Part2-1

AR(1) and AR(2) Process Modeling

AR(1) Process

For $n \leq T_0$, the signal x[n]x[n]x[n] follows the AR(1) model:

$$x[n] = \rho \cdot x[n-1] + w[n] \tag{1}$$

where ρ is the AR(1) coefficient, and $w[n] \sim \mathcal{N}(0, \sigma_\alpha^2)$ is white noise.

- $x[n]$ depends on its previous value $x[n-1]$ and the noise $w[n]$.
- The pole location ρ indicates the autocorrelation of the signal; the closer ρ is to 1, the stronger the temporal dependence of the signal.

AR(2) Process

For $n > T_0$, the signal $x[n]$ follows the AR(2) model:

$$x[n] = a_1 \cdot x[n-1] + a_2 \cdot x[n-2] + w[n] \tag{2}$$

where:

$$a_1 = \rho \cdot \cos(\phi), \quad a_2 = -\frac{\rho^2}{4} \tag{3}$$

are the AR(2) coefficients, and $w[n] \sim \mathcal{N}(0, \sigma_\beta^2)$ is white noise.

- This process uses two previous values, $x[n-1]$ and $x[n-2]$, to determine the current value $x[n]$, which indicates a stronger dependence of the signal.
- The pole location $\rho/2 \cdot exp(\pm j\phi)$defines the oscillatory characteristics of the signal.

---

2. MAP Estimation

The goal is to detect the change point T_0 by observing the signal $x[n]$ and estimate this point using Maximum A Posteriori (MAP) estimation.

We need to compute the conditional probability of the observed data X, which is the posterior probability:

$$P(T_0|X) \propto P(X|T_0)P(T_0) \tag{4}$$

where:

- $P(X \mid T_0)$ is the likelihood function, representing the probability of observing the data X given the change point T_0.
- $P(T_0)$ is the prior probability of the change point T_0, which is typically assumed to follow a uniform distribution.

Assuming the change at T_0 involves a shift from AR(1) to AR(2), the data model is:

- For $n \leq T_0$, we calculate the residuals using the AR(1) model:

$$e_{AR1} = x[n] - \rho \cdot x[n-1] \tag{5}$$

- For $n > T_0$, we calculate the residuals using the AR(2) model:

$$e_{AR2} = x[n] - a1 \cdot x[n-1] - a2 \cdot x[n-2] \tag{6}$$

The Residual Sum of Squares (RSS) is used as a measure of the likelihood:

$$RSS = \sum_{n=1}^{N} e[n]^2 \tag{7}$$

The posterior probability is determined by the residuals and the noise model:

$$P(T_0|X) \propto \exp\left(-\frac{1}{2} \cdot (RSS_1 + RSS_2)\right) \tag{8}$$

3. Change Point Detection

Using the MAP method, we calculate the posterior probability $P(T_0 \mid X)$ for each possible change point T_0 and find the position corresponding to the maximum posterior probability. The specific steps are as follows:

1. **Calculate the posterior probability for each possible change point T_0**:
   - For each T_0, compute the residuals for the AR(1) and AR(2) segments.
   - Calculate the posterior probability for each T_0.
2. **Smooth the posterior probability**:
   - Apply a moving average (e.g., 5-point moving average) to smooth the posterior probability and reduce noise effects.
3. **Find the maximum posterior point**:
   - Identify the T_0 corresponding to the maximum posterior probability as the MAP estimate, i.e., the estimated change point.

---

4. Mean Squared Error (MSE) Calculation

To assess the accuracy of the change point estimation, we use the Mean Squared Error (MSE) as an evaluation metric:

