**Batch: A2 Roll No.: 16010122041**

**Experiment No. 10**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **Title: : Study experiment on Bi-directional Associative Memory, example** |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Aim :** To understand the Bi-Directional Associative Memory

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

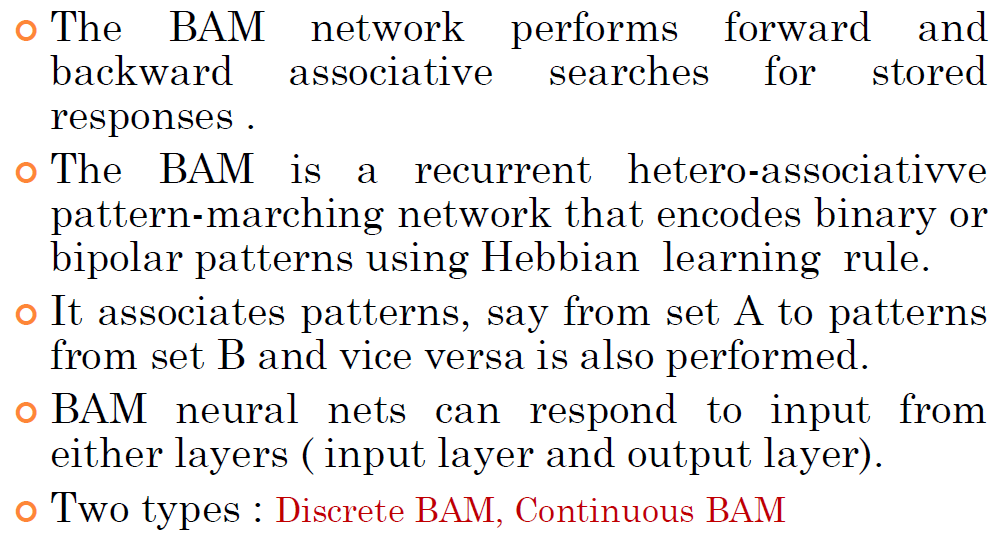
**Expected Outcome of Experiment:**

**CO3 :** Understand perceptron’s and counter propagation networks **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

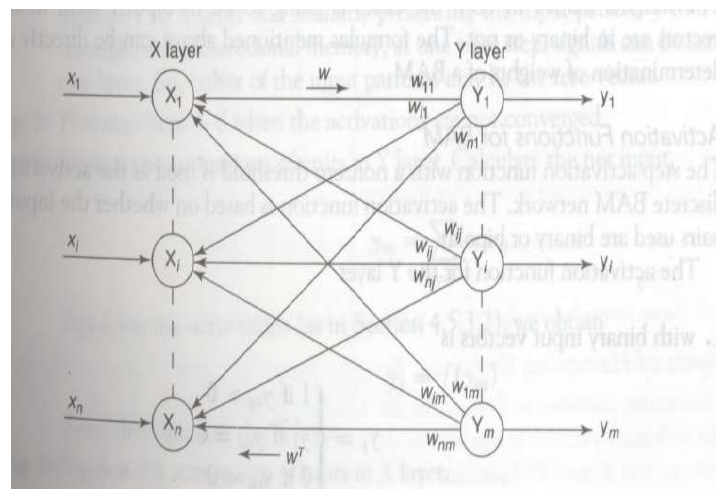
**Books/ Journals/ Websites referred:**

Principles of Soft Computing – Wiley

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
**Theory**:



**Architecture of BAM:**



**Training and Testing of Discrete BAM**

Discrete BAM works with binary or bipolar data. Each pattern in the input and output is a binary or bipolar vector (e.g., 0/1 or -1/1). Discrete BAM stores the association between pairs of binary vectors using a weight matrix and performs recall by updating states based on the weight matrix.

**Training of Discrete BAM**

1. **Initialize Weight Matrix:**
   * The weight matrix WWW is initialized to zero.
   * WWW has dimensions m×nm \times nm×n, where mmm is the length of the input vector and nnn is the length of the output vector.
2. **Learning Rule (Hebbian Learning):**
   * For each pair of input and output patterns (x,y)(x, y)(x,y), update the weight matrix as: W=W+xTyW = W + x^T yW=W+xTy
   * Here, xxx is the input vector and yyy is the output vector. xTyx^T yxTy is the outer product of xxx and yyy.
3. **Symmetric Weight Matrix:**
   * After updating for all training pairs, ensure that the weight matrix is symmetric: W=W+WTW = W + W^TW=W+WT

**Testing of Discrete BAM**

1. **Pattern Recall (Forward Association):**
   * Given an input vector xxx, compute the corresponding output vector yyy by: y=sign(Wx)y = \text{sign}(Wx)y=sign(Wx)
   * The sign function is applied element-wise to the result, converting positive values to 1 and negative values to -1.
2. **Backward Association:**
   * Similarly, given an output vector yyy, recall the corresponding input vector by: x=sign(WTy)x = \text{sign}(W^T y)x=sign(WTy)
3. **Iterative Update (if necessary):**
   * The process may be repeated iteratively until the patterns converge to a stable state.

