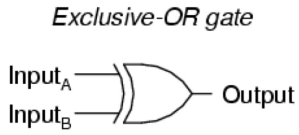
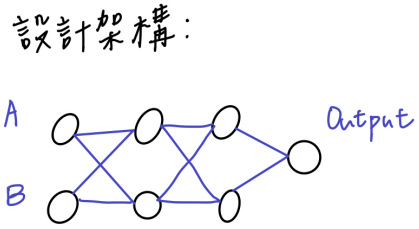
Homework 3　311512040 林胤宏

1. (Optimization algorithm for NN's training) As the statement in class, the single-layer perceptron cannot treat the XOR logic problem. Please use the genetic algorithm (GA) or particle swarm optimization (PSO) to train a neural network with at least one hidden layer for solving the XOR logic (two-input) problem. Hint: input/output of NN are (A, B)/output, respectively. ↵



A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0



file: PSO_XOR.ipynb

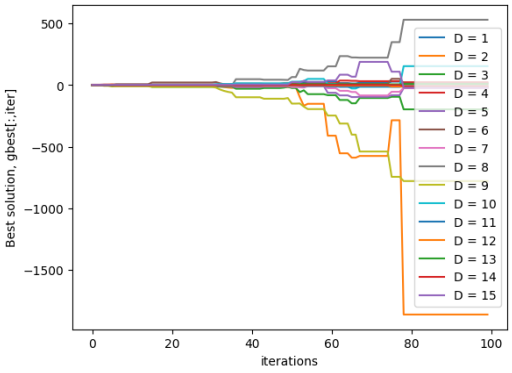
- 設計網路架構為兩層隱藏層，每層個兩個節點，加上bias共15個可調參數，加上設計30組particles，所以先隨機產生30*15的矩陣，考量到sigmoid活化函數的特性，設計隨機參數的範圍為-2~2。

```
pop_size = 30              #總共想設計幾組參數組合
dimemsion = 15             #每個組合的參數數量
x = np.random.uniform(-2,2,(pop_size,dimemsion))   #隨機產生-2~2的值
```

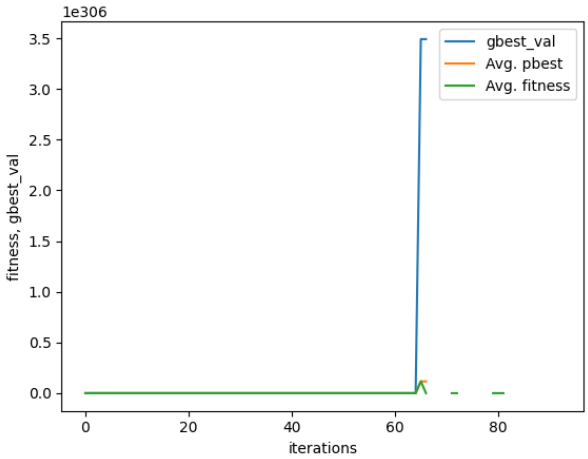
- PSO演算法的迭代次數設為100次，每次迭代一開始先透過 `forward(x)` function回傳30個particles的error，再將fitness設為1/error，以此作為根據來選出pbest及gbest。
- `pbest_val` 為大小30的矩陣，存放30個particles的最佳fitness值；`pbest` 為30*15的矩陣，存pbest的所有參數。`gbest_val` 為大小100的矩陣，存放每次迭代最佳fitness值；`gbest` 為100*15的矩陣，存每次迭代的gbest參數。

```
# pbest and pbest_val update
ind = np.argwhere(fitness > pbest_val)   # indices where current fitness value set is greater than pbset
pbest_val[ind] = fitness[ind]   # update pbset_val at those particle indices where fit > pbest_val
pbest[ind,:] = x[ind,:]     # update pbest for those particle indices where fit > pbest_val
```

```
# gbest and gbest_val update
ind2 = np.argmax(pbest_val)              # index where the fitness is maximum
gbest_val[iter] = pbest_val[ind2]        # 所有個體最佳解的最大值及為群體最佳解
gbest = pbest[ind2,:]                    # global best solution, gbest
```