$$MSE = \frac{1}{MC} \sum_{mc=1}^{MC} (T_{\hat{0},mc} - T_0)^2 \tag{9}$$

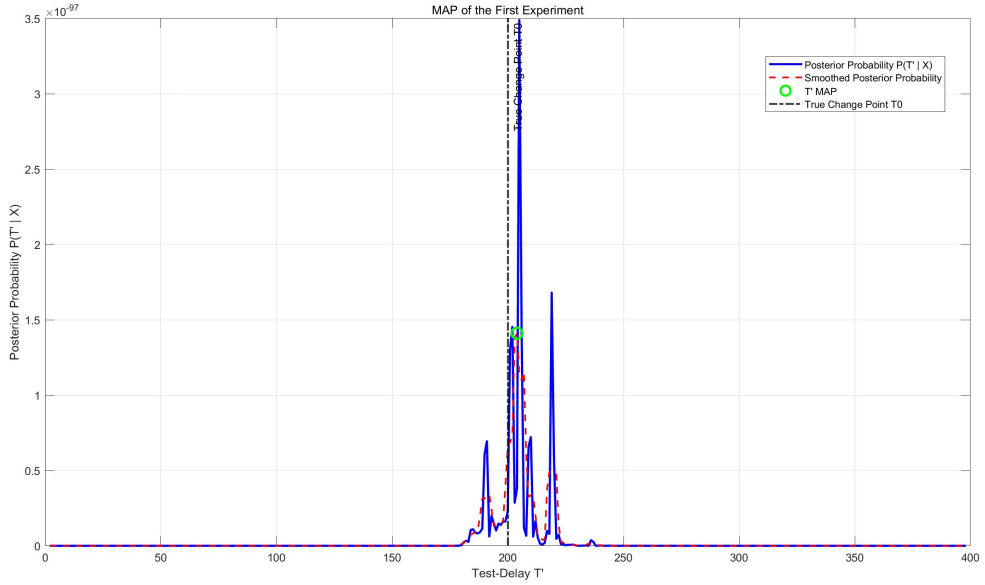
where $T_{\hat{0},mc}$ is the estimated change point from the mcmcmc-th experiment, and T_0 is the true change point.

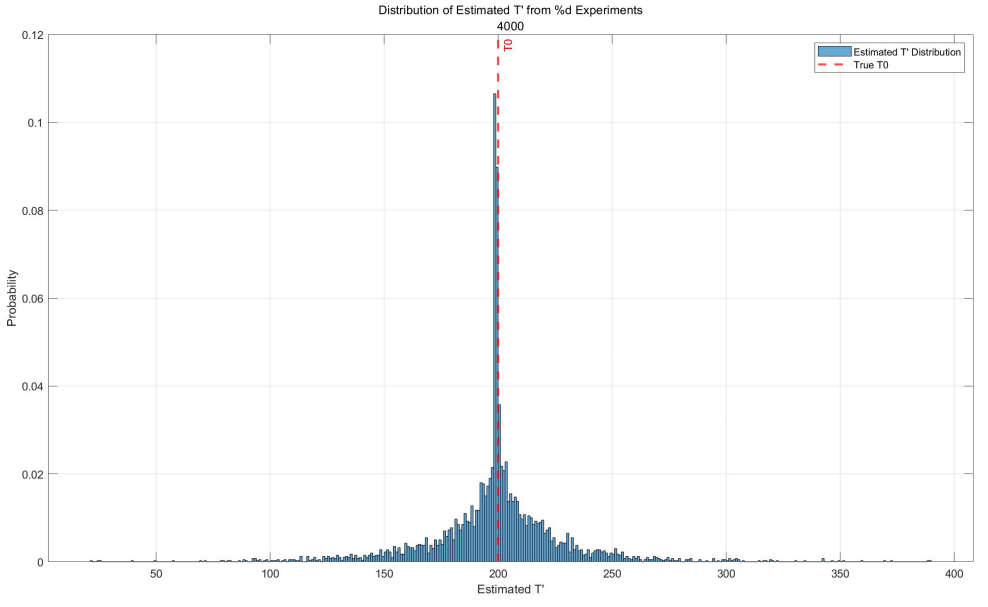
Through Monte Carlo simulations, we compute the MSE between the estimated change points and the true change point to evaluate the precision of the detection method.

Summary

We modeled the change point signal using AR(1) and AR(2) processes and employed the MAP method to detect the change point T_0. By calculating the posterior probability using the Residual Sum of Squares (RSS) as a likelihood measure, we found the MAP estimate of the change point. We evaluated the method's accuracy through MSE in multiple Monte Carlo simulations and demonstrated the distribution of the estimated change points.

