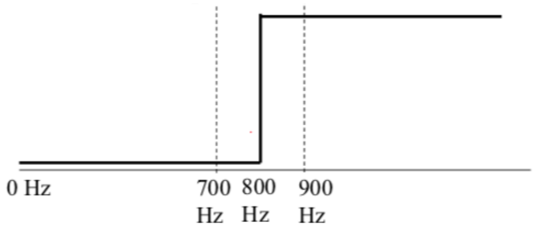
(1) Design a Mini-max highpass FIR filter such that (40 scores)
Filter length = 19, Sampling frequency fs = 4000Hz,
Pass Band 800~2000Hz , Transition band: 700~900 Hz,
Weighting function: W(F) = 1 for passband, W(F) = 0.5 for stop band
Set Δ = 0.0001 in Step 5.



(a) the Matlab program, (b) the frequency response,
(c) the impulse response h[n], and (d) the maximal error for each iteration.

Ans:
(a) Python program

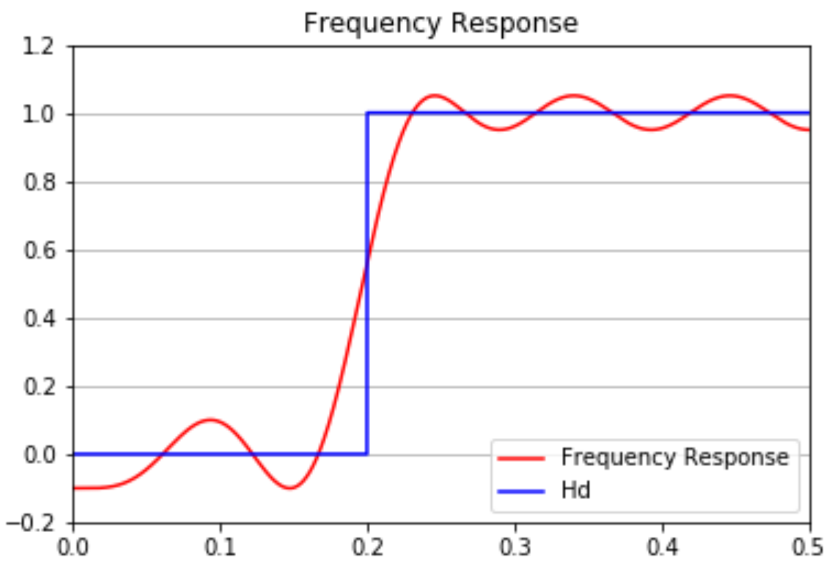
```
import numpy as np
from numpy.linalg import inv

import matplotlib.pyplot as plt
from matplotlib.pyplot import figure


def Weight(F):

    w_low = np.where(F <= transition_low, W_stop, 0)
    w_high = np.where(F >= transition_high, W_pass, 0)
    W = w_low + w_high

    return W

def get_M(F):

    W = Weight(F)
    M = np.cos(F.reshape(k+2,1) * np.arange(k+1) * 2 * np.pi)
    W_inv = np.reciprocal(W) * np.array([(-1) ** x for x in range(k+2)])
    M = np.concatenate((M, W_inv.reshape(k+2,1)), axis = 1)

    return M

def get_Hd(F):

    H = np.array(F > mid) * 1

    return H

def get_R(F,s):

    n = len(F)
    R = np.cos(F.reshape(n, 1) * np.arange(k+1) * 2 * np.pi)
    R = np.matmul(R, s)

    return R

def get_err(F, s):

    Hd = get_Hd(F)
    W = Weight(F)
    R = get_R(F, s)
    err = (R - Hd) * W

    return err
```

```python
def find_peak(err):

    err_shiftRight = np.concatenate(([0], err[:-1]))
    err_shiftLeft = np.concatenate((err[1:], [0]))
    peak_check = (err - err_shiftLeft) * (err - err_shiftRight)
    F_peak = np.array([x for x,y in enumerate(peak_check) if y > 0], dtype = 'uint32')
    P = F_peak[1:-1] # avoid boundary first
    number_select = len(P)

    Boundary_right = n_sample - 1

    if number_select == k:
        P = F_peak
    elif number_select == k + 1:
        if F_peak[0] == 0 and F_peak[-1] == Boundary_right:
            x = 0 if abs(err[0]) > abs(err[Boundary_right]) else Boundary_right
            P = np.append(P, x)
            P = np.sort(P)

        elif F_peak[0] != 0 and F_peak[-1] == Boundary_right:
            P = np.concatenate((F_peak[0], P))

        else:
            P = np.append(P, F_peak[-1])
    else:
        P = P[-(k + 2):] # pick the last k+2 elements from passband to stopband

    return F_sample[P]


def plot(curve, title=''):

    figure(figsize=(6, 4))
    plt.title(title)

    if title == 'Frequency Response':

        x = F_sample
        Hd_F = get_Hd(F_sample)
        plt.plot(x, curve, color='red', label='Frequency Response')
        plt.plot(x, Hd_F, color='blue', label='Hd')
        plt.xlim(0, 0.5)
        plt.ylim(-0.2, 1.2)
        plt.xlabel('Normalized Frequency')
        plt.grid(axis='y')
        plt.legend()
        plt.savefig("Frequency_Response.png")
        plt.show()


    elif title == 'Impulse Response':
        x = list(range(N))
        plt.stem(x, curve, linefmt=None, markerfmt=None, basefmt=None, use_line_collection=True, label='h[n]')
        plt.ylim(-0.4, 0.7)
        plt.xlabel('N')
        plt.grid(axis='y')
        plt.xticks(x)
        plt.legend()
        plt.savefig("Impulse_Response.png")
        plt.show()
```

