

# Linear Algebra

## HW3 Cosine Transform

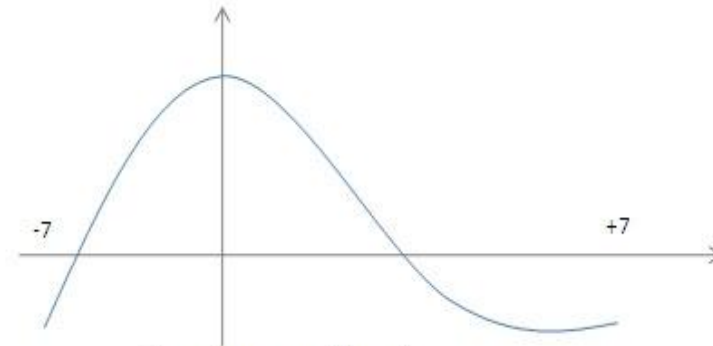
吳思霖

# Outline

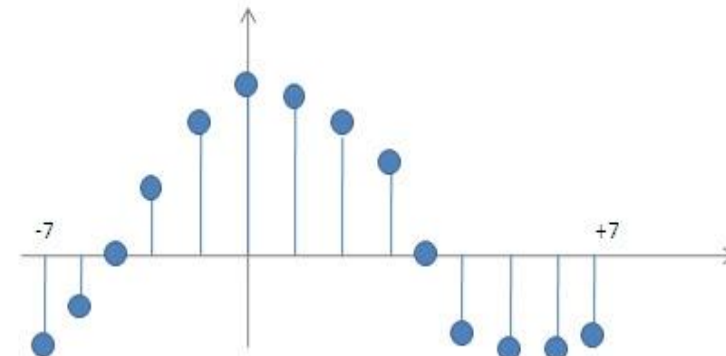
- What is signal
- Fourier Transform
- Cosine Transform
- HW3
  - Input&output
  - Code&util function
  - Rules

# What is signal

- 傳遞有關一些現象的行為或屬性的資訊的函數
- Example
  - $f(t)$ : 音訊
  - $f(x, y)$ : 圖片
- Type
  - Continuous
  - Discrete
- In this homework, we use
  - discrete signal



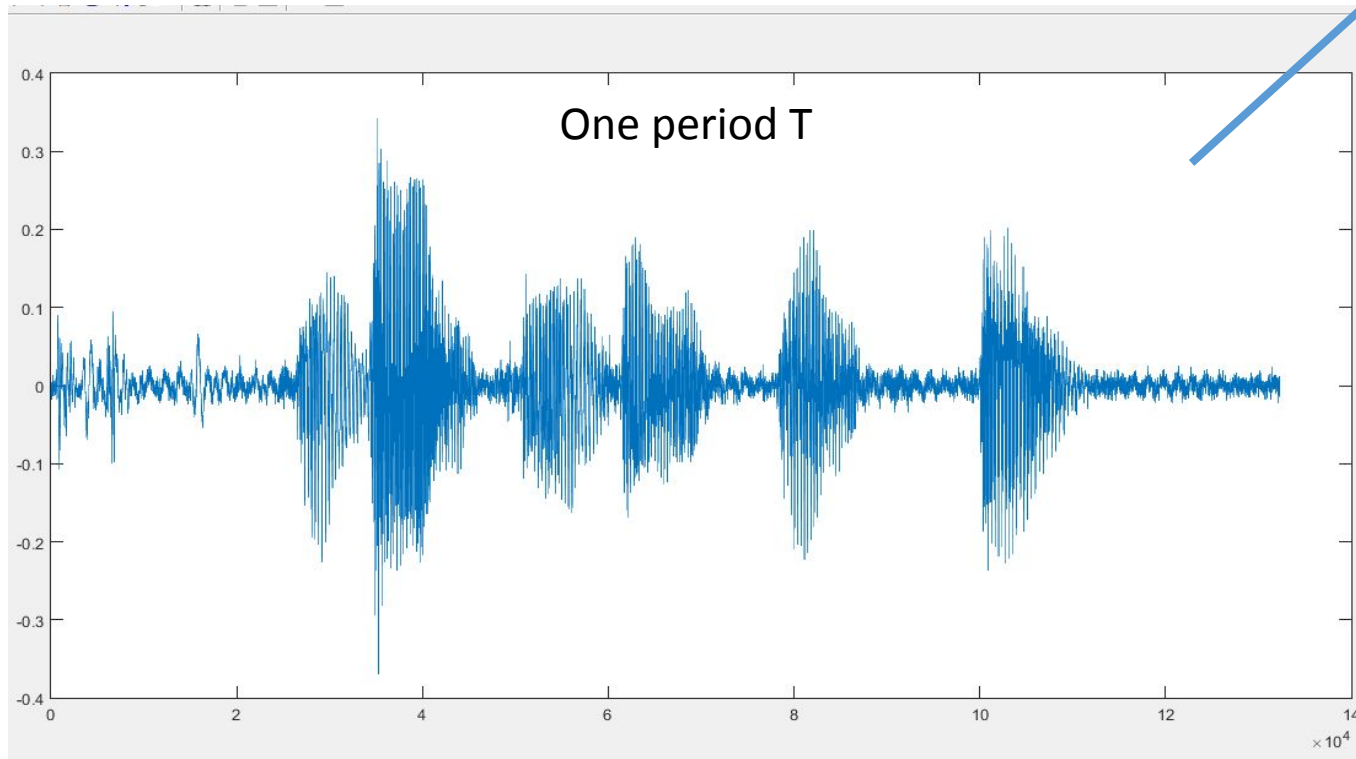
Continuous Signal  
(takes values in the set  $[-7, 7]$ )



