Applied Stochastic Processes (30515) - Assignment 2

Andrea Lisci

April 2024

1 Part I

1.1 The invariant distribution of X_t

By considering a Markov chain (X_t) with state space $\chi = \{1, 2, \ldots, n\}$ and transition matrix P, in full generality: let $\pi = \{\pi_1, \pi_2, \ldots, \pi_{|\chi|}\}$ be a vector of probabilities. We say that π is a stationary or invariant distribution for the markov chain (X_t) with transition matrix P if:

$$(\pi_j = \sum_{i=1}^{n} \pi_i P_{ij})$$

For all $j \in \chi$. Or, in matrix form:

$$\pi P = \pi \tag{1}$$

So, π is a left eigenvector of P with eigenvalue 1.

1.2 The limiting distribution of X_t

The limiting distribution of a Markov chain X_t is defined as the distribution reached by the Markov chain as time t approaches infinity, so:

$$\pi_j = \lim_{t \to \infty} P_{ij}^t$$

1.3 The asymptotic frequencies/distribution of X_t

The asymptotic frequencies or distribution of a Markov chain, instead of having a purely theoretical definition, relies more on the practical aspect of the chain, by applying the transition matrix a high number of times and calculating the proportion of time spent on each node of the chain.

$$\pi_i = \lim_{n \to \infty} \frac{Z_t}{n} \tag{2}$$

Where Z_t is the number of visits of (X_t) to state i before time n.

This three different approaches are trying to solve the same problem, finding a "stable" distribution that can help us understand how the Markov chain behaves. This "stable" and unique distribution exists only when some assumptions are met, the assumptions are part of the convergence theorem:

1.3.1 The Convergence Theorem

For a Markov chain X_t in finite state space χ with transition matrix P:

1. There exist a distribution π on χ such that $\pi P = \pi$
2. If X_t is irreducible, such stationary distribution π is unique

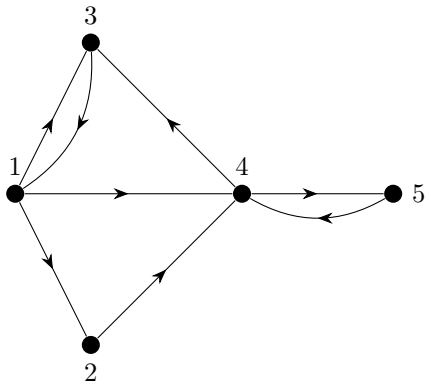
So: Let X_t be an irreducible (and aperiodic) Markov chain in finite state space χ with transition matrix P. Then:

$$P_{ij}^t \to \pi_j$$

as $t \to \infty$ for all $i, j \in \chi$, where π is the (unique) stationary distribution of (X_t). When the convergence theorem holds, we call π the limiting/equilibrium distribution of (X_t), and we say that (X_t) converges to stationarity.

2 Part II

In this second section we are dealing with a specific Markov chain model, called Pagerank, described by this directed graph:



2.1 Describe X_t by providing its state space and transition matrix

For this computation part I used Python, by analyzing the directed graph above, we can see that the state space is given by the total number of nodes, so $\chi = \{1, 2, 3, 4, 5\}$, I calculated the transition probabilities and I created a

python array that behaves like a 1 step transition matrix:

$$P = \begin{bmatrix} 0 & 1/3 & 1/3 & 1/3 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

(X_t) is certainly aperiodic, it has a finite state space and since there are not looping classes, it is irreducible. The convergence theorem holds, so there exists a unique π distribution for this Markov chain.

We now know that this Markov chain has a unique asymptotic distribution, we can find it by using our three main methods:

1. Invariant distribution method
2. Limiting distribution method
3. Asymptotic distribution method

2.2 Invariant distribution method

In order to find the invariant distribution of (X_t) with the Invariant distribution method I had to first find the eigenvectors of the transition matrix, then I had to find the left eigenvector with eigenvalue of 1. In this particular case, the transition matrix had some row with probability values of 1/3, since Python can't compute eigenvalues with the precise value of 1/3, the eigenvector that we are interested in had an eigenvalue of approximately 1 (Figure 1).

