Printed M out:

```
M
[[0.  0.  0.6 0.  0.  0.  0.1 0.  0.  0. ]
 [0.  0.  0.  0.  0.3 0.  0.  0.  0.  0. ]
 [0.  0.2 0.  0.  0.3 0.1 0.  0.  0.1 0. ]
 [0.  0.  0.  0.  0.  0.1 0.  0.5 0.4 0. ]
 [0.  0.  0.  0.  0.  0.4 0.  0.  0.  0. ]
 [0.6 0.  0.  0.  0.  0.  0.5 0.  0.  0. ]
 [0.  0.  0.  0.  0.  0.  0.4 0.  0.1 0.5]
 [0.4 0.5 0.  0.2 0.3 0.  0.  0.  0.  0. ]
 [0.  0.  0.  0.4 0.  0.  0.  0.5 0.  0. ]
 [0.  0.3 0.4 0.4 0.1 0.4 0.  0.  0.4 0.5]]
size
10
```

A:

Eigen Analysis
```
[-0.12035   +0.j         -0.0974722 -0.0964816j  -0.0974722 +0.0964816j
 -0.05636152+0.31121603j -0.05636152-0.31121603j  0.33275266+0.20436279j
  0.33275266-0.20436279j -0.3779016 +0.j         -0.38715739+0.01149969j
 -0.38715739-0.01149969j]
norm 1.0
sum (-0.9147284788397201+0j)
```

State Propagation
```
[0.04832012 0.0175804  0.04136312 0.06662768 0.05860132 0.14650329
 0.23502244 0.05902418 0.05616316 0.2707943 ]
norm 0.41096793674714016
sum 1.0000000000000016
```

Matrix Power
```
[0.04832012 0.0175804  0.04136312 0.06662768 0.05860132 0.14650329
 0.23502244 0.05902418 0.05616316 0.2707943 ]
norm 0.41096793674714016
sum 1.0000000000000016
```

Random Walk
```
[0.02 0.02 0.03 0.08 0.06 0.15 0.25 0.09 0.03 0.27]
norm 0.4226109321823088
sum 1.0
```

B:

Run matrix power and state propagation techniques with q0 as a distribution. For what value of t is required to get as close to the true answer as the older initial state?

Intro:
Assume the markov chain converges at t=2048, when $q^* = M \times q^*$.
Here is the L1 norm: 0.41096793674713994
I will increment t, and record the delta norm value from the L1 norm above. If the difference is less than 0.0001, that is when it "gets as close to the true answer as the older initial state".

**************State Propagation: **************
converged case when t == 2048
[0.04832012 0.0175804 0.04136312 0.06662768 0.05860132 0.14650329
0.23502244 0.05902418 0.05616316 0.2707943]
norm: 0.41096793674713994

iterate from t = 1 to 100
case when t == 1 , delta from converged result: 0.17493434590763998
case when t == 2 , delta from converged result: 0.11351543995478222
case when t == 3 , delta from converged result: 0.05867270887367954
case when t == 4 , delta from converged result: 0.031501864835394536
case when t == 5 , delta from converged result: 0.017631111365310943
case when t == 6 , delta from converged result: 0.011501514966775376
case when t == 7 , delta from converged result: 0.007024912382931624
case when t == 8 , delta from converged result: 0.003792159405752857
case when t == 9 , delta from converged result: 0.0019485119459941543
case when t == 10 , delta from converged result: 0.0011257986797107047
case when t == 11 , delta from converged result: 0.000747147340373778
case when t == 12 , delta from converged result: 0.00045823964798244433
case when t == 13 , delta from converged result: 0.00024472348751521594
case when t == 14 , delta from converged result: 0.0001249565563763112
case when t == 15 , delta from converged result: 7.331760200230261e-05
case when t == 16 , delta from converged result: 4.8121418777239945e-05

~~~~~~

**************Matrix Power: **************
~~~~~~