实验结果





Monte Carlo computed MSE: 870.8987

Part2-2

Estimation of AR Models

Parameter Estimation for AR(1) Process

Using **Least Squares Estimation (LSE)**, the parameter $\hat{\rho}$ is obtained as:

$$\hat{\rho} = \frac{\sum_{n=2}^{T'} x[n]x[n-1]}{\sum_{n=2}^{T'} x[n-1]^2} \tag{10}$$

This formula minimizes the residual sum of squares (RSS):

$$RSS_1 = \sum_{n=2}^{T'} (x[n] - \hat{\rho}x[n-1])^2 \tag{11}$$

For the second segment modeled as an AR(2) process:

$$x[n] = a_1 x[n-1] + a_2 x[n-2] + v[n] \tag{12}$$

We can formulate a regression equation and use **Least Squares Estimation (LSE)** to estimate $\hat{a_1}, \hat{a_2}$:

$$\begin{bmatrix} x[T'+2] \\ x[T'+3] \\ \vdots \\ x[N] \end{bmatrix} = \begin{bmatrix} x[T'+1] & x[T'] \\ x[T'+2] & x[T'+1] \\ \vdots & \vdots \\ x[N-1] & x[N-2] \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} + \begin{bmatrix} v[T'+2] \\ v[T'+3] \\ \vdots \\ v[N] \end{bmatrix} \tag{13}$$

Define:

$$Y = \begin{bmatrix} x[T'+2] \\ x[T'+3] \\ \vdots \\ x[N] \end{bmatrix}, \quad X = \begin{bmatrix} x[T'+1] & x[T'] \\ x[T'+2] & x[T'+1] \\ \vdots & \vdots \\ x[N-1] & x[N-2] \end{bmatrix} \tag{14}$$

Then, the estimated parameters are given by:

$$\begin{bmatrix} \hat{a_1} \\ \hat{a_2} \end{bmatrix} = (X^T X)^{-1} X^T Y \tag{15}$$

Similarly, the residual sum of squares (RSS) is:

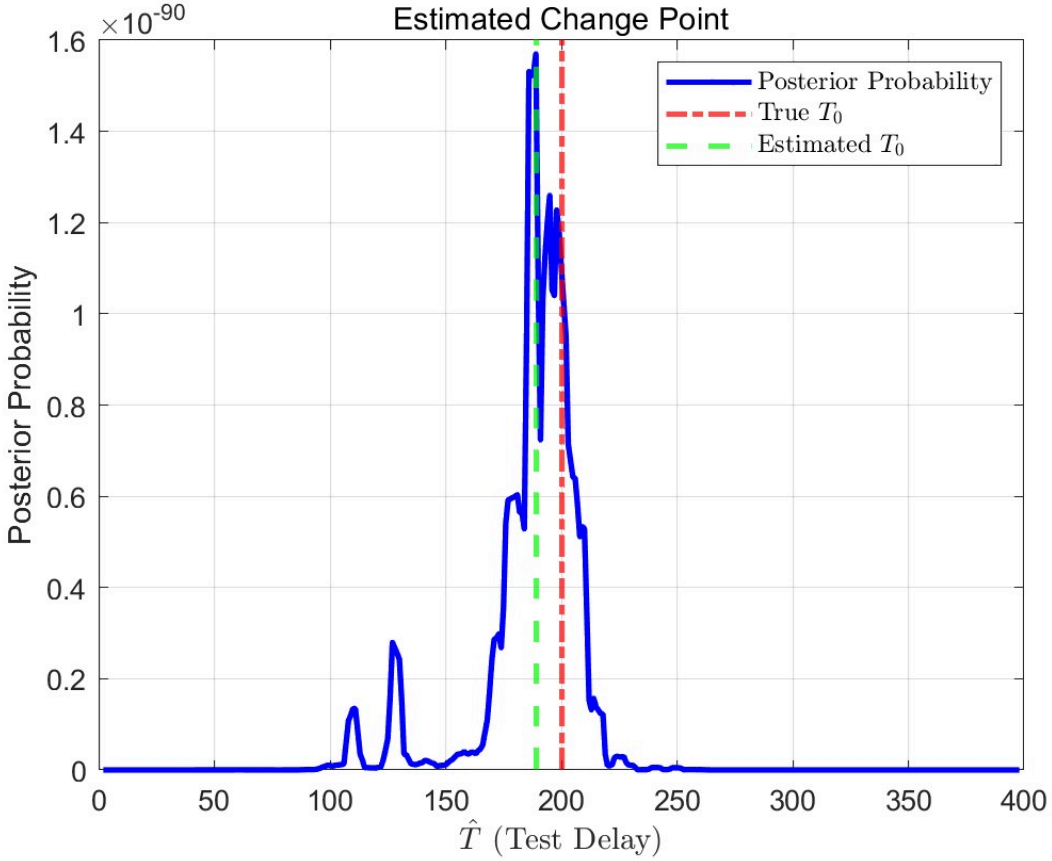
$$RSS_2 = \sum_{n=T'+3}^{N} (x[n] - \hat{a_1}x[n-1] - \hat{a_2}x[n-2])^2 \tag{16}$$

