ลิสต์แบบหลายมิติ และ การใช้ไลบรารี

ภาควิชาวิศวกรรมคอมพิวเตอร์
จุฬาลงกรณ์มหาวิทยาลัย
๒๕๕๘

Topics

- **ส่วนที่ 1 ลิสต์แบบหลายมิติ**
- ลิสต์หลายมิติใน Python
- ไลบรารี Numpy
  - การสร้างอาร์เรย์
  - การเข้าถึง
  - การใช้งานทางคณิตศาสตร์ (element-wise)
  - การใช้งานด้วยฟังก์ชั่นเฉพาะ เช่น sum, mean, max, min
  - การคูณเมทริกซ์ (dot operator)
  - enumerate vs. ndenumerate
  - การเขียนอ่าน text file
- Application1 kNN
  - โจทย์ Iris
  - นำผลลัพธ์มา plot Histogram

Topics (ต่อ)

- **ส่วนที่ 2 การใช้ไลบรารีอื่นนอกจาก Numpy**
- ไลบรารีอื่นๆ
  - ความแตกต่างระหว่าง from vs. import
  - ไลบรารี Matplotlib กับการวาดกราฟ
  - ไลบรารี Scipy กับการจัดการรูปภาพ

- Application2: Image Convolution
- Application3: Face Detection
- Application4: Linear Transformation

ส่วนที่ 1 ลิสต์แบบหลายมิติ

ภาควิชาวิศวกรรมคอมพิวเตอร์
จุฬาลงกรณ์มหาวิทยาลัย
๒๕๕๘

ลิสต์ใน Python

```
x = [1,2,3,4,5,6,7,8,9,10]
y = [1]*5 + [2]*5
sum = [0]*len(x)
for i in range(len(x)):
    sum[i] = x[i] + y[i]
```

ลองทำ print(x+y)

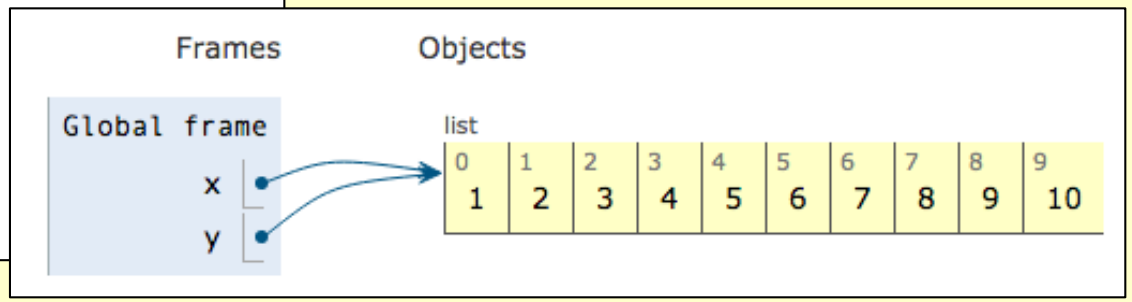
```
>>> print(sum)
[2, 3, 4, 5, 6, 8, 9, 10, 11, 12]
>>> print(sum[0])
2
>>> print(sum[-1])
12
>>> print(sum[8:10])
[11, 12]
```

คำสั่งน่ารู้

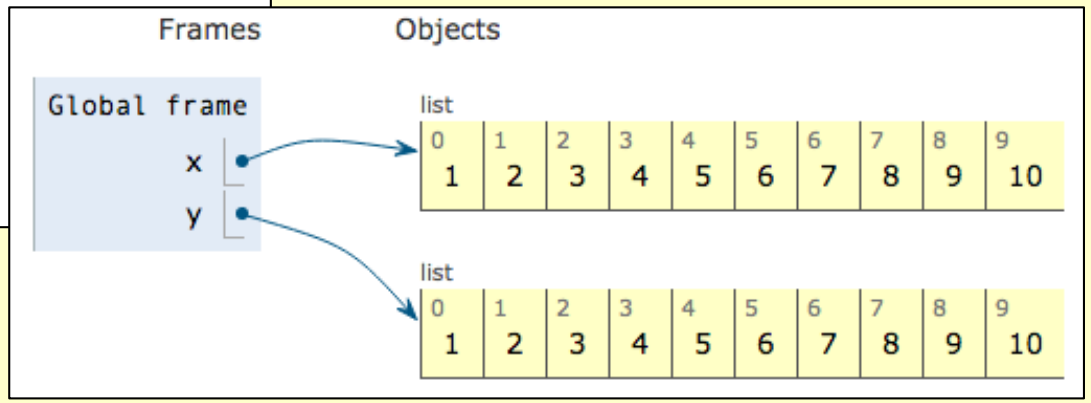
```
x.append(e)
x.insert(i, e)
x.pop(i)
x.remove(e)
x.count(e)
x.sort()
x.reverse()
```

ลิสต์ใน Python (cont.)

```
>>> x = [1,2,3,4,5,6,7,8,9,10]
>>> y = x
>>> print(x == y)
True
>>> print(x is y)
True
```