上圖為15個變數經過迭代的變化。



上圖為經過迭代fitness值的變化，在60次迭代以後fitness已趨近無限大。

```
[[0.]
 [1.]
 [1.]
 [0.]]
```

迭代100次後的gbest參數能正確分類XOR問題。

2. (Optimization algorithm for controller design) The most commonly used model to describe the dynamics of chemical processes is the *First-Order Plus Time Delay Model*. By proper choice of τ_{DT} and τ, this model can be made to represent the dynamics of many industrial processes. Consider the **PID controller** design problem for the plant with transfer function $G(s) = \frac{Ke^{-\tau_{DT}s}}{\tau s+1} = \frac{e^{-s}}{10s+1}$ and $C(s) = (k_p + k_d s + k_i/s)$, this means that select the suitable parameters to optimize performance index. In addition, the usually utilized time response performance indices are rise time, setting time, overshoot, sum of square error, etc and gain margin/phase margin for time response and frequency response, respectively. (a) Please formulate an optimization problem by performance indices introduced above. (b) Use the PSO or GA to treat it and demonstrate your results. ↵

file:PSO_PID.ipynb

(a)設定理想的最大超越量、安定時間、穩態誤差，並設定error function(optimization problem)。

```
ideal_Mo=0.05
ideal_ts=2
ideal_Ess=0
error = (3*(Mo-ideal_Mo)**2+1*(ts-ideal_ts)**2+2*(ess-ideal_Ess)**2)**(1/2)  #定義loss function
```

- error function: 各項比重經由實驗來設計，穩態誤差項的尺度較小，但對於響應的影響也不大，因為大部分實驗的穩態誤差都收斂到可接受範圍，所以設計參數2；不希望產生過多的最大超越量以及震盪，所以設計最大的懲罰比重。

$$\sqrt{3*(Mo-ideal\_Mo)^2+1*(ts-ideal\_ts)^2+2*(ess-ideal\_Ess)^2}$$↵

(b)PID參數優化，利用在python中import `control` 這個套件，它可以方便創造出轉移函數，以及得到系統的響應等。time delay利用一階`control.pade`的方式來近似，一開始用一階近似，但模擬延遲的效果不佳，後來選擇用五階數學式近似，但因為引進了右半面的零點，所以在t=0附近出現低射現象，響應則分析T=1~10內的範圍。

```
(num,den)  =  control.pade(1,n=5)       # 用pade近似延遲器
time_delay = control.tf(num,den)

t, y = control.step_response(plant,T=10) # step_response()方法求單位步階響應
```

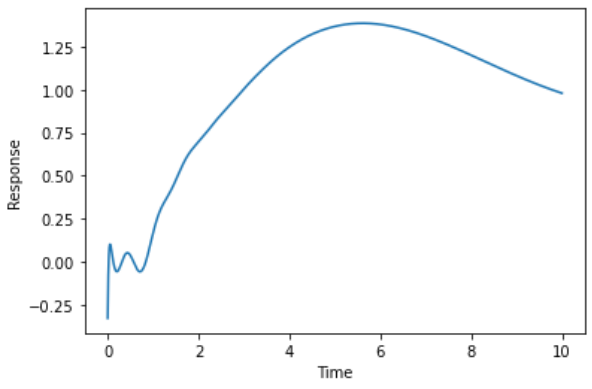
利用下列三個function來取得所需指標，fitness則一樣定為1/error。

```
def max_overshoot(t,y):
    max_value = y[np.argmax(y)]
    if max_value < 1:
        return 0
    else:
        return max_value-1

def settling_time(t,y,percentage):          #回傳安定時間(從最後的index找回來)
    for i in range(y.size):
        if (y[y.size-1-i] > 1+percentage or y[y.size-1-i] < 1-percentage) :
            return t[y.size-1-i]

def steady_state_error(t,y):    #穩態誤差
    ess = (1-y[y.size-1])
    return abs(ess)
```

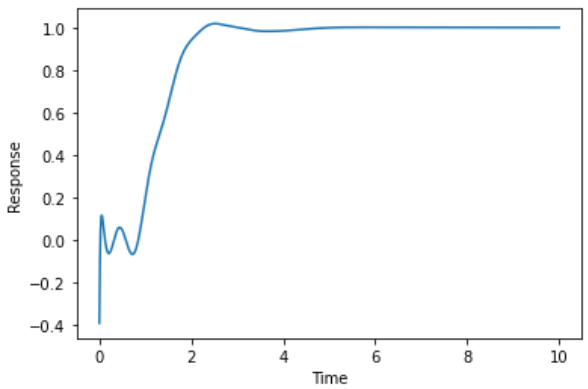
PSO參數優化的部分跟上一題類似，但因為是調整PID參數，所以設置了一些上下限的條件，一開始隨機產生的參數範圍設在0~5，經由PSO迭代優化期間，將超過15的參數定為15；0以下的參數定為0，迭代次數則設為30。

```
x = np.random.uniform(0,5,(pop_size,dimemsion))   #隨機產生0~5的值

for i in range(pop_size):
  for k in range(dimemsion): #限制參數範圍0~15
    if (x[i,k]<0):
      x[i,k]=0
    if (x[i,k]>15):
      x[i,k]=15
```

