LAB 4: Eigenvalues and Eigenvectors

(Math 250: Sections C3)

Name: Aryan Malhotra

Last 4 digits of RUID: 8256

Date: 10/20/2024

Learning Outcomes

- The geometric meaning of eigenvalues and eigenvectors of a matrix
- Determination of eigenvalues and eigenvectors using the characteristic polynomial of a matrix
- Use of eigenvectors to transform a matrix to diagonal form.
- Steady-state eigenvector for a transition matrix
- Applications of eigenvalues and eigenvectors to study Markov chains.

Table of Contents

Name: Aryan Malhotra....1
Last 4 digits of RUID: 8256....1
Date: 10/20/2024....1
Learning Outcomes....1
Random Seed....1
Question 1. Graphic Demonstration of Eigenvectors and Eigenvalues (Total Points - 5)....2
Question 2. Characteristic Polynomial (Total Points - 4)....4
Question 3. Eigenvectors and Diagonalization (Total Points - 5)....5
Question 4. Steady-State Eigenvector for a Transition Matrix (Total Points - 4)....9
Question 5. Markov Chains (Total Points - 5)....12

Random Seed

Initialize the random number generator by typing the following code, where abcd are the last four digits of your RUID

```
rng('default');
rng(abcd, 'twister');
```

```
% Enter your code here
rng('default');
rng(8256, 'twister');
```

This will ensure that you generate your own particular random vectors and matrices.

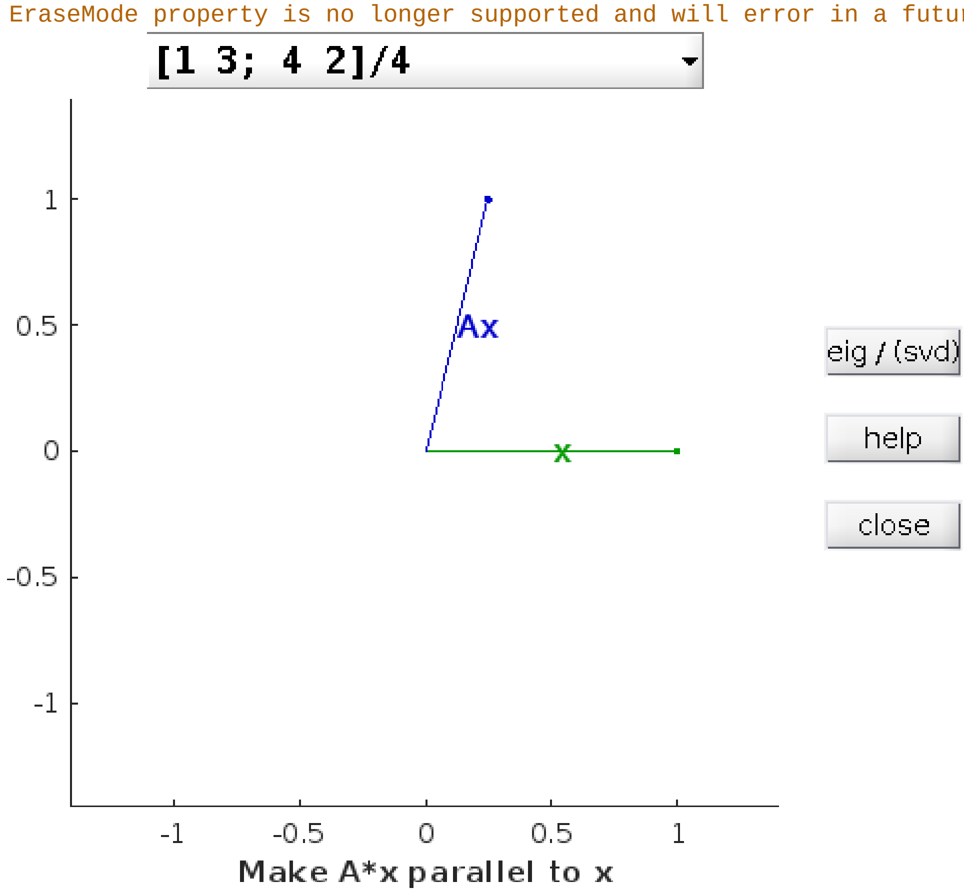
The lab report that you hand in must be your own work. The following problems use randomly generated matrices and vectors, so the matrices and vectors in your lab report will not be the same as those of other students doing the lab. Sharing of lab report files is not allowed in this course.

