02471 Machine Learning for Signal Processing

Problem Set 3

Emilie Sofie Lissner (s194341)

CONTENTS

1 Problem 3.1: Sparse signal representations

We consider a discrete-time signal x_n represented by the following model:

$$x_n = \sum_{j=1}^{K} a_j \cos\left(\frac{\pi(2m_j - 1)n}{2l}\right), \quad n = 0, \ldots, l-1$$

where $l = 2^9$ represents the total length of the signal, and $K \ll l$ implies that the signal is sparse, with only a few non-zero coefficients. We have access to a subset of the original samples, specifically 2^5 out of 2^9 possible samples. The indices of these samples are known.

1.1 Problem 3.1.1

The signal $\mathbf{x} \in \mathbb{R}^{2^9}$ is K-sparse in the Discrete Cosine Transform (DCT) basis $\Psi \in \mathbb{R}^{2^9 \times 2^9}$:

$$\mathbf{x} = \Psi\mathbf{a}.$$

The sampled measurements $\mathbf{y} \in \mathbb{R}^{2^5}$ are described by:

$$\mathbf{y} = \mathbf{C}\mathbf{x},$$

where the matrix $\mathbf{C} \in \mathbb{R}^{2^5 \times 2^9}$ is the measurements matrix, constructed using the indices of the measurements. The objective is to find the sparsest vector $\mathbf{a}$ that is consistent with the measurements $\mathbf{y}$:

$$\mathbf{y} = \mathbf{C}\Psi\mathbf{a} = \Theta\mathbf{a}.$$

Given that the system is underdetermined, there are infinitely many solutions. However, our objective is to find the sparsest solution by minimizing the ℓ_1-norm:

$$\hat{\mathbf{a}} = \arg\min_{\mathbf{a}} ||\mathbf{a}||_1 \quad \text{subject to:} \quad \mathbf{y} = \Theta\mathbf{a}$$

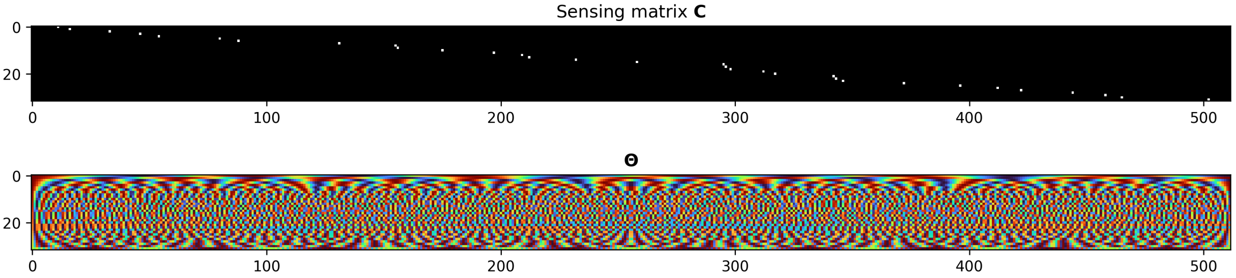

Figure 1. Visualization of the sensing matrix **C** and the matrix Θ.

1.2 Problem 3.1.2

The LASSO LARS algorithm was chosen to solve the problem for its ability to promote sparsity in solutions. This algorithm is particularly suitable for under-determined systems and computational efficient. The estimated number of non-zero parameters is $K = 4$. The indices and corresponding values of these parameters are summarized in Table 1. The parameters and the reconstructed solution is visualized in Figure 2

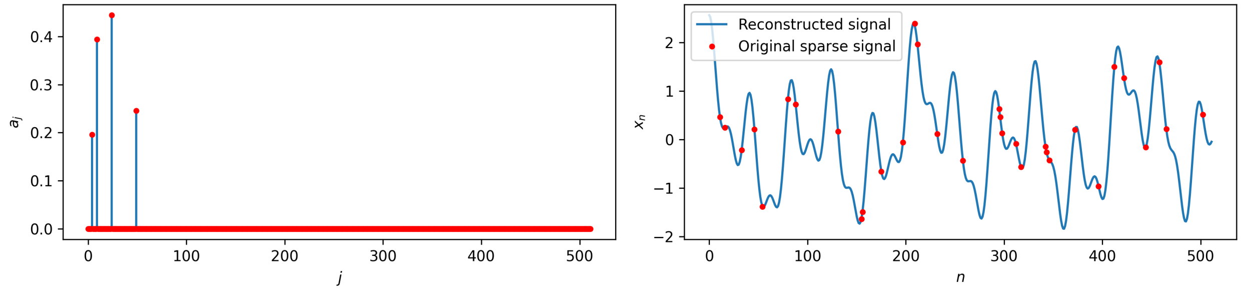

Figure 2. Visualization of the estimated values for $\mathbf{a}$.

j	m_j	a_j
1	5	0.20
2	10	0.40
3	25	0.44
4	50	0.25

Table 1. Estimated values for a_j and their indices.

```python
from scipy.fftpack import dct, idct
from sklearn.linear_model import LassoLars
import matplotlib.pyplot as plt
import numpy as np
from scipy.io import loadmat

# Load the .mat file
data = loadmat('problem3_1.mat')
n = data['n'].flatten() - 1 # Sample indices
y = data['x'].flatten() # Sample values

l = 2**9 # Signal length
N = len(n) # Number of samples

# Construct the sensing matrix from indices and compute Theta
C = np.zeros((N, l))
for i in range(N):
    C[i, n[i]] = 1
Theta = idct(C)

# Fit LassoLars model
model = LassoLars(alpha=0.01, fit_intercept=False, max_iter=1000)
model.fit(Theta, y)
a = model.coef_ # Extract solutions
x = dct(a)  # Reconstruct signal
```