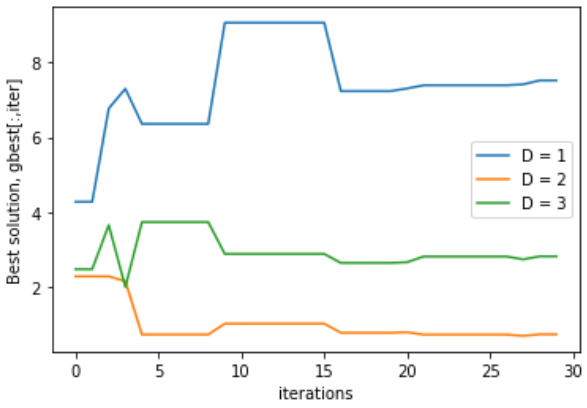
經過30次迭代後產生的[Kp,Ki,Kd]=[7.52069123 0.73857838 2.82219986]，下面兩張圖比較一開始的gbest響應，以及最後迭代完的gbest響應。



上圖為一開始的gbest響應。

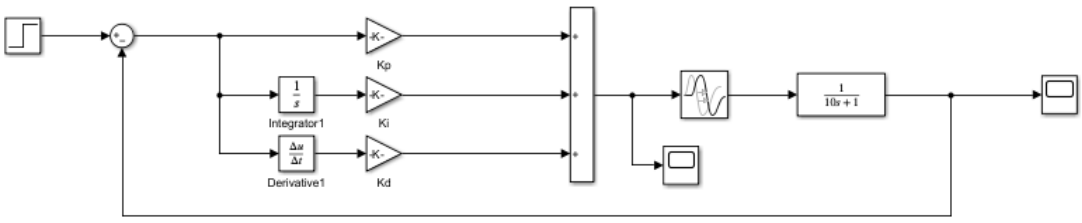


上圖最後迭代完的gbest響應。

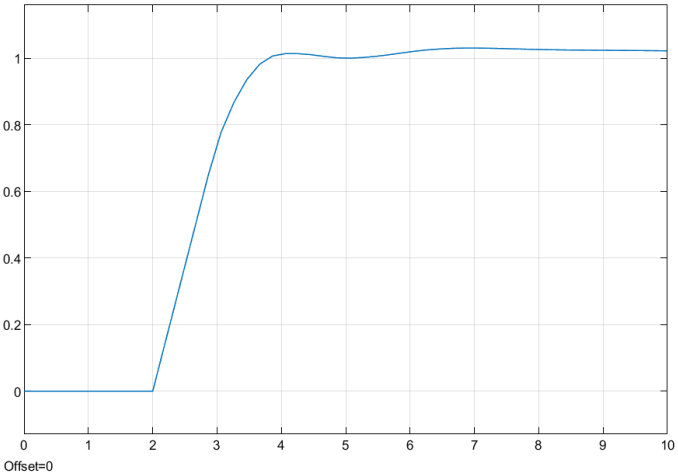


上圖為三個參數(Kp,Ki,Kd)隨著迭代過程的變化。

- 將最後得出的參數帶入simulink模擬:



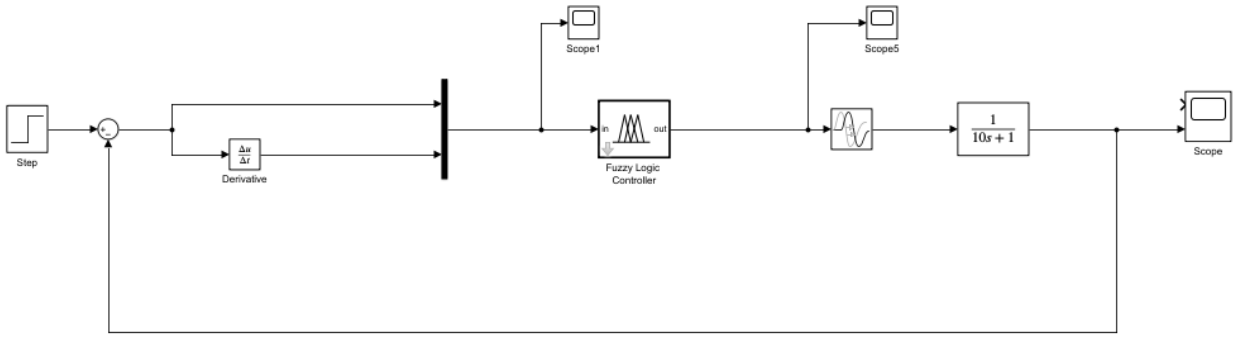
得到的響應:



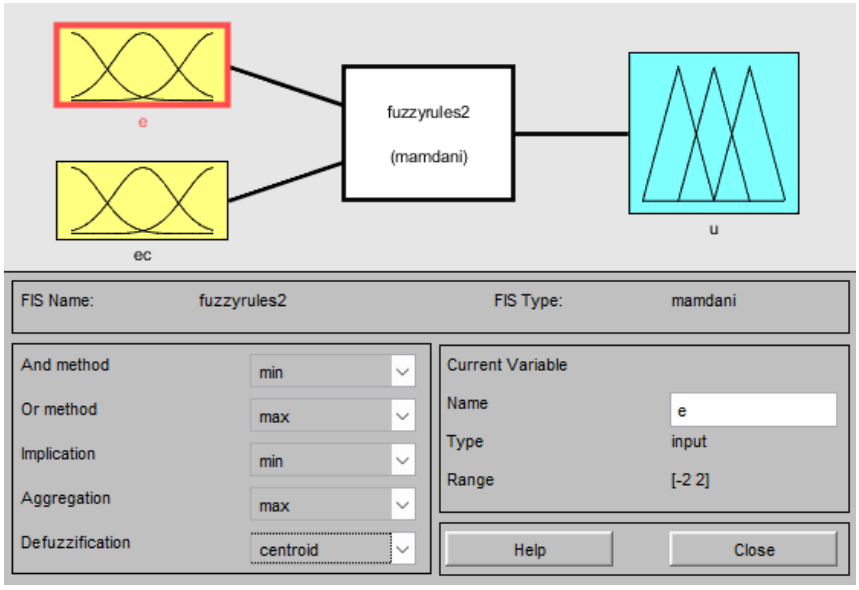
3. (Fuzzy control) Continue the above controller design problem, the utilized controller is replaced by fuzzy control. That is the *First-Order Plus Time Delay Model* is controlled by a fuzzy controller, please give your design and implementation results using simulation. ↵

file: fuzzyrules2.fis & fuzzy_control.slx

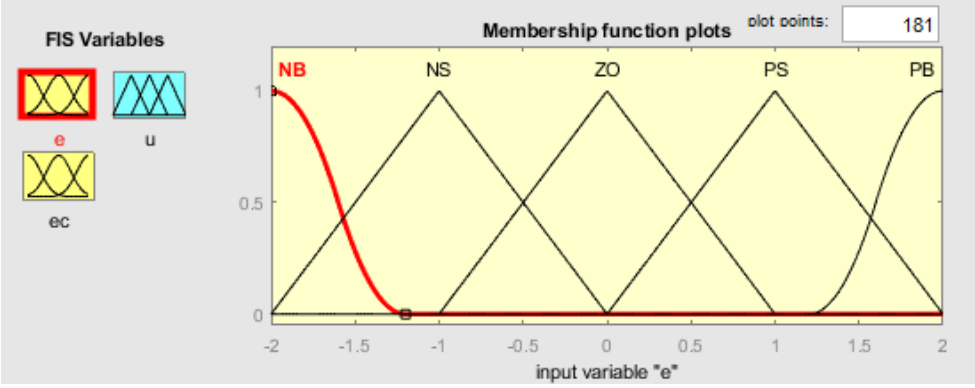
- 建立一個二為模糊控制器，控制器輸入定為系統偏差量e，以及偏差變化量ec，輸出為控制訊號u，整體架構如下圖。