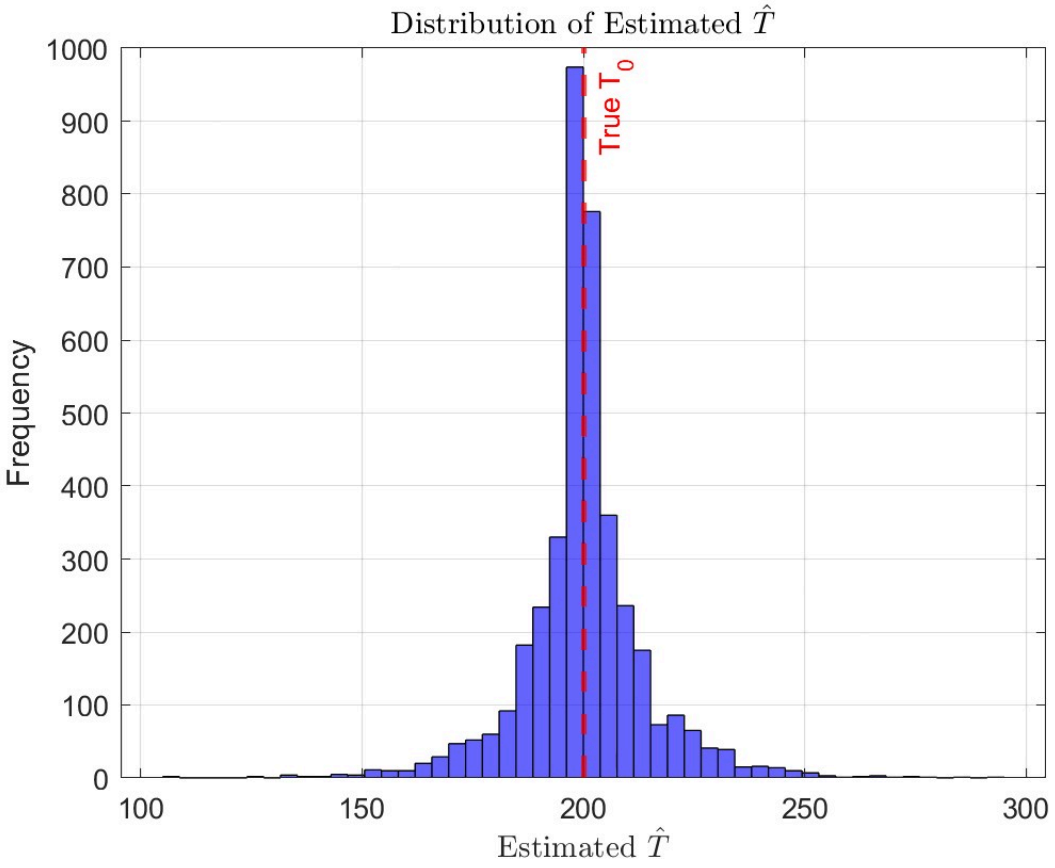
Simulation results

- **Estimated AR(1) parameter**:
  - ρ_{mean} = 0.7924
- **Estimated AR(2) parameter**:
  - $\hat{a}_{1mean}$ = 0.4955
- **Estimated AR(2) parameter**:
  - $\hat{a}_{2mean}$ = -0.2988