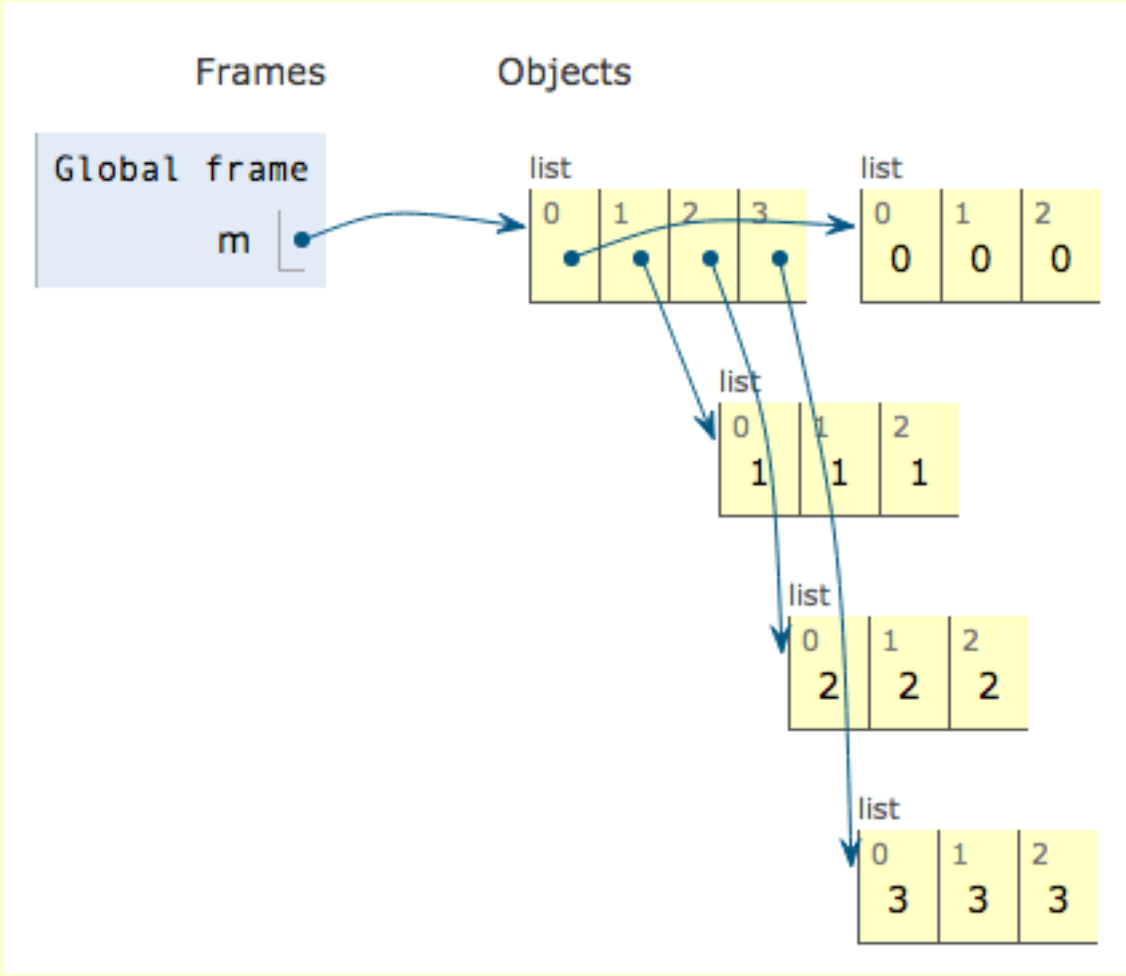



```
>>> x = [1,2,3,4,5,6,7,8,9,10]
>>> y = list(x)
>>> print(x == y)
True
>>> print(x is y)
False
```



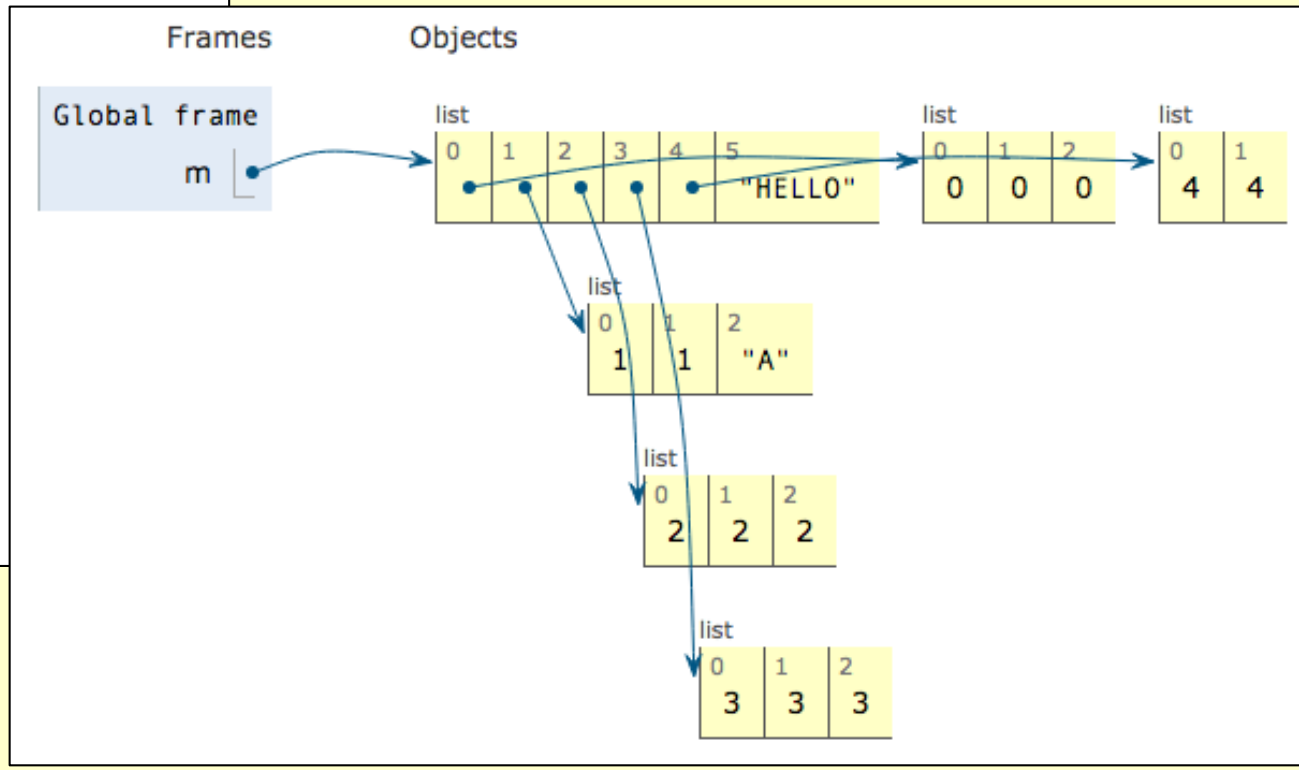
ลิสต์ของลิสต์ใน Python

```
m = [[0, 0, 0],
     [1, 1, 1],
     [2, 2, 2],
     [3, 3, 3]]
>>> len(m)
4
>>> m[1]
[1, 1, 1]
>>> m[1][:]
[1, 1, 1]
>>> m[1][0]
1
>>> len(m[1])
3
```