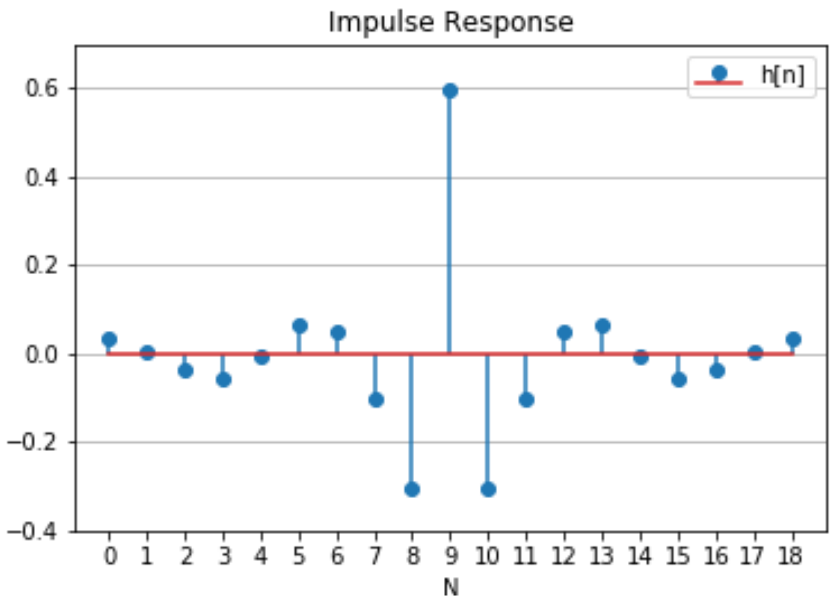
```python
# ### Initialization

N = 19 #filter lenth
k = int((N - 1) / 2) # mid point index = 9
n_sample = 5001
F_sample = np.linspace(0, 0.5, n_sample)
fs = 4000 #sampling frequency

# Transition band

transition_low = 700
transition_high = 900
mid = (transition_low + transition_high) / 2

# Normalization

transition_low = transition_low / fs # 0.175
transition_high = transition_high / fs # 0.225
mid = mid / fs # 0.2

# Weighting function
W_pass = 1
W_stop = 0.5

delta = 0.0001
print(transition_low, transition_high, mid)


# ### Find k+2 Extreme Points

# Step 1, Choose arbitrary k+2 extreme frequencies in the range of (0, 0.5), transtion band excluded

n_extreme = int(k+2)
n_low = int(n_extreme / 2)
n_high = int(n_extreme / 2) if n_extreme%2==0 else int(n_extreme / 2) + 1

Fm_low = np.linspace(0, transition_low, n_low)
Fm_high = np.linspace(transition_high, 0.5, n_high)
Fm = np.concatenate((Fm_low, Fm_high))

E1 = float("inf")
iteration = 0
while(True):
    iteration += 1

    # Step 2: Compute s[n]
    Hd = get_Hd(Fm)
    M = get_M(Fm)
    s = np.matmul(inv(M), Hd)[:-1]
    R = get_R(Fm,s)

    # Step 3: Compute err(F) for 0 <= F <= 0.5, exclude the transition band.

    err_F = get_err(F_sample, s)

    # Step 4: Find k+2 local maximal (or minimal) points of err(F)

    ExtremePoints = find_peak(err_F)

    # Step 5:

    E0 = np.amax(abs(err_F))
    print("Iteration %d , Max Error = %.4f" %(iteration, E0))

    if E1 - E0 <= delta and E1 - E0 >= 0:
        break
    else:
        E1 = E0
        Fm = ExtremePoints
```

```
# ### Print The Result

# Step 6: Calculate h[n], print Frequency Response and Impulse Response

R_F  = get_R(F_sample, s)
plot(R_F, title='Frequency Response')

hn = np.zeros(N)
hn[k] = s[0]

for i in range(1,k+1):
    hn[k+i] = s[i] / 2
    hn[k-i] = s[i] / 2

plot(hn, title='Impulse Response')
print(hn)
```

