A(i, j)	the (i,j)-entry of A.
A(i, :)	the i^{th} row of A.
A(:, j)	the j^{th} column of A.
A + B	matrix addition.
A – B	matrix subtraction.
t * A	scalar multiplication if t is a real number, i.e. tA.
A * B	matrix multiplication.
A^n	raising a square matrix A to a positive integral power n, i.e. A^n.
A′	transpose of A, i.e. A^T
inv(A)	inverse of an invertible square matrix A, i.e. A^{-1}.
zeros(n)	the $n \times n$ zero matrix, i.e. $\mathbf{0}_{n\times n}$.
zeros(m, n)	the $m \times n$ zero matrix, i.e. $\mathbf{0}_{m\times n}$.
eye(n)	the $n \times n$ identity matrix, i.e. $\boldsymbol{I}_n$.
rref(A)	the reduced row echelon form of A.
det(A)	the determinant of A, i.e. det(A).

format short	scaled fixed point format with 5 digits (default setting).
format long	scaled fixed point format with 15 digits.
format shorte	floating point format with 5 digits.
format longe	floating point format with 15 digits.
format rat	approximate fractions.

ezplot('E(x, y)')	plot points satisfying the equation $\mathsf{E}(x,y)$ for $-2\pi < x < 2\pi$.
ezplot('E(x, y)', [xmin, xmax])	plot points satisfying the equation $\mathsf{E}(x,y)$ for $\mathsf{xmin} < x < \mathsf{xmax}$.
ezplot('f(x)')	plot the function $y = f(x)$ for $-2\pi < x < 2\pi$.
ezplot('f(x)', [xmin, xmax])	plot the function $y = f(x)$ for $\mathsf{xmin} < x < \mathsf{xmax}$.
ezplot('x(t)', 'y(t)')	plot points $(x(t),\ y(t))$ for $-2\pi < t < 2\pi$.
ezplot('x(t)', 'y(t)', [tmin, tmax])	plot points $(x(t),\ y(t))$ for $\mathsf{tmin} < t < \mathsf{tmax}$.

The command "hold on" tells MATLAB to keep all the old lines when a new ezplot command is executed

The command "clf" clears all the graphical figures

ezsurf('f(x, y)')
draw the graph of the function $z = f(x,y)$ for $-2\pi < x < 2\pi$ and $-2\pi < y < 2\pi$.

ezsurf('f(x, y)', [xmin, xmax], [ymin, ymax])
draw the graph of the function $z = f(x,y)$ for $\mathsf{xmin} < x < \mathsf{xmax}$ and $\mathsf{ymin} < y < \mathsf{ymax}$.

ezsurf('x(s, t)', 'y(s, t), 'z(s, t)')
draw the surface with points $(x(s,t),\ y(s,t),\ z(s,t))$ for $-2\pi < s < 2\pi$ and $-2\pi < t < 2\pi$.

ezsurf('x(s, t)', 'y(s, t), 'z(s, t)', [smin, smax], [tmin, tmax])
draw the surface with points $(x(s,t),\ y(s,t),\ z(s,t))$ for $\mathsf{smin} < s < \mathsf{smax}$ and $\mathsf{tmin} < t < \mathsf{tmax}$.

Remark:
- In general, the row operation R2 ± kR1 creates a new second line which when k increases, moves away from the original second line of the system (∗) but moves towards the first line.
- However, the intersection point, i.e. the solution of the system, remains the same.

Remark:
- In general, the row operation R2 ±kR1 creates a new second plane which when k increases, moves away from the original second plane of the system (∗∗) but moves towards the first plane.
However, the intersection line, i.e. the solutions of the system, remains the same.

linear combination : rref([A u]) inconsistent – not linear combi

linear independence : $Ax = 0$, w = zeros(m,1)

① rref([A w]) → non-pivot col exists ⇒ free param ⇒ infinite sol'ns

② rref(A)

redundant vectors : non-pivot col of rref(A) ⇒ entry on non-pivot col = linear combi of pivot col

diag([a, b, c])" to enter a 3 × 3 diagonal matrix with diagonal entries a, b, c

C(2,1) = 0.4: to change the (2,1)-entry of C.