2.1. Invariant distribution method

```
In [89]: values, left = scipy.linalg.eig(P, right = False, left = True)

         for i in range(len(values)):
             print("Left eigenvector for eigenvalue {}:".format(values[i]))
             print(left[:,i])
             print()
```

```
Left eigenvector for eigenvalue (-0.7886751345963761+0j):
[ 0.66402333 -0.28064928 -0.52369869  0.38337405 -0.24304941]

Left eigenvector for eigenvalue (-0.21132486540177595+0j):
[-0.41768125  0.65882964  0.08826643  0.24114838 -0.5705632 ]

Left eigenvector for eigenvalue (-1.0001875870404818e-12+0j):
[ 2.12171828e-12 -7.07106781e-01 -2.12264390e-24 -1.41447885e-12
  7.07106781e-01]

Left eigenvector for eigenvalue (0.9999999999991535+0j):
[0.48038446 0.16012815 0.48038446 0.64051262 0.32025631]

Left eigenvector for eigenvalue 0j:
[ 0.         -0.70710678  0.          0.          0.70710678]
```

Figure 1: Invariant distribution method

So, in this list of left eigenvectors we are interested in the one that has eigenvalue approximately equal to 1, so we take it and normalize it in order to obtain the correct invariant distribution (figure 2):

The normalized probabilities are:

```
In [5]: for i in range(len(values)):
            if (values[i] < 1.001) & (values[i] > 0.999):
                print(left[:,i]/sum(left[:,i]))

[0.23076923 0.07692308 0.23076923 0.30769231 0.15384615]
```

Figure 2: Normalized invariant distribution

$$\pi = [0.23076923, 0.07692308, 0.23076923, 0.30769231, 0.15384615]$$

2.3 Limiting distribution method

In order to obtain the limiting distribution we have to apply the 1-step transition probabilities matrix P to the initial distribution vector for a big enough number of times. We want to start from the first node, so our initial distribution vector has the form: $v = \{1, 0, 0, 0, 0\}$, I then applied the transition matrix P by using a while cycle for 1,000,000 times (figure 3):

2.2. Limiting distribution method

```
In [6]: x_0 = [1,0,0,0,0]
        x_1 = x_0
        i = 0
        while i < 1000000:
            x_1 =np.matmul(x_1,P)
            i = i+1

In [7]: x_1

Out[7]: array([0.23076904, 0.07692301, 0.23076904, 0.30769205, 0.15384602])
```

Figure 3: Limiting distribution starting from the first node

$$\pi = [0.23076904, 0.07692301, 0.23076904, 0.30769205, 0.15384602]$$

We can also try to start from a different node, in order to see if the limiting distribution changes, let's start from node 5 ($v = \{0, 0, 0, 0, 1\}$) (figure 4):

$$\pi = [0.23076904, 0.07692301, 0.23076904, 0.30769205, 0.15384602]$$

So, the limiting distribution of this Markov chain doesn't change if we start from different nodes.

```
In [8]: x_0 = [0,0,0,0,1]
        x_1 = x_0
        i = 0
        while i < 1000000:
            x_1 =np.matmul(x_1,P)
            i = i+1

In [9]: x_1

Out[9]: array([0.23076904, 0.07692301, 0.23076904, 0.30769205, 0.15384602])
```

Figure 4: Limiting distribution starting from the fifth node

2.4 Asymptotic distribution method

For the asymptotic distribution method we just have to let the process run for $N = 1000$ times or longer and compute the asymptotic frequencies by calculating the proportion of time spent on each node, so I created a list called X_t that represents the node we pass by during our process, I decided to start from the first node, so the initial list is $X_t = [1]$. After that I initialized a for cycle that gives us the next node (the next element of the X_t list is drawn randomly from the nodes according to the 1 - step transition probabilities of the transition matrix P). I decided to reiterate the cycle for 10,000 times, after that I did a new for cycle that calculates the proportion of times we passed on each node (figure 5):

