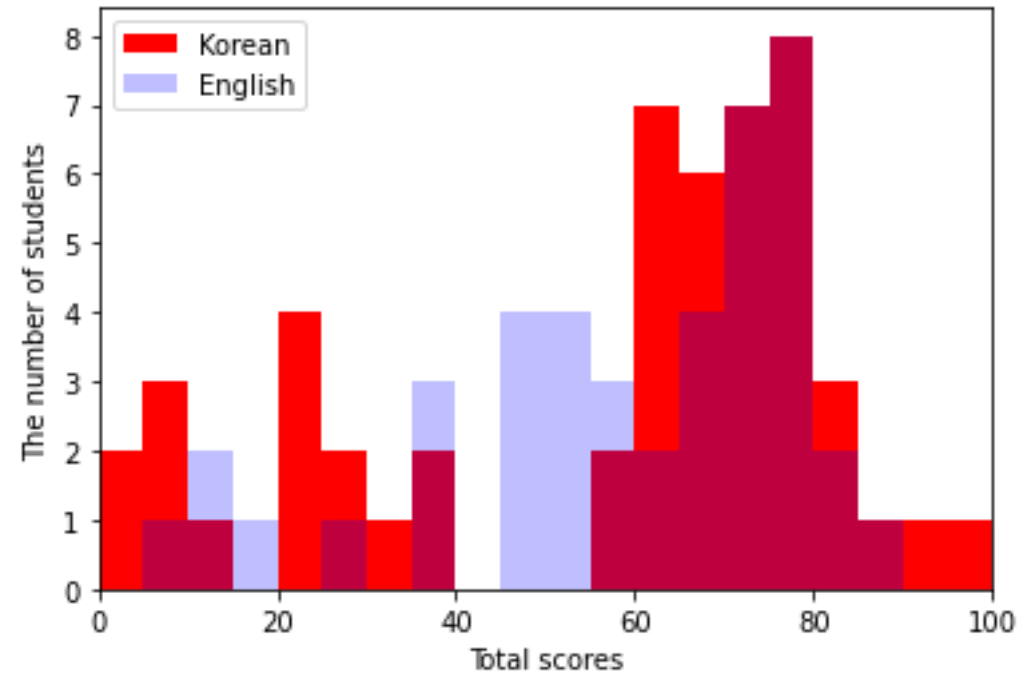
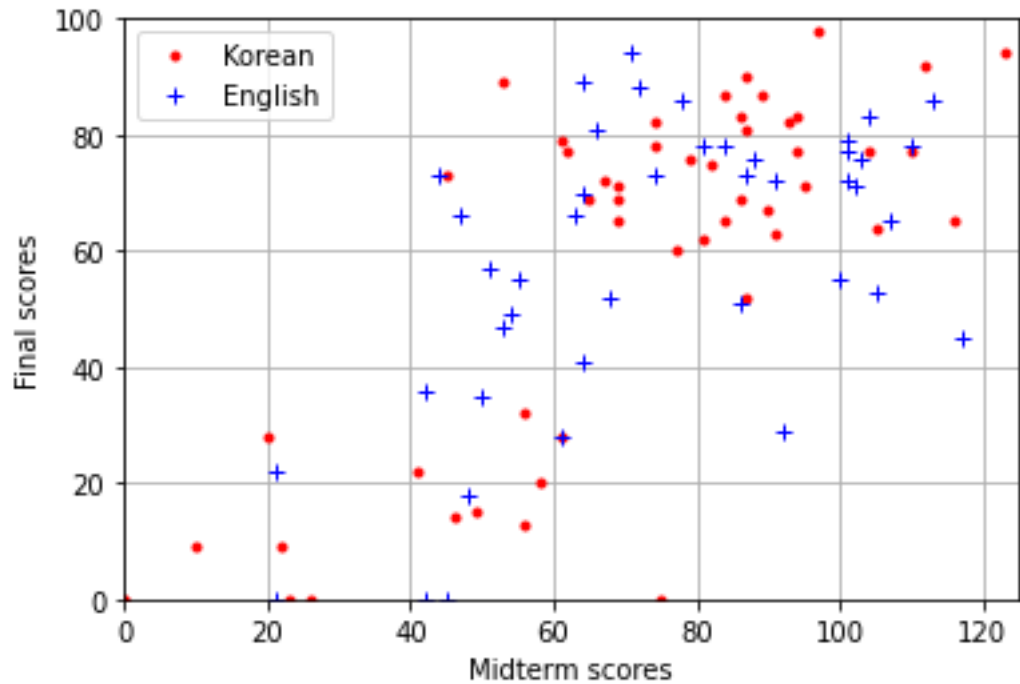


The screenshot shows an Excel spreadsheet titled 'class\_score\_en.csv'. The data is organized into two columns: '# midterm (max 125)' in column A and 'final (max 100)' in column B. There are 13 rows of data, numbered 1 to 13. The bottom of the spreadsheet shows a tab labeled 'class\_score\_en' and a status bar with the text '준비'.