Listing 1. Code for exercise 3.1.2

2 Problem 3.2: Bayesian inference and the EM algorithm

We consider a regression problem,

$$y_n = \theta^T \mathbf{x}_n + \eta_n.$$

where the noise η_n is i.i.d. with a Gaussian distribution, $\eta_n \sim \mathcal{N}(0, \sigma_\eta^2)$. We additionally assume that the elements in the θ vector are i.i.d. with a Gaussian distribution, $\theta_i \sim \mathcal{N}(0, \sigma_\theta^2)$. The parameters θ can be found by solving the optimization problem

$$\hat{\theta} = \arg\min_{\theta} \left(\frac{1}{2\sigma_n^2} \|\mathbf{y} - \mathbf{X}\theta\|_2^2 + \frac{1}{2\sigma_\theta^2} \|\theta\|_2^2 \right).$$

2.1 Problem 3.2.1

This problem solves a Bayesian linear regression model that corresponds to using ridge regression. In this setup, both the likelihood and the prior are assumed to be Gaussian. The regularization parameter λ in ridge regression is inversely related to the variance of the prior on the coefficients and can be expressed as $\lambda = \sigma_\eta^2 / \sigma_\theta^2$, where σ_η^2 is the variance of the noise in the likelihood model and σ_θ^2 is the variance of the prior on the regression coefficients. The role of the terms are described below:

- **Role of first term:** This term represents the likelihood of the observed data. The expression $\|\mathbf{y} - \mathbf{X}\theta\|_2^2$ is the squared Euclidean norm (sum of squared residuals), which quantifies the fit of the model to the data. Minimizing this term alone would lead to ordinary least squares regression. The factor $\frac{1}{2\sigma_n^2}$ scales the term by the variance of the noise, weighting the importance of the fit relative to the noise level.

- **Role of second term:** This term serves as a regularization that imposes a penalty on the size of the regression coefficients. The expression $\|\theta\|_2^2$ penalizes larger values of the coefficients, which helps in controlling overfitting. The factor $\frac{1}{2\sigma_\theta^2}$ reflects the strength of the prior belief in smaller θ-values, controlling the amount of shrinkage applied to the coefficients.

- **Role of noise variance σ_η:** A smaller value of σ_η increases the weight of the fit term $\|\mathbf{y} - \mathbf{X}\theta\|_2^2$ in the optimization problem, emphasizing a better fit to the data as the observed data is assumed to be more precise (less noisy). Conversely, a larger σ_η decreases this term's influence, which might lead to a less accurate fit as the model becomes less sensitive to the noise in the data.

- **Role of coefficient variance σ_θ:** A smaller σ_θ increases the weight of the regularization term $\|\theta\|_2^2$, promoting regularization. A larger σ_θ reduces the impact of the regularization term, allowing for larger values of θ which could be necessary if the true underlying model has larger coefficients.

2.2 Problem 3.2.2

We will now use an inverse gamma prior on θ_η^2. Given the probability density function of an inverse gamma distributed random variable z with parameters a and b, the log of the density function is:

$$\ln p(z \mid a, b) = \ln \left(\frac{b^a}{\Gamma(a)} z^{-a-1} \exp\left(-\frac{b}{z} \right) \right)$$

Expanding and simplifying using logarithmic properties, we have:

$$\ln p(z \mid a,b) = a \ln b - \ln(\Gamma(a)) - (a+1)\ln z - \frac{b}{z}$$

Omitting terms that do not depend on z, the expression reduces to:

$$\ln p(z \mid a,b) = -(a+1)\ln z - \frac{b}{z}.$$

2.3 Problem 3.2.3

We employ Bayes' theorem to derive the posterior distribution for the parameters θ and σ_η^2 given the observed data $\mathbf{y}$. Bayes' theorem relates the conditional and marginal probabilities of these parameters and data as follows:

$$p(\theta, \sigma_\eta^2 \mid \mathbf{y}) = \frac{p(\theta, \sigma_\eta^2, \mathbf{y})}{p(\mathbf{y})}$$

where $p(\theta, \sigma_\eta^2, \mathbf{y})$ denotes the joint probability of the parameters and the data, and $p(\mathbf{y})$ represents the marginal probability of the data. By applying the product rule, the joint probability can be expressed in terms of the likelihood of the data given the parameters and the joint probability of the parameters:

$$p(\theta, \sigma_\eta^2, \mathbf{y}) = p(\mathbf{y} \mid \theta, \sigma_\eta^2) \cdot p(\theta, \sigma_\eta^2)$$

Here, $p(\mathbf{y} \mid \theta, \sigma_\eta^2)$ is the likelihood of the data conditioned on the parameters. Assuming conditional independence between θ and σ_η^2, the joint parameter distribution can be decomposed as follows:

$$p(\theta, \sigma_\eta^2) = p(\theta \mid \sigma_\eta^2) \cdot p(\sigma_\eta^2)$$