ลิสต์ของลิสต์ใน Python (cont.)

```
m = [[0, 0, 0],
     [1, 1, 1],
     [2, 2, 2],
     [3, 3, 3]]
m.append([4,4])
m.append("HELLO")
m[1][2] = 'A'
```




```
>>> m
[[0, 0, 0], [1, 1, 'A'], [2, 2, 2], [3, 3, 3], [4, 4], 'HELLO']
```

ลองเขียนดู : Matrix Transpose

เขียนโปรแกรมรับค่าเมตริกจากผู้ใช้ขนาด 3 * 3 โดยรับค่าเมตริกทีละแถว จากนั้นให้คำตอบเป็น Matrix Transpose

```
Input matrix row1: 1 2 3
Input matrix row2: 4 5 6
Input matrix row3: 7 8 9

Matrix: [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
Matrix.T: [[1, 4, 7], [2, 5, 8], [3, 6, 9]]
```

Numerical Python (Numpy)

- อาร์เรย์หรือลิสต์ใน Python ไม่สามารถคำนวณ arithmetic operations (+,-,*,/)

```
>>> a = [1,3,5,7,9]
>>> b = [3,5,6,7,9]
>>> c = a + b
>>> print c
[1, 3, 5, 7, 9, 3, 5, 6, 7, 9]
```

ผลลัพธ์ผิด!

- ดังนั้นจึงต้องการไลบรารี Numerical Python (Numpy) เป็นลิสต์ที่มีความสามารถทางคณิตศาสตร์เพิ่มขึ้นมา (อีกมาก!!!)

```
import numpy
```

NumPy คืออะไร

- Numpy เป็นไลบรารีหนึ่งของไพธอน โดยเป็นไลบรารีหลักที่ใช้ในการคำนวณทางวิทยาศาสตร์
- Numpy มากับโครงสร้างข้อมูลที่เรียกว่าอาเรย์หลายมิติ (ndarray) และมีฟังก์ชั่นพื้นฐานที่ใช้ในการจัดการและวิเคราะห์ข้อมูลอาเรย์เหล่านี้
  - เช่น การคำนวณทางคณิตศาสตร์ การปรับมิติของอาเรย์ การทำ discrete Fourier transform การคำนวณ linear algebra การคำนวณทางสถิติพื้นฐาน เป็นต้น
- Numpy ถูกพัฒนาด้วยภาษา C ดังนั้นจึงสามารถประมวลผลได้เร็ว ซึ่งเป็นส่วนผสมของไลบรารี NumArray และ Numeric ในอดีต (ปัจจุบันใช้แต่ Numpy)

