Technical University of Denmark

02443 - Stochastic Simulation

Project 1: Task 2 Resubmission

Gunnhildur Katrín Erlendsdóttir - s212510

Helga Dís Halldórsdóttir - s212457

David Miles-Skov - s204755

Patrick Jensen Martins - s204748

Professor:

Bo Friis Nielsen

Contents

1 Task 2

In order to find the distribution over the states after 120 months ($\mathbf{p}_{120}$) produced by our simulations, the state of 10^3 women at $t = 120$ was simulated via Markov Chain Monte Carlo, with the algorithm terminating at month 120.

This simulated frequency distribution was compared with that of the formula:

$$\mathbf{p}_t = \mathbf{p}_0(\mathbf{P}^t) \tag{1}$$

Where $\mathbf{P}$ is given and $\mathbf{p}_0$ is determined to be $[1, 0, 0, 0, 0]$, as the initial state is always 1. The results are represented below.

Table 1: Comparison between simulated and analytical state frequency distributions at $t = 120$

State (i)	1	2	3	4	5
Simulated	364	152	170	68	246
Analytical	359	159	166	67	248

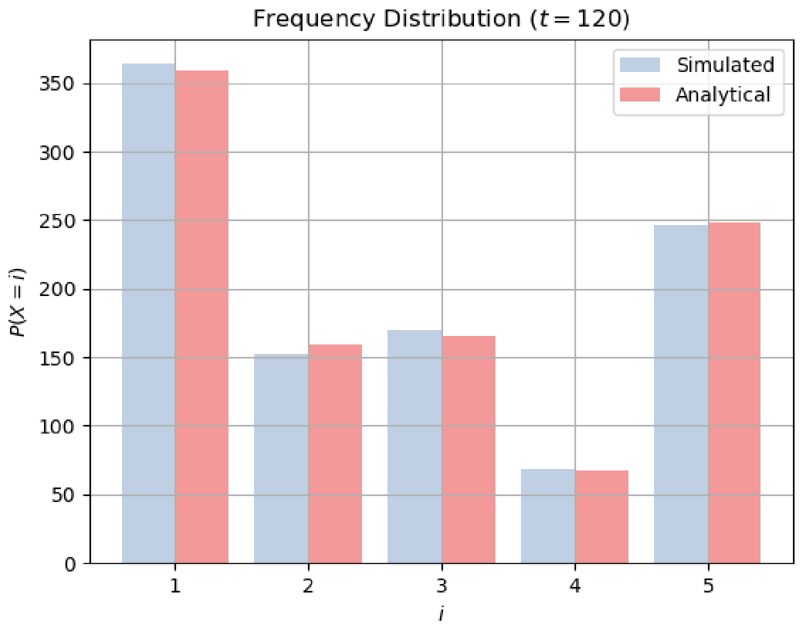

Figure 1: Comparison between simulated and analytical state distributions at $t = 120$

In order to evaluate the similarity of distributions, a χ^2 test was conducted, yielding the following results:

Table 2: Results of χ^2 test

Statistical Quantity	**Value**
χ^2 test statistic	0.486
P-value	0.975

A high p-value obtained from the χ^2 test, coupled with the visually similar distributions leads us to conclude that there is no significant difference between the two distributions.

2 Code

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats


def get_analytical_state_vector(P, t):


    P_t = np.linalg.matrix_power(P, t)
    p0 = np.array([1,0,0,0,0])

    return np.dot(p0, P_t)




# Markov Chain Monte Carlo

def get_state_at_120_MCMC():

    P = np.array([[0.9915, 0.005, 0.0025, 0, 0.001],
                  [0, 0.986, 0.005, 0.004, 0.005],
                  [0, 0, 0.992, 0.003, 0.005],
                  [0, 0, 0, 0.991, 0.009],
                  [0, 0, 0, 0, 1]])

    N = len(P[0])

    # initial state

    states = [0]

    for _ in range(120): # t = 0 -> 120

        currrent_state = states[-1] # Current state (as defined by previous
    transition)

        # performing transition

        transition_probabilities = P[currrent_state]
        new_state = np.random.choice(list(range(N)), p =
    transition_probabilities) # New column

        # Updating

        states.append(new_state)

    return states[-1] # State at t = 120

def task_2(n):
    """
    Comparing distribution over states at t = 120 for both
    analytical and simulated approach
    """

    # Generating analytical distribution

    t = 120
    P = np.array([[0.9915, 0.005, 0.0025, 0, 0.001],
                  [0, 0.986, 0.005, 0.004, 0.005],
                  [0, 0, 0.992, 0.003, 0.005],
                  [0, 0, 0, 0.991, 0.009],
                  [0, 0, 0, 0, 1]])

    analytical = 1000*get_analytical_state_vector(P, t)

    simulated = np.zeros(5)

    for i in range(n): # Simulating n women

        if i%100==0:
            print(f"simulated {i} samples")

        state_at_120 = get_state_at_120_MCMC()
        simulated[state_at_120] += 1 # Counting number of times in each
    state

    simulated = simulated # Normalizing
    print(f"Simulated: {simulated}")
    print(f"analytical: {analytical}")

    x = np.array(list(range(1, len(simulated)+1)))

    plt.bar(x-0.2, simulated, color="lightsteelblue", label="Simulated",
    alpha=0.8, width=0.4, align='center')
    plt.bar(x+0.2, analytical,  color="lightcoral", label="Analytical",
    alpha=0.8, width=0.4, align='center')
    plt.xticks(x)
    plt.xlabel(r"$i$")
    plt.ylabel(r"$P(X = i)$")
    plt.title("Frequency Distribution "+r"$(t=120)$")
    plt.legend()
    plt.grid()
    plt.show()




    t, p = stats.chisquare(f_obs=simulated, f_exp=analytical)
    print("chi-2 test:")
    print(f"Test statistic: {t}")
    print(f"P-value: {p}")




if __name__=="__main__":
    task_2(1000)
```