# EECE 5644 Assignment 3 Jingcheng Wang

*Repository:* https://github.com/tomwang777/2025-Fall-EECE-5644-Machine-Learning/tree/main/Assignment%203

## Question 1

I first define the specific parameters of the Gaussian data distribution to meet the requirements of the problem. Then, I generate multiple training sets and one test set. Next, I generate the error probability of the theoretically optimal classifier according to the requirements of the problem. Then, I define the basic forward propagation layer through the activation function and SoftMax output. I then use cross-entropy loss to train the MLP and K-fold loop to obtain the optimal perceptron P value. Finally, I implement gradient descent to minimize the loss. The code uses Matlab's Neural Network Toolbox (patternnet) by configurating it properly according to the requirements of the assignment.

```
N_train = [100, 500, 1000, 5000, 10000];
N_test = 100000;
P_options = [1, 2, 3, 4, 5, 6, 8, 10, 12, 15, 20, 25, 30];
K_folds = 10;
```

### Theoretical optimal performance (benchmark)

*Class-conditional Gaussian pdf*

$$p(\,x \mid C_k\,) = \frac{1}{(2\pi)^{3/2} \mid \Sigma_k \mid^{1/2}} \exp!\left(-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k)\right)$$

*Uniform prior*

$$P(C_k) = \frac{1}{4}$$

*Joint distribution*

$$p(x, C_k) = p(\,x \mid C_k\,)P(C_k)$$

$$p(x, C_k) = \frac{1}{4} p(\,x \mid C_k\,)$$

$$p(x, C_k) = \frac{1}{4} \cdot \frac{1}{(2\pi)^{3/2} \mid \Sigma_k \mid^{1/2}} \exp!\left(-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k)\right)$$

*Posterior probability*

$$p(\,C_k \mid x\,) = \frac{p(\,x \mid C_k\,)P(C_k)}{\sum_{j=1}^{4} p(\,x \mid C_j\,)P(C_j)}$$

*MAP classifier (Decision Rule)*

$$\hat{C}(x) = \arg\max_{k} p\,(\,x \mid C_k\,)$$

Discriminant function under Gaussian conditional density (with the same covariance)

$$g_c(x) = \ln \pi_c - \frac{1}{2}\, \mu_c^{\top} \Sigma^{-1} \mu_c + \mu_c^{\top} \Sigma^{-1} x$$

empirically estimate the probability of error for this theoretically optimal classifier on the test dataset

$$\widehat{P}_{\mathrm{err}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\{\hat{y}_i \neq y_i\}$$

While calculating the probability of error on the test dataset, for every test sample:

1. Calculate the Gaussian likelihood for each class $p(\,x \mid c\,)$.
2. Calculate $gc(x) = ln\pi c + lnp(\,x \mid c\,)$
3. Get $y = argmaxcgc(x)$ and calculate $\hat{P}_{e}rr$

## Model structure selection (cross-validation)

*MLP Forward Propagation: Hidden Layer (ReLU)*

Linear transformation of the hidden layer (weighted input)

$$\mathbf{a}^{(1)} = \mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}$$

The activation output of the hidden layer (ReLU activation function)

$$\mathbf{h} = \mathrm{ReLU}\big(\mathbf{a}^{(1)}\big) = \max\big(\mathbf{0}, \mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}\big)$$

*Softmax output layer*

$$y_k = \frac{\exp(a_k)}{\sum_{j=1}^{4} \exp(a_j)}$$

## Model training and evaluation

*Cross-Entropy Loss*

Regarding with all samples:

$$L = -\sum_{i=1}^{N} \sum_{k=1}^{4} t_{ik} \log y_{ik}$$

*The gradient of cross-entropy with respect to logits*

$$\frac{\partial L}{\partial a_k} = y_k - t_k$$

*Parameter update (gradient descent)*

$$\theta \leftarrow \theta - \eta \frac{\partial L}{\partial \theta}$$

## Results

```
Data Distribution Defined:
- Classes: 4
- Dimensions: 3
- Priors: Uniform (0.25 each)

   Training set 1: 100 samples
   Training set 2: 500 samples
   Training set 3: 1000 samples
   Training set 4: 5000 samples
   Training set 5: 10000 samples
   Test set: 100000 samples

  Theoretical Optimal P(error) = 0.0789 (7.89%)

Training MLPs with Cross-Validation...
Perceptron options: [1     2     3     4     5     6     8    10    12    15    20    25    30]
K-fold: 10, Random reinitializations: 5

  Training Set 1 (N=100)
   Cross-validation complete. Best P = 5 (CV error = 0.1400)
   Training final MLP with P=5...
   Test P(error) = 0.1109 (11.09%)

  Training Set 2 (N=500)
   Cross-validation complete. Best P = 4 (CV error = 0.0800)
   Training final MLP with P=4...
   Test P(error) = 0.0898 (8.98%)

  Training Set 3 (N=1000)
   Cross-validation complete. Best P = 5 (CV error = 0.0790)
   Training final MLP with P=5...
   Test P(error) = 0.0815 (8.15%)

  Training Set 4 (N=5000)
   Cross-validation complete. Best P = 10 (CV error = 0.0774)
   Training final MLP with P=10...
   Test P(error) = 0.0795 (7.95%)
```

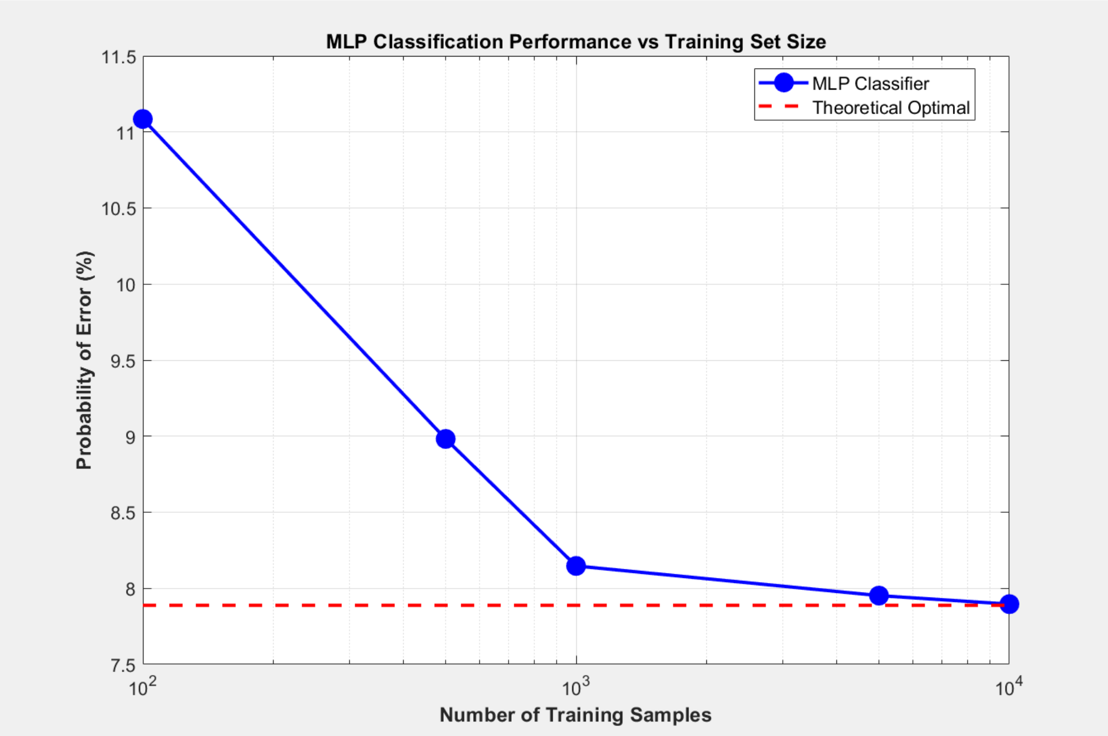
```
 Training Set 5 (N=10000)
   Cross-validation complete. Best P = 15 (CV error = 0.0791)
   Training final MLP with P=15...
   Test P(error) = 0.0790 (7.90%)
```