Question 1. Graphic Demonstration of Eigenvectors and Eigenvalues (Total Points - 5)

Type ***eigshow*** at the Matlab prompt. A graphics window should open. Underneath the graph the statement "*Make A*x parallel to x"* should appear (if it does not, then click on the eig button to get this statement) (DO NOT CLOSE IT).

```
% Enter your code here
eigshow
```

```
Warning: The EraseMode property is no longer supported and will error in a future release.
Warning: The EraseMode property is no longer supported and will error in a future release.
Warning: The EraseMode property is no longer supported and will error in a future release.
Warning: The EraseMode property is no longer supported and will error in a future release.
Warning: The EraseMode property is no longer supported and will error in a future release.
Warning: The EraseMode property is no longer supported and will error in a future release.
```



(a) Click on the pull-down bar above the graph and select the matrix [1 3; 4 2]/4. Move the cursor onto the vector x, and make x go around a full circle. The transformed vector Ax then moves around an ellipse. Search for the *special lines* through zero that contain both Ax and x. When x lies on such a line, it is an eigenvector of the matrix A (the word *eigen* means *special* in German).

For any x lying on these *special lines*, $Ax = \lambda x$, where λ is an *eigenvalue* of A. Since x is a unit vector, the length of Ax is $|\lambda|$. If Ax points in the same direction as x, then $\lambda > 0$. If Ax points in the opposite direction to x, then $\lambda < 0$.

From your graphical experimentation answer the following questions (no algebraic calculations needed):

Caution: Be careful in counting eigenvalues; if x is an eigenvector with eigenvalue λ, then $-x$ is also an eigenvector with the **same** eigenvalue λ.

(i) ***Exercise (Points - 1):*** How many different positive eigenvalues does A have? (This occurs when the Ax arrow points in the same direction as the x arrow.)

Answer: 2

(ii) ***Exercise (Points - 1):*** How many different negative eigenvalues does A have? (This occurs when the Ax arrow points in the opposite direction to the x arrow.)

Answer: 2

(iii) ***Exercise (Points - 1):*** What are the (approximate) numerical values of the eigenvalues? (Estimate these values using the relative lengths of the x arrow and the Ax arrow.)

Answer: It seems like the eigenvalues seems to be around 1.25 and -0.5 (5 of the lengths of difference between Ax and x fits for one eigenvalue and for the other, the Ax vector seems about half of that of x but in opposite direction)

(b) Click on pull-down matrix selection bar again and select [3 1; -2 4]/4. Move x around the circle with the cursor and observe what happens, as in part (a). Use your graphical experimentation to answer the following questions (no algebraic calculations needed):

(i) ***Exercise (Points - 1):*** Are there any lines through zero that contain both Ax and x?

Answer: No, x and Ax are never along the same line passing through the origin

(ii) ***Exercise (Points - 1):*** Does A have any real eigenvectors or eigenvalues? Explain.

Answer: Since Ax is never on the same line as x, it is never a linear combination/scaled version of x and hence, there are no eigen values.

Question 2. Characteristic Polynomial (Total Points - 4)

At the Matlab prompt type ***A = [1 3; 4 2]/4*** (this is the matrix in part (a) of Question 1). The eigenvalues of A are the roots of the characteristic polynomial of A.

```
% Enter your code here
A = [1 3; 4 2]/4
```

```
A = 2×2
    0.2500    0.7500
    1.0000    0.5000
```

(a) Use Matlab to calculate its characteristic polynomial p(t) by

```
syms t; I = eye(2); p = det(A - t*I)
```

```
% Enter your code here
syms t; I = eye(2); p = det(A - t*I)
```

p =

$$t^2-\frac{3\,t}{4}-\frac{5}{8}$$

(i) ***Exercise (Points - 1):*** Verify by hand calculation that the constant term in the polynomial p(t) is det(A).

Answer: 0.5*0.25 - 1.0*0.75 = -5/8

(b) Use the Matlab command ***solve(p)*** to get the roots of p(t) (the eigenvalues of A).

```
% Enter your code here
solve(p)
```

ans =

$$\begin{pmatrix} -\frac{1}{2} \\ \frac{5}{4} \end{pmatrix}$$

(i) ***Exercise (Points - 1):*** Compare these values with your graphical estimates for the eigenvalues from Question 1(a).