คุณสมบัติของอาเรย์ใน NumPy

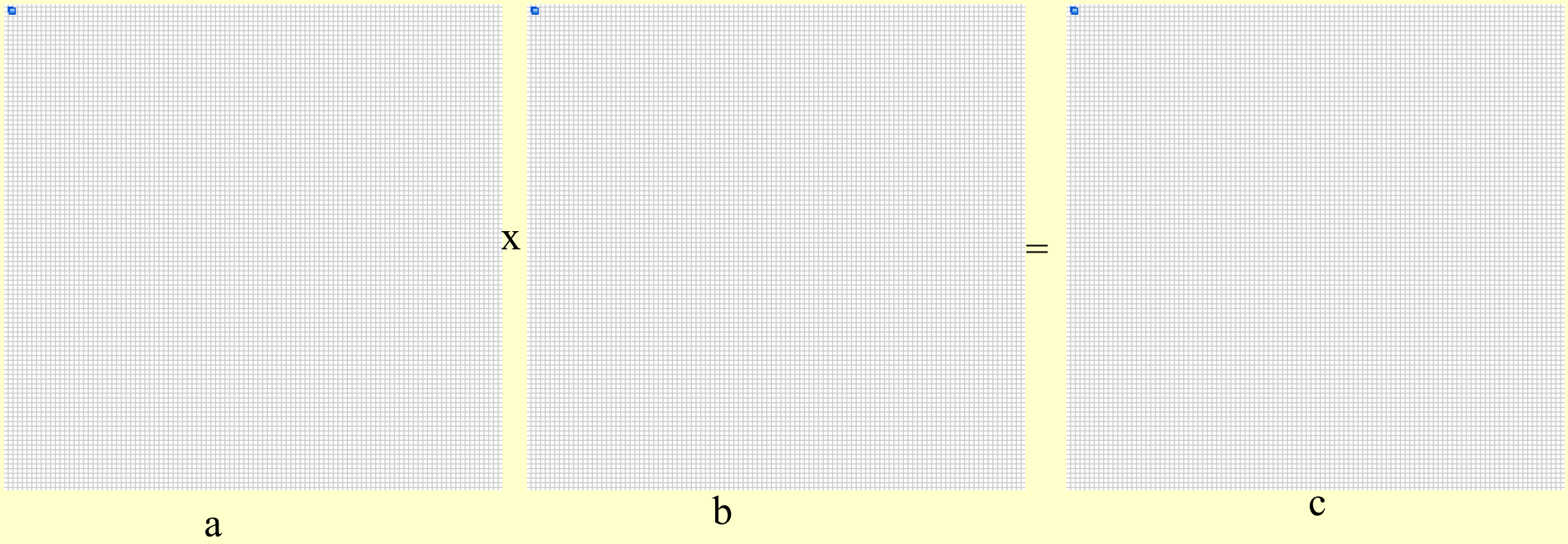
- สร้างแล้วมีขนาดตายตัว เปลี่ยนแปลงไม่ได้ ไม่เหมือน ลิสต์ที่มากับภาษาไพธอน
- ค่าในอาเรย์หนึ่งๆ ต้องเป็นประเภทเดียวกันทั้งหมด (ยกเว้นอาเรย์ของออบเจค ยังไม่ได้เรียนออบเจค)
- มีฟังก์ชั่นพื้นฐานที่ใช้ในการจัดการและวิเคราะห์ข้อมูลอาเรย์ ที่มีประสิทธิภาพสูง

เมทริกซ์ใน Numpy

- อาเรย์สองมิติใน Numpy เรียกว่าเมทริกซ์
- จำนวนของมิติของอาเรย์ใน Numpy เรียกว่าแร้งค์ (rank)
- รูปร่าง (shape) ของอาเรย์ถูกแสดงโดย tuple โดยบอกขนาดของอาเรย์ในแต่ละมิติ

ตัวอย่างประสิทธิภาพของ NumPy

มีอาเรย์ 2 มิติ ขนาดเท่ากัน 10,000 x 10,000



ถ้าต้องการอาเรย์ c โดยที่ $c_{ij} = a_{ij} * b_{ij}$ ต้องทำอย่างไร

หา $c_{ij} = a_{ij} * b_{ij}$ แบบที่ 1 ใช้ลิสต์ (ไม่ใช้ Numpy)

```
import time
test_size = 10000
t1 = time.time()
Matrix_a = [[2 for x in range(test_size)] for x in range(test_size)]
Matrix_b = [[3 for x in range(test_size)] for x in range(test_size)]
Matrix_c = [[0 for x in range(test_size)] for x in range(test_size)]
t2 = time.time()
print("time used for list initialization = ", t2-t1)

for i in range(test_size):
    for j in range(test_size):
        Matrix_c[i][j] = Matrix_a[i][j] * Matrix_b[i][j]
t3 = time.time()
print("time used for for loop = ", t3-t2)
```

```
time used for list initialization = 15.21 seconds
time used for for loop = 32.69 seconds
```

หา $c_{ij} = a_{ij} * b_{ij}$ แบบที่ 2 ใช้อาเรย์ใน NumPy

```
import numpy as np
import time
test_size = 10000
t1 = time.time()
Matrix_a = 2 * np.ones(shape=(10000,10000))
Matrix_b = 3 * np.ones(shape=(10000,10000))
t2 = time.time()
print("time used for Matrices initialization = ", t2-t1)

Matrix_c = Matrix_a * Matrix_b
t3 = time.time()
print("time used for Matrix calculation = ", t3-t2)
```

```
time used for Matrices initialization =  1.26
time used for Matrix calculation =  0.37
```

การสร้างอาร์เรย์ใน Numpy (1 มิติ)

```
# as vectors from lists
>>> import numpy as np
>>> a = np.array([1,3,5,7,9])
>>> b = np.array([3,5,6,7,9])
>>> c = a + b
>>> c
array([ 4,  8, 11, 14, 18])
>>> print(c)
[ 4 8 11 14 18]
>>> type(c)
(<type 'numpy.ndarray'>)
>>> c.shape
(5,)
```

การสร้างอาร์เรย์ใน Numpy (2 มิติ)

```
>>> M = np.array([[1, 2, 3], [3, 6, 9], [2, 4, 6]])
>>> print(M)
[[1 2 3]
 [3 6 9]
 [2 4 6]]
>>> M.shape
(3, 3)
>>> print(M.dtype)  # get type of an array
Int64
```

```
#only one type
>>> M[0,0] = 'A'
Traceback (most recent call last):
  File "<pyshell#8>", line 1, in <module>
    M[0,0] = 'A'
ValueError: invalid literal for int() with base 10: 'A'
```

การสร้างอาเรย์ใน NumPy ด้วยฟังก์ชัน

```
import numpy as np

x = np.zeros((2, 3))# สร้างอาเรย์ 2 มิติ 2 บรรทัด 3 คอลัมน์
ค่าเริ่มต้นทุกช่องเป็น 0
```

```
y = np.ones((2, 3)) สร้างอาเรย์ 2 มิติ 2 บรรทัด 3 คอลัมน์ ค่า
เริ่มต้นทุกช่องเป็น 1
```

```
z1 = np.arange(10) # สร้างอาเรย์ 1 มิติที่มีค่าเริ่มจาก 0 และเพิ่ม
ค่าที่ละ 1
```

```
z2 = np.arange(2,10,dtype=np.float)# สร้างอาเรย์ 1 มิติที่มี
ค่าเริ่มจาก 2.0 ถึง 9.0 เป็นเลขทศนิยม และเพิ่มค่าที่ละ 1
```

```
z3 = np.arange(2,3,0.1)
```