(b)



(c)



(d)

Iteration	1	2	3	4	5	6	7
Max_Error	0.1595	0.1048	0.1668	0.1120	0.0505	0.0500	0.0500

(2) Suppose that X(f) is the discrete-time Fourier transform of x(nΔt). Also suppose that we have known that Δt = 0.001 sec and
X(f) = 1 for | f | < 200 and X(f) = 0 for 200 < | f | < 500
Determine (a) X(900), (b) X(-1900), (c) X(6100). (10 scores)

Ans:

We know that

$$X(f) = \sum_{n=-\infty}^{\infty} x[n]e^{-j2\pi fn\Delta t}$$

and

$$X(f + \frac{1}{\Delta t}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j2\pi(f+\frac{1}{\Delta t})n\Delta t} = \sum_{n=-\infty}^{\infty} e^{-j2\pi n}x[n]e^{-j2\pi fn\Delta t}$$

Since multiplying by $e^{-j2\pi n}$ means a full rotation,

we then have

$$X(f + \frac{1}{\Delta t}) = X(f)$$

So the period of frequency domain is $\frac{1}{\Delta t} = \frac{1}{0.001} = 1000$

(a) X(900) = X(900 - 1000) = X(-100) = 1

(b) X(-1900) = X(-1900 +2000) = X(100) = 1

(c) X(6100) = X(6100 - 6000) = X(100) = 1

(3) From the view point of implementation, what are the disadvantages of the discrete Fourier transform? (5 scores)

Ans:

1. 運算積分範圍太大，負無限大到正無限大，無法分析一個訊號在局部的特性。
2. 實數的信號經過轉換後，也會變成複數運算，計算量變大。
3. 經常是無理數運算，只要用有限的bit來近似它，就會有誤差

(4) Suppose that x[n] = y(0.0002n) and the length of x[n] is 15000 and X[m] is the FFT of x[n]. Find m_1 and m_2 such that X[m_1] and X[m_2] correspond to the 200Hz and -300Hz components of y(t), respectively. (10 scores)

Ans:

$$f_s = \frac{1}{\Delta t} = \frac{1}{0.0002} = 5000$$

We know

$$f = m\frac{f_s}{N} \text{ , } for\ \ m \le \frac{N}{2}$$

$$f = (m - N)\frac{f_s}{N} \text{ , } for\ \ m > \frac{N}{2}$$

So

$$m_1 = f_1\frac{N}{f_s} = 200 \times \frac{15000}{5000} = 600$$

$$m_2 - N = f_2\frac{N}{f_s} = -300 \times \frac{15000}{5000} = -900$$

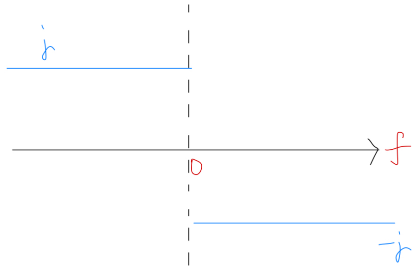
then

$$m_2 = -900 + 15000 = 14100$$

(5) Which of the following filters are odd? (i) bandpass filter, (ii) edge detector, (iii) differentiation 2 times, (iv) integration 3 times, (v) particle filter, (vi) the Hilbert. (10 scores)

Ans:

bandpass filter: even
Differentiation 2 times: equivalent to multiplying $j2\pi f$ twice, so it's even
Integration 3 times: equivalent to multiplying $\dfrac{1}{j2\pi f}$, so it's odd.
edge detection: we do differentiation to detect edges, so it's odd.
particle filter: for prediction, no such symmetric property.
Hilbert Transform:



So, (ii)edge detector, (iv) integration 3 times, and (vi) the Hilbert are odd.