*Final Results*

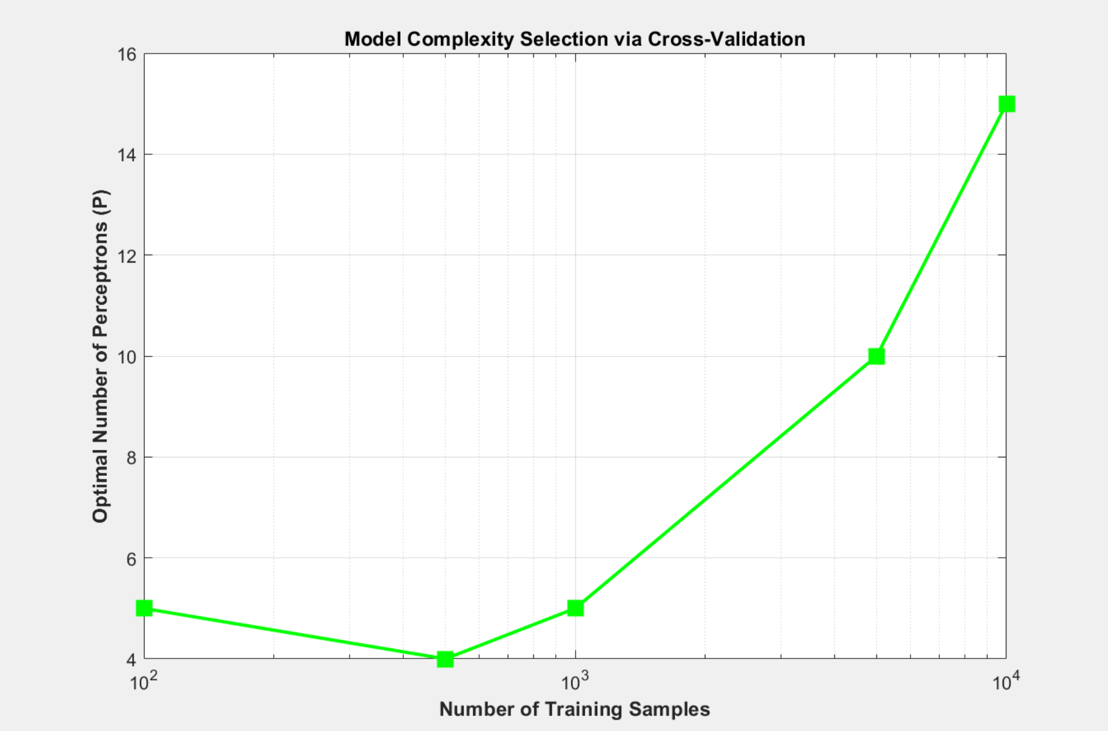
```
Theoretical Optimal P(error): 0.0789 (7.89%)

MLP Results:
N_train          P_optimal        Test P(error)
-------           ---------         -------------
100               5                 0.1109 (11.09%)
500               4                 0.0898 (8.98%)
1000              5                 0.0815 (8.15%)
5000              10                0.0795 (7.95%)
10000             15                0.0790 (7.90%)
```