การสร้างอาเรย์ใน NumPy ด้วยฟังก์ชัน (ต่อ)

```
import numpy as np

x = np.array([[1, 2, 3], [4, 5, 6]], float)

y = np.zeros_like(x)

y = np.ones_like(x)

z = np.identity(4,dtype=float)
```

การอ้างอิงข้อมูลใน Numpy

```
>>> print(M)
[[1 2 3]
 [3 6 9]
 [2 4 6]]
>>> print(M[0])  # this is just like a list of lists
[1 2 3]
>>> print(M[1, 2])  # comma separated indices
9
>>> print(M[1, 1:3])  # and slices
[6 9]
```

```
>>> M[1, 2] = 7
>>> print(M)
[[1 2 3]
 [3 6 7]
 [2 4 6]]
>>> M[:, 0] = [0, 9, 8]
>>> print(M)
[[0 2 3]
 [9 6 7]
 [8 4 6]]
```

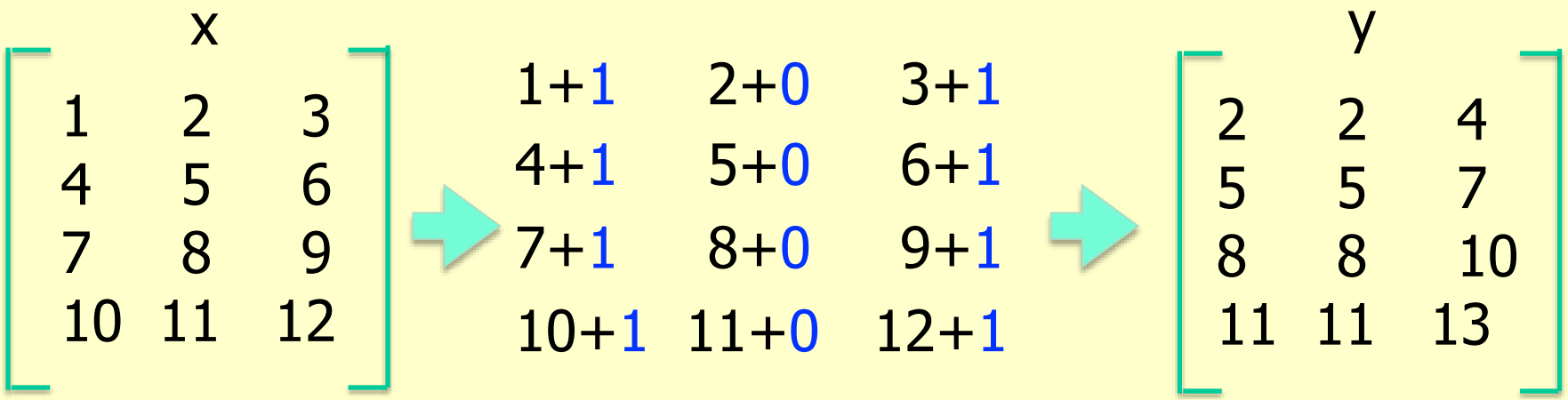
การบวก ลบ คูณ หาร อาเรย์ (ค่าต่อค่า)

```
import numpy as np
x = np.array([[1,2],[3,4]])
y = np.array([[5,6],[7,8]])

print(x+y)
print(np.add(x,y))

print(x-y)
print(np.subtract(x,y))

print(x*y)     # * element wise multiplication
print(np.multiply(x,y))

print(x/y)
print(np.divide(x,y))

print(np.sqrt(x))
```

บวก ลบ คูณ หาร ค่าต่อค่า อาเรย์สองตัวต้องมีรูปร่างเดียวกัน

ถ้าอาเรย์มีรูปร่างไม่เหมือนกัน จะทำบวก ลบ คูณ หาร อย่างไร??

เพิ่มค่าในอาเรย์โดยใช้ค่าจากเวคเตอร์ (แบบลูป)




```
import numpy as np
x = np.array([[1,2,3],[4,5,6],[7,8,9],[10,11,12]])
v = np.array([1,0,1])
y = np.empty_like(x) #สร้างเมทริกซ์ว่าง ๆที่มีรูปร่างแบบ x
for i in range(4):
    y[i,:] = x[i, :] + v
print(y)
```

บอร์ดคาสติ้ง (Broadcasting)

- บอร์ดคาสติ้งใช้ในการอธิบายวิธีการทำงานของโอเปอร์เรเตอร์ทางคณิตศาสตร์ระหว่างอะเรย์ที่มีจำนวนมิติ รูปร่าง ขนาด ต่างกันอย่างไร
- โดยทั่วไปอาเรย์ที่มีขนาดเล็กกว่าจะถูกบอร์ดคาสไปยังอาเรย์ที่มีขนาดใหญ่กว่า

```
x = np.array([1,2,3])
x = x + [1]
print(x)
```

```
x = np.array([[1,2,3],[4,5,6]])
x[0] = x[0] + [1]
print(x)
x[:,0] = x[:,0] + [1]
print(x)
```


http://cs231n.github.io/python-numpy-tutorial/#numpy

ตัวอย่างบอร์ดคาสติ้ง

```
# Good example
import numpy as np
x = np.array([[1,2,3],[4,5,6],[7,8,9],[10,11,12]])
v = np.array([1,0,1])
y = x + v
print(y)
```

```
# Error example
import numpy as np
x = np.array([1,2,3],float)
y = np.array([4,5], float)
z = x + y
print(z)
```