Answer: My estimates were actually correct!

(c) Now at the Matlab prompt type ***A = [3 1 ; -2 4]/4*** (the matrix in part (b) of Question 1).

```
% Enter your code here
A = [3 1 ; -2 4 ]/4
```

```
A = 2×2
    0.7500    0.2500
   -0.5000    1.0000
```

(i) ***Exercise (Points - 1):*** Calculate the characteristic polynomial p(t) of A as in part (a) and find its roots as in part (b).

```
% Enter your code here
syms s; H = eye(2); q = det(A - s*H)
```

q =

$$s^2 - \frac{7\,s}{4} + \frac{7}{8}$$

```
solve(q)
```

ans =

$$\begin{pmatrix} \frac{7}{8} - \frac{\sqrt{7}\,\mathrm{i}}{8} \\ \frac{7}{8} + \frac{\sqrt{7}\,\mathrm{i}}{8} \end{pmatrix}$$

(ii) ***Exercise (Points - 1):*** Are the eigenvalues of A real? How does this explain what you observed Question 1(b).

Answer: These eigenvalues are real. This corresponds well to what we observe since there are no vectors in the real space that get scaled as eigenvalues. But there do seem to be complex eigenvectors.

Question 3. Eigenvectors and Diagonalization (Total Points - 5)

(a) Generate a random 3 × 3 integer matrix and test whether its eigenvalues are all real by the commands

```
A = rmat(3,3), z = eig(A) - real(eig(A))
```

```
% Enter your code here
A = rmat(3,3), z = eig(A) - real(eig(A))
```

```
A = 3×3
     1     2     5
     4     2     0
     4     1     2
z = 3×1
     0
     0
     0
```

If any entry in the vector z is not zero, then the eigenvalues of A are not all real. In this case repeat these commands until you get an A for which z has all zeros. If you generated any intermediate matrices with complex eigenvalues **do not delete them**. They should be part of the lab report.

Now calculate the characteristic polynomial p(t) of your matrix A by

```
syms t; I = eye(3); p = det(A - t*I)
```

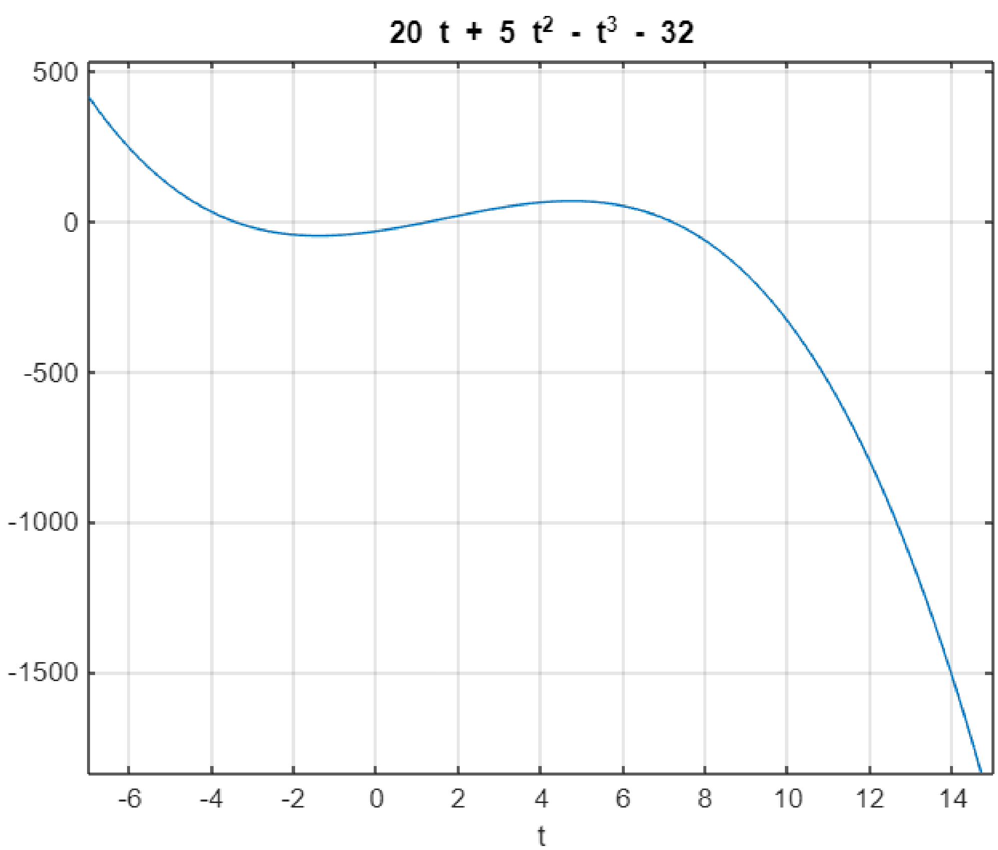
```
% Enter your code here
syms t; I = eye(3); p = det(A - t*I)
```

