Detailed Explanation of Finite Markov Chains

A **Markov Chain** is a stochastic process where the future state depends only on the current state and not on the sequence of past states. This is known as the **Markov Property**. Finite Markov Chains are a special type of Markov Chain where the state space (the set of all possible states) is finite.

Key Concepts

1. State Space

The **state space** is the set of all possible states of the Markov Chain. Let the state space be denoted by:

$$S = \{s_1, s_2, \ldots, s_n\}.$$

2. Transition Probabilities

The **transition probability** is the probability of moving from one state to another in one step. It is denoted by:

$$P(X_{t+1} = s_j \mid X_t = s_i) = P_{ij}$$

Where:

- P_{ij} is the probability of transitioning from state s_i to state s_j.

  These probabilities are typically represented in a **transition matrix**:

$$P = \begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1n} \\ P_{21} & P_{22} & \cdots & P_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ P_{n1} & P_{n2} & \cdots & P_{nn} \end{bmatrix}$$

Where:

- $P_{ij} \geq 0$ for all i, j,
- $\sum_{j=1}^{n} P_{ij} = 1$ (Each row sums to 1).

3. Initial Distribution

The **initial distribution** specifies the probabilities of the chain starting in each state at time $t = 0$. It is denoted by:

$$\pi^{(0)} = [\pi_1^{(0)}, \pi_2^{(0)}, \ldots, \pi_n^{(0)}]$$

Where $\pi_i^{(0)} = P(X_0 = s_i)$ and $\sum_{i=1}^{n} \pi_i^{(0)} = 1$.

4. n-Step Transition Probabilities

The n-step transition probabilities represent the probabilities of transitioning from one state to another in n steps:

$$P^{(n)} = P^n$$

Where P^n is the n-th power of the transition matrix.

5. Stationary Distribution

A **stationary distribution** is a probability distribution that remains unchanged as the chain evolves. It satisfies:

$$\pi P = \pi$$

Where $\pi = [\pi_1, \pi_2, \ldots, \pi_n]$ and $\sum_{i=1}^{n} \pi_i = 1$.

6. Classification of States

1. **Recurrent States**: A state s_i is recurrent if the chain is guaranteed to return to it at some point.

2. **Transient States**: A state s_i is transient if there is a non-zero probability of never returning to it.

3. **Absorbing States**: A state s_i is absorbing if $P_{ii} = 1$, meaning once the chain enters this state, it never leaves.

7. Ergodicity

A Markov Chain is **ergodic** if:

1. It is irreducible (every state is reachable from every other state),

2. It is aperiodic (the chain does not cycle between states in fixed intervals).

An ergodic Markov Chain has a unique stationary distribution.

Examples

Example 1: Weather Model

Consider a weather system with two states:

- $S_1 = \text{Sunny}$
- $S_2 = \text{Rainy}$

The transition probabilities are:

$$P = \begin{bmatrix} 0.8 & 0.2 \\ 0.5 & 0.5 \end{bmatrix}$$

Where:

- $P_{11} = 0.8$: Probability of sunny today and sunny tomorrow.
- $P_{12} = 0.2$: Probability of sunny today and rainy tomorrow.

If the initial distribution is $\pi^{(0)} = [1, 0]$ (starts sunny), the probabilities for future days can be computed by multiplying $\pi^{(0)}$ by P:

$$\pi^{(1)} = \pi^{(0)} P$$

Example 2: Stationary Distribution

Find the stationary distribution for the weather model above:

$$\pi P = \pi \quad \text{and} \quad \pi_1 + \pi_2 = 1$$

This leads to the system of equations:

$$\pi_1 = 0.8\pi_1 + 0.5\pi_2$$

$$\pi_2 = 0.2\pi_1 + 0.5\pi_2$$

Solving these equations gives:

$$\pi = [0.714, 0.286]$$

Python Implementation

Transition Matrix and n-Step Transition Probabilities

```
import numpy as np

# Transition matrix
P = np.array([
    [0.8, 0.2],
    [0.5, 0.5]
])

# Initial distribution
pi_0 = np.array([1, 0])  # Starts sunny

# Compute probabilities after 3 steps
n = 3
pi_n = np.dot(pi_0, np.linalg.matrix_power(P, n))
print(f"State probabilities after {n} steps: {pi_n}")
```

Stationary Distribution

```
# Find stationary distribution
eigvals, eigvecs = np.linalg.eig(P.T)
stationary = eigvecs[:, np.isclose(eigvals, 1)]  # Eigenvector corresponding to eigenvalue 1
stationary = stationary / stationary.sum()  # Normalize
print(f"Stationary Distribution: {stationary.flatten()}")
```

Key Applications

1. **Weather Prediction**: Modeling weather transitions over time.
2. **Queueing Systems**: Modeling customer arrivals and service processes.
3. **PageRank Algorithm**: Ranking webpages based on a Markov Chain of link structures.
4. **Economics**: Modeling economic states or stock market trends.