Under the independence assumption, where the distribution of θ does not depend on σ_η^2, this simplifies to:

$$p(\theta \mid \sigma_\eta^2) = p(\theta)$$

Consequently, the joint parameter distribution simplifies to:

$$p(\theta, \sigma_\eta^2) = p(\theta) \cdot p(\sigma_\eta^2)$$

Finally, substituting these expressions back into Bayes' theorem, we obtain the posterior probability:

$$p(\theta, \sigma_\eta^2 \mid \mathbf{y}) = \frac{p(\mathbf{y} \mid \theta, \sigma_\eta^2) \cdot p(\theta) \cdot p(\sigma_\eta^2)}{p(\mathbf{y})}$$

Taking the logarithm of both sides, we obtain the log-posterior:

$$\begin{aligned} \ln p(\theta, \sigma_\eta^2 \mid \mathbf{y}) &= \ln p(\mathbf{y} \mid \sigma_\eta^2, \theta) + \ln p(\sigma_\eta^2) + \ln p(\theta) - \ln p(\mathbf{y}) \\ &\propto \ln p(\mathbf{y} \mid \sigma_\eta^2, \theta) + \ln p(\sigma_\eta^2) + \ln p(\theta)), \end{aligned}$$

where:

- The likelihood $p(\mathbf{y} \mid \sigma_\eta^2, \theta)$ is assumed to follow a normal distribution, specifically $\mathcal{N}(\mathbf{X}\theta, \sigma_\eta^2 \mathbf{I})$, leading to (this was derived in exercise 9.1.2):

$$\begin{aligned}\ln p(\mathbf{y} \mid \sigma_\eta^2, \theta) &= -\frac{1}{2\sigma_\eta^2}\|\mathbf{y} - \mathbf{X}\theta\|^2 - \frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\sigma_\eta^2) \\ &\propto -\frac{1}{2\sigma_\eta^2}\|\mathbf{y} - \mathbf{X}\theta\|^2 - \frac{n}{2}\ln(\sigma_\eta^2)\end{aligned}$$

- The prior on σ_η^2, following an inverse gamma distribution Inv-Gamma(a,b), that was derived in the previous problem:

$$\ln p(\sigma_\eta^2) \propto -(a+1)\ln \sigma_\eta^2 - \frac{b}{\sigma_\eta^2}$$

- The prior on θ, assuming a normal distribution $\mathcal{N}(\mathbf{0}, \sigma_\theta^2 \mathbf{I})$, is expressed as:

$$\begin{aligned}\ln p(\theta) &= -\frac{1}{2\sigma_\theta^2}\|\theta\|^2 - \frac{k}{2}\ln(2\pi) - \frac{k}{2}\ln(\sigma_\theta^2) \\ &\propto -\frac{1}{2\sigma_\theta^2}\|\theta\|^2 - \frac{k}{2}\ln(\sigma_\theta^2)\end{aligned}$$

Thus, the log-posterior is:

$$\ln p(\theta, \sigma_\eta^2 \mid \mathbf{y}) \propto -\frac{1}{2\sigma_\eta^2}\|\mathbf{y} - \mathbf{X}\theta\|^2 - \frac{n}{2}\ln(\sigma_\eta^2) - (a+1)\ln \sigma_\eta^2 - \frac{b}{\sigma_\eta^2} - \frac{1}{2\sigma_\theta^2}\|\theta\|^2 - \frac{k}{2}\ln(\sigma_\theta^2)$$

3 Problem 3.3: Estimation of ICA solution

The Independent Component Analysis (ICA) model is formulated as:

$$\mathbf{x} = A\mathbf{s}$$

where $\mathbf{x}$ is a random vector and we have observed N realizations of $\mathbf{x}$. The goal is to recover the mixing matrix A used to mix the sources $\mathbf{s}$. We assume that A is a square invertible matrix, allowing us to express the transformation as:

$$\mathbf{z} = W\mathbf{x} = WAs$$

where $W = A^{-1}$ when perfect recovery is achieved. Throughout this problem, we specify the matrix A as:

$$A = \begin{bmatrix}\mathbf{a}_1 & \mathbf{a}_2\end{bmatrix}, \quad \mathbf{a}_1 = \begin{bmatrix}3 \\ 1\end{bmatrix}, \quad \mathbf{a}_2 = \begin{bmatrix}1 \\ 1\end{bmatrix}.$$

3.1 Problem 3.3.1

Assume **s** follows a uniform distribution $U(0,1)$. The sources are generated using the following code:

```python
def generate_sources(num_samples):
    return np.random.uniform(0, 1, (2, num_samples))
```

The program below generates data and estimates A based on ICA:

```python
def perform_experiment(num_samples, A, mu, components, iterations):
    s = generate_sources(num_samples)  # Generate sources
    x = (A @ s).T  # Mix sources
    x -= np.mean(x, axis=0)  # Centering the data

    W = ICA(x, mu, components, iterations, 'subGauss') # Run ICA
    A_hat = np.linalg.inv(W) # Estimated mixing matrix
    W = np.divide(W, np.max(W)) # Normalize unmixing matrix
    z = (W @ x.T).T # Compute unmixed signals
    return A_hat, x, s, z
```

Figure 3 displays an example of the sources, the observations, and the estimated sources.