p = $-t^3+5\,t^2+20\,t-32$

(i) ***Exercise (Points - 1):*** Plot the characteristic polynomial of A in a graphics window by the following commands and Save the graph with a range that shows all three real roots.

```
figure; ezplot(p, [-10, 10]), grid
```

```
% Enter your code here
figure; ezplot(p, [-7, 15]), grid
```



IMPORTANT: Your plot may or may not show all three real roots of p(t). If it doesn't, you must adjust the horizontal range of the plot (change ***[−10, 10]*** as needed) to show that p(t) has three real roots (**zoom as necessary**, using the magnifying glass button on the top of the graph) .

(ii) ***Exercise (Points - 1):*** Use the graph to obtain approximate values for the three real roots of p(t).

Answer: -3.5, 1.5, 7.25

(b) Use the following Matlab command to generate a matrix P and a diagonal matrix D:

```
[P D] = eig(A)
```

```
% Enter your code here
[P D] = eig(A)
```

```
P = 3×3
   -0.6385   -0.7264    0.1618
   -0.4943    0.5323   -0.9130
   -0.5899    0.4348    0.3746
D = 3×3
    7.1675         0         0
         0   -3.4584         0
         0         0    1.2909
```

(i) ***Exercise (Points - 1):*** Compare the diagonal entries of D with your graphical estimates for the eigenvalues of A in part (a).

Answer: They are comparable to each other.

Use Matlab to define the columns of P:

```
p1 = P(:,1), p2 = P(:,2), p3 = P(:,3)
```

```
% Enter your code here
p1 = P(:,1), p2 = P(:,2), p3 = P(:,3)
```

```
p1 = 3×1
   -0.6385
   -0.4943
   -0.5899
p2 = 3×1
   -0.7264
    0.5323
    0.4348
p3 = 3×1
    0.1618
   -0.9130
    0.3746
```

Calculate

```
A*p1 - D(1,1)*p1, A*p2 - D(2,2)*p2, A*p3 - D(3,3)*p3
```

```
% Enter your code here
A*p1 - D(1,1)*p1, A*p2 - D(2,2)*p2, A*p3 - D(3,3)*p3
```

```
ans = 3×1
```

10^{-14} ×

```
   -0.1776
    0.2220
    0.1776
ans = 3×1
10^-14 ×
    0.0888
    0.3553
    0.3553
ans = 3×1
10^-14 ×
    0.1693
   -0.1110
   -0.0167
```

(ii) *Exercise (Points - 1):* What does this calculation tell you about the eigenvalues and eigenvectors of A? (Using the definiation of eigenvalues and eigenvectors)

Answer: p1 is an eigenvector of A with respect to the eigenvalue D(1,1); p2 is an eigenvector of A with respect to the eigenvalue D(2,2); p3 is an eigenvector of A with respect to the eigenvalue D(3,3).

(c) *Exercise (Points - 1):* Let A, P, D be as in part (b). Verify by Matlab that A = P*D*inv(P).

```
% Enter your code here
A, P*D*inv(P)
```

```
A = 3×3
     1     2     5
     4     2     0
     4     1     2
ans = 3×3
    1.0000    2.0000    5.0000
    4.0000    2.0000    0.0000
    4.0000    1.0000    2.0000
```

Use this formula for A to express A^5 and A^{10} symbolically in terms of P , P^{-1} , D^5 and D^{10} . Verify your answer to this question numerically using Matlab.