2.3. Asymptotic distribution method ¶

```
In [153]: chi = [1,2,3,4,5]
          X_t = []
          X_t.append(1)
          for N in range(1,10000):
              prob = P[X_t[N-1]-1,:].tolist()
              x_t = random.choices(chi,weights = prob,k=1)
              X_t.append(x_t[0])

In [154]: prob_fin = []
          i = 0
          for i in chi:
              x_i = X_t.count(i)/len(X_t)
              prob_fin.append(x_i)
              i = i +1
          prob_fin

Out[154]: [0.2373, 0.0817, 0.2372, 0.2991, 0.1447]
```

Figure 5: Asymptotic distribution

$$\pi = [0.2373, 0.0817, 0.2372, 0.2991, 0.1447]$$

Now we can also plot the cumulative averages for each node. I did this by crating five new lists composed of only zeroes and ones (one list for every node,there is a 1 when the n-th node appears on the X_t list, zero when it doesn't)(figure 6).

```
In [157]: cumu1 = []
          cumu2 = []
          cumu3 = []
          cumu4 = []
          cumu5 = []
          for i in X_t:
              if i == 1:
                  cumu1.append(1)
              else:
                  cumu1.append(0)
              if i == 2:
                  cumu2.append(1)
              else:
                  cumu2.append(0)
              if i == 3:
                  cumu3.append(1)
              else:
                  cumu3.append(0)
              if i == 4:
                  cumu4.append(1)
              else:
                  cumu4.append(0)
              if i == 5:
                  cumu5.append(1)
              else:
                  cumu5.append(0)
```

Figure 6: Cumulative lists

After that I plotted the lists:

1. Plot for node 1: figure 13
2. Plot for node 2: figure 14
3. Plot for node 3: figure 15
4. Plot for node 4: figure 16
5. Plot for node 5: figure 17

2.5 Report the final ranking obtained with the three methods

After using the three methods, it's clear that node 4 is the most "visited one" as we increase the number of steps, then there are nodes 1 and 3, and finally node 2: figure 7.

3 Part III

In this section we are working with a much bigger dataset, in the previous part the nodes were only 5, now we have 50.

3.1 Obtain the PageRank transition matrix associated to the directed network described by Net

We are interested in determining the PageRank transition matrix for this network, so, after importing the file in my python environment, I decided to sum

3. Ranking

```
In [123]: states = list(range(1,len(chi)+1))
          df2 = pd.DataFrame({'Probabilities':prob_fin}, states)
          df2.head()
```

Out[123]:

	Probabilities
1	0.2295
2	0.0786
3	0.2295
4	0.3077
5	0.1547

```
In [125]: df2.sort_values('Probabilities', ascending = False)
```

Out[125]:

	Probabilities
4	0.3077
1	0.2295
3	0.2295
5	0.1547
2	0.0786

Figure 7: Rank of the 5 nodes

the values of every row and divide the row by that value using a for cycle. By doing this I obtained the transition matrix (after computing the dataset I summed up the values of the matrix by row, in order to check if every row summed up to 1)(figure 8).

```
In [105]: i = 0
          P_net = []
          for i in range(0,len(som)):
              y = net.iloc[i]/som[i]
              P_net.append(y)
              i = i +1
          P_net = pd.DataFrame(P_net)
          P_net.head()
```

Out[105]:

	1	2	3	4	5	6	7	8	9	10	...	41	42	43	44	45	46	47	48	49	50
0	0.0	0.0	0.0	0.25	0.000000	0.0	0.0	0.0	0.0	0.25	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.00	0.000000	0.0	0.0	0.0	0.0	0.00	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.00	0.500000	0.0	0.0	0.0	0.0	0.00	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.00	0.000000	0.0	0.0	0.0	0.0	0.00	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.00	0.333333	0.0	0.0	0.0	0.0	0.00	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5 rows × 50 columns

```
In [106]: P_net.sum(axis = 1).head()
```

Out[106]:

```
0    1.0
1    1.0
2    1.0
3    1.0
4    1.0
dtype: float64
```

Figure 8: Transition matrix for Net

In order to better visualize the values of the transition matrix, we can use a sparse matrix visualization to see where the transition probabilities are different from 0: figure 9)