Discrete Signal  
(takes values at the integers  $\{-7, -6, \dots, 0, \dots, 6, 7\}$ )

# Basis

- How to use basis in signal analysis
- Given a speech signal
  - Can we find basis to describe this signal ?



$$\mathbf{x} = [x_0, x_1, \dots, x_{N-1}]$$

$$\mathbf{B} = \{\mathbf{b}_0, \mathbf{b}_1, \dots, \mathbf{b}_{N-1}\}$$
$$[\mathbf{x}]_{\mathbf{B}} = [a_0, a_1, \dots, a_{N-1}] = \mathbf{a}$$
$$\mathbf{x} = \sum_{k=0}^{N-1} a_k \mathbf{b}_k$$

exists  $\mathbf{b}$  that is easy to analysis?  
(basis is also a signal)

# Joseph Fourier

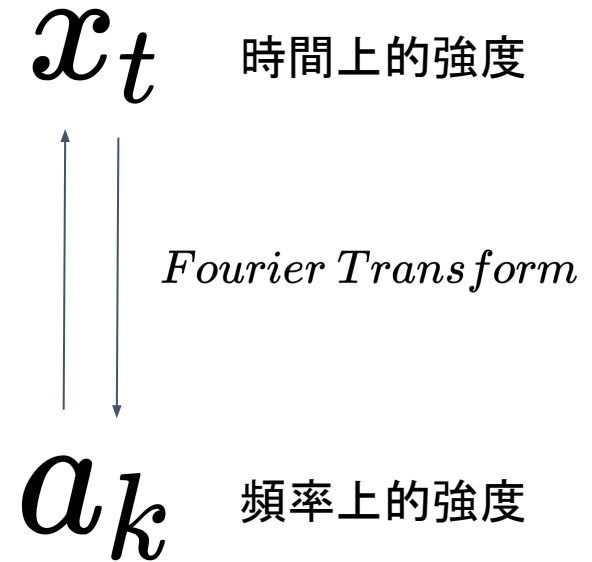
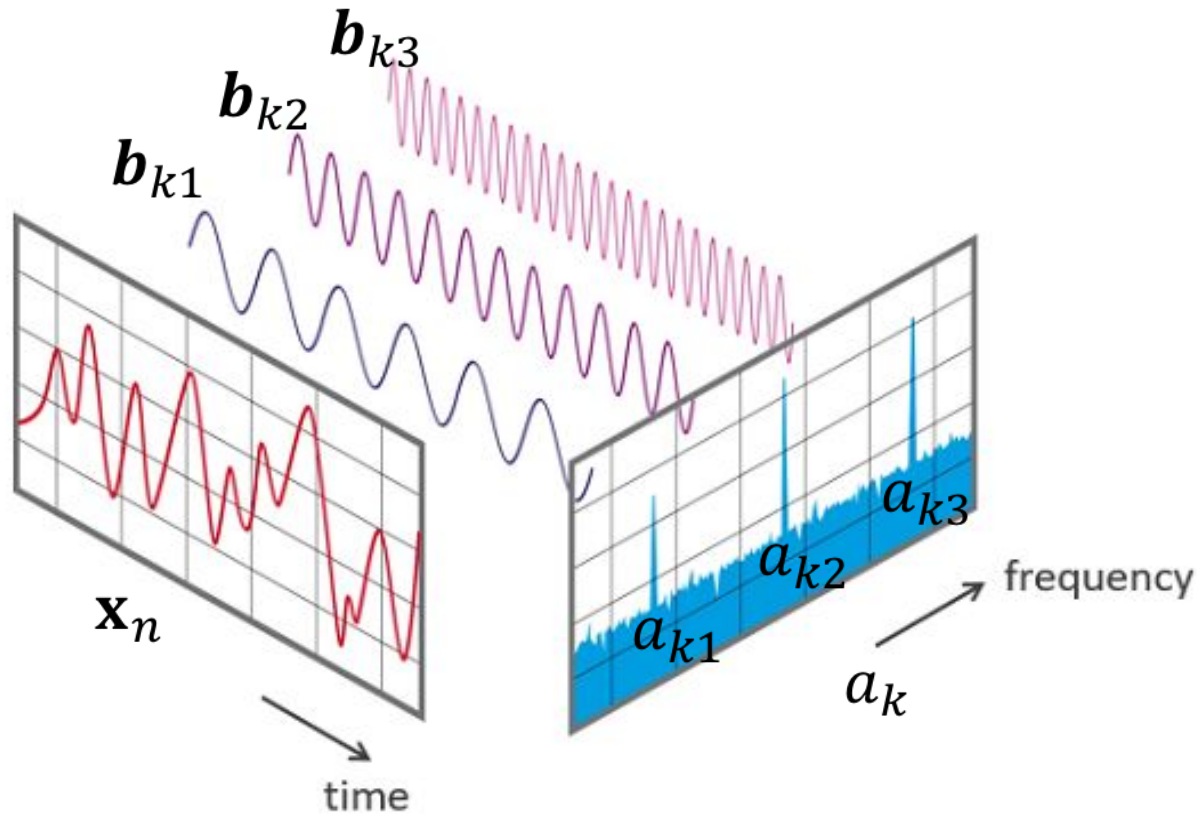


Any **periodic** signal  
can be represented  
as a sum of  
sinusoids.

# Fourier Transform

假設 $x$ 主要由三個basis vector組成  
→三個頻率的cosine signal組成

$$\mathbf{x} = a_{k1}\mathbf{b}_{k1} + a_{k2}\mathbf{b}_{k2} + a_{k3}\mathbf{b}_{k3} + \cdots$$
$$0 \leq k1, k2, k3 \leq N - 1$$



# Cosine Transform

- Fourier Transform includes complex number computation
  - We use cosine transform instead
- Cosine Transform Formula
  - Given a discrete signal  $x = [x_0, x_1, \dots, x_n, \dots, x_{N-1}]$  with N length
  - Basis Matrix:
- $\mathbf{B} = \{\mathbf{b}_0, \mathbf{b}_1, \dots, \mathbf{b}_{N-1}\}$

$$\mathbf{b}_{k,n} = \begin{cases} \frac{1}{\sqrt{N}}, & \text{if } k = 0 \\ \frac{\sqrt{2}}{\sqrt{N}} \cos \frac{(n + 0.5)k\pi}{N}, & \text{else} \end{cases}$$