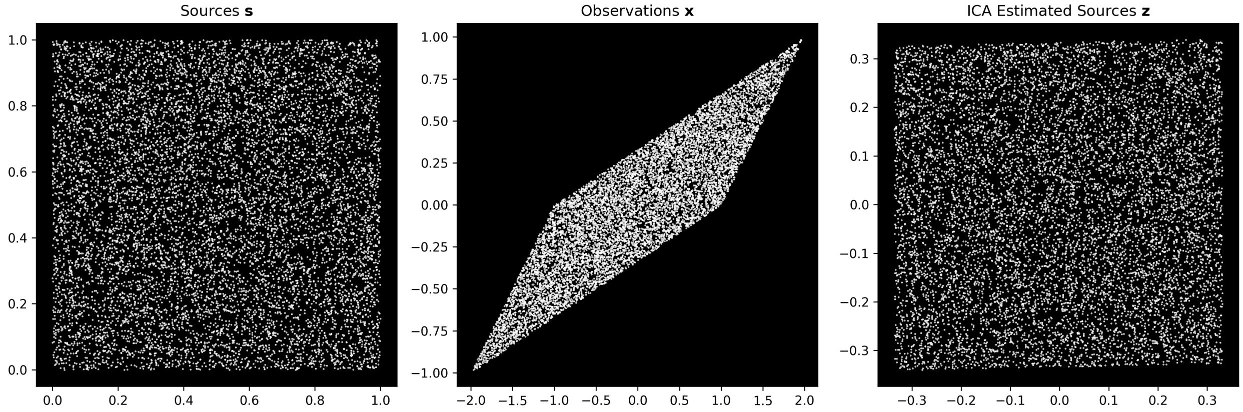

Figure 3. Sources, observations, and estimated sources using ICA.

To analyze the error between the estimated matrix $\hat{A}$ and the true matrix A, we use a measure that can handle differences in the order of basis elements and in their magnitudes. This is done by normalizing the matrices and then computing the difference of their Frobenius norms. The code below shows the error function:

```python
def compute_error(A, A_hat):
    A_norm = A / np.max(A)  # Normalize true mixing matrix
    A_hat_norm = A_hat / np.max(A_hat)  # Normalize estimated mixing matrix
    return np.linalg.norm(A_norm - A_hat_norm, 'fro')
```

The experiment has been run 100 times to analyze the errors of the estimates of A:

```python
# Initialize lists to store results
A_hats = []
errors = []

# Run experiments 100 times
```

```
for i in range(num_experiments):
    A_hat, x, s, z = perform_experiment(num_samples, A, mu, components, iterations)
    A_hats.append(A_hat)
    errors.append(compute_error(A, A_hat))
```

The errors are visualized in Figure 4, showing a distribution closely centered around zero.

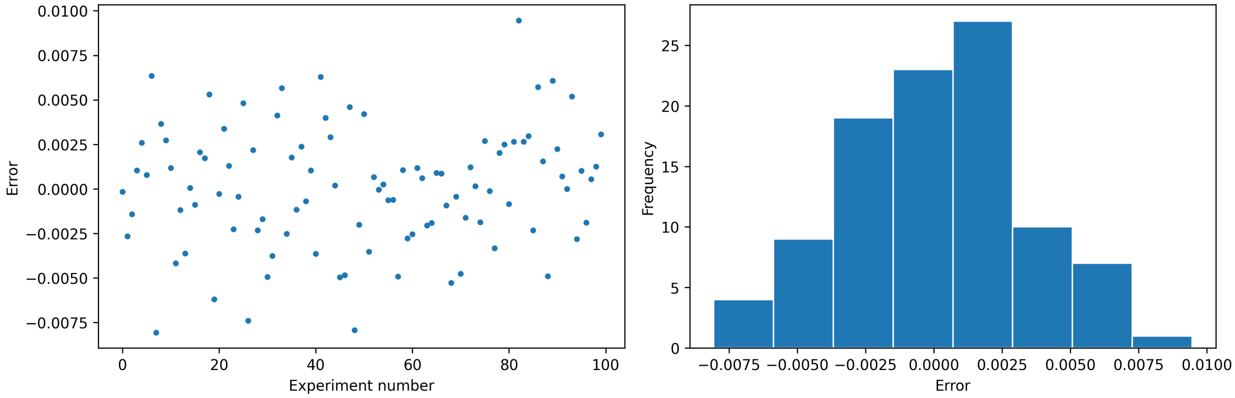

Figure 4. Distribution of errors over 100 experiments.

3.2 Problem 3.3.2

The experiment is repeated for the following cases:

- **Case 1:** s_1 is drawn from a uniform distribution $U(0,1)$, and s_2 is drawn from a beta distribution $B(0.1,0.1)$.
- **Case 2:** s_1 is drawn from a uniform distribution $U(0,1)$, and s_2 is drawn from a normal distribution $N(0,1)$.
- **Case 3:** $\mathbf{s}$ is drawn from a multivariate normal distribution with mean vector $\mu = (0,1)$ and covariance matrix $\Sigma = \begin{bmatrix} 2 & 0.25 \\ 0.25 & 1 \end{bmatrix}$.

  In cases 1 and 2, it is possible to estimate the mixing matrix A because the components involve non-Gaussian distributions or at most one Gaussian component, which is conducive to the assumptions of ICA. However, in case 3, ICA fails to properly separate the components. ICA relies on the assumption that each component it identifies should be statistically independent from the others, i.e., $p(z_i, z_j) = p(z_i)p(z_j)$ for $i \neq j$.