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: operands could not be broadcast together with shapes (3,) (2,)
```

ตย ฟังก์ชันทางคณิตศาสตร์ที่น่าสนใจอื่นๆใน NumPy

abs, sign, sqrt, log, log10, exp, sin, cos, tan, arcsin, arccos, arctan, sinh, cosh, tanh, arcsinh, arccosh, และ arctanh

ทำงานเป็นค่าต่อค่า (element-wise) เหมือน บวก ลบ คูณ หาร

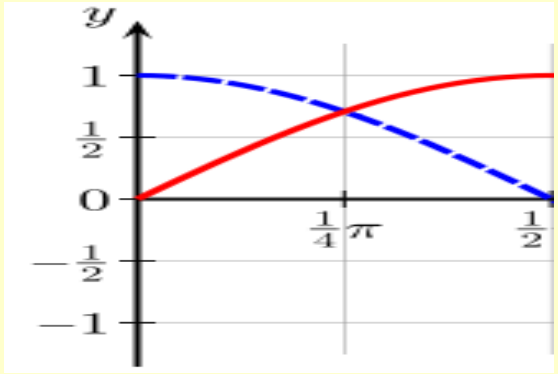
```
import numpy as np
a = np.array([1, 4, 9], float)
np.sqrt(a)
```

CH08_1

- การหาอนุพันธ์ (derivative) ทำได้ดังสมการด้านล่าง ซึ่งสามารถใช้ในการหาทิศทางการเปลี่ยนแปลงของฟังก์ชัน โดยหากผลอนุพันธ์เป็นบวก หมายถึงฟังก์ชันกำลังเพิ่มขึ้น และหากผลอนุพันธ์เป็นลบ หมายถึงฟังก์ชันกำลัง

$$f'(x_i) = \frac{f(x_i + \Delta x) - f(x_i - \Delta x)}{2\,\Delta x}$$

- จงหาอนุพันธ์และคำนวณผลรวมของทิศทางการเปลี่ยนแปลง (เครื่องหมาย) ของฟังก์ชัน sin และ ประมวลผลเช่นเดียวกันฟังก์ชัน cos
- กำหนดให้ $x_i = [0, \pi/2]$ และกำหนดให้ $\Delta x = 0.1$
- ไม่ต้องรวมทุก x_i เริ่มต้นและสิ้นสุด (Hint: x_i มี 15 ค่า)
- **ห้ามใช้ loop!**



CH08_2

- การถดถอยโลจิสติก (logistic regression) สามารถใช้ทำนาย (predict) ความน่าจะเป็น (probability) จากตัวแปรทำนาย (predictors)
- จงใช้สมการถดถอยด้านล่าง เพื่อ**ทำนาย**ความน่าจะเป็นในการสอบผ่านของนิสิต 5 คน เมื่อกำหนดให้มีตัวแปรทำนาย 2 ตัวได้แก่ คะแนนสอบกลางภาค (score) และ เกรด (GPA)
- นิสิตจะมีโอกาสผ่าน (True) เมื่อ p > 0.5
- **ห้ามใช้ loop! ให้ใช้ numpy อาร์เรย์ 2 มิติ ชื่อ "data" ในการเก็บข้อมูลนิสิตเท่านั้น!!!**

	Score	GPA
1	15	3.78
2	29	2.00
3	10	2.50
4	25	2.85
5	30	3.96

$$logit(p_i) = -3.98 + 0.2 * score_i + 0.5 * GPA_i$$

$$p(x_i) = \frac{1}{1 + e^{-logit(p_i)}}$$

การคูณเมทริกซ์ (dot operator)

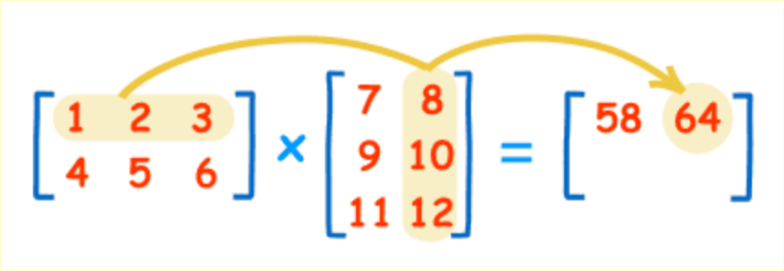


Figure source: https://www.mathsisfun.com/algebra/matrix-multiplying.html

```
import numpy as np
x = np.array([[1,2,3],[4,5,6]])
y = np.array([[7,8],[9,10],[11,12]])
print(x.dot(y))
np.dot(x,y)
```

การคูณเมทริกซ์ (dot operator) (CH08_3)

ร้านขายอาหารตามสั่งมีราคาอาหารคือ
ข้าวแกง 25 บาท
ข้าวผัด 30 บาท
สุกี้ทะเล 45 บาท
ในสัปดาห์ที่ผ่านมาที่ร้านขายอาหารได้ตามนี้

	จันทร์	อังคาร	พุธ	พฤหัส	ศุกร์
ข้าวแกง	75	120	70	90	80
ข้าวผัด	80	90	100	70	50
สุกี้ทะเล	50	45	70	65	50

ในสัปดาห์ที่ผ่านมาร้านอาหารมีรายได้เท่าไหร่

ฟังก์ชัน sum() ใน Numpy

	จันทร์	อังคาร	พุธ	พฤหัส	ศุกร์
ข้าวแกง	75	120	70	90	80
ข้าวผัด	80	90	100	70	50
สุกี้ทะเล	50	45	70	65	50

โค้ดจากหน้าที่แล้ว