**Training and Testing of Continuous BAM**

Continuous BAM deals with continuous input/output values, typically within a specific range like [0,1][0, 1][0,1] or [−1,1][-1, 1][−1,1]. The key difference from discrete BAM is that continuous BAM uses continuous activation functions and updates the patterns iteratively until convergence.

**Training of Continuous BAM**

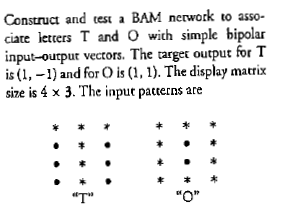
1. **Initialize Weight Matrix:**
   * Similar to discrete BAM, initialize the weight matrix WWW to zero.
   * The weight matrix dimensions will depend on the size of the input and output vectors.
2. **Learning Rule:**
   * For each training pair (x,y)(x, y)(x,y), update the weight matrix as: W=W+xTyW = W + x^T yW=W+xTy
3. **Normalization:**
   * Normalize the weight matrix if required, depending on the range of values in the continuous domain.

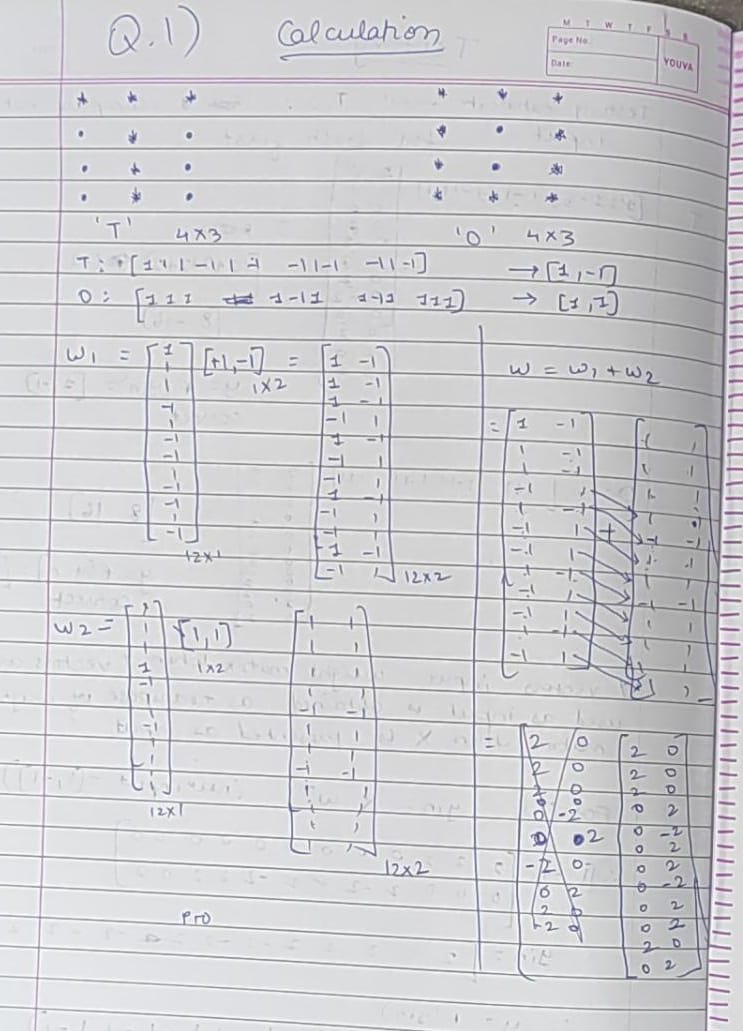