	A	B	C	D	E	F	G	H	I
1	# midterm	final (max 100)							
2	113	86							
3	104	83							
4	110	78							
5	101	79							
6	101	77							
7	103	76							
8	71	94							
9	102	71							
10	88	76							
11	101	72							
12	81	78							
13	84	78							

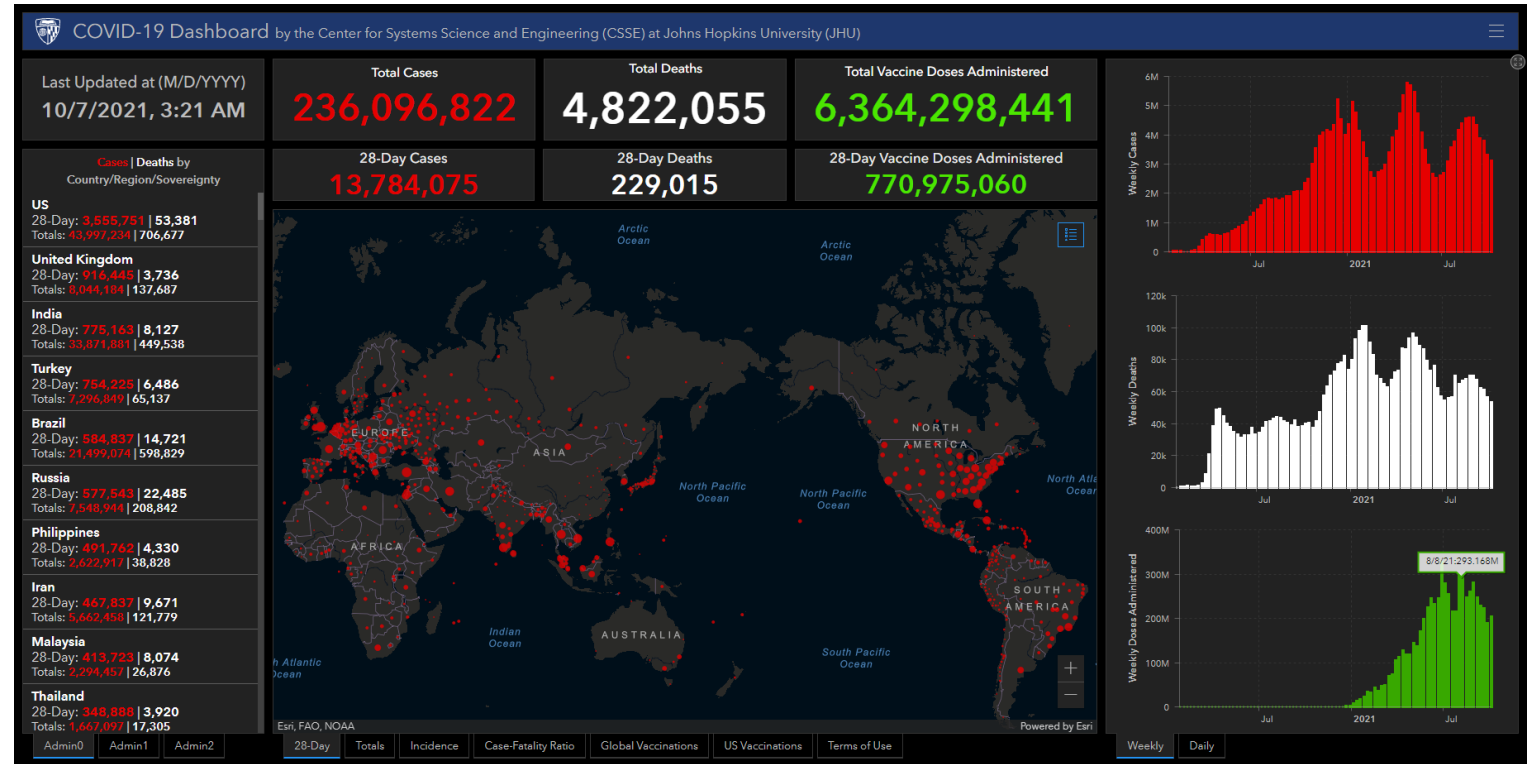
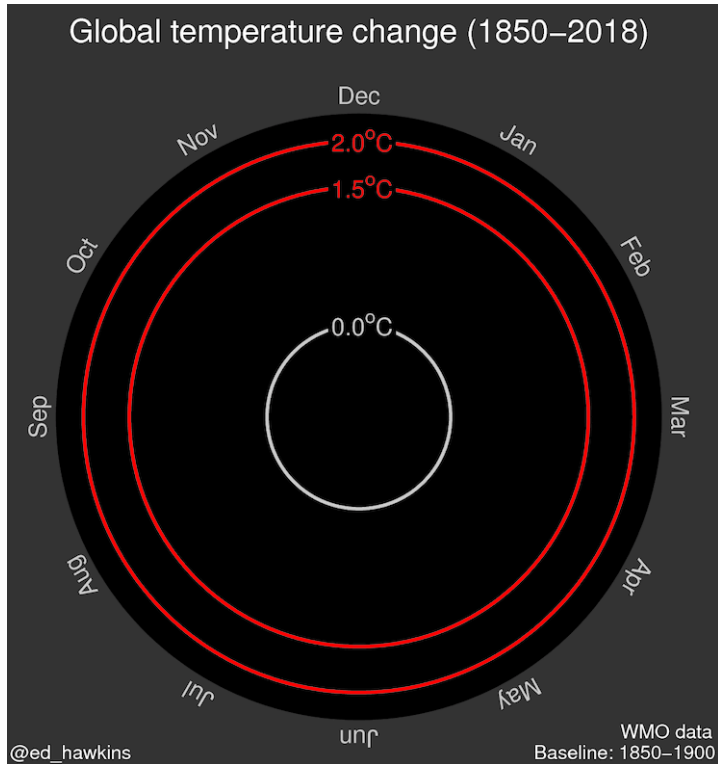
# Matplotlib: A Plotting Library in Python