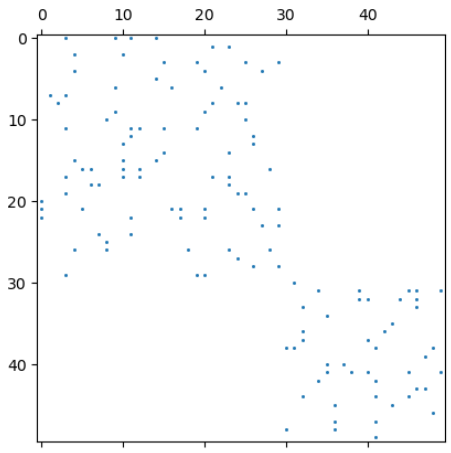


Figure 9: Sparse matrix for Net

3.2 Calculate the asymptotic frequencies of X_t first by setting $X_0 = 1$ and then by setting $X_0 = 50$

In this part's instructions we said that, since the network is fairly big, we can only use the asymptotic distribution method.

In order to calculate the asymptotic probabilities I followed the same steps used in 2.4 (figure 10).

2. Calculate the asymptotic frequencies of first by setting $X_0 = 1$ and then by setting $X_0 = 50$

In [107]:
```
chi = list(range(0,len(P_net)))
X_t = []
X_t.append(0)
for N in range(1,10000):
    prob = P_net.iloc[X_t[N-1]]
    x_t = random.choices(chi,weights = prob,k=1)
    X_t.append(x_t[0])
prob = []
i = 0
for i in chi:
    prob_i = X_t.count(i)/len(X_t)
    prob.append(prob_i)
    i = i +1
prob
```

Figure 10: Asymptotic frequencies for Net

The list that contains the probabilities has 50 entries, so instead of writing it down, i plotted it for the two cases (starting from the first node and starting from the 50th node):

1. Plot starting from the first node (figure 18)
2. Plot starting from the 50th node (figure 19)

The two limiting probabilities are different if we start from different nodes, so the convergence theorem does not apply to this case.

3.3 Implement the PageRank algorithm for a suitable choice of α. Find and report the ranking of the 50 nodes/pages

For a reducible (X_t) with transition matrix P and state space $\{1, 2, \ldots, n\}$ a modified PageRank algorithm can be proposed. For the same state space the modified PageRank transition matrix P^* is defined:

$$P^*_{ij} = \alpha P_{ij} + (1 - \alpha)\frac{1}{n} \tag{3}$$

for every $i, j \in \chi$ and for $\alpha \in (0, 1)$, tipically $\alpha = 0.95$. Basically with probability α the web server moves according to P, with probability $1 - \alpha$ they pick a page uniformly at random. By implementing this new algorithm with $\alpha = 0.95$ to the transition matrix we get this new matrix: figure 11.

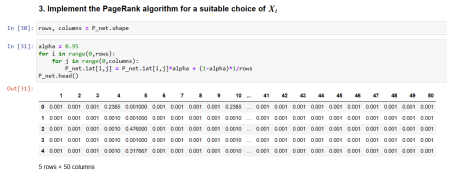


Figure 11: New transition Matrix

We can now use this new matrix to calculate the asymptotic probabilities, I followed the same steps as before. I decided to calculate them starting from the first and the 50th node, like we did before. I then plotted the results: figure 20 and figure 21. The two plots are basically the same, unlike before. So it doesn't matter where we start from, the asymptotic distribution with this new matrix seems to be unique. We can now rank the different nodes by asymptotic probability, the first 5 nodes in terms of probabilities are the ones in figure 12.

Figure 12: Ranking of the first 5 nodes

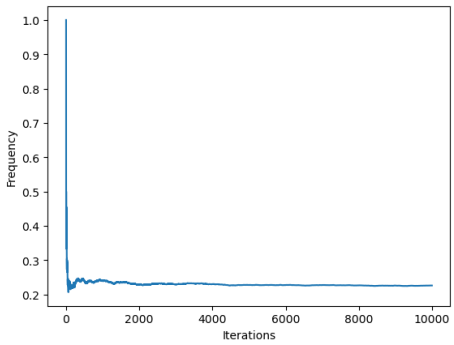


Figure 13: Plot for node 1

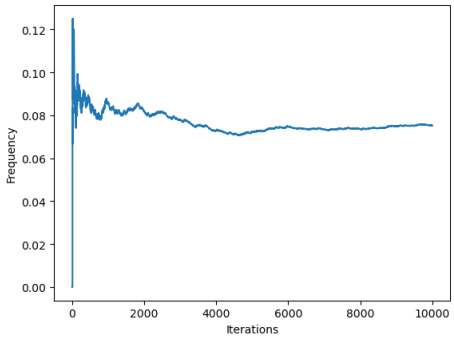


Figure 14: Plot for node 2

3.4 At least for a few states inspect empirically whether the estimated asymptotic probabilities have converged

The last thing we can do is inspect empirically whether the estimated asymptotic probabilities have converged for some states that we can choose randomly. I decided to inspect the states 10, 25 and 40.