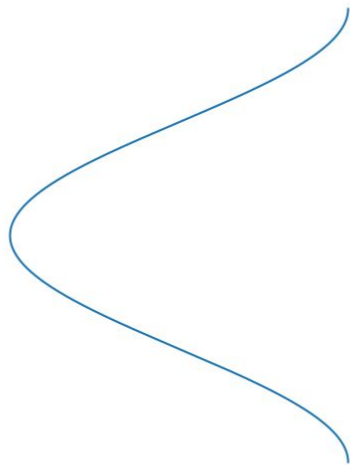
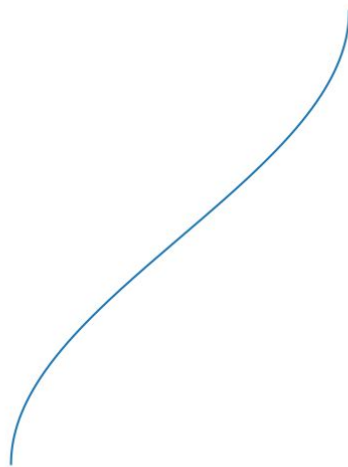
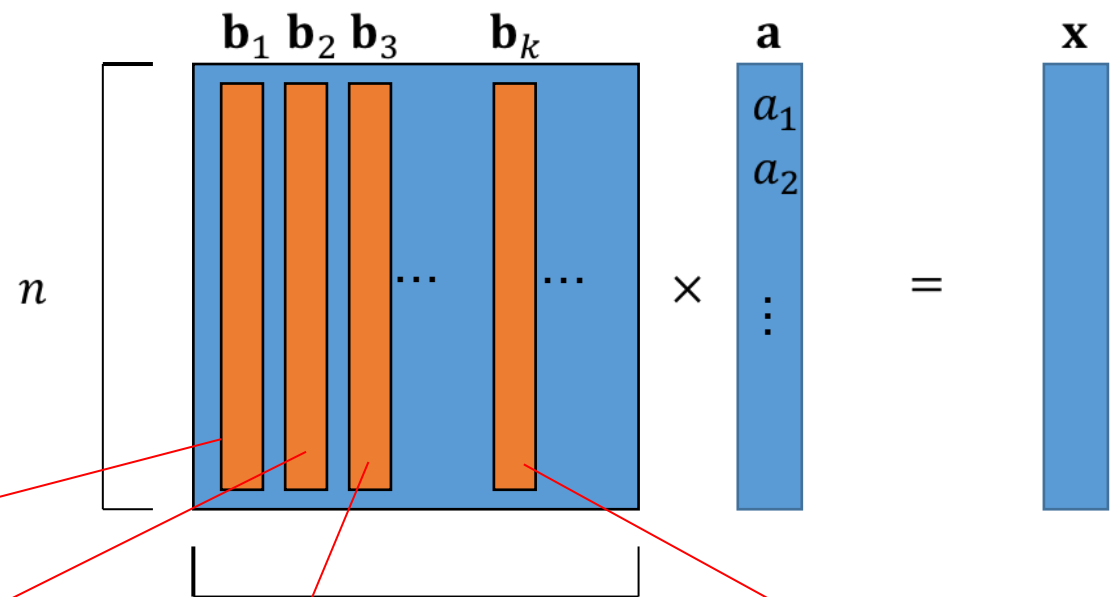
How to get a



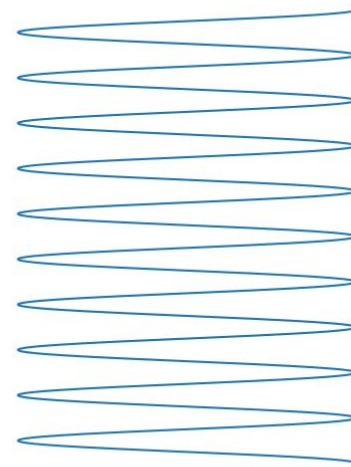
$$\begin{aligned} \mathbf{x} &= \mathbf{B}\mathbf{a} \\ \mathbf{a} &= \mathbf{B}^{-1}\mathbf{x} \end{aligned}$$

n: 時間上的index  
k: 頻率上的index

$$\mathbf{b}_{k,n} = \begin{cases} \frac{1}{\sqrt{N}}, & \text{if } k = 0 \\ \frac{\sqrt{2}}{\sqrt{N}} \cos \frac{(n + 0.5)k\pi}{N}, & \text{else} \end{cases}$$