- Example) Plotting midterm and final scores
  - Practice) Plot the [scatter plot](#) of midterm and final scores
  - Practice) Plot the [histogram](#) of total scores
  - What can you discover from two graphs?



# Visualization

- Visualization is very important not only for your data but also for your programs, systems, and society.

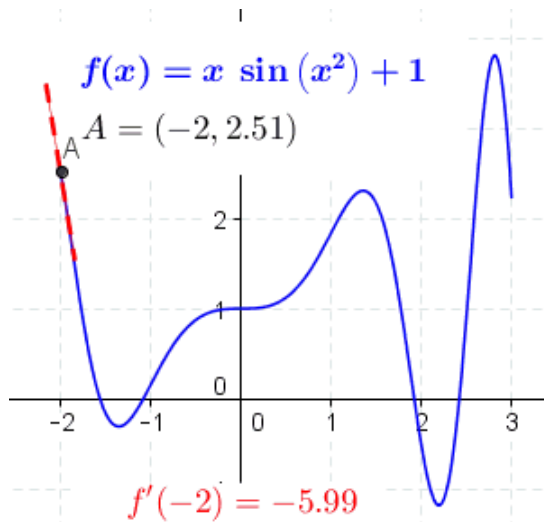


# Calculus

- Calculus (originally called *infinitesimal calculus*, 미적분학 in Korean) is the mathematical study of continuous change.
- Two major tools

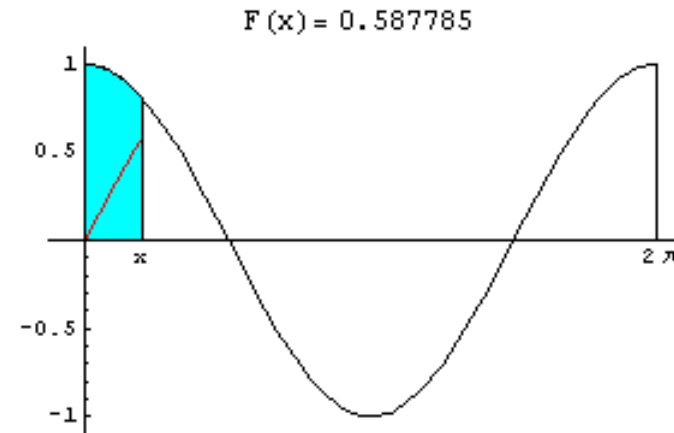
## Differentiation:

Finding a derivative of a function



## Integration:

Calculating numbers to a function by combining infinitesimal increments





# Calculus Differentiation

- [Differentiation](#) is the process of finding a **derivative** of a function.