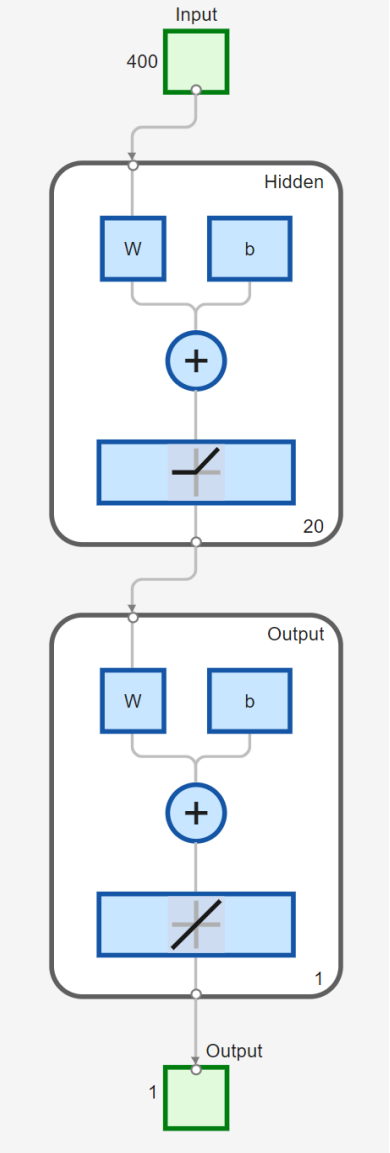
Monte Carlo Simulation

- **Estimated MSE**: 226.1943





Part 2-3



Let the input data be x, the weight matrix be $W^{(1)}$, the bias be $b^{(1)}$, and the activation function of the hidden layer be f. The output of the hidden layer is computed as:

$$z^{(1)} = W^{(1)}x + b^{(1)} \tag{17}$$

$$h = f(z^{(1)}) \tag{18}$$

Where:

h is the activation value of the hidden layer (dimension m).

f is the nonlinear activation function, such as ReLU or Sigmoid.

The output of the hidden layer h is linearly transformed by the output layer weights $W^{(2)}$ and bias $b^{(2)}$, then passed through the output layer activation function (such as a linear function or Sigmoid) to compute the final predicted value $\hat{y}$.

$$z^{(2)} = W^{(2)}h + b^{(2)} \tag{19}$$

$$\hat{y} = g(z^{(2)}) \tag{20}$$

Where:

g is the activation function of the output layer (for regression problems, it is usually a linear function, i.e., $g(x) = x$).

$W^{(2)}$ is the weight matrix from the hidden layer to the output layer (dimension $1 \times m$).

$b^{(2)}$ is the bias term of the output layer (scalar).

Finally, the predicted value of the neural network is $\hat{y}$.

The goal of training a neural network is to minimize errors, which are measured by the **loss function**. For regression problems, the most commonly used loss function is the **Mean Squared Error (MSE)**:

$$\text{MSE} = \frac{1}{M} \sum_{i=1}^{M} (Y_i - \hat{Y}_i)^2 \tag{21}$$

Where:

M is the number of training samples.

Y_i is the **ground truth** (actual value).

$\hat{Y}_i$ is the **predicted value** by the neural network.

The loss function measures the **error between the neural network's output $\hat{y}$ and the true value y**. The smaller the error, the more accurate the model is.

To minimize the loss, we need to **update the network's weights**, which involves the **Backpropagation** algorithm.

Computing the Gradient for the Output Layer:

$$\delta^{(2)} = \frac{\partial \mathcal{L}}{\partial z^{(2)}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z^{(2)}} \tag{22}$$

For **Mean Squared Error (MSE)** loss function:

$$\frac{\partial \mathcal{L}}{\partial \hat{y}} = -2(Y - \hat{Y}) \tag{23}$$

Thus:

$$\delta^{(2)} = -2(Y - \hat{Y}) \tag{24}$$

Computing the Gradient for the Hidden Layer:

Using the **chain rule**, propagate the gradient **backwards** from the output layer to the hidden layer:

$$\delta^{(1)} = \left(W^{(2)}\right)^T \delta^{(2)} \cdot f'(z^{(1)}) \tag{25}$$

Where:

$f'(z^{(1)})$is the derivative of the hidden layer activation function (if using ReLU, $f'(x) = 1$ when $x > 0$, otherwise 0).

$W^{(2)}$ is the weight matrix from the hidden layer to the output layer.

Gradient Descent

Update the **neural network weights** using **gradient descent**, based on the computed gradients from backpropagation:

$$W^{(1)} = W^{(1)} - \eta \cdot \frac{\partial \mathcal{L}}{\partial W^{(1)}} \tag{26}$$

$$W^{(2)} = W^{(2)} - \eta \cdot \frac{\partial \mathcal{L}}{\partial W^{(2)}} \tag{27}$$

$$b^{(1)} = b^{(1)} - \eta \cdot \frac{\partial \mathcal{L}}{\partial b^{(1)}} \tag{28}$$

$$b^{(2)} = b^{(2)} - \eta \cdot \frac{\partial \mathcal{L}}{\partial b^{(2)}} \tag{29}$$

Where:

η is the **learning rate**, which controls the step size of weight updates.