1. Plot for state 10: figure 22
2. Plot for state 25: figure 23
3. Plot for state 40: figure 24

From the plots we see that the probabilities converge.

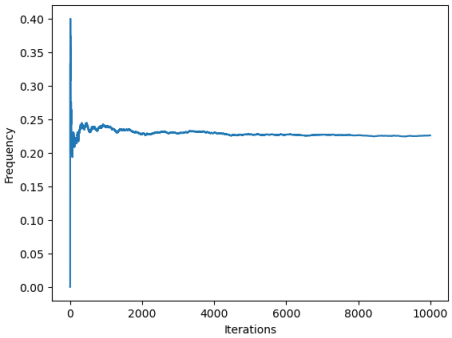


Figure 15: Plot for node 3

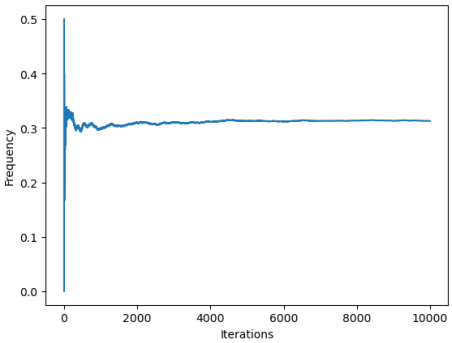


Figure 16: Plot for node 4

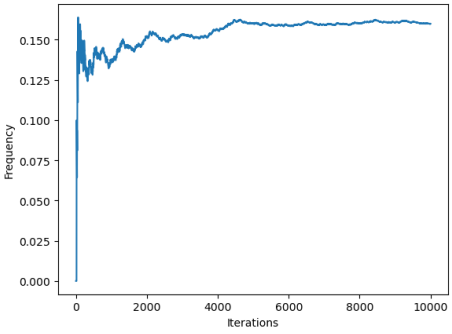


Figure 17: Plot for node 5

```
In [68]: chi = list(range(0,len(P_net)))
         X_t = []
         X_t.append(0)
         for N in range(1,10000):
             prob = P_net.iloc[X_t[N-1]]
             x_t = random.choices(chi,weights = prob,k=1)
             X_t.append(x_t[0])
         prob = []
         i = 0
         for i in chi:
             prob_i = X_t.count(i)/len(X_t)
             prob.append(prob_i)
             i = i +1

In [69]: plt.plot(prob)
         plt.xlabel('Nodes')
         plt.ylabel('Probabilities')

Out[69]: Text(0, 0.5, 'Probabilities')
```

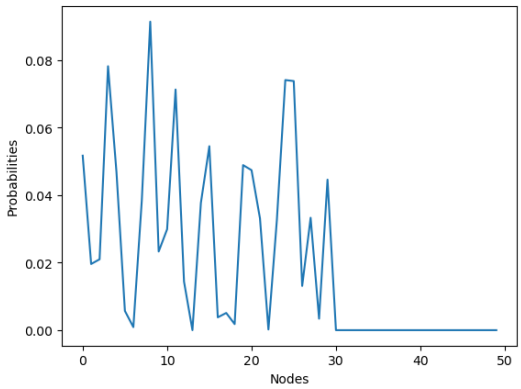


Figure 18: Asymptotic frequencies starting from the first node

```
In [70]: chi = list(range(0,len(P_net)))
         X_t = []
         X_t.append(49)
         for N in range(1,10000):
             prob = P_net.iloc[X_t[N-1]]
             x_t = random.choices(chi,weights = prob,k=1)
             X_t.append(x_t[0])
         prob = []
         i = 0
         for i in chi:
             prob_i = X_t.count(i)/len(X_t)
             prob.append(prob_i)
             i = i +1

In [71]: plt.plot(prob)
         plt.xlabel('Nodes')
         plt.ylabel('Probabilities')

Out[71]: Text(0, 0.5, 'Probabilities')
```