- **Derivative** [\[Wikipedia\]](#)

- Definition

- Change of the function value (**output value**) with respect to the change in its argument (**input value**)

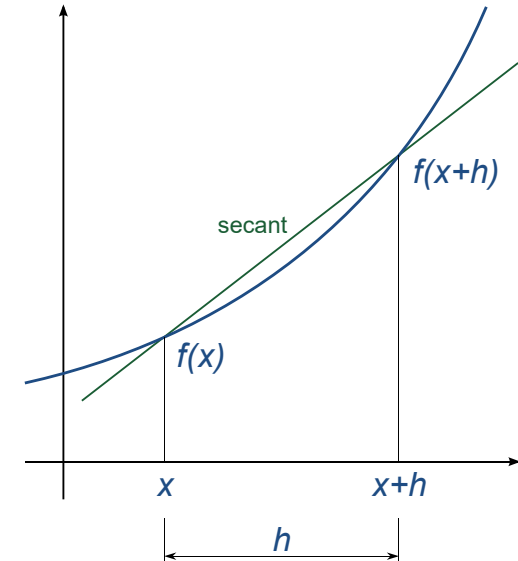
$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

- Meaning

- The **slope** of a tangent line (plane) to the given function
- The tangent line ~ a **linear approximation** (Note: 1st-order [Taylor series](#))

- Notation

- Leibniz's notation:  $\frac{df}{dx}(x), \frac{d^2f}{dx^2}(x)$
- Lagrange's notation:  $f'(x), f''(x)$
- Newton's notation:  $\dot{y}, \ddot{y}$  where  $y = f(x)$
- Euler's notation:  $D_x f(x), D_x^2 f(x)$



# Differentiation

- Rules of computing a derivative [\[Wikipedia\]](#)
  - The polynomial functions such as  $f(x) = x^n$ 
    - $f'(x) = nx^{n-1}$
  - Logarithmic and exponential functions
    - $\frac{d}{dx} \ln x = \frac{1}{x}$
    - $\frac{d}{dx} (e^x) = e^x$  (Note:  $e$  - [Euler's number](#), 자연상수 in Korean)
  - Too many functions ...
  - Differentiation is linear.**
    - $(f(x) + g(x))' = f'(x) + g'(x)$  and  $(af(x))' = af'(x)$
  - The **product rule** of  $h(x) = f(x)g(x)$ 
    - $h'(x) = f'(x)g(x) + f(x)g'(x)$
  - The **chain rule** of  $h(x) = f(g(x))$ 
    - $h'(x) = \frac{dh(x)}{dx} = \frac{df(g(x))}{dg(x)} = f'(g(x)) \cdot g'(x) = \frac{df(g(x))}{dg(x)} \cdot \frac{dg(x)}{dx}$

