In [219]:
```
import numpy as np
def train(times, set_size, tau):
    out_minus_in = 0
    for T in range(times):
        print(T, end='\r')
        ## get x_set, y_set
        x_set = np.sort(np.random.uniform(-1, 1, set_size))
        y_set = np.zeros([set_size])
        num_negative = 0
        for i in range(set_size):
            y_set[i] = np.sign(x_set[i])
            if y_set[i]==0:
                y_set[i] = -1
            temp = np.random.uniform(0, 1)
            if temp<=tau:
                y_set[i] *= -1
            if y_set[i]==-1:
                num_negative += 1
        ##dp_table, E_in
        dp_table = np.zeros([2, set_size])
        dp_table[0][0] = set_size-num_negative ## s=-1 預測錯的個數
        dp_table[1][0] = num_negative ## s=1 預測錯的個數
        for i in range(1, set_size):
            if y_set[i-1]==-1:
                dp_table[0][i] = dp_table[0][i-1] + 1
                dp_table[1][i] = dp_table[1][i-1] -1
            elif y_set[i-1]==1:
                dp_table[0][i] = dp_table[0][i-1] - 1
                dp_table[1][i] = dp_table[1][i-1] + 1
        E_in = np.min(dp_table) / set_size
        ##s, theta
        index = np.unravel_index(np.argmin(dp_table, axis=None), dp
_table.shape)
        if index[0] == 0:
            s = -1
        elif index[0] == 1:
            s = 1
        which_theta = index[1]
        if which_theta == 0:
            theta = -1
        else:
            theta = (x_set[which_theta-1] + x_set[which_theta]) / 2
         ##E_out
        if s == -1:
            E_out = 1 - (0.5 * abs(theta))
        elif s == 1:
            E_out = 0.5 * abs(theta)
        E_out = E_out*(1-(2*tau))+tau

        out_minus_in += (E_out-E_in)
    mean = out_minus_in/times
    return mean
```

In [220]:
```
mean_16 = train(10000, 2, 0)
print("mean_16", mean_16)
```

```
mean_16 0.2907575302757793
```

In [221]:
```
mean_17 = train(10000, 20, 0)
print("mean_17", mean_17)
```

```
mean_17 0.024181985529829673
```

In [222]:
```
mean_18 = train(10000, 2, 0.1)
print("mean_18", mean_18)
```

```
mean_18 0.36722770808211214
```

In [223]:
```
mean_19 = train(10000, 20, 0.1)
print("mean_19", mean_19)
```

```
mean_19 0.05157750106317468
```

In [224]:
```
mean_20 = train(10000, 200, 0.1)
print("mean_20", mean_20)
```

```
mean_20 0.005457546034295806
```