converged case when t == 2048
[0.04832012 0.0175804 0.04136312 0.06662768 0.05860132 0.14650329
 0.23502244 0.05902418 0.05616316 0.2707943]
norm: 0.41096793674716386

iterate from t = 1 to 100
case when t == 1 , delta from converged result: 0.17493434590765075
case when t == 2 , delta from converged result: 0.11351543995478824
case when t == 3 , delta from converged result: 0.05867270887367867
case when t == 4 , delta from converged result: 0.031501864835386126
case when t == 5 , delta from converged result: 0.017631111365297596
case when t == 6 , delta from converged result: 0.01150151496676444
case when t == 7 , delta from converged result: 0.007024912382926099
case when t == 8 , delta from converged result: 0.0037921594057542885
case when t == 9 , delta from converged result: 0.0019485119460033926
case when t == 10 , delta from converged result: 0.0011257986797240814
case when t == 11 , delta from converged result: 0.0007471473403844537
case when t == 12 , delta from converged result: 0.0004582396479874991
case when t == 13 , delta from converged result: 0.0002447234875132012
case when t == 14 , delta from converged result: 0.00012495655636648542
case when t == 15 , delta from converged result: 7.331760198900188e-05
case when t == 16 , delta from converged result: 4.812141876691809e-05

Conclusion:

t = 14 for state propagation and matrix power

C.

Technique	**PRO**	**CON**
State Propagation	Produces accurate results in reasonable efficiency. Useful for medium sized data.	Slower than matrix power as it is linearly multiplying the matrix with q_.
Matrix Power	Decent when the probability transition matrix does not change often, as it precomputes the matrix power by t before multiplying it by q_0. It is excellent for cases when you need to compute something about all states, as you can just pre-power the matrix and multiply it by all q_ It is more efficient than state propagation when uses dynamic programming to pre-power the matrix	Not helpful if your probability transition matrix is changing all the time, as the precomputed result would quickly render useless.
Eigen Analysis	Very straightforward to implement, useful for small data when there is no need to multiply the Probability Transition Matrix for 2000 times. For example, compute your relationship with people in a ten people group chat.	Impossible for large scale pieces of data as Eigenvector Decomposition is slow for large matrices. Say, describe the relationship between all webpages on this planet.
Random Walk	The most efficient solution. Great for large datasets. Because it does not require any matrix arithmatics, you can feed an adjacency list representation of the graph to this technique. (adjacency list is excellent for sparse graphs like website links)	Unstable result. Each run can return different results, requiring large t to produce accurate results.

D.

It is ergodic.

An ergodic markov chain cannot be:
1. cyclic
Part B shows that q* slowly converges because $q^* = M \times q^*$.
2. has absorbing and transient states
Part B shows with distribution q_0, all states are accessible.
3. disconnected nodes
The p* computed from all four techniques shows all values in q* are above 0, meaning the intial state shown in q_0 has a path to all states.

E.

	0	1	2	3	4	5	6	7	8	9
0	0	0	0.6	0	0	0	0.1	0	0	0
1	0	0	0	0	0.3	0	0	0	0	0
2	0	0.2	0	0	0.3	0.1	0	0	0.1	0
3	0	0	0	0	0	0.1	0	0.5	0.4	0
4	0	0	0	0	0	0.4	0	0	0	0
5	0.6	0	0	0	0	0	0.5	0	0	0
6	0	0	0	0	0	0	0.4	0	0.1	0.5
7	0.4	0.5	0	0.2	0.3	0	0	0	0	0
8	0	0	0	0.4	0	0	0	0.5	0	0
9	0	0.3	0.4	0.4	0.1	0.4	0	0	0.4	0.5

What nodes can be reached from node 5 in one step, and with what probabilities?
node 5 can reach node 2 at Pr=0.1, node 3 at Pr=0.1, node 4 at Pr=0.4, node 9 at Pr=0.4