Theoretical Computer Science Cheat Sheet	
$\pi$	Calculus
Wallis' identity: $\pi = 2 \cdot \frac{2}{1} \cdot \frac{2}{3} \cdot \frac{4}{5} \cdot \frac{4}{7} \cdot \frac{6}{9} \cdots$ Brouncker's continued fraction expansion: $\frac{\pi}{4} = 1 + \frac{1^2}{2 + \frac{3^2}{2 + \frac{5^2}{2 + \frac{7^2}{2 + \cdots}}}}$ Gregory's series: $\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots$ Newton's series: $\frac{\pi}{6} = \frac{1}{2} + \frac{1}{2 \cdot 3 \cdot 2^3} + \frac{1 \cdot 3}{2 \cdot 4 \cdot 5 \cdot 2^5} + \cdots$ Sharp's series: $\frac{\pi}{6} = \frac{1}{\sqrt{3}} \left( 1 - \frac{1}{3^1 \cdot 3} + \frac{1}{3^2 \cdot 5} - \frac{1}{3^3 \cdot 7} + \cdots \right)$ Euler's series: $\frac{\pi^2}{6} = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \frac{1}{5^2} + \cdots$ $\frac{\pi^2}{8} = \frac{1}{1^2} + \frac{1}{3^2} + \frac{1}{5^2} + \frac{1}{7^2} + \frac{1}{9^2} + \cdots$ $\frac{\pi^2}{12} = \frac{1}{1^2} - \frac{1}{2^2} + \frac{1}{3^2} - \frac{1}{4^2} + \frac{1}{5^2} - \cdots$	Derivatives: 1. $\frac{d(cu)}{dx} = c \frac{du}{dx}$ ,    2. $\frac{d(u+v)}{dx} = \frac{du}{dx} + \frac{dv}{dx}$ ,    3. $\frac{d(uv)}{dx} = u \frac{dv}{dx} + v \frac{du}{dx}$ , 4. $\frac{d(u^n)}{dx} = nu^{n-1} \frac{du}{dx}$ ,    5. $\frac{d(u/v)}{dx} = \frac{v \frac{du}{dx} - u \frac{dv}{dx}}{v^2}$ ,    6. $\frac{d(e^{cu})}{dx} = ce^{cu} \frac{du}{dx}$ , 7. $\frac{d(c^u)}{dx} = (\ln c)c^u \frac{du}{dx}$ ,    8. $\frac{d(\ln u)}{dx} = \frac{1}{u} \frac{du}{dx}$ , 9. $\frac{d(\sin u)}{dx} = \cos u \frac{du}{dx}$ ,    10. $\frac{d(\cos u)}{dx} = -\sin u \frac{du}{dx}$ , 11. $\frac{d(\tan u)}{dx} = \sec^2 u \frac{du}{dx}$ ,    12. $\frac{d(\cot u)}{dx} = -\csc^2 u \frac{du}{dx}$ , 13. $\frac{d(\sec u)}{dx} = \tan u \sec u \frac{du}{dx}$ ,    14. $\frac{d(\csc u)}{dx} = -\cot u \csc u \frac{du}{dx}$ , 15. $\frac{d(\arcsin u)}{dx} = \frac{1}{\sqrt{1-u^2}} \frac{du}{dx}$ ,    16. $\frac{d(\arccos u)}{dx} = \frac{-1}{\sqrt{1-u^2}} \frac{du}{dx}$ , 17. $\frac{d(\arctan u)}{dx} = \frac{1}{1+u^2} \frac{du}{dx}$ ,    18. $\frac{d(\operatorname{arccot} u)}{dx} = \frac{-1}{1+u^2} \frac{du}{dx}$ , 19. $\frac{d(\operatorname{arcsec} u)}{dx} = \frac{1}{u\sqrt{1-u^2}} \frac{du}{dx}$ ,    20. $\frac{d(\operatorname{arccsc} u)}{dx} = \frac{-1}{u\sqrt{1-u^2}} \frac{du}{dx}$ , 21. $\frac{d(\sinh u)}{dx} = \cosh u \frac{du}{dx}$ ,    22. $\frac{d(\cosh u)}{dx} = \sinh u \frac{du}{dx}$ , 23. $\frac{d(\tanh u)}{dx} = \operatorname{sech}^2 u \frac{du}{dx}$ ,    24. $\frac{d(\coth u)}{dx} = -\operatorname{csch}^2 u \frac{du}{dx}$ , 25. $\frac{d(\operatorname{sech} u)}{dx} = -\operatorname{sech} u \tanh u \frac{du}{dx}$ ,    26. $\frac{d(\operatorname{csch} u)}{dx} = -\operatorname{csch} u \coth u \frac{du}{dx}$ , 27. $\frac{d(\operatorname{arcsinh} u)}{dx} = \frac{1}{\sqrt{1+u^2}} \frac{du}{dx}$ ,    28. $\frac{d(\operatorname{arccosh} u)}{dx} = \frac{1}{\sqrt{u^2-1}} \frac{du}{dx}$ , 29. $\frac{d(\operatorname{arctanh} u)}{dx} = \frac{1}{1-u^2} \frac{du}{dx}$ ,    30. $\frac{d(\operatorname{arcoth} u)}{dx} = \frac{1}{u^2-1} \frac{du}{dx}$ , 31. $\frac{d(\operatorname{arsech} u)}{dx} = \frac{-1}{u\sqrt{1-u^2}} \frac{du}{dx}$ ,    32. $\frac{d(\operatorname{arcsch} u)}{dx} = \frac{-1}{ u \sqrt{1+u^2}} \frac{du}{dx}$ .
<b>Partial Fractions</b> Let $N(x)$ and $D(x)$ be polynomial functions of $x$ . We can break down $N(x)/D(x)$ using partial fraction expansion. First, if the degree of $N$ is greater than or equal to the degree of $D$ , divide $N$ by $D$ , obtaining $\frac{N(x)}{D(x)} = Q(x) + \frac{N'(x)}{D(x)},$ where the degree of $N'$ is less than that of $D$ . Second, factor $D(x)$ . Use the follow-	