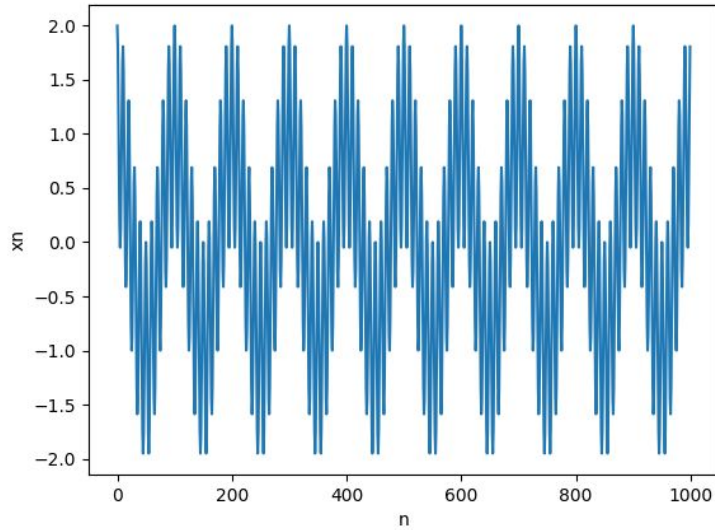


$k$



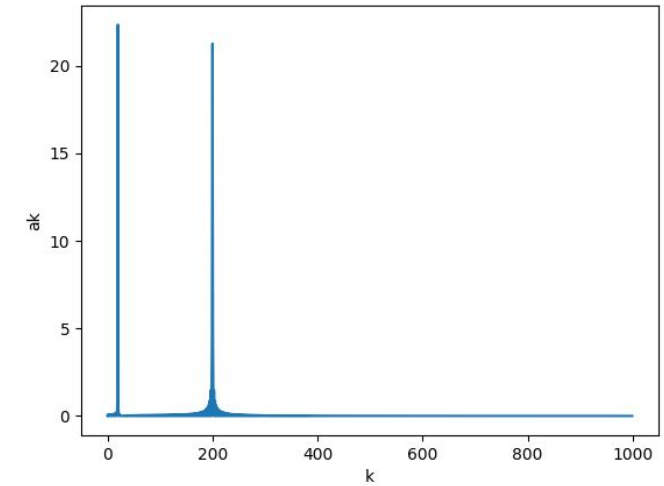


If we want to get the low frequency signal of a mixed signal...



# Cosine transform

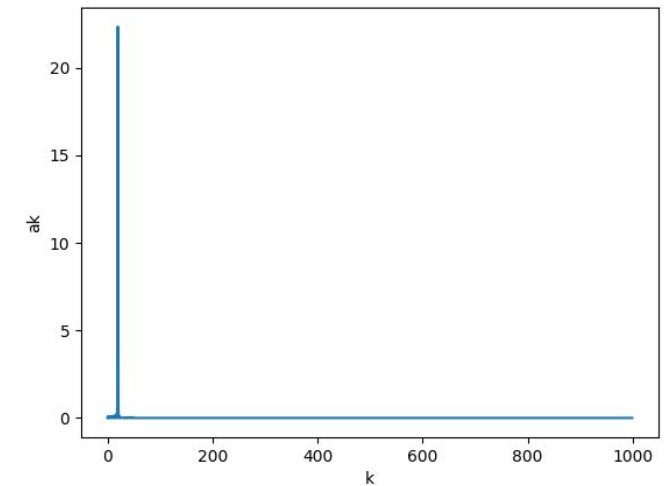
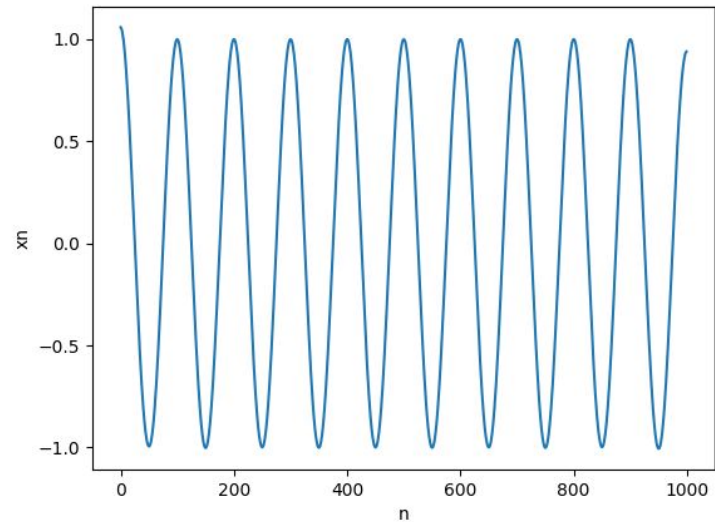
$$\mathbf{a} = \mathbf{B}^{-1}\mathbf{x}$$