  In case 3, not only is this condition of independence not met - due to the multivariate normal distribution's non-diagonal covariance matrix indicating correlation between sources - but the presence of multiple Gaussian components. This hinders the ICA's ability to uniquely identify and separate the source signals, as the sum of Gaussian random variables remains Gaussian, thus lacking the distinct non-Gaussian features required for effective separation by ICA.

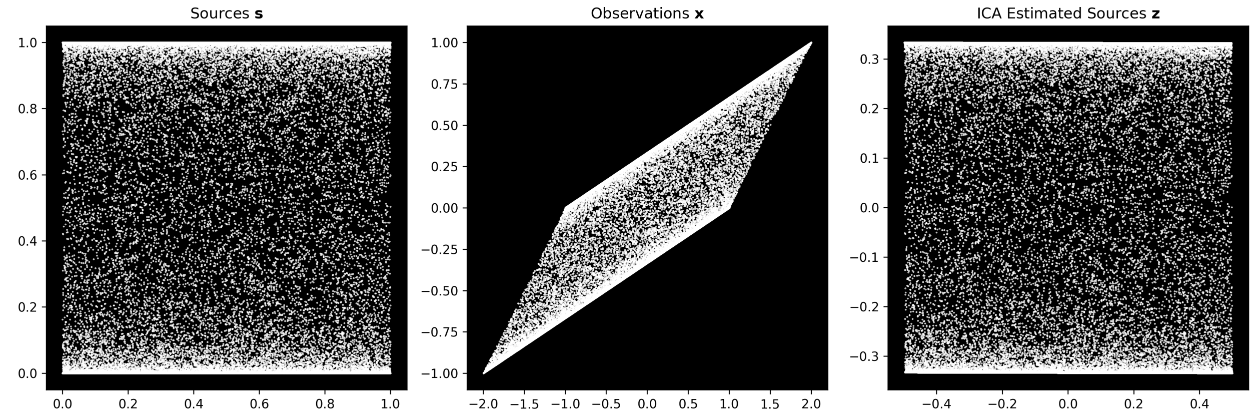

Figure 5. Case 1.

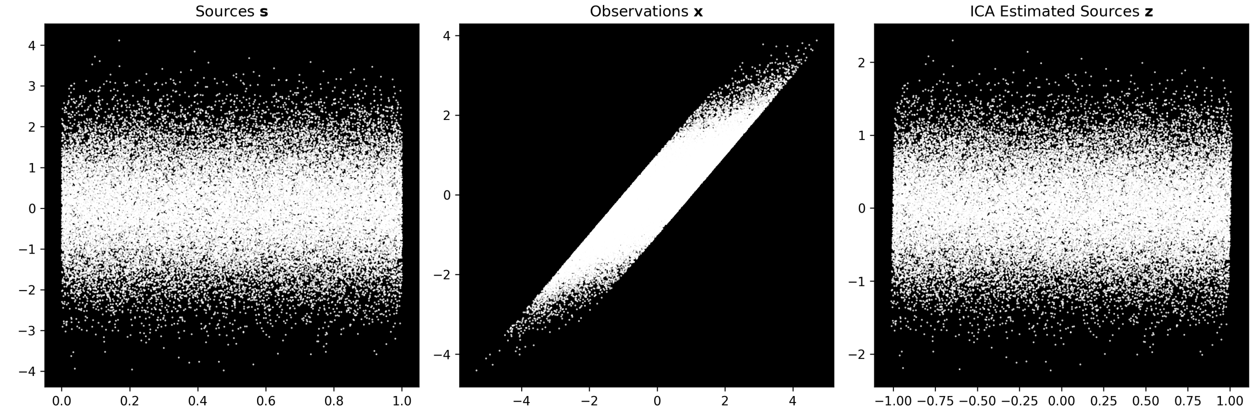

Figure 6. Case 2.

4 Problem 3.4: Hidden Markov Models

Consider a system characterized by two internal states, denoted s_1 and s_2, and three possible activities, denoted a_1, a_2, and a_3. The internal states of the system are not directly observable; instead, they must be estimated by observing the activities.

4.1 Problem 3.4.1

The configuration for the Hidden Markov Model includes:

- **Number of States**: $K = 2$
- **Initial State Probabilities**:
  - Probability of starting in state s_1, $P_1 = 0.60$
  - Probability of starting in state s_2, $P_2 = 0.40$
- **Transition Probabilities**:
  - Probability of staying in state s_1, $P_{11} = 0.90$

  - Probability of transitioning from state s_1 to s_2, $P_{12} = 0.10$
  - Probability of transitioning from state s_2 to s_1, $P_{21} = 0.35$
  - Probability of staying in state s_2, $P_{22} = 0.65$

- **State Emission Probabilities**:
  - Probability of emitting a_1 from state s_1, $p(a_1 \mid s_1) = 0.60$
  - Probability of emitting a_2 from state s_1, $p(a_2 \mid s_1) = 0.30$
  - Probability of emitting a_3 from state s_1, $p(a_3 \mid s_1) = 0.10$
  - Probability of emitting a_1 from state s_2, $p(a_1 \mid s_2) = 0.10$
  - Probability of emitting a_2 from state s_2, $p(a_2 \mid s_2) = 0.60$
  - Probability of emitting a_3 from state s_2, $p(a_3 \mid s_2) = 0.30$