## Data Visualization

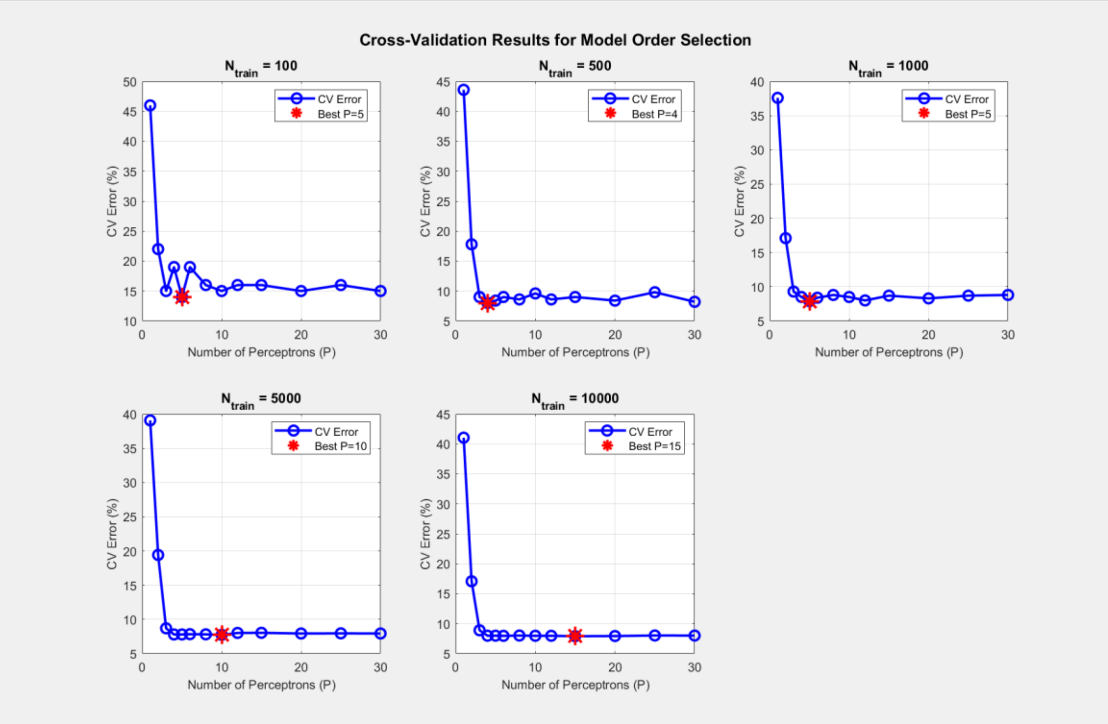
*Plot of the empirically estimated test P(error) for each trained MLP versus number of training samples used*



*Plot of model order selection based on perceptrons versus number of training samples used*



*Plot of the cross-validation result for every model order selection based on different number of training samples used*



As the number of training samples increases, the empirical error probability of the MLP classifier for each validation sample set continuously approaches the original theoretical optimal value. This indicates that training using MLP forward propagation and K-fold cross-validation loops can effectively sense gradient descent and optimize loss to obtain the minimized classification error probability.

## Question 2

I first set up a 4-component GMM dataset with overlapping components, then estimated the parameters of the GMM using the EM algorithm, setting RegularizationValue to 0.01, Replicates to 3, and the maximum number of iterations to 500 to achieve relatively perfect convergence within a controllable time period. Subsequently, I performed a K-fold Cross-Validation iteration process and verified the log-likelihood. Finally, I performed 100 repetitions of training to obtain the visualization results.

N_samples = [10, 100, 1000];
n_experiments = 100;
K_folds = 10;

### GMM Design

*GMM Model Definition*

$$p\left(x \mid \Theta\right) = \sum_{m=1}^{M} \pi_m \, \mathcal{N}\left(x \mid \mu_m, \Sigma_m\right)$$

$$\left(\textstyle\sum_{m=1}^{M} \pi_m = 1, \quad \pi_m \geq 0\right)$$

*Gaussian components*

$$\mathcal{N}(x \mid \mu_m, \Sigma_m) = \frac{1}{(2\pi)^{d/2}|\Sigma_m|^{1/2}} exp\,!\left(-\frac{1}{2}(x-\mu_m)^T \Sigma_m^{-1}(x-\mu_m)\right)$$

### Data generation

*Dataset complete likelihood*

$$p\left(X \mid \Theta\right) = \prod_{i=1}^{N} \sum_{m=1}^{M} \pi_m \, \mathcal{N}\left(x_i \mid \mu_m, \Sigma_m\right)$$

*Log-likelihood*

$$\log p\left(X \mid \Theta\right) = \sum_{i=1}^{N} \log\left(\sum_{m=1}^{M} \pi_m \, \mathcal{N}\left(x_i \mid \mu_m, \Sigma_m\right)\right)$$

### Model Evaluation and Selection

*EM Algorithm: E-step*

$$\gamma_{im} = \frac{\pi_m \, \mathcal{N}\left(x_i \mid \mu_m, \Sigma_m\right)}{\sum_{j=1}^{M} \pi_j \, \mathcal{N}\left(x_i \mid \mu_j, \Sigma_j\right)}$$