$\partial L / \partial W$ is the partial derivative of the loss function with respect to the weights.

Part 2-4

Derivation of MSE and ρ Relationship

As ρ increases, the signal becomes more autocorrelated, making the AR(1) and AR(2) models fit the data better. This leads to smaller residuals and more concentrated posterior probabilities, improving change point detection accuracy and reducing MSE.

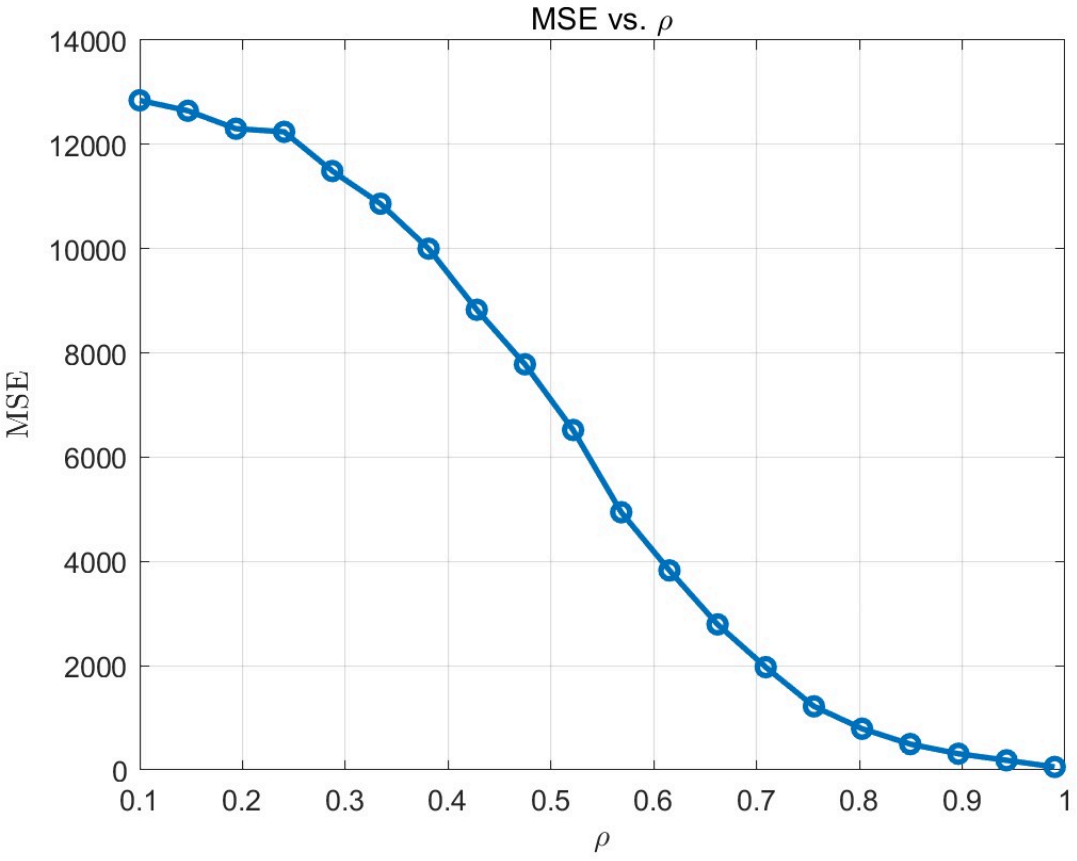
Conversely, when ρ is smaller, the signal exhibits more fluctuation, making the model fit worse, increasing residuals, and spreading out the posterior probability distribution, thus increasing MSE.

Theoretical Conclusion for MSE

- **As ρ increases**, the fit of the AR(1) and AR(2) models improves, reducing residuals, and increasing detection accuracy, which results in a lower MSE.
- **As ρ decreases**, the model fit worsens, residuals increase, and detection accuracy decreases, leading to a higher MSE.

Conclusion

- **MSE vs ρ**: MSE decreases as ρ increases because higher ρ enhances autocorrelation, improving model fitting and change point detection accuracy.
- **Theoretical Explanation**: As ρ increases, the model fit improves, residuals decrease, and posterior probability distributions become more concentrated, leading to more accurate change point detection and reduced MSE.


MSE vs. ρ
14000
12000
10000
8000
6000
4000
2000
0
MSE
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
ρ