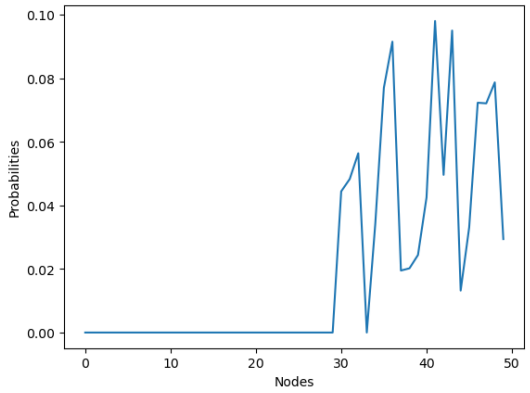


Figure 19: Asymptotic frequencies starting from the 50th node

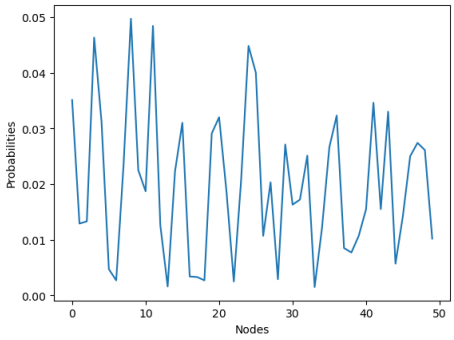


Figure 20: Asymptotic probabilities with the new matrix starting from node 1

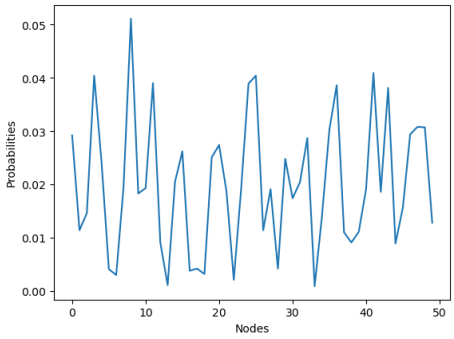


Figure 21: Asymptotic probabilities with the new matrix starting from node 50

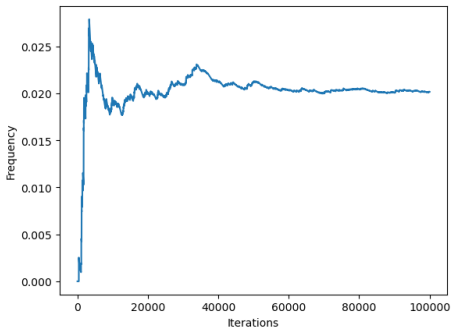


Figure 22: Asymptotic frequency of the state number 10

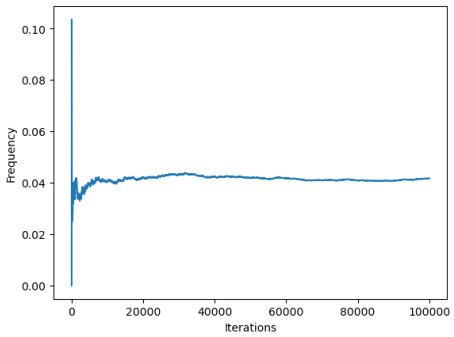


Figure 23: Asymptotic frequency of the state number 25

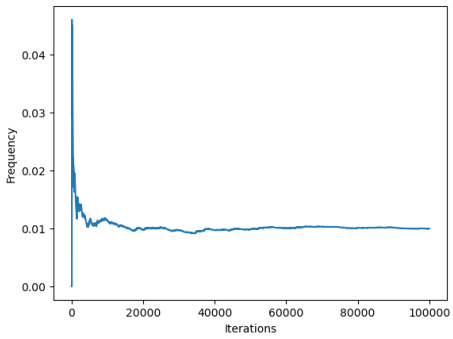


Figure 24: Asymptotic frequency of the state number 40