4.2 Problem 3.4.2

The filtering recursion updates the probability distribution of the current state based on all available information up to the present. The recursion is expressed as:

$$\alpha(\mathbf{x}_n) = p(\mathbf{y}_n \mid \mathbf{x}_n) \sum_{\mathbf{x}_{n-1}} \alpha(\mathbf{x}_{n-1}) P(\mathbf{x}_n \mid \mathbf{x}_{n-1})$$

where: $p(\mathbf{y}_n \mid \mathbf{x}_n)$ is the probability of observing $\mathbf{y}_n$ given the state $\mathbf{x}_n$, $\alpha(\mathbf{x}_{n-1})$ represents the posterior probability of the state $\mathbf{x}_{n-1}$ given all observations up to time $n-1$ and $P(\mathbf{x}_n \mid \mathbf{x}_{n-1})$ denotes the transition probability from state $\mathbf{x}_{n-1}$ to state $\mathbf{x}_n$.

For $n = 1$, we have observed $y_1 = a_1$, which leads to the equations:

$$\begin{aligned}\alpha(x_1 = s_1) &= p(y_1 = a_1 | x_1 = s_1) p(x_1 = s_1)\\ &= 0.60 \cdot 0.60\\ &= \mathbf{0.360}\end{aligned}$$

$$\begin{aligned}\alpha(x_1 = s_2) &= p(y_1 = a_1 | x_1 = s_2) p(x_1 = s_2)\\ &= 0.10 \cdot 0.40\\ &= \mathbf{0.040}\end{aligned}$$

For $n = 2$, we have observed $y_2 = a_2$, which leads to the equations:

$$\begin{aligned}\alpha(x_2 = s_1) &= p(y_2 = a_2 | x_2 = s_1) \cdot (\alpha(x_1 = s_1) p(x_2 = s_1 | x_1 = s_1) + \alpha(x_1 = s_2) \cdot P(x_2 = s_1 | x_1 = s_2))\\ &= 0.30 \cdot (0.360 \cdot 0.90 + 0.040 \cdot 0.35)\\ &= \mathbf{0.101}\end{aligned}$$

$$\begin{aligned}\alpha(x_2 = s_2) &= p(y_2 = a_2 | x_2 = s_2) \cdot (\alpha(x_1 = s_1) p(x_2 = s_2 | x_1 = s_1) + \alpha(x_1 = s_2) \cdot P(x_2 = s_2 | x_1 = s_2))\\ &= 0.60 \cdot (0.360 \cdot 0.10 + 0.040 \cdot 0.65)\\ &= \mathbf{0.037}\end{aligned}$$

For $n = 3$, we have observed $y_2 = a_1$, which leads to the equations:

$$\begin{aligned}\alpha(x_3 = s_1) &= p(y_3 = a_1|x_3 = s_1) \cdot (\alpha(x_2 = s_1)p(x_3 = s_1|x_2 = s_1) + \alpha(x_2 = s_2) \cdot P(x_3 = s_1|x_2 = s_2))\\ &= 0.60 \cdot (0.101 \cdot 0.90 + 0.037 \cdot 0.35)\\ &= \mathbf{0.063}\end{aligned}$$

$$\begin{aligned}\alpha(x_3 = s_2) &= p(y_3 = a_1|x_3 = s_2) \cdot (\alpha(x_2 = s_1)p(x_3 = s_2|x_2 = s_1) + \alpha(x_2 = s_2) \cdot P(x_3 = s_2|x_2 = s_2))\\ &= 0.10 \cdot (0.101 \cdot 0.10 + 0.037 \cdot 0.65)\\ &= \mathbf{0.003}\end{aligned}$$

To compute the conditional probability $P(x_2 = s_1 \mid y_{1:2})$, we use the normalized forward probabilities. The general formula for computing the posterior probability of state $\mathbf{x}_n$ given the sequence of observations up to time n is:

$$P(\mathbf{x}_n \mid Y_{1:n}) = \frac{\alpha(\mathbf{x}_n)}{p(Y_{1:n})} = \frac{\alpha(\mathbf{x}_n)}{\sum_{\mathbf{x}_n} \alpha(\mathbf{x}_n)}.$$

For the specific case of x_2 and observations $y_{1:2}$, the probability that x_2 is in state s_1 is computed as:

$$P(x_2 = s_1 \mid y_{1:2}) = \frac{\alpha(x_2 = s_1)}{\alpha(x_2 = s_1) + \alpha(x_2 = s_2)} = \frac{0.101}{0.101 + 0.037} = \mathbf{0.732}.$$

4.3 Problem 3.4.3

The probability of observing y_3 given the state x_2 can be derived by marginalizing over the possible states at time 3, x_3. Assuming the Markov property (each state depends only on the previous state) and the independence between the state transitions and the observations, we have:

$$P(y_3 \mid x_2) = \sum_{x_3} P(y_3 \mid x_3)P(x_3 \mid x_2).$$

For the specific cases we get

$$\begin{aligned}P(y_3 = a_1|x_2 = s_1) &= P(y_3 = a_1 \mid x_3 = s_1)P(x_3 = s_1 \mid x_2 = s_1) + P(y_3 = a_1 \mid x_3 = s_2)P(x_3 = s_2 \mid x_2 = s_1)\\ &= 0.60 \cdot 0.90 + 0.10 \cdot 0.10\\ &= \mathbf{0.550}\end{aligned}$$

and

$$\begin{aligned}P(y_3 = a_1|x_2 = s_2) &= P(y_3 = a_1 \mid x_3 = s_1)P(x_3 = s_1 \mid x_2 = s_2) + P(y_3 = a_1 \mid x_3 = s_2)P(x_3 = s_2 \mid x_2 = s_2)\\ &= 0.60 \cdot 0.35 + 0.10 \cdot 0.65\\ &= \mathbf{0.275}.\end{aligned}$$

