stat201ahw

Question 1

a.

The matrix is defined as follows:

$$\begin{pmatrix} 0.2 & 0.7 & 0.1 \\ 0.2 & 0.5 & 0.3 \\ 0.2 & 0.4 & 0.4 \end{pmatrix}$$

b.

```python
import numpy as np

transition_matrix = np.array([
    [0.2, 0.7, 0.1],
    [0.2, 0.5, 0.3],
    [0.2, 0.4, 0.4]
])

pi_0 = np.array([1, 0, 0])

pi_1 = np.dot(pi_0, transition_matrix)

pi_1
```

```
array([0.2, 0.7, 0.1])
```

```python
import numpy as np
import random

P = np.array([
    [0.2, 0.7, 0.1],
    [0.2, 0.5, 0.3],
    [0.2, 0.4, 0.4]
])

current_state = 0

num_steps = 10

# Markov Chain
states = [current_state + 1]
for _ in range(num_steps):
    current_state = np.random.choice([0, 1, 2], p = P[current_state])
    states.append(current_state + 1)

print(states)
```

```
[1, 2, 1, 1, 2, 2, 3, 2, 2, 2, 1]
```

The above code shows how the chain changes based on 10 steps.

Question 2

a.

```python
from scipy.linalg import eig

P = np.array([
    [0.2, 0.7, 0.1],
    [0.2, 0.5, 0.3],
    [0.2, 0.4, 0.4],
])


pmatrixt = P.T
```

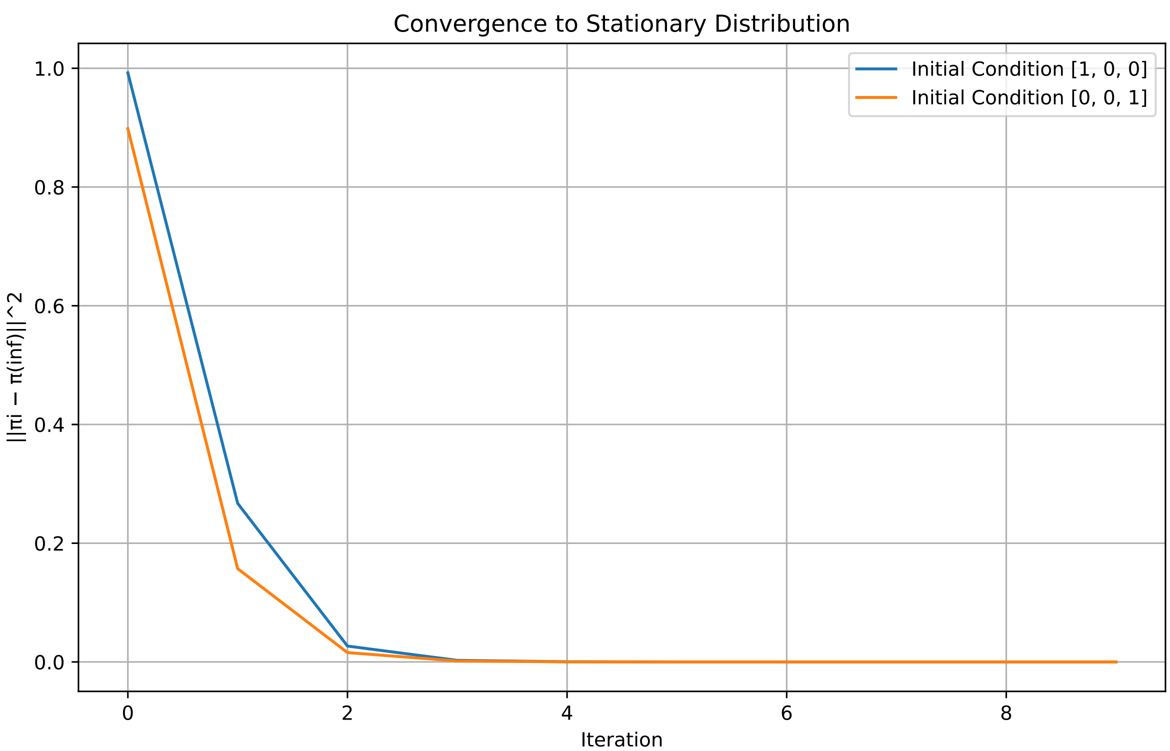
```
eig_val, eig_vec = eig(pmatrixt)


eig_vec = eig_vec[:, np.isclose(eig_val, 1, atol=1e-8)]

stationary_distribution = eig_vec / np.sum(eig_vec)

stationary_distribution.real
```

```
array([[0.2       ],
       [0.51111111],
       [0.28888889]])
```