```
import numpy as np
x = np.array([25,30,45])
y =
np.array([[75,120,70,90,80],[80,90,100,70,50],[50,45
,70,65,50]])
print(np.sum(y))
print(np.sum(y, axis=0)) # บวกแนวตั้ง
print(np.sum(y, axis=1)) # บวกแนวนอน
```

จำนวนจานทั้งหมดที่ขายได้แต่ละวัน

อาหารแต่ละประเภทที่ขายได้ใน 1 สัปดาห์

ฟังก์ชัน mean(), std(), max(), argmax()ใน Numpy

	จันทร์	อังคาร	พุธ	พฤหัส	ศุกร์
ข้าวแกง	75	120	70	90	80
ข้าวผัด	80	90	100	70	50
สุกี้ทะเล	50	45	70	65	50

โคดจากหน้าที่แล้ว

```
import numpy as np
x = np.array([25,30,45])
y =
np.array([[75,120,70,90,80],[80,90,100,70,50],[50,45
,70,65,50]])
```

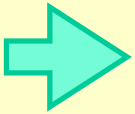
ประเภทของอาหารที่ขายได้โดยเฉลี่ยต่อวัน

```
print(np.mean(y, axis=1)) # ค่าเฉลี่ยแนวนอน
print(np.std(y, axis=1))
print(np.max(y, axis=1))# 120, 100, 70
print(np.argmax(y, axis=1)) # 1, 2, 2
```

เมทริกซ์ทรานสโพส (matrix transpose)

```
import numpy as np
y =
np.array([[75,120,70,90,80],[80,90,100,70,50],[50,45,70,65,50]])
print(y)
print(y.T)
```

	จันทร์	อังคาร	พุธ	พฤหัส	ศุกร์
ข้าวแกง	75	120	70	90	80
ข้าวผัด	80	90	100	70	50
สุกี้ทะเล	50	45	70	65	50



	ข้าวแกง	ข้าวผัด	สุกี้ทะเล ทะเล
จันทร์	75	80	50
อังคาร	120	90	45
พุธ	70	100	70
พฤหัส	90	70	65
ศุกร์	80	50	50

ลิสต์อินูเมอเรต (enumerate)

```
>>years = ['Freshy','Sophomore','Junior','Senior']
>>list(enumerate(years))
[(0, 'Freshy'), (1, 'Sophomore'), (2, 'Junior'), (3, 'Senior')]
>>list(enumerate(years, start=1))
[(1, 'Freshy'), (2, 'Sophomore'), (3, 'Junior'), (4, 'Senior')]
```

```
for i, year in enumerate(years):
    print(i, year)
```

```
0 Freshy
1 Sophomore
2 Junior
3 Senior
```

ได้ผลออกมาเป็นลิสต์ของ tuple โดยมี index ของตำแหน่งข้อมูล และ ค่าของข้อมูล

เอ็นดีอินูเมอเรต (ndenumerate)

```
import numpy as np
a = np.array([[1,2],[3,4]])
for index,x in np.ndenumerate(a):
    print(index,x)
```

```
(0, 0) 1
(0, 1) 2
(1, 0) 3
(1, 1) 4
```

ได้ผลออกมาเป็น

- ลิสต์ของ tuple ลำดับข้อมูลในอาร์เรย์
- และ ค่าของข้อมูล

การอ่านข้อมูลจากไฟล์เข้าอาเรย์

ตัวอย่างไฟล์ข้อมูล ArrayData.csv

```
# this is a header
# this file is generated on Oct 7, 2015
# it is just an example file for NumPy I/O
1;1;1;1;1;1;1;1;1;1
1;2;1;2;1;2;1;2;1;2;1
2;2;2;2;2;2;2;2;2;2;2
3;3;3;3;3;3;3;3;3;3;3
4;5;6;6;7;8;9;5;1;3;2
```

การอ่านข้อมูลจากเท็กซ์ไฟล์เข้าอาเรย์ (ต่อ)

ใช้ **loadtxt** อ่านข้อมูลทั้งหมด **(ทุกคอลัมน์)**

```
import numpy as np
data = np.loadtxt("ArrayData.csv",skiprows=3, \
         delimiter=";")
print(data)
```

การอ่านไฟล์เลือกเอามาเฉพาะบางคอลัมน์

```
import numpy as np
data = np.loadtxt("ArrayData.csv",skiprows=3, \
         delimiter=";",usecols=(0,1,2))
print(data)
```

การเขียนข้อมูลอาเรย์ไปเท็กซ์ไฟล์

ใช้ **savetxt** เขียนข้อมูลอาร์เรยใน **Numpy** ใส่ในไฟล์

```
import numpy as np

Matrix_a = 2 * np.ones(shape=(10000,10000))
np.savetxt("Matrix_a.csv", Matrix_a, fmt='%.1e',\
    newline="\n", delimiter=";")
```

ลองเขียนดู (CH08_04)

อ่านไฟล์คะแนนเก็บของนิสิตและทำการคำนวณหาคะแนนรวมโดยใช้ loadtxt() และ savetxt()