mask [ 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  
...]

## Inverse cosine transform

$$\mathbf{x} = \mathbf{a}^T \mathbf{B}$$



# Application

- Filter
  - Human voice
    - Man: 85-180Hz
    - Woman: 165-255Hz
- Remove high frequency noise from speech signal

# Input & Output

- $x = \sum_{i=1}^5 \text{Cosine}(2\pi f_i)$ 
  - $f_1 < f_2 < f_3 < f_4 < f_5$
- Input data
  - Please download the file of <student\_id>.txt
  - Total 1000 lines, one value per line.
- Output  $f_1$  and  $f_3$  signal to following file
  - $f_1$  : <student\_id>\_f1.txt ( ex: b01901118\_f1.txt )
  - $f_3$  : <student\_id>\_f3.txt
  - Use `numpy.savetxt` to output the answer signal
  - Same format as input: 1000 lines
- Output the picture of **a** of input signal
  - <student\_id>\_freq.png
  - Use util function
- You can use test.txt for testing.
  - Only two cosine signal with different freq are mixed.

```
1 2.0000000000000000e+00
2 1.807043722803219010e+00
3 1.301131695689425438e+00
4 6.732702563537411589e-01
5 1.595661667536837358e-01
6 -4.894348370484646882e-02
7 1.207594915133041180e-01
8 5.958100580910720145e-01
9 1.185323674418810924e+00
10 1.653344919876962527e+00
11 1.809016994374947451e+00
12 1.579530237150736927e+00
13 1.037985621796358338e+00
14 3.755301115537416079e-01
15 -1.715930046262574837e-01
16 -4.122147477075268629e-01
17 -2.731901993959511277e-01
18 1.727366797267690379e-01
19 7.347962859400198887e-01
20 1.177141547059625148e+00
21 1.309016994374947451e+00
22 1.057706881539801413e+00
23 4.963983089606727184e-01
24 -1.836837608106426378e-01
25 -7.462264748456348684e-01
26 -9.99999999999998890e-01
27 -8.718075139042609223e-01
28 -4.343502279392522092e-01
29 1.216356797892221842e-01
30 5.603271072100921568e-01
31 6.909830056250526598e-01
32 4.408924416902699206e-01
33 -1.167622971901245421e-01
34 -7.907706684766655503e-01
```