*EM Algorithm: M-step*

Update mixing coefficients

$$N_m = \sum_{i=1}^{N} \gamma_{im}$$

$$\pi_m = \frac{N_m}{N}$$

Update mean value

$$\mu_m = \frac{1}{N_m}\sum_{i=1}^{N}\gamma_{im}x_i$$

Update covariance matrix

$$\Sigma_m = \frac{1}{N_m}\sum_{i=1}^{N}\gamma_{im}(x_i - \mu_m)(x_i - \mu_m)^T$$

*Log-likelihood of the K-fold cross-validation*

$$\mathcal{L}_{k}(M) = \sum_{i\in \mathrm{Val}_k} \log p\left( x_i \,\middle|\, \widehat{\Theta_{M,k}} \right)$$

Mean score of all folds

$$\mathcal{L}(M) = \frac{1}{K}\sum_{k=1}^{K}\mathcal{L}_{k}(M)$$

## Repeated experiments and reports

Model order selection

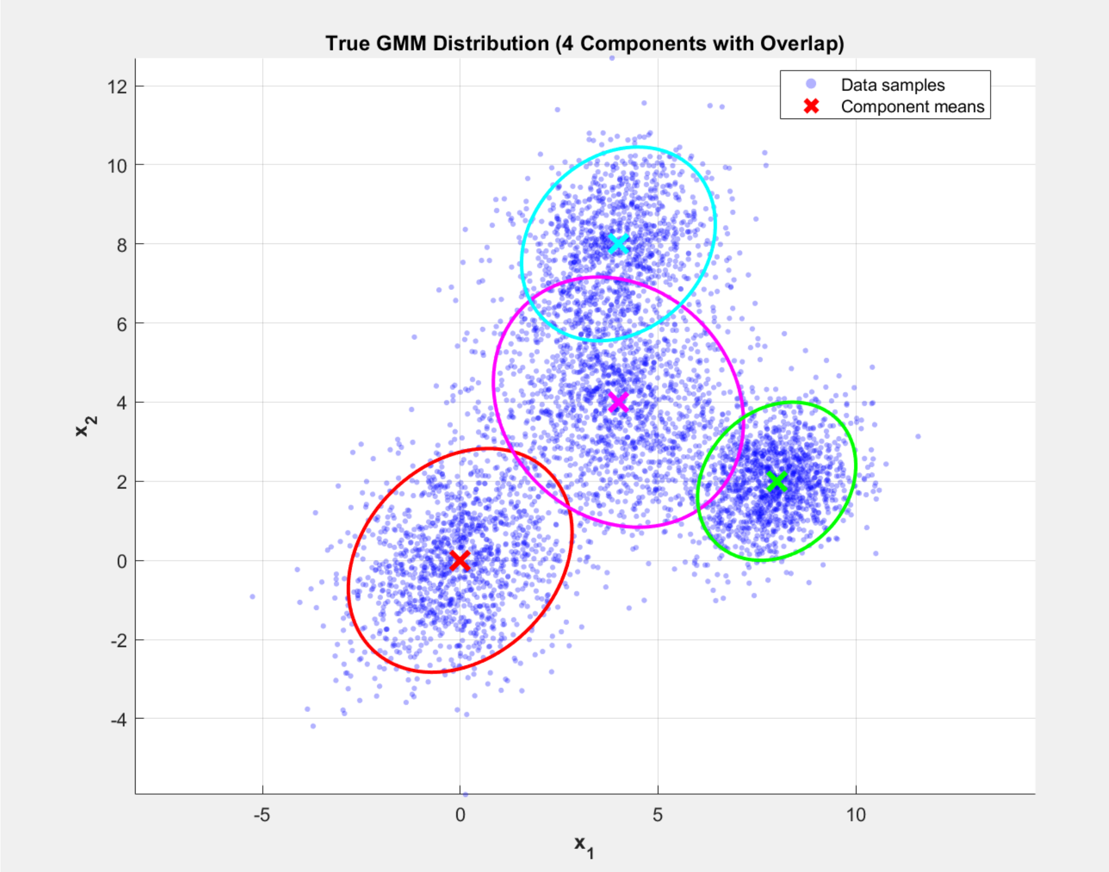
$$M^* = \arg\max_M \mathcal{L}\,(M)$$

## Results

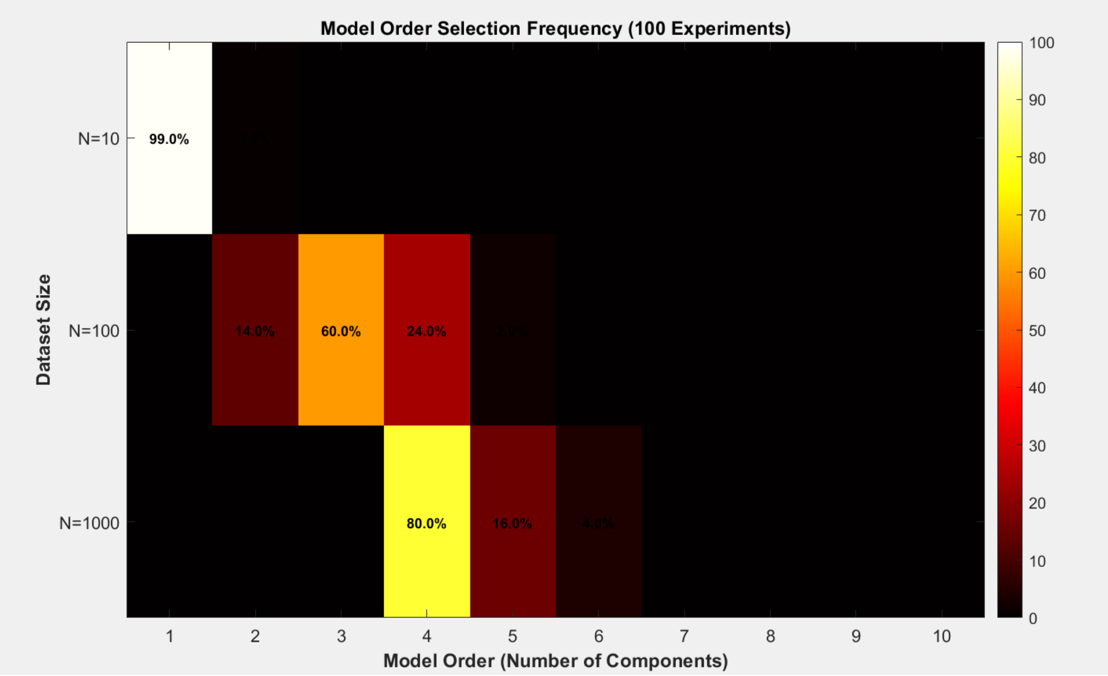
```
Dataset Size N = 10:
Model Order       Count              Frequency (%)
-----------       -----              -------------
1                 99                 99.00
2                 1                  1.00

Statistics:
  Mean selected order: 1.01
  Std deviation: 0.10
  Most frequently selected: 1 (99.0% of experiments)

Dataset Size N = 100:
Model Order       Count              Frequency (%)
-----------       -----              -------------
2                 14                 14.00
3                 60                 60.00
4                 24                 24.00
5                 2                  2.00

Statistics:
    Mean selected order: 3.14
    Std deviation: 0.67
    Most frequently selected: 3 (60.0% of experiments)

Dataset Size N = 1000:
Model Order         Count               Frequency (%)
-----------         -----               -------------
4                   80                  80.00
5                   16                  16.00
6                   4                   4.00

Statistics:
    Mean selected order: 4.24
    Std deviation: 0.51
    Most frequently selected: 4 (80.0% of experiments)
```