```
# scores of students in IT521 class
# year: 2557/1
STUDENT_ID,hw1,hw2,quiz1,exam1,quiz2,exam2
5622770071,4,4,2,28,1,38
5622770790,5,3,1,30,2,28
5622771079,2,4,3,12,3,34
5622771152,3,5,4,13,4,23
```

ผลลัพธ์

```
77.00
69.00
58.00
52.00
```

Application 1: kNN

http://glowingpython.blogspot.com/2012/04/k-nearest-neighbor-search.html

Application1: kNN

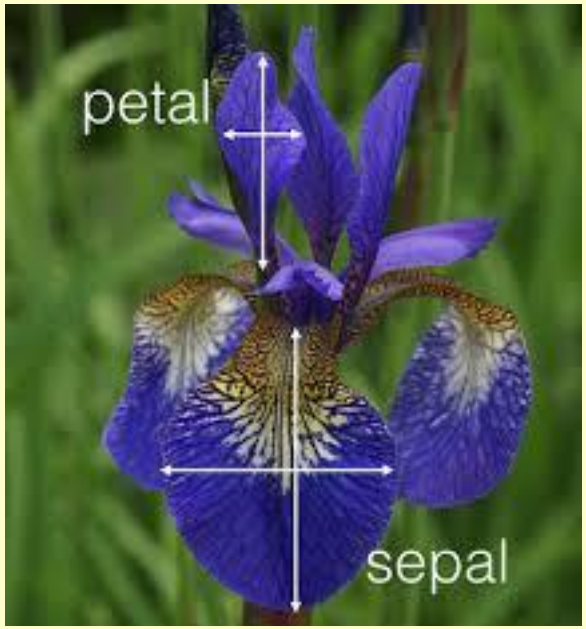
- k-Nearest Neighbors (kNN) เป็นวิธีการทำนายประเภทข้อมูลที่ได้รับความนิยม และสามารถพัฒนาได้ง่าย
- **ขั้นตอนที่ 1:** นำข้อมูลที่ต้องการทำนายประเภทมาหาระยะห่าง (distance) ดังสูตรด้านล่าง กับข้อมูลตัวอย่างเรียนรู้ (training data) **ทุกๆ อัน**

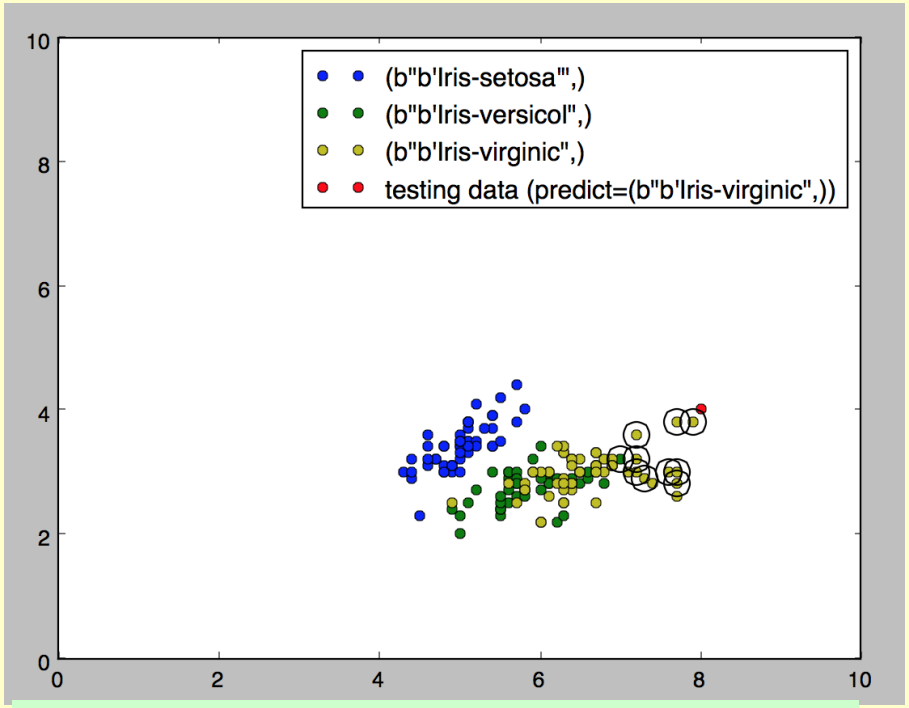
$$d(p,q) = d(q,p)$$
$$= \sqrt{(q_1 - p_1)^2 + (q_3 - p_3)^2 + \cdots + (q_n - p_n)^2}$$
$$= \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

- **ขั้นตอนที่ 2:** จากนั้นจะเลือกข้อมูลที่มีความใกล้เคียงที่สุด k ตัวอย่าง มาเพื่อ**โหวต** ซึ่งผลการทำนายจะคือ ประเภทข้อมูลที่ได้รับโหวตสูงสุด

Application1: kNN (ต่อ)

- งานชิ้นนี้ผู้ใช้กรอก (1) ความยาวกลีบดอก และ (2) ความกว้างกลีบดอก
- จากนั้นนำมา**ทำนาย**ประเภทของดอก iris ซึ่งมี 3 ประเภทได้แก่ “Iris-setosa”, “Iris-versicolor”, “Iris-virginica” **ด้วย kNN (k=10)**
- ข้อมูลตัวอย่างสามารถโหลดได้จากไฟล์ “iris.csv” โดยประกอบไปด้วยข้อมูล 150 ตัวอย่าง (ประเภทละ 50 ตัวอย่าง)
  - Link: http://tinyurl.com/2110101resource






```
Enter length & width: 8 4
(b"b'Iris-virginic",)
```