**Testing of Continuous BAM**

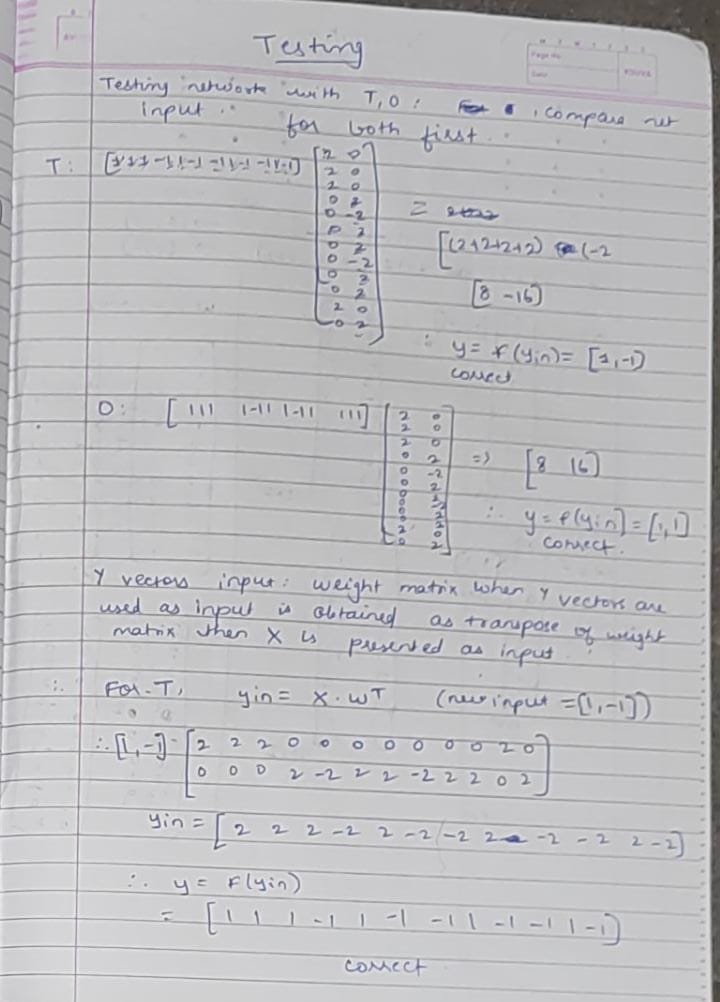
1. **Continuous Activation Function:**
   * Given an input vector xxx, calculate the corresponding output vector yyy using a continuous activation function, such as the sigmoid: y=sigmoid(Wx)y = \text{sigmoid}(Wx)y=sigmoid(Wx)
   * The sigmoid function is defined as: sigmoid(z)=11+e−z\text{sigmoid}(z) = \frac{1}{1 + e^{-z}}sigmoid(z)=1+e−z1​
   * The output values will be in the range (0,1)(0, 1)(0,1).
2. **Bidirectional Recall:**
   * Perform both forward and backward association until the patterns converge. The backward association is done using: x=sigmoid(WTy)x = \text{sigmoid}(W^T y)x=sigmoid(WTy)
3. **Iterative Update:**
   * Update the patterns iteratively by feeding back the output vector into the input until the patterns converge.

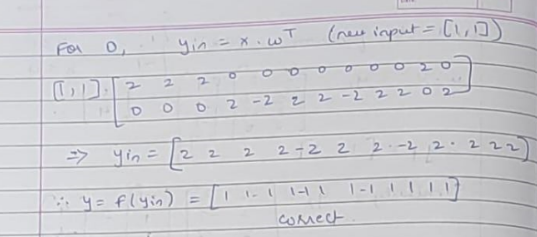
**Numerical to solve (handwritten solution) :**