(6) Estimate the length of the digital filter if both the passband ripple and the stopband ripple are smaller than 0.01, the sampling interval $\Delta t = 0.0002$, and the transition band is from 1600Hz to 1800Hz. (10 scores)

Ans:

passband ripple $\leq \delta_1 = 0.01$
stopband ripple $\leq \delta_2 = 0.01$

width of transition band $\leq \Delta F$

where $\Delta F = \dfrac{(f_1 - f_2)}{f_s} = (f_1 - f_2) \times \Delta t = (1800 - 1600) \times 0.0002 = 0.04$

Therefore,

$$\mathrm{N} = \frac{2}{3}\frac{1}{\Delta F} log_{10}(\frac{1}{10\delta_1\delta_2}) = \frac{2}{3}\frac{1}{0.04} log_{10}(\frac{1}{10 \times 0.01 \times 0.01}) = 50$$

(7) Use the MSE method to design the 9-point FIR filter that approximates the lowpass filter of $H_d(F) = 1$ for $|F| < 0.2$ and $H_d(F) = 0$ for $0.2 < |F| < 0.5$.

Ans:

We know

$$s[0] = \int_{-\frac{1}{2}}^{\frac{1}{2}} H_d(F)\, dF$$

and

$$s[n] = 2\int_{-\frac{1}{2}}^{\frac{1}{2}} cos(2\pi nF) H_d(F)\, dF$$

Given $H_d(F) = 1$ for $|F| < 0.2$, and $H_d(F) = 0$ for $0.2 < |F| < 0.5$

So

$$s[0] = \int_{-0.2}^{0.2} 1\, dF = 0.4$$

and

$$s[n] = 2\int_{-0.2}^{0.2} cos(2\pi nF)\, dF$$

$$= 2 \times [\frac{sin(2\pi n(0.2))}{2\pi n} - \frac{sin(2\pi n(-0.2))}{2\pi n}]$$

$$= \frac{2 \times sin(0.4\pi n)}{\pi n}$$

Finally, set $h[k] = s[0] = 0.4$

$$h[k+n] = h[k-n] = s[n]/2 \quad \text{for n} = 1, 2, 3, \ldots, \text{k},$$

$$h[n] = 0 \text{ for n} < 0 \text{ and n} \geq \text{N}$$

Here, N = 9 , and k = 4

So, the impulse response of the designed filter is

$$h[n] = \frac{sin(0.4\pi(4-n))}{\pi(4-n)} \qquad \text{for n} = 0, 1, 2, 3$$

$$h[n] = 0.4 \qquad \text{for n} = 4$$

$$h[n] = \frac{sin(0.4\pi(n-4))}{\pi(n-4)} \qquad \text{for n} = 5, 6, 7, 8$$

$$h[n] = 0 \qquad \text{for n} < 0 \text{ and n} \geq 9$$

(Extra): Answer the questions according to your student ID number. (ended with 0, 1, 2, 5, 6, 7) (15 scores)

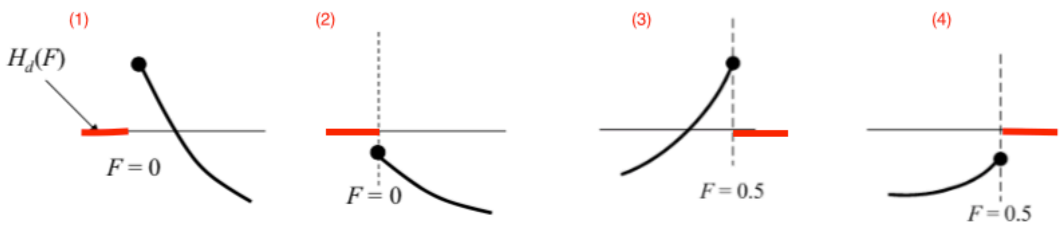
60

(4) Extreme points 判斷的規則：

(a) The local peaks or local dips that are not at boundaries must be extreme points.

boundaries: $F = 0$, $F =$ 0.5, 以及 transition band 的兩端

(b) For boundary points



Add a zero to the outside and conclude whether the point is a local maximum or a local minimum.

Ans:

Extreme points 必須比左右兩邊的點都大，或是比左右兩邊都小，

由圖可知，由左而右 (1), (3)為extreme points, (2), (4) 不是extreme points