Table of Contents


```
% Final Project --- Luis Kligman
% Markov Chains --- Part 1
clear all;
close all;
clc;

% Define the transition matrix
P = [.10, .05, 0.0, .25, .33;
     .20, .35, 0.0, .25, .32;
     .30, .10, .35, .25, 0.0;
     .15, .40, .55, .25, 0.0;
     .25, .10, .10, 0.0, .35];

% Ensure that each column sums to 1 (Markov property)
col_sums = sum(P,1);
disp('Column sum of P (should all be 1):');
disp(col_sums);
```

```
Column sum of P (should all be 1):
     1     1     1     1     1
```

(a.) What is the probability that an individual at site 2 (the initial state vector is (0, 1, 0, 0, 0)T) will move to site 5 in three steps?

Start with a single individual at site 2. This can be represented as a state vector: x0 = [0; 1; 0; 0; 0]; This means 0% at site1, 100% at site2, ...

```
x0_single = [0; 1; 0; 0; 0];

% 3-Step transition matrix is P^3 (matrix multiplication, Not element-wise)
% The distribution after 3 steps is:
x3_single = P^3 * x0_single;

disp('Probability of going from site 2 to site 5 in 3 steps:');
disp(x3_single(5));
```

```
Probability of going from site 2 to site 5 in 3 steps:
    0.1162
```

(b.) Suppose 100 individuals are uniformly distributed at the five sites initially. How will the individuals be distributed after four steps?

Uniformly distributed means the individuals are split equally between sites

```
x0_pop = [20; 20; 20; 20; 20];

% After 4 steps x4 = P^4 * x0
x4_pop = P^4 * x0_pop;

disp('Populastion distribution after 4 steps (counts at each site):');
disp(x4_pop)

% Check that total population is still 100 (Markov chain converses total)
disp('Total population after 4 steps (should be 100):');
disp(sum(x4_pop));
```

```
Populastion distribution after 4 steps (counts at each site):
   14.1466
   22.1620
   21.2874
   30.0624
   12.3417

Total population after 4 steps (should be 100):
   100
```

(c.) Find the steady state vector of P.

The steady state vector pi satisfies: P * pi = pi sum(pi) = 1

```
% In Markov-chain content: pi is an eigenvector of P with eigenvalue 1
% in MATLAB, we can use eig(p) and then normalize

% Compute eigenvalues and eigenvectors of P
[V, D] = eig(P);

% The diagonal of D contains eigenvalues
eigenvalues = diag(D);

% Find index of eigenvalue closest to 1
[~, idx] = min(abs(eigenvalues - 1));

% Extract corresponding eigenvector
steady_vec = V(:, idx);
```

```
% Eigenvectors are determined up to scaling; we normalize to sum to 1
steady_vec = steady_vec / sum(steady_vec);

disp('Steady state vector (long-run probabilities at each site):');
disp(steady_vec);
```

```
Steady state vector (long-run probabilities at each site):
    0.1408
    0.2195
    0.2154
    0.3032
    0.1211
```

Published with MATLAB® R2025a