**Question 1:**



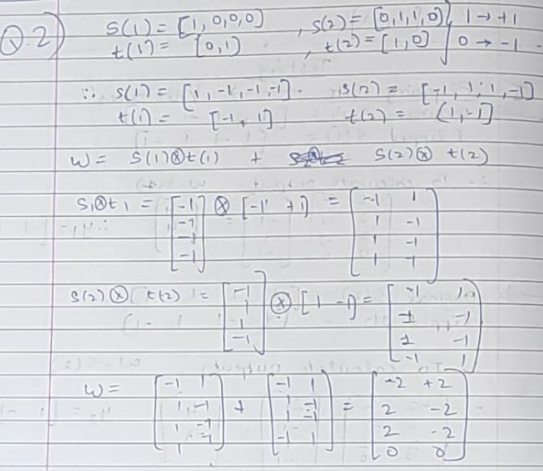


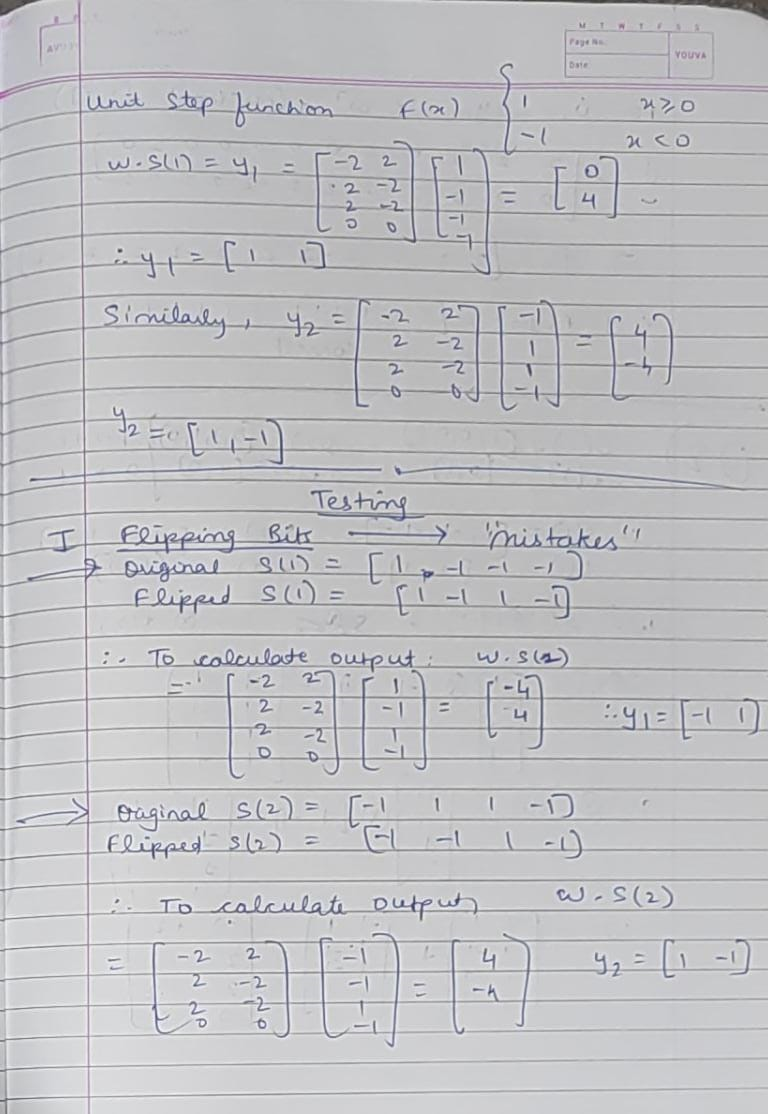


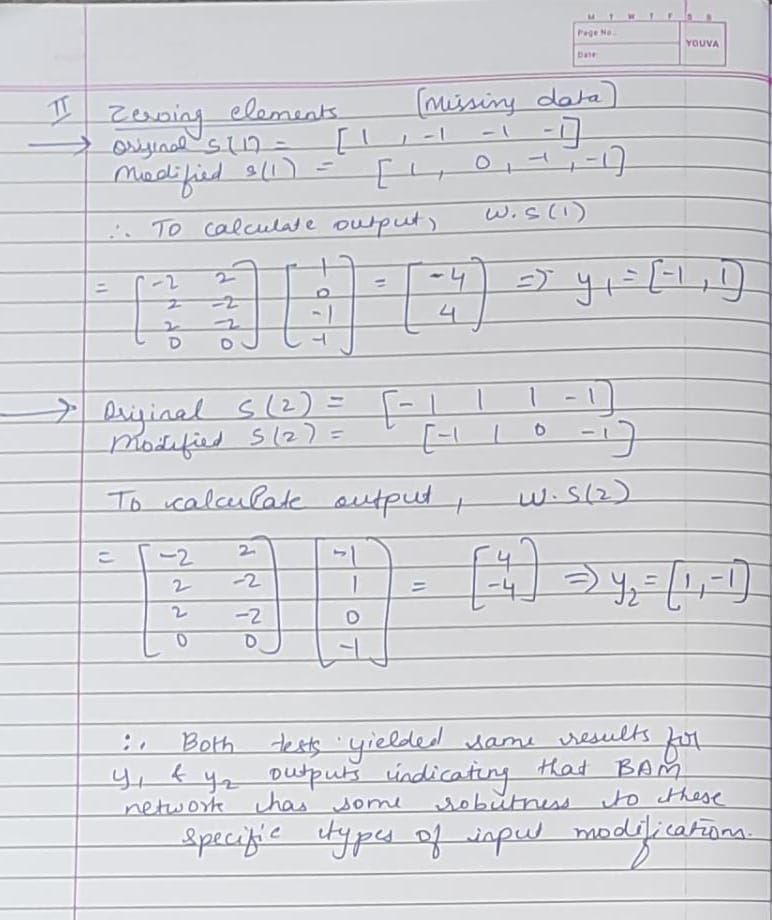


**Question 2:**

****

****





**Date: 04/11/2024 Signature of faculty in-charge**