b.

```
import matplotlib.pyplot as plt
import numpy as np

# Define the transition matrix P
pmatrix = np.array([
    [0.2, 0.7, 0.1],
    [0.2, 0.5, 0.3],
    [0.2, 0.4, 0.4]
])

stationary_vector_real = np.array([0.2, 0.51111111, 0.28888889])

num_iterations = 10


def calculate_pi_i(pi_0, pmatrix, i):
    return pi_0.dot(np.linalg.matrix_power(pmatrix, i))

pi0_a = np.array([1, 0, 0])
pi0_b = np.array([0, 0, 1])

norms_a = []
norms_b = []

for i in range(num_iterations):
```

```
    pi_i_a = calculate_pi_i(pi0_a, pmatrix, i)
    pi_i_b = calculate_pi_i(pi0_b, pmatrix, i)

    norm_a = np.linalg.norm(pi_i_a - stationary_vector_real, 2)
    norm_b = np.linalg.norm(pi_i_b - stationary_vector_real, 2)

    norms_a.append(norm_a)
    norms_b.append(norm_b)

# Plotting
plt.figure(figsize=(10, 6))
plt.plot(range(num_iterations), norms_a, label='Initial Condition [1, 0, 0]')
plt.plot(range(num_iterations), norms_b, label='Initial Condition [0, 0, 1]')
plt.xlabel('Iteration')
plt.ylabel('|| i -  (inf)||^2')
plt.title('Convergence to Stationary Distribution')
plt.legend()
plt.grid(True)
plt.show()
```