Answer: $A^5 = P\ D^5\ P^{-1}$; $A^{10} = P\ D^{10}\ P^{-1}$.

```
% Enter your code here
A^5, P*D^5*inv(P), A^10, P*D^10*inv(P)
```

```
ans = 3×3
        7797        4703        8095
        6476        3556        5860
        7648        4256        7072
ans = 3×3
10^3 ×
    7.7970    4.7030    8.0950
    6.4760    3.5560    5.8600
    7.6480    4.2560    7.0720
ans = 3×3
   153160397    87845479   147924135
   118339308    68041924   114703300
```

```
   141279968    81201312   136863904
ans = 3×3
10^8 ×
     1.5316    0.8785    1.4792
     1.1834    0.6804    1.1470
     1.4128    0.8120    1.3686
```

Question 4. Steady-State Eigenvector for a Transition Matrix (Total Points - 4)

The square matrix A is called a *transition matrix* if its entries are nonnegative and the sum of the entries in each column is one.

(a) Generate a random 2×2 transition matrix A by

```
A = eye(2); B = rand(2);
A(:,1) = B(:,1)/sum(B(:,1)); A(:,2) = B(:,2)/sum(B(:,2))
```

```
% Enter your code here
A = eye(2); B = rand(2);
A(:,1) = B(:,1)/sum(B(:,1)); A(:,2) = B(:,2)/sum(B(:,2))
```

```
A = 2×2
     0.7704    0.7410
     0.2296    0.2590
```

Calculate ***[1 1]*A.***

```
% Enter your code here
[1 1]*A
```

```
ans = 1×2
      1     1
```

(i) ***Exercise (Points - 1):*** Explain (by a hand calculation) why the answer to this calculation shows that A is a transition matrix.

Answer: [1(0.7704)+1(0.2296) 1(0.7410)+1(0.2590)] = [1 1]

(b) Transition matrices such as A with all entries positive are called *regular*. For a regular transition matrix, 1 is the largest eigenvalue, and the corresponding eigenspace is one-dimensional.

Use the T-codes ***nulbasis*** to calculate a normalized eigenvector for the matrix A you generated in part (a):

```
u = nulbasis(A - eye(2)); v = u/sum(u)
```

```
% Enter your code here
u = nulbasis(A - eye(2)); v = u/sum(u)
```

```
v = 2×1
     0.7635
     0.2365
```

The vector v should have components that are positive and sum to 1.

(i) ***Exercise (Points - 1):*** Verify by Matlab that v is an eigenvector for A with eigenvalue 1 (called the *steady-state vector* for A).

```
% Enter your code here
A*v, v
```

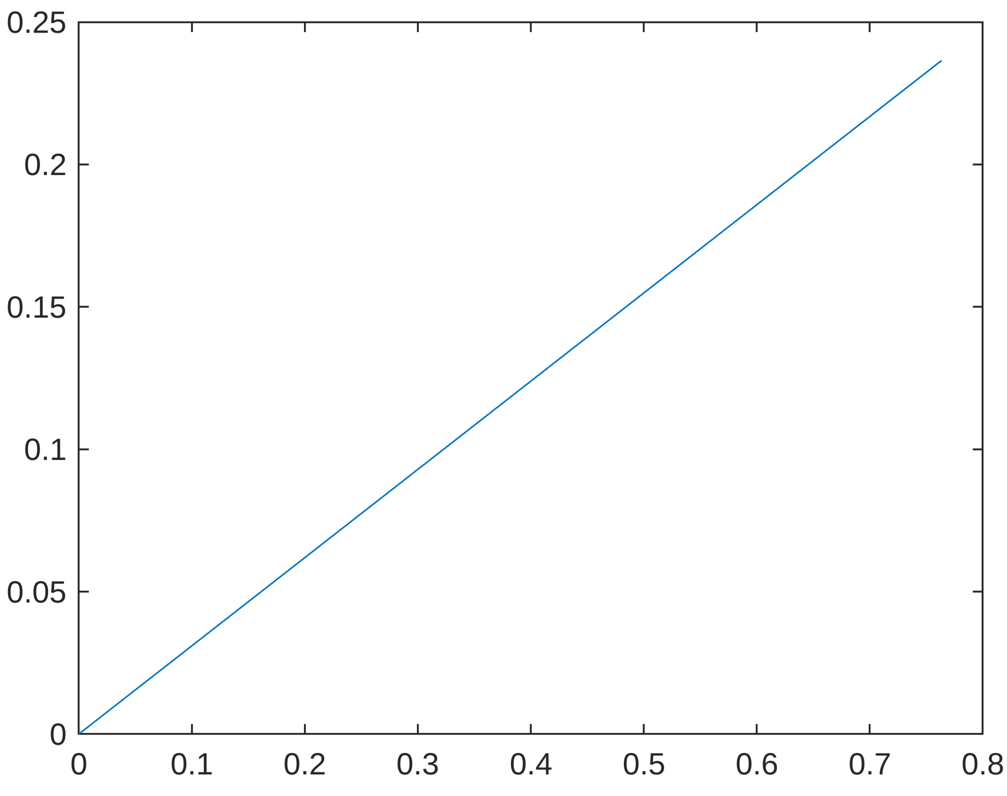
```
ans = 2×1
    0.7635
    0.2365
v = 2×1
    0.7635
    0.2365
```