4.4 Problem 3.4.4

We compute $P(x_2 = s_1 \mid y_{1:3})$ using the Bayes' rule and the law of total probability:

$$\begin{aligned}P(x_2 = s_1 \mid y_{1:3}) &= \frac{P(y_3 = a_1 \mid x_2 = s_1)P(x_2 = s_1 \mid y_{1:2})}{P(y_3 = a_1 \mid x_2 = s_1)P(x_2 = s_1 \mid y_{1:2}) + P(y_3 = a_1 \mid x_2 = s_2)P(x_2 = s_2 \mid y_{1:2})}\\ &= \frac{0.550 \cdot 0.732}{0.550 \cdot 0.732 + 0.275 \cdot (1 - 0.732)}\\ &= \mathbf{0.845}.\end{aligned}$$

5 Problem 3.5: Kalman Filter

5.1 Problem 3.5.1

In this problem, we consider the movement of a train in a one-dimensional space. The train's motion is modeled using a Kalman filter that tracks both the position and velocity but only observes the position. The state of the train at any time n is described by the following state vector:

$$\mathbf{x}_n = \begin{bmatrix} p_n \\ v_n \end{bmatrix}$$

where p_n and v_n represent the position and velocity of the train, respectively. The state transition model, which predicts the next state based on the current state, is given by:

$$\mathbf{x}_n = \mathbf{F}_n \mathbf{x}_{n-1} + \mathbf{w}_n$$

with the state transition matrix:

$$\mathbf{F}_n = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$$

where Δt is the time step, and $\mathbf{w}_n$ is the process noise, assumed to be Gaussian. The observation model relates the observable data to the state vector:

$$y_n = \mathbf{H}_n \mathbf{x}_n + v_n$$

with the observation matrix: $\mathbf{H}_n = \begin{bmatrix} 1 & 0 \end{bmatrix}$. Here, v_n represents the measurement noise, which is also assumed to be Gaussian with variance, $R = 0.1$, i.e. $v_n \sim \mathcal{N}(0, 0.1)$. The process noise is characterized by its covariance matrix $\mathbf{Q}$ where the variances are also assumed to be 0.1 for both position and velocity:

$$\mathbf{Q} = \begin{bmatrix} 0.1 & 0.0 \\ 0.0 & 0.1 \end{bmatrix}.$$

The initial state $\mathbf{x}_0$ and the initial covariance matrix $\mathbf{P}_0$ are set to:

$$\mathbf{x}_0 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \mathbf{P}_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The Kalman filter proceeds through a series of prediction and update steps. For each time step:

- **Prediction:**

$$\hat{\mathbf{x}}_{n|n-1} = \mathbf{F}_n \hat{\mathbf{x}}_{n-1|n-1},$$
$$\mathbf{P}_{n|n-1} = \mathbf{F}_n \mathbf{P}_{n-1|n-1} \mathbf{F}_n^T + \mathbf{Q}$$

- **Update:**

$$\mathbf{K}_n = \mathbf{P}_{n|n-1} \mathbf{H}_n^T (\mathbf{H}_n \mathbf{P}_{n|n-1} \mathbf{H}_n^T + \mathbf{R})^{-1}$$
$$\hat{\mathbf{x}}_{n|n} = \hat{\mathbf{x}}_{n|n-1} + \mathbf{K}_n (y_n - \mathbf{H}_n \hat{\mathbf{x}}_{n|n-1})$$
$$\mathbf{P}_{n|n} = (\mathbf{I} - \mathbf{K}_n \mathbf{H}_n) \mathbf{P}_{n|n-1}$$

The tracking performance is shown in Figure 7

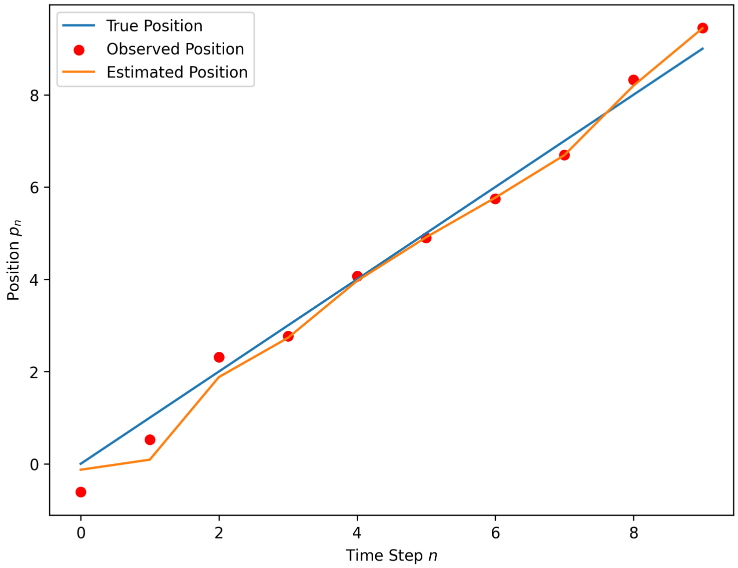

Figure 7. Tracking performance.