Note) The quotient rule of  $h(x) = \frac{f(x)}{g(x)} = f(x)g(x)^{-1}$

$$\bullet \quad h'(x) = \frac{f'(x)g(x) - g'(x)f(x)}{g^2(x)}$$

# Differentiation

- Example) Find the slope and tangent line of  $y = 0.1x^3 - 0.8x^2 - 1.5x + 5.4$  at  $x = 2$  ( $y = 0$ )
  - Derivative function:  $\frac{dy}{dx}(x) = 0.3x^2 - 1.6x - 1.5$
  - Derivative value at  $x = 2$  ( $y = 0$ ):  $\frac{dy}{dx}(2) = -3.5$
  - Tangent line at  $x = 2$  ( $y = 0$ ):  $y - 0 = -3.5(x - 2) \rightarrow y = -3.5x + 7$

- Example) Plotting the above results

```
import matplotlib.pyplot as plt
```

```
scale = 10
```

```
xs = [x/scale for x in range(-4*scale, 10*scale)]
```

```
ys = [0.1*x**3 - 0.8*x**2 - 1.5*x + 5.4 for x in xs]
```

```
yt = [-3.5*x + 7 for x in xs]
```

```
plt.plot(xs, ys, 'r-', label='y')
```

```
plt.plot(xs, yt, 'b--', label='tangent line at x=2')
```

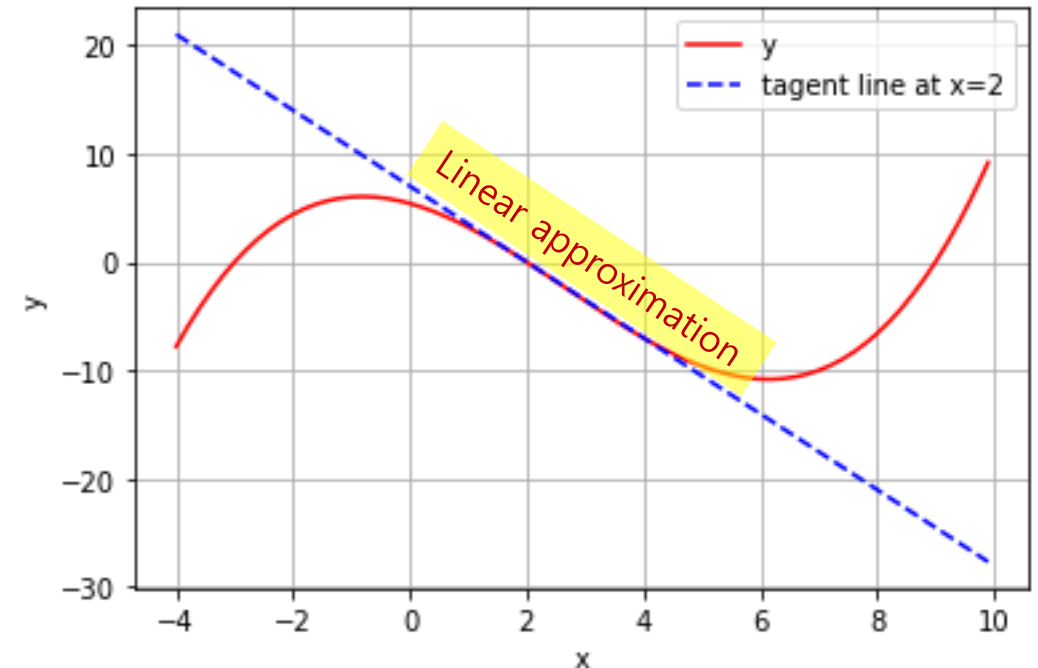
```
plt.xlabel('x')
```

```
plt.ylabel('y')
```

```
plt.grid()
```

```
plt.legend()
```

```
plt.show()
```



# Differentiation

- Example) A turtle moving on  $y = 0.1x^3 - 0.8x^2 - 1.5x + 5.4$  (turtle\_animation.py)