# Code & Data link

- [Link](#)
- Code
  - hw3.py
- Data
  - <student\_id>.txt
  - test.txt

# HW3 Code

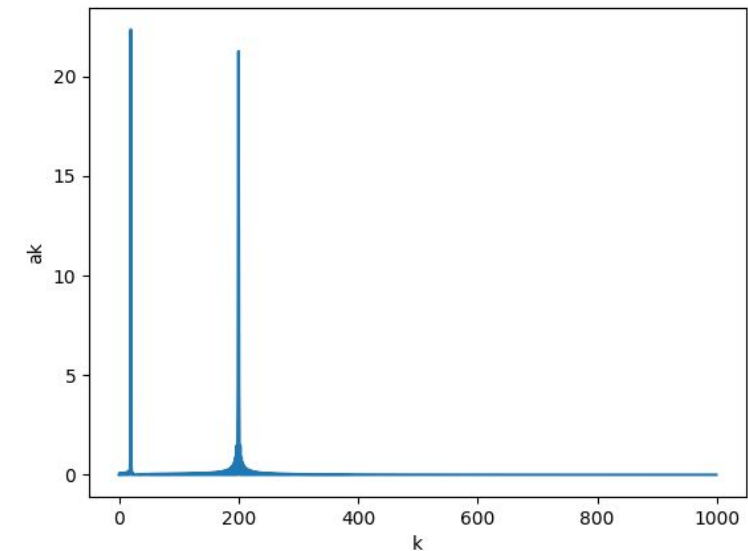
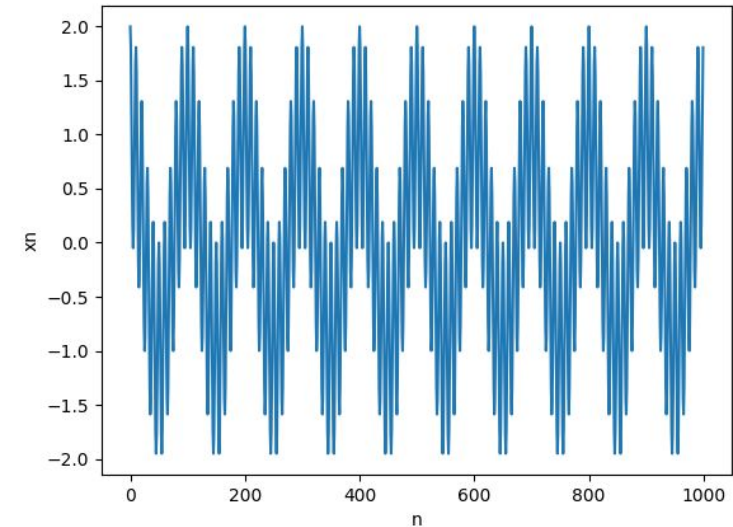
```
5 def CosineTrans(x, B):  
6     # TODO  
7     # implement cosine transform  
8     return  
9  
10 def InvCosineTrans(a, B):  
11     # TODO  
12     # implement inverse cosine transform  
13     return  
14  
15 def gen_basis(N):  
16     # TODO  
17     return  
18  
19 if __name__ == '__main__':  
20  
21     signal_path = sys.argv[1]
```

# Run the code

- `python hw3.py <input_signal_txt>`
- Your code should generate 2 txt files and 1 png file.
  - `<student_id>_f1.txt`
  - `<student_id>_f3.txt`
  - `<student_id>_freq.png`
  - These three files should be **in the same folder** with `hw3.py`

# Some util function

```
9
10 def plot_wave(x, path = './wave.png'):
11     # util function
12     plt.gcf().clear()
13     plt.plot(x)
14     plt.xlabel('n')
15     plt.ylabel('xn')
16     plt.savefig(path)
17
18 def plot_ak(a, path =  './freq.png' ):
19     # util function
20     plt.gcf().clear()
21
22     # Only plot the mag of a
23     a = np.abs(a)
24     plt.plot(a)
25     plt.xlabel('k')
26     plt.ylabel('ak')
27     plt.savefig(path)
28
```



# Scoring

1. Plot the figure of  $a_k$ . (2%)
2. Output correct  $f_1$  signal (2%)
3. Output correct  $f_3$  signal (2%)



# Submit

- Code you download  
|-- hw3.py
- Code you **submit** should be put in a folder and compressed in a **zip** file  
r07922072\_hw3.zip  
|-- ./r07922072\_hw3  
|-- hw3.py  
|-- r07922072\_f1.txt  
|-- r07922072\_f3.txt  
|-- r07922072\_freq.png

# Standard Rules

- 不要抄作業，不要交別人的答案，作弊一律0分計算
- 上傳 zip 檔案到 CEIBA
- 注意繳交的資料夾學號開頭英文用**小寫**
- **DEADLINE: 2018/11/15(四) 23:59 (GMT+8:00)**
- **遲交每過一天: 分數 $\times 0.8$  (per day)**
- **格式、檔案、各種奇怪的錯誤讓我無法改作業: 分數 $\times 0.8$**

# Code Rules

- You **can't**
  - Use cosine transform formula in appendix to generate seperated signal
    - 只准使用inverse matrix的方法產生指定的三個檔案
    - 可以實作, 但僅限於檢查inverse matrix的方法是否正確
  - import scipy
    - Or other cosine transform package