## Data Visualization

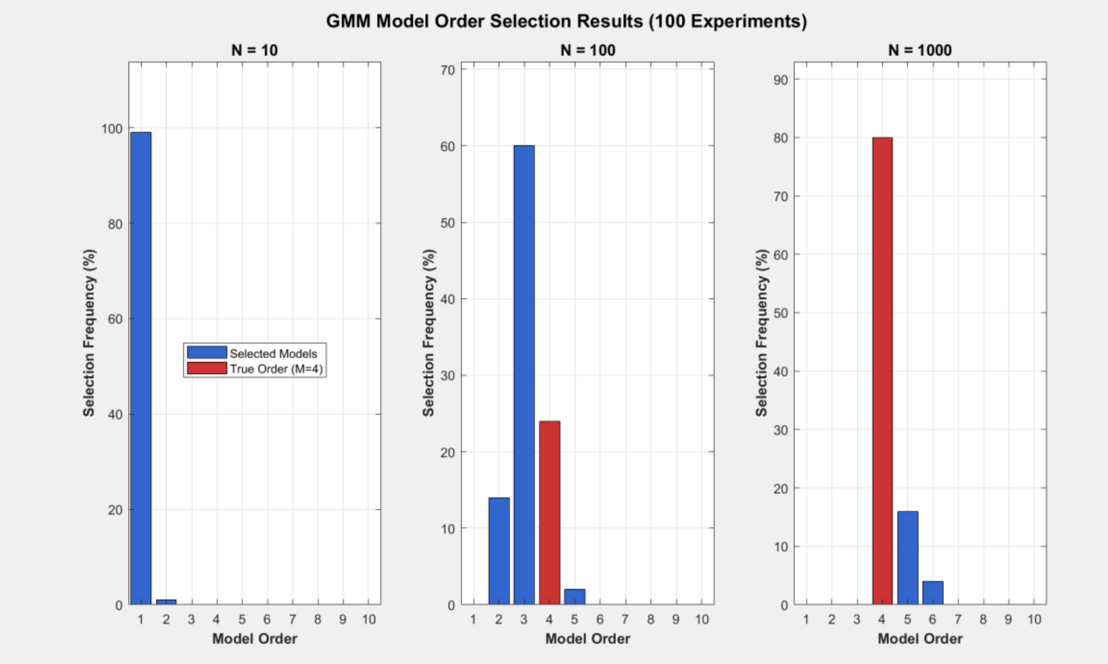
*Plot of True GMM Distribution*



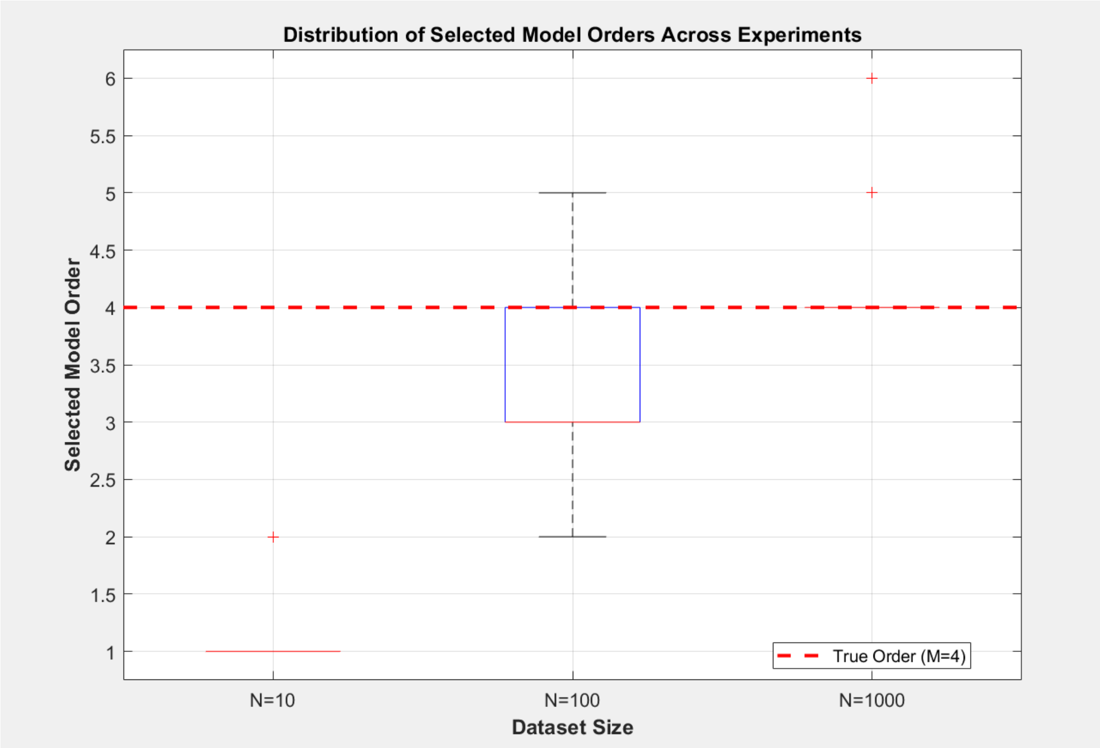
*Plot of Selection Frequency Heatmap*



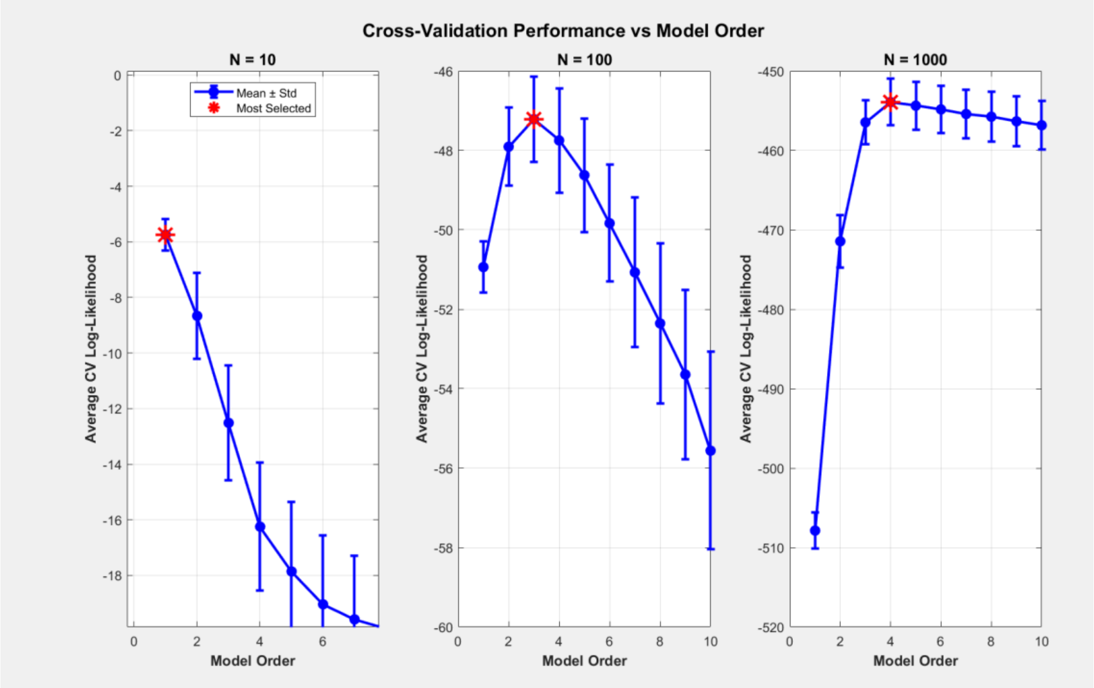
*Plot of selection frequencies for each dataset size*



*Plot of distribution of selected model orders across experiments*



*Plot of Cross-Validation Curves*



The experiment shows that when the sample size is small, although the cross-validation loss is not large, it is impossible to have a good understanding of all orders. Therefore, model mismatch and unstable selection will occur. When the sample size is slightly larger, the model selection will be gradually optimized, but it still deviates from the ideal value due to lack of training. Until the sample size reaches a relatively large value (such as 1000), true GMM model order (such as 4 in this case) will be selected accurately with a smaller deviation.

Citation

1. Course recording
2. Course notes
3. Course codes provided on Canvas
4. Discussion with classmates
5. Generative AI models
6. Training tools from Matlab source