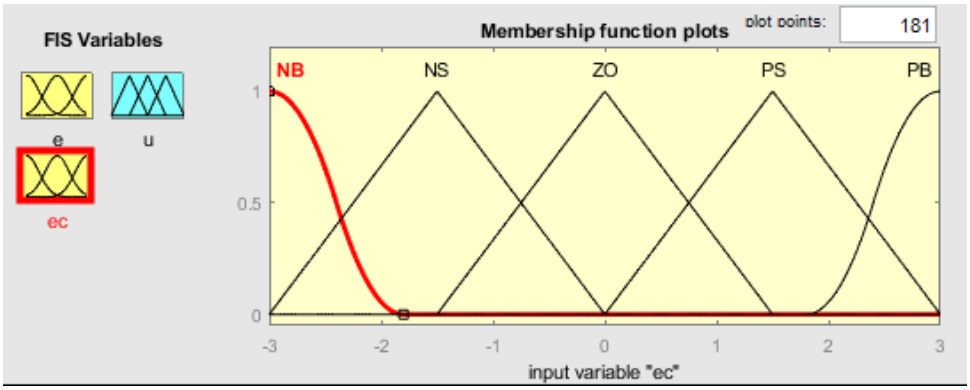


- 模糊控制器如下，解模糊使用重心法。

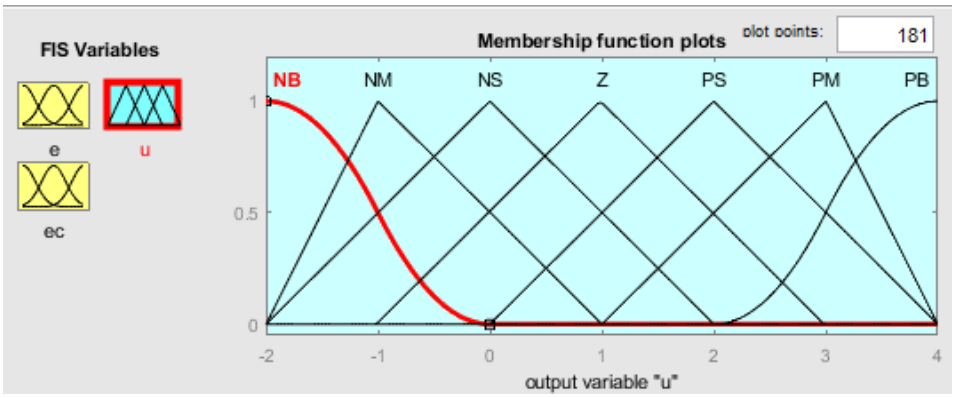


- 系統偏差量e及偏差變化量ec的Membership function如下圖。

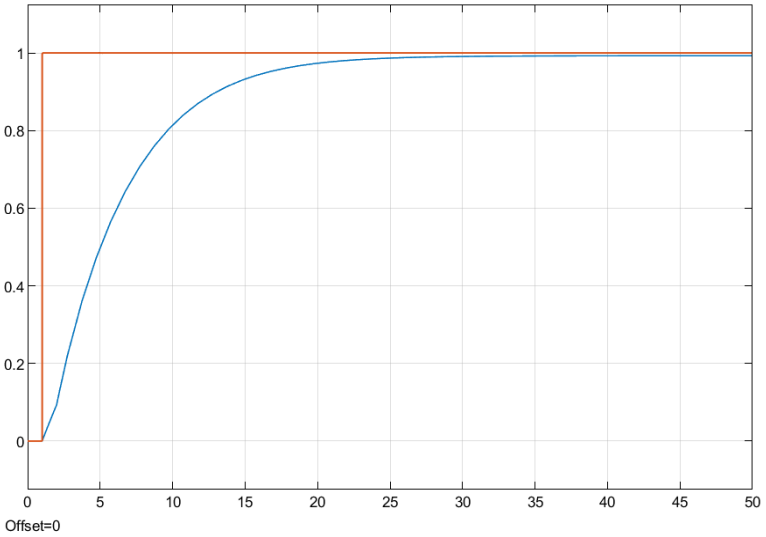




- 輸出u的Membership function的值一開始按照網路資料設置，但由於受控場不一樣會導致穩態值無法收斂至1，後來參考了第二題受控廠前的訊號，來設定模糊控制器的輸出u的範圍[-2 4]。



最後的響應如下圖:



4. (Final Project planning) Please describe the **implementation plan for your final project**, includes problem, plant, datasets,…↵

此篇論文想解決的問題為利用train model的方式，利用automated optical inspection (AOI)的影像訓練網路，達到缺陷偵測的效果，實現的部分將參考論文的方式利用Faster-RCNN+ResNet50作為baseline，在引入論文內用的GARPN,FPN來提升網路效率，試著比較幾種模型的準確率及效率，dataset的部分則用北京大學所提供的PCB-datasets共10000張左右的數據做為訓練測試集。

參考資料

▼ 第一題

Solving complex problems with Particle Swarm Optimization

Python 實作 Particle Swarm Optimization , PSO(粒子群最佳化)

Particle-Swarm-Optimization-PSO-using-Python

▼ 第二題

python控制系統仿真庫control（一）伯德圖

Getting Started with Transfer Functions

Python 的控制系統套件：control(slycot)

Python Control Systems Library

【Simulink】PSO优化算法整定PID控制器参数（一）———高阶不稳定系统

粒子群算法（PSO）整定PID参数

基於粒子群最佳化法之控制系統_基於粒子群最佳化法之PID控制

▼ 第三題

matlab模糊控制工具箱使用和模糊控制pid实例参考

模糊控制pid控制

模糊控制——（2）模糊系統和模糊控制器

模糊 PID 自整定控制算法简单理解