Since Av = v, v is an eigenvector with eigenvalue 1.

Plot this vector (as a solid line) by

```
figure; plot([0,v(1)], [0, v(2)]), hold on
```

```
% Enter your code here
figure; plot([0,v(1)], [0, v(2)]), hold on
```



Leave the graphic window open for the next part.

(c) A general result about *regular transition matrices* asserts that if p is any initial choice of a probability vector in $\mathbb{R}^2$, then the sequence of vectors $A^k p$ converges to the *steady-state vector* v as k $\rightarrow \infty$. To demonstrate this graphically for your matrix A, generate a random initial probability vector

```
w = rand(2,1); p = w/sum(w)
```

```
% Enter your code here
w = rand(2,1); p = w/sum(w)
```

```
p = 2×1
    0.6588
    0.3412
```

Graph the vector Ap (as a dotted line) in the same window from part (b):

```
p = A*p, plot([0,p(1)], [0, p(2)], ':'), hold on
```

```
% Enter your code here
p = A*p, plot([0,p(1)], [0, p(2)], ':'), hold on
```

```
p = 2×1
    0.7604
    0.2396
```

(i) ***Exercise (Points - 1):*** To plot the sequence of vectors A^2p, A^3p, A^4p, . . . in the same window, just repeat the last command.

```
% Enter your code here
p = A*A*p, plot([0,p(1)], [0, p(2)], ':'), hold on
```

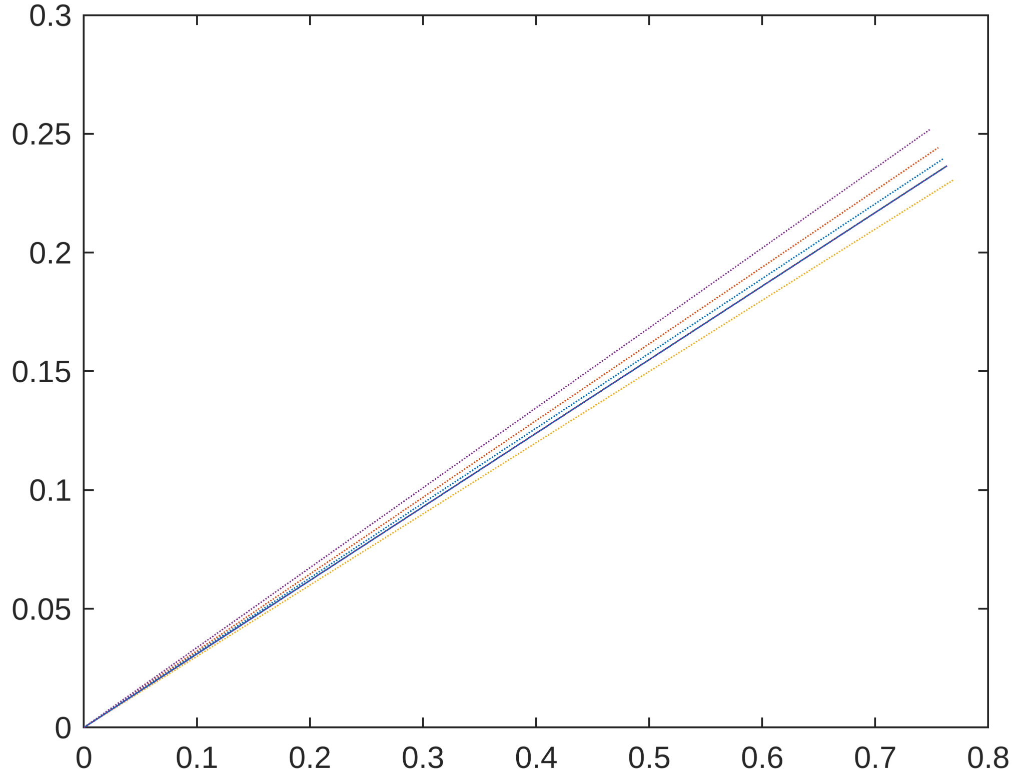
```
p = 2×1
    0.7635
    0.2365
```

```
p = A*A*A*p, plot([0,p(1)], [0, p(2)], ':'), hold on
```

```
p = 2×1
    0.7635
    0.2365
```

```
p = A*A*A*A*p, plot([0,p(1)], [0, p(2)], ':'), hold on
```

```
p = 2×1
    0.7635
    0.2365
```