6 Problem 3.6: Kernel Methods

6.1 Problem 3.6.1

Figure 8 shows a chirp signal. The center location of the chirp is located at time index 26, which corresponds to $t = 5.2$.

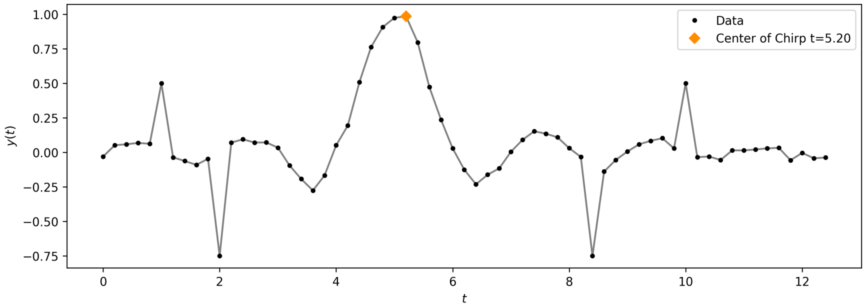

Figure 8. Raw signal

6.2 Problem 3.6.2

Kernel Ridge Regression (KRR) has been employed to analyze the signal, utilizing a Gaussian kernel with parameters set at $C = 1 \times 10^{-2}$ and $\sigma = 1$. The results from this regression are depicted in Figure 9, showcasing the fitted values against the observed data.

The SNR was calculated using the following formulas:

$$\text{SNR} = 10\log_{10}\left(\frac{\text{Signal Power}}{\text{Noise Power}}\right)$$

where the Signal Power is computed as the variance of the predicted values, and the Noise Power is derived from the variance of the residuals. The SNR obtained from the analysis was 7.62 dB. The center location of the chirp, identified as the peak of the estimated signal, was determined to be at $t = 5$.

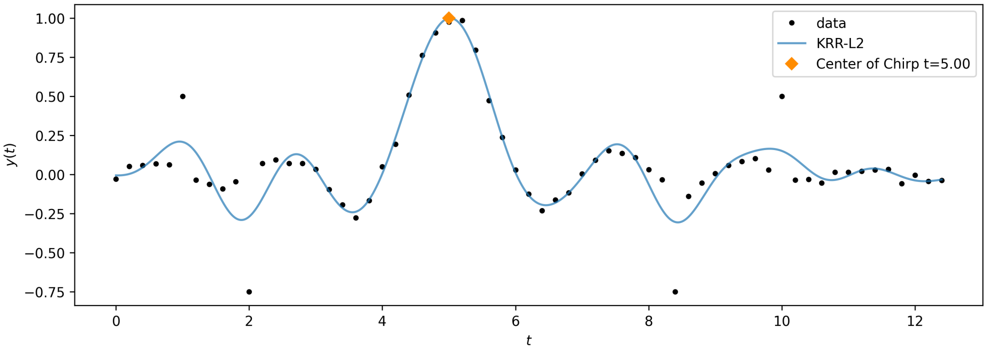

Figure 9. Kernel Ridge Regression fit and observed data.

6.3 Problem 3.6.3

Support Vector Regression (SVR) was also applied using a Gaussian kernel to analyze the signal. The SVR parameters were set as follows: $C = 1$, $\gamma = 1$, and $\varepsilon = 0.003$. The resulting fit is visualized in Figure 10.

A simple rule was implemented to identify outliers in the observed data: any data points lying beyond the 95th percentile of the residuals were classified as outliers. The SNR was computed excluding the identified outliers, ensuring a more accurate representation of the signal quality. The computed SNR value is 20.70 dB, indicating a good separation between the signal and the noise, hence a high quality of the reconstructed signal. The center of the chirp was determined to be at $t = 5.0$.

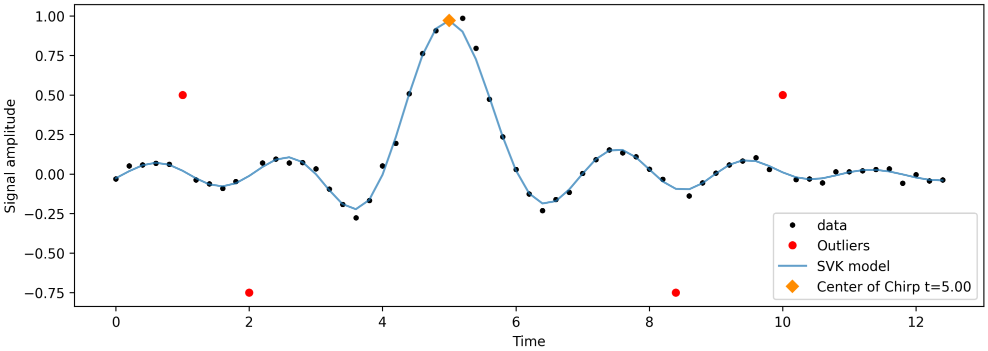

Figure 10. SVR fit and observed data.