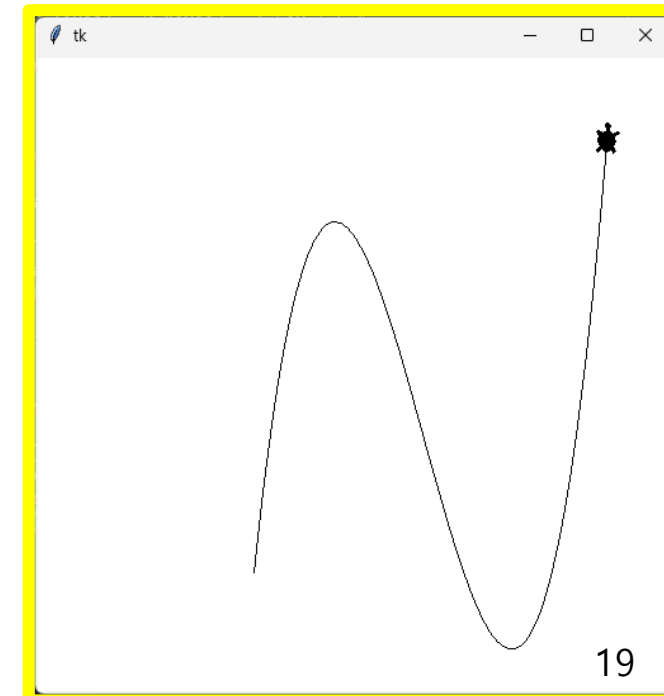
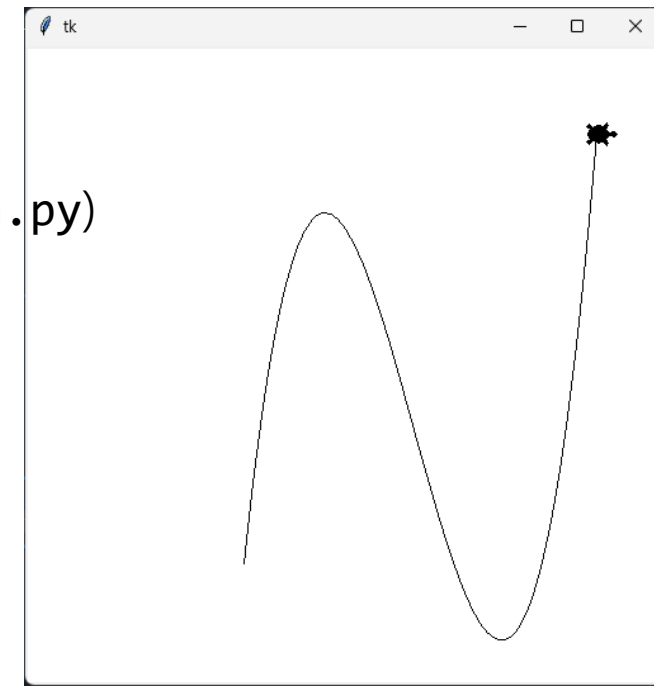
```
import tkinter as tk
import turtle, math

# Generate 'y = 0.1*x**3 - 0.8*x**2 - 1.5*x + 5.4'
scale = 10
xs = [x/scale for x in range(-4*scale, 10*scale)]
ys = [0.1*x**3 - 0.8*x**2 - 1.5*x + 5.4 for x in xs]
yd = [0.3*x**2 - 1.6*x - 1.5 for x in xs]

# Prepare 'screen' and 'actor'
root = tk.Tk()
canvas = tk.Canvas(root, width=500, height=500)
canvas.pack()
screen = turtle.TurtleScreen(canvas)
actor = turtle.RawTurtle(screen, 'turtle')
actor.radians() # Use radian unit for angle and rotation

# Draw the function
zoom = 20
actor.penup()
actor.setpos(zoom*xs[0], zoom*ys[0]) # Put at the initial point
actor.pendown()
for (x, y, slope) in zip(xs, ys, yd):
    actor.setpos(zoom*x, zoom*y)
    actor.setheading(math.atan2(slope, 1))
screen.mainloop()
```

How to derive a **derivative function** of  
a complex function?



# Differentiation

- Example) Find the derivative function of  $y = 0.1x^3 - 0.8x^2 - 1.5x + 5.4$  using [SymPy](#)

```
import sympy as sp
```

```
x, y = sp.symbols('x y')
```

```
y = 0.1*x**3 - 0.8*x**2 - 1.5*x + 5.4
```

```
yd = sp.diff(y, x)
```

```
print(yd)
```

```
# 0.3*x**2 - 1.6*x - 1.5
```

```
print(float(yd.subs({x: 2})))
```

```
# -3.5 / Casting from Float object to float
```

## Note) More Examples with [SymPy](#)

- Example) Solve the above equation (find  $x$  when  $y = 0$ ) using [SymPy](#)

```
roots = sp.solve(y, x)
print(roots) # FiniteSet(-3.0, 2.0, 9.0)
r0 = float(roots[0]) # -3.0 / Casting from Float object to float
```

- Example) Factorize the above equation using [SymPy](#)

```
sp.pprint(sp.factor(y)) # 5.4*(0.11...x - 1.0)*(0.33...x + 1.0)*(0.5x - 1.0)

y = (x**3 - 8*x**2 - 15*x + 54)/10 # Make the coefficients rational numbers
sp.pprint(sp.factor(y)) # (x - 9)*(x - 2)*(x + 3)
# -----
# 10
# (x - 9)*(x - 2)*(x + 3)/10

print(sp.factor(y))
```