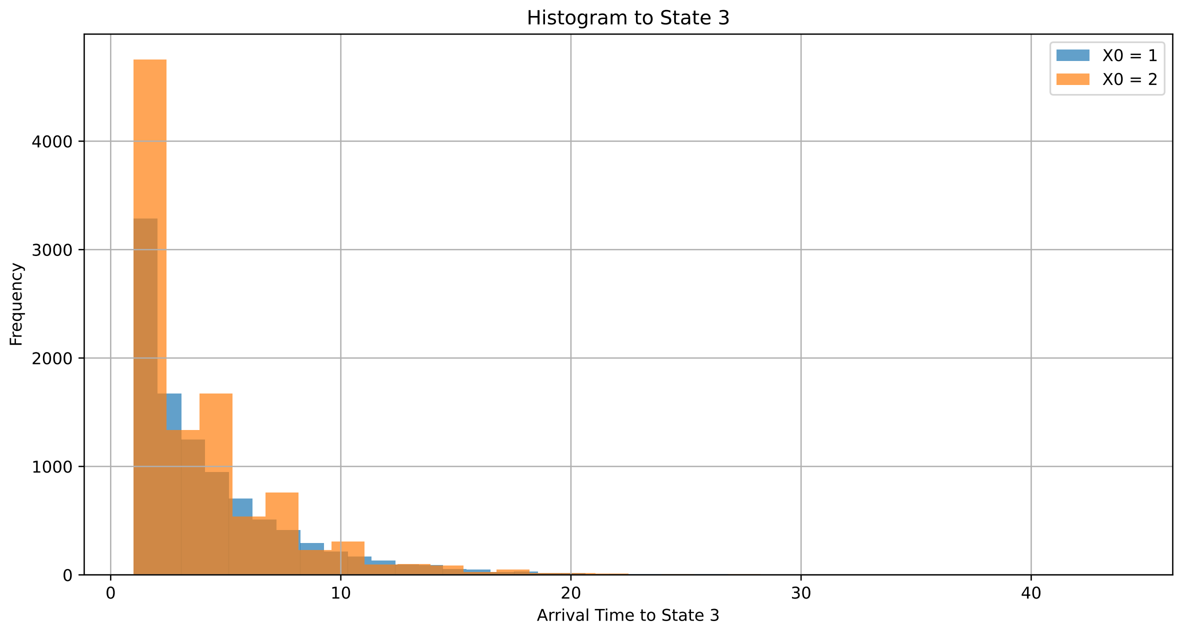


Question 3

a.

```python
import matplotlib.pyplot as plt
import random
random.seed(123)

def simulate_until_state_3(pmatrix, start_state):
    """Simulate the Markov chain until it reaches state 3."""
    current_state = start_state
    time_steps = 0

    while current_state != 3:
        current_state = np.random.choice([1, 2, 3], p=pmatrix[current_state - 1])
        time_steps += 1

    return time_steps


num_simulations = 10000

# Simulate for X0 = 1 and X0 = 2
arrival_times_1 = [
    simulate_until_state_3(pmatrix, start_state=1)
    for _ in range(num_simulations)
]
arrival_times_2 = [
    simulate_until_state_3(pmatrix, start_state=2)
    for _ in range(num_simulations)
]

mean_time_1 = np.mean(arrival_times_1)
mean_time_2 = np.mean(arrival_times_2)

# Plot histograms
plt.figure(figsize=(12, 6))
plt.hist(arrival_times_1, bins=30, alpha=0.7, label='X0 = 1')
plt.hist(arrival_times_2, bins=30, alpha=0.7, label='X0 = 2')
plt.xlabel('Arrival Time to State 3')
plt.ylabel('Frequency')
```

```
plt.title('Histogram to State 3')
plt.legend()
plt.grid(True)
plt.show()

print(f"Mean arrival time from state 1 is {mean_time_1}")
print(f"Mean arrival time from state 2 is {mean_time_2}")
```




```
Mean arrival time from state 1 is 4.6376
Mean arrival time from state 2 is 3.8359
```

b.

The theoretical and simulated mean arrival times are very similar.

The theoretical mean arrival times (depending on the simulation) was approx. 3.846 for mu_2 and 4.615 for mu_1, which is very similar to the simulated arrival times. These values are close to the simulated values found in question 3a.

The theoretical mean arrival times were calculated as follows:

$$u_1 = 1 + \sum_{j=1}^{3} P_{1j} u_j$$

$$u_1 = 1 + P_{11} u + P_{12} u_2 + \cancel{P_{13} u_3}$$

$$u_2 = 1 + P_{21} u + P_{22} u_2 + \cancel{P_{23} u_3}$$

$$u_3 = 0$$

$$\begin{bmatrix} 0.2\, P_{11} & 0.7\, P_{12} & 0.1\, P_{13} \\ 0.2\, P_{21} & 0.5\, P_{22} & 0.3\, P_{23} \\ 0.2\, P_{31} & 0.4\, P_{32} & 0.4\, P_{33} \end{bmatrix}$$

$$u_1 = 1 + 0.2 u_1 + 0.7 u_2$$

$$u_2 = 1 + 0.2 u_1 + 0.5 u_2$$

$$0.8 u_1 - 0.7 u_2 = 1$$

$$(-0.2 u_1 + 0.5 u_2 = 1) \cdot 4$$

$$-0.8 u_2 + 2 u_2 = 4$$

$$\begin{array}{rl} & \cancel{0.8 u_1} - 0.7 u_2 = 1 \\ +) & -\cancel{0.8 u_2} + 2 u_2 = 4 \\ \hline & 1.3 u_2 = 5 \end{array}$$

$$u_2 = \frac{5}{1.3} = 3.84615$$

$$0.8 u_1 - (0.7)(3.84615) = 1$$

$$u_1 = 4.615384$$