EE 550 - Homework 2

Implementation of Binary Hopfield Model

Sezen Perçin – 2018401000

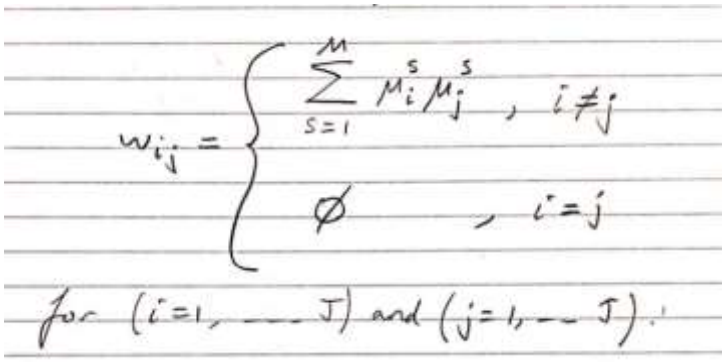
1) Numbers I generated with -1 and 1 values:

Number 1: **Number 4:** **Number7:** **Number9:**

2) Converting each sample matrix to vector to form sample vectors:

```
n1=reshape(number1,64,1);
n4=reshape(number4,64,1);
n7=reshape(number7,64,1);
n9=reshape(number9,64,1);
```

3) I used the formula below to form the weight matrix where the learning takes place in Hopfield Model:



(Taken from the course notes 12.04.2022)

4) Introducing the noises and iterating for each number and standard deviation value:
To get the output, I have changed the "numberx" parameter (where x=1,4,7,9) representing the sample with which noise is introduced and the value of standard deviation "std"(0.4, 0.7, 1) in the below parts contained in the code:

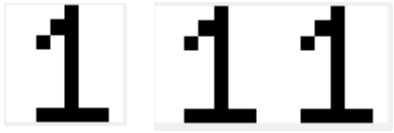
```
std=1;
noise=std*randn(8,8)+0;
im_in=sign(number1+noise);
in=reshape(im_in,64,1);
```

During the iterations a memory parameter called "ymemory" is created and the output of the previous step is stored which is consistent with the allocative memory nature of the Hopfield model. Also, the activation function is given as below (since sign() function also assigns the value 0):

```
ynew=((w*in)>=0)*2-1;
ymemory=zeros(64,1);
```

The output values: (Noisy Image - Iterations)

Number 1 – std=0.4



Number 1-std=0.7

Number1 – std=1

Number 4 – std=0.4

Number 4-std=0.7

Number 4 – std=1

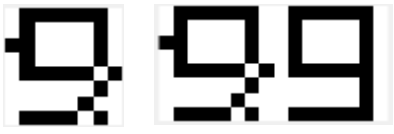
Number 7 – std=0.4

Number 7-std=0.7

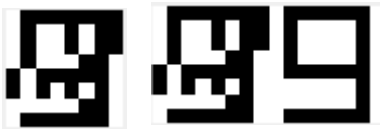
Number 7 – std=1

Number 9 – std=0.4

Number 9-std=0.7



Number 9 – std=1



Noise is not very visible when the standard deviation is 0.4.