# Appendix - Cosine Transform Formula

- Cosine transform

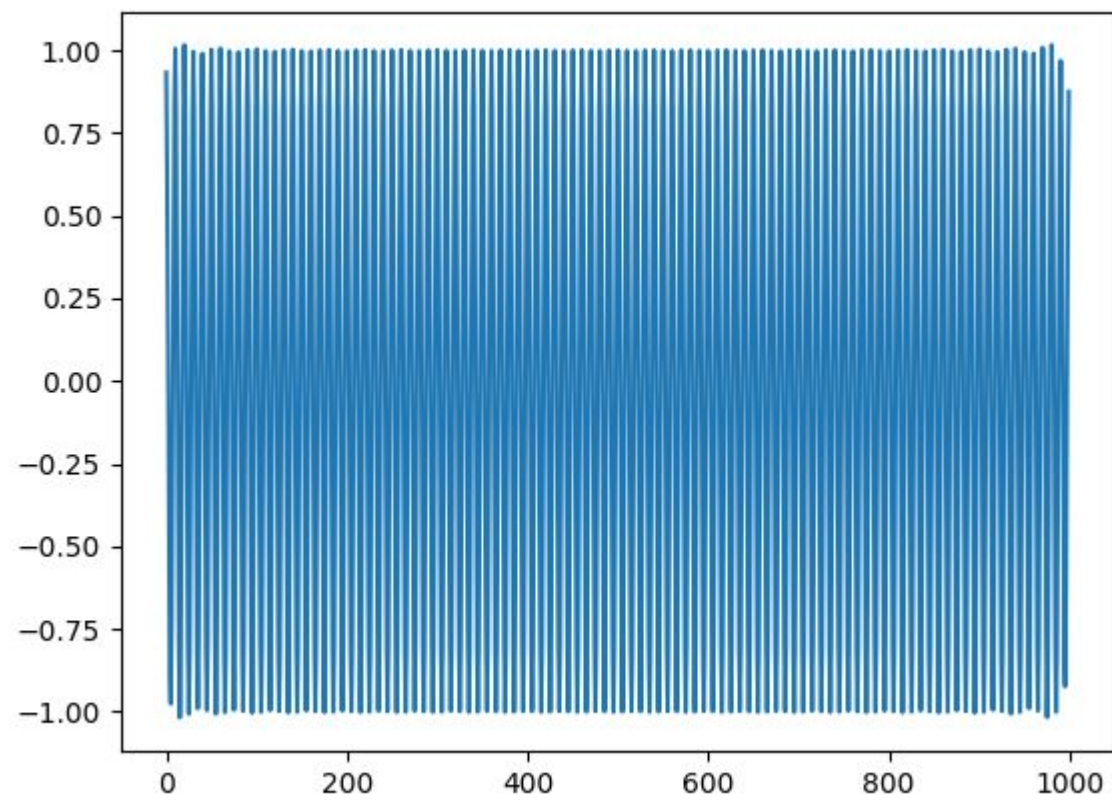
$$a_k = s_k * 2 \sum_{n=0}^{N-1} x_n \cos\left(\frac{\pi}{N} k \left(n + \frac{1}{2}\right)\right)$$

$$\boxed{\begin{array}{l} \text{if } k = 0, s_0 = \sqrt{\frac{1}{4N}} \\ \text{else, } s_i = \sqrt{\frac{1}{2N}} \end{array}}$$

- Inverse Cosine transform

$$x_n = \frac{1}{\sqrt{N}} a_0 + \frac{\sqrt{2}}{\sqrt{N}} \sum_{k=1}^{N-1} a_k \cos\left(\frac{\pi}{N} k \left(n + \frac{1}{2}\right)\right)$$

# Update - test.txt的高頻波型



# Update - 改作業的流程

假設同學的學號是r07922072, 且對應的波檔是r07922072.txt

我會run

```
>>python hw3.py r07922072.txt
```

在程式中, r07922072.txt這個字串是由sys.argv[1]傳進程式裡的

同學的檔案要可以生出r07922072\_f1.txt、r07922072\_f3.txt、r07922072\_freq.png, 且在同一個資料夾下

# Update - inverse matrix

同學可以直接用`numpy.linalg.inv`這個函式拿inverse matrix

[Ref](#)