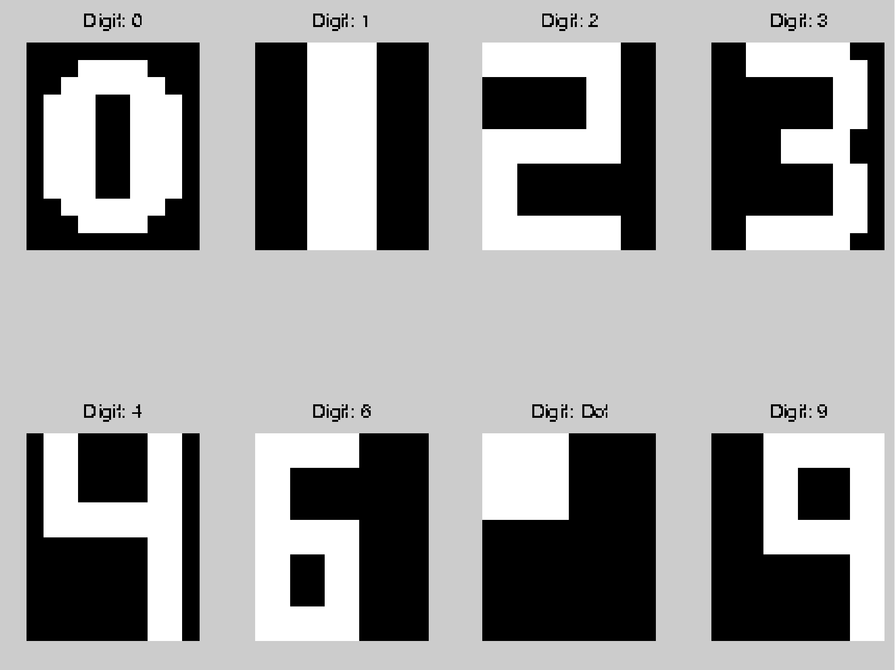
Programming task 2

Character recognition with
Associative Networks

One layer Associative Networks

In this task we want to store a set of patterns in a memory. This memory will be build as a single layer Neural Network with signum functions as non-linear neuron functions. We will access the memory by presenting as an input a distorted version of a pattern. The network is then expected to give out the pattern, from the memorized patterns, which resembles the input pattern. The stored patterns are bipolar and we will use a coding where black colour Is coded as –1 and white colour is coded as +1.

The patterns which we will use in this task are the digits ”0”, ”1”,”2”, ”3”, ”4”, ”6”, ” ` ”, ”9”. Each digit are sampled in an 12-by-10 array of bipolar numbers. The digits can be plotted as:



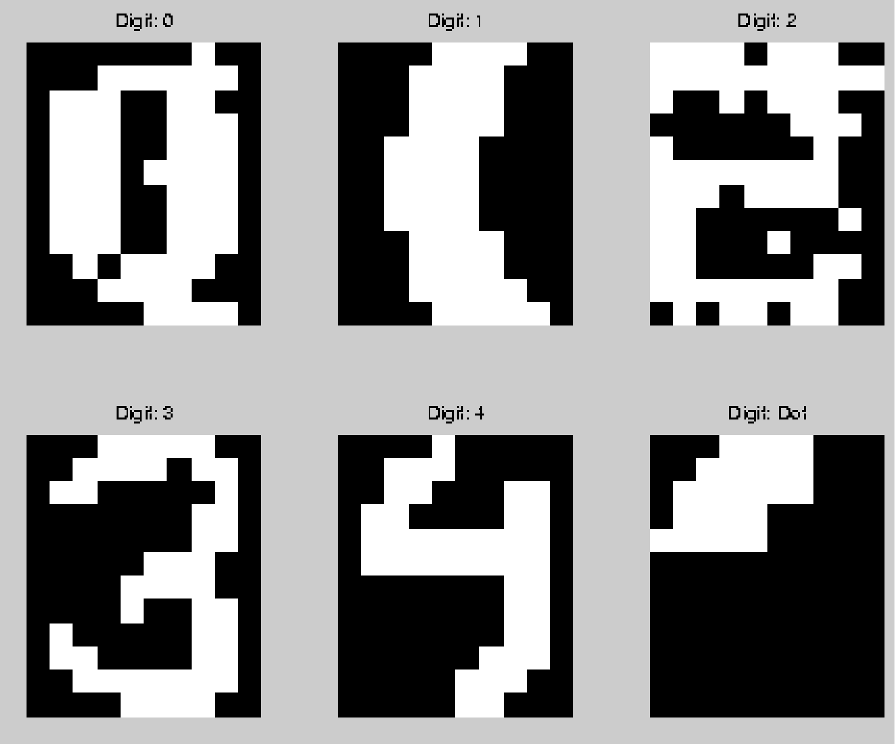
Each digit can be compactly represented as a matrix of size 12-by-10. For example the digit ”4” can be written as:

-1	1	1	-1	-1	-1	-1	1	1	-1
-1	1	1	-1	-1	-1	-1	1	1	-1
-1	1	1	-1	-1	-1	-1	1	1	-1
-1	1	1	-1	-1	-1	-1	1	1	-1
-1	1	1	1	1	1	1	1	1	-1
-1	1	1	1	1	1	1	1	1	-1
-1	-1	-1	-1	-1	-1	-1	1	1	-1
-1	-1	-1	-1	-1	-1	-1	1	1	-1
-1	-1	-1	-1	-1	-1	-1	1	1	-1
-1	-1	-1	-1	-1	-1	-1	1	1	-1
-1	-1	-1	-1	-1	-1	-1	1	1	-1
-1	-1	-1	-1	-1	-1	-1	1	1	-1

In the formulation of memory storage we have adopted, the memorized patterns should be column vectors. This can however be easily constructed from the matrix formulation by writing the elements of the matrix as a column vector (each column of the matrix is added to the previous one).

a) We will use an auto-associative memory consisting of 120 (=12*10) neurons in one layer where each neuron have 120 corresponding weights, i.e. the weight matrix will be of size (120-by-120). Each neuron should have the signum function at the output.

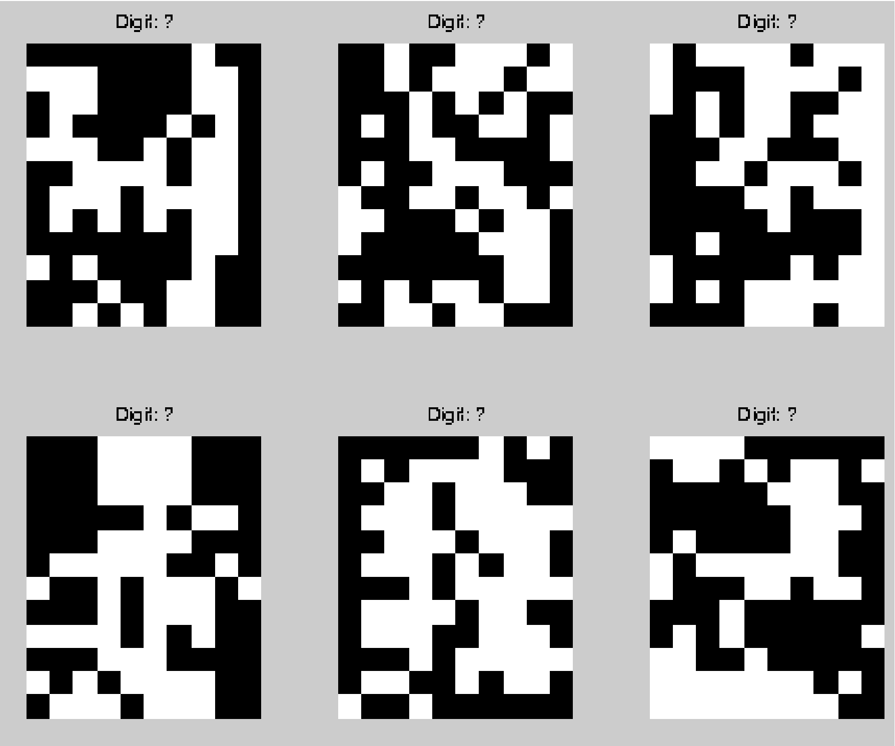
- Find the weight matrix, W, by using the so called Hebbian paradigm. This is done by creating the correlation memory matrix from the digits which should be memorized.
- The following distorted characters should be presented to the network. What is the output from your network? Make a plot similar to the one below with the outputs at each entry. Use the same coding as described above.



- Use the pseudo-inverse memory, which means that the weight matrix should be found by taking the pseudo-inverse instead of the transpose in the Hebbian formulation. Present the output in a plot similar to the one above.
- Which one of the memory methods above gives the best memory recoverage?

b) In this section we will use a recurrent network – the Hopfield Network.

- Create a Hopfield Network which should keep the same digits in memory as in the previous task. The network should take an initial state vector as an input and iterate according to the Hopfiel model and give the final states as an output. Keep the same coding and arrangement of the stored patterns as in the previous section.
- Use the same corrupted digits as before and plot the output when using the Hopfield Network.
- Use the heavily distorted digits presented below as initial state vectors and plot the resulting states after convergence.



- Is the Hopfield Network a good candidate as an associative network, compared to the feedforward one in part a) of this task?