(ii) ***Exercise (Points - 1):*** Do this as many times as needed until the vector p has converged numerically (to three decimal places in each component) to the *steady-state vector* v that you plotted in part (b). Check by Matlab.

Answer: As we can see, p converges to the same value as v

```
% Enter your code here
p,v
```

```
p = 2×1
    0.7635
    0.2365
v = 2×1
    0.7635
    0.2365
```

Question 5. Markov Chains (Total Points - 5)

Read the section of Markov Chains (Section 5.9) and look at Example 1 and Practice Problem 1 and its solution.

Answer the following questions:

Suppose that a particular region with a constant population is divided into three areas: the city, suburbs, and country. Enumerate the states as 1 = city, 2 = suburbs, 3 = country.

The probability that a person living in the city moves to the suburbs (in one year) is .10, and the probability that someone moves to the country is .50. The probability is .20 that a person living in the suburbs moves to the city and is .10 for a move to the country. The probability is .20 that a person living in the country moves to the city and is .20 for a move to the suburbs.

Suppose initially that 50% of the people live in the city, 30% live in the suburbs, and 20% live in the country.

(a) ***Exercise (Points - 1):*** Determine the *migration matrix* (transition matrix) A for the above three states and enter it into your Matlab workspace:

```
% Enter your code here
A = transpose([0.40, 0.10, 0.50;
     0.20, 0.70, 0.10;
     0.20, 0.20, 0.60]);
```

Let u be the row vector [1 1 1]. Verify that each column of A sums to 1 by calculating that u * A = u:

```
% Enter your code here
u = [1, 1, 1];
column_sums = u * A
```

```
column_sums = 1×3
     1     1     1
```

(b)

(i) ***Exercise (Points - 1):*** Determine the initial probability vector p which represents the percentage of people living in the city, suburbs, and country. Verify that the entries of p sum to 1 by calculating that u * p = 1.

```
% Enter your code here
p = [0.50; 0.30; 0.20]; entry_sum = u * p
```

```
entry_sum =
1
```

(ii) ***Exercise (Points - 1):*** Now use powers of the matrix A and the vector p to find the percentage of people living in the city, suburbs, and country after 1, 2, 3, 5, and 8 years.

```
% Enter your code here
A*p
```

```
ans = 3×1
    0.3000
    0.3000
    0.4000
```

```
A^2*p
```

```
ans = 3×1
    0.2600
    0.3200
    0.4200
```

```
A^3*p
```

```
ans = 3×1
    0.2520
    0.3340
    0.4140
```

```
A^5*p
```

```
ans = 3×1
    0.2501
    0.3459
    0.4041
```

```
A^8*p
```

```
ans = 3×1
    0.2500
    0.3495
    0.4005
```

Answer: Initially, the distribution for city, suburb and country is

50-30-20

After 1 year: 30-30-40

After 2 years: 26-32-42

After 3 years: 25.2-33.4-41.4

After 5 years: 25.01-34.95-40.41

After 8 years: 25-34.95-40.05

(c) The *steady-state probability vector* v is an eigenvector for A with eigenvalue 1.

(i) ***Exercise (Points - 1):*** Use Matlab to find v by the same method as Question 4(b).

```
% Enter your code here
u = nulbasis(A - eye(3)); v = u/sum(u)
```

```
v = 3×1
    0.2500
    0.3500
    0.4000
```

(ii) ***Exercise (Points - 1):*** What is the relation between the vector v and the population distribution vector in part (b) after 8 years?

Answer: population after 8 years tends to approach the values in vector v. (I suspect, the population stabalizes at the values in vector v, described as the "steady-state" in part c) of the previous question)

Helper Function (Do Not Edit)

```
function A = rmat(m,n)
A = fix(10*rand(m,n));
end
```