# Differentiation

- Example) Find the derivative function of a composite function using [SymPy](#)

```
import sympy as sp
```

```
from random import random
```

```
perceptron = lambda x, w, b: sp.tanh(w * x + b)
```

```
f = lambda x: perceptron(x, random(), random())
```

```
x, y = sp.symbols('x y')
```

```
y = f(f(f(f(f(f(f(x)))))))
```

```
yd = sp.diff(y, x)
```

```
print(yd)
```

```
#0.005*
```

```
 #(1 - tanh(0.12*x + 0.81)**2)*
```

```
 #(1 - tanh(0.97*tanh(0.12*x + 0.81) + 0.62)**2)*
```

```
 #(1 - tanh(0.75*tanh(0.97*tanh(0.12*x + 0.81) + 0.62) + 0.97)**2)*
```

```
 #(1 - tanh(0.68*tanh(0.75*tanh(0.97*tanh(0.12*x + 0.81) + 0.62) + 0.97) + 0.38)**2)*
```

```
 #(1 - tanh(0.99*tanh(0.68*tanh(0.75*tanh(0.97*tanh(0.12*x + 0.81) + 0.62) + 0.97) + 0.38) + 0.91)**2)*
```

```
 #(1 - tanh(0.49*tanh(0.99*tanh(0.68*tanh(0.75*tanh(0.97*tanh(0.12*x + 0.81) + 0.62) + 0.97) + 0.38) + 0.91) + 0.27)**2)*
```

```
 #(1 - tanh(0.16*tanh(0.49*tanh(0.99*tanh(0.68*tanh(0.75*tanh(0.97*tanh(0.12*x + 0.81) + 0.62) + 0.97) + 0.38) + 0.91) + 0.27) + 0.07)**2)*
```



# Differentiation

- [Numerical differentiation](#) is an algorithm to approximate the derivative value of a function using the values of the function. (Note: It does not derive a derivative function.)

- The definition of derivative:  $f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$
- Two-point estimation:  $f'(x_0) = \frac{f(x_0+h) - f(x_0)}{h}$  or  $f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h}$

- Example) Find the slope and tangent line of  $y = 0.1x^3 - 0.8x^2 - 1.5x + 5.4$  at  $x = 2$

- Derivative at  $x = 2$  ( $y = 0$ ):  $-3.5$

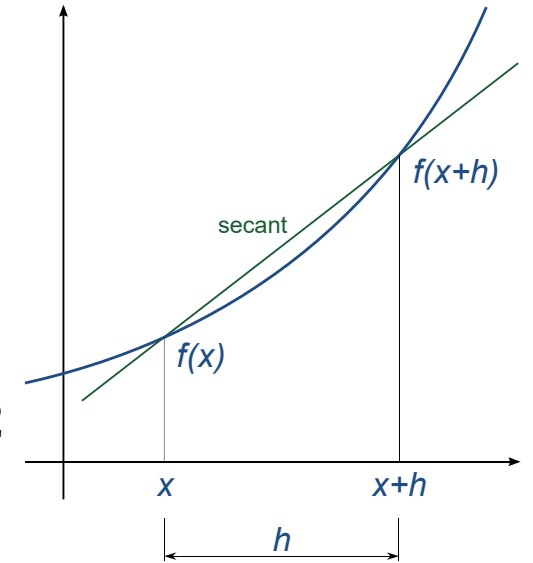
```
y = lambda x: 0.1*x**3 - 0.8*x**2 - 1.5*x + 5.4
```

```
x0 = 2
```

```
h = 0.001
```

```
d1 = (y(x0+h) - y(x0)) / h # -3.5002
```

```
d2 = (y(x0+h/2) - y(x0-h/2)) / h # -3.5000
```



# Summary

- Building **mathematical intuition**: 1) **Imagine**, 2) **Ask why**

- Visualization

- [Matplotlib](#): A plotting library in Python
- **Visualization is very important** not only for your data but also for your programs, systems, and society.

- ~~Calculus~~ Differentiation

- Derivative definition:  $f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$
- Meaning
  - The **slope** of a tangent line (plane) to the given function
  - The tangent line ~ a **linear approximation**
- How to get a derivative
  - ~~Your hand-on derivation using [rules](#)~~
  - Analytical differentiation using [SymPy](#) (a symbolic mathematical engine in Python)
  - Numerical differentiation (two-point estimation)