Span : rref([C D]) & rref([D C]) both consistent ⇒ span

D ⊆ C

C ⊆ D ⇒ consistent ⇒ subset

d_1 d_2 d_3 …

are all non-pivot

⇒ ✓ subset

Find Basis : rref(A)

basis = pivot cols of A

entries of non-pivot col = linear combi of pivot cols

Dimensions : dim(V) = no. of vectors in basis

Check Basis : (if T is a basis for V)

① linearly indep

② V ⊆ span(T)

③ span(T) ⊆ V

Coordinate vector : (T is a basis) rref([T h])

e.g. $h = h_1 + 2h_2 + 3h_3$

$(h)_T = (1,2,3)$

We shall find the reduced row-echelon form (RREF) of $(\boldsymbol{A} \mid \boldsymbol{b})$:

```
>> rref([A b])
```

$$\text{ans} = \begin{array}{cccccc} x_1 & x_2 & x_3 & x_4 & x_5 & b \\ 1 & 0 & -2 & 1 & 0 & 1 \\ 0 & 1 & 1 & -1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ & & s & t & & \end{array}$$

$x_1 - 2x_3 + x_4 = 1$ ⇒ $x_1 = 1 + 2s - t$

$x_2 + x_3 - x_4 = 2$ ⇒ $x_2 = 2 - s + t$

$x_5 = 1$

inv(A) * b = solution

Only applicable for square matrices which are invertible

(Here `[A b]` is the matrix obtained by combining `A` and `b` to obtain the augmented matrix. The separator `|` should be omitted in the MATLAB command.)

One sees that the 1st, the 2nd and the 5th columns are pivot columns.

Set $x_3 = s$ and $x_4 = t$ to be arbitrary parameters, and solve other variables:

$$x_1 = 2s - t + 1, \quad x_2 = -s + t + 2, \quad x_5 = 1.$$

Indeed, we can verify that $\boldsymbol{x} = \begin{pmatrix} 2s-t+1 \\ -s+t+2 \\ s \\ t \\ 1 \end{pmatrix}$ is a solution.

We first declare that s and t are parameters.

```
>> syms s t
```

Then define

```
>> x = [2*s-t+1; -s+t+2; s; t; 1]
x =  2*s - t + 1
     t - s + 2
     s
     t
```

Note that $\boldsymbol{x}$ is a solution if and only if $\boldsymbol{Ax} = \boldsymbol{b}$. So we evaluate $\boldsymbol{Ax}$ and compare it with $\boldsymbol{b}$:

```
>> A * x
ans =  -2
       -2
       3
```

2.1.1. *Use Row Vectors.*

View each $v_1, \ldots, v_k$ as a row vector. Then the nonzero rows of any row-echelon form of the matrix $\begin{pmatrix} v_1 \\ \vdots \\ v_k \end{pmatrix}$ form a basis for V.

For example, let $S = \{v_1, v_2, v_3, v_4, v_5\}$, where

$v_1 = (1,1,1,1,1)$, $v_2 = (1,-1,1,-1,1)$, $v_3 = (2,0,2,0,2)$, $v_4 = (1,-2,4,-8,16)$, $v_5 = (0,4,-6,16,30)$.

(i) Input $v_1, \ldots, v_5$ into MATLAB as row vectors:

```
>> v1 = [1 1 1 1 1];   v2 = [1 -1 1 -1 1];   v3 = [2 0 2 0 2];
>> v4 = [1 -2 4 -8 16];   v5 = [0 4 -6 16 -30];
```

(ii) Find the reduced row-echelon form of the matrix $\begin{pmatrix} v_1 \\ \vdots \\ v_5 \end{pmatrix}$.

```
>> rref([v1; v2; v3; v4; v5])
ans =  1   0   0   2  -4
       0   1   0   1   0
       0   0   1  -2   5
       0   0   0   0   0
       0   0   0   0   0
```

Its nonzero rows $\{(1,0,0,2,-4), (0,1,0,1,0), (0,0,1,-2,5)\}$ form a basis for $V = \text{span}(S)$. Note that the vectors in the basis are not necessarily in S.

2.1.2. *Use Column Vectors.* View each $v_1, \ldots, v_k$ as column vectors. Find the pivot columns of any row-echelon form of the matrix $\begin{pmatrix} v_1 & \cdots & v_k \end{pmatrix}$. Then the corresponding vectors in S form a basis S' for V. Note that $S' \subseteq S$.

(i) Input $v_1, \ldots, v_5$ into MATLAB as column vectors. In Section 2.1.1, $v_1, \ldots, v_5$ are defined as vectors. Their transposes $v_1^T, \ldots, v_5^T$ (`v1'`, …, `v5'`) are the required column vectors.

(ii) Find the reduced row-echelon form of the matrix $\begin{pmatrix} v_1 & \cdots & v_5 \end{pmatrix}$.

```
>> rref([v1' v2' v3' v4' v5'])
ans =  1   0   1   0   1
       0   1   1   0   1
       0   0   0   1  -2
       0   0   0   0   0
       0   0   0   0   0
```

Its 1st, 2nd and 4th columns are pivot. Then $\{v_1, v_2, v_4\} = \{(1,1,1,1,1), (1,-1,1,-1,1), (1,-2,4,-8,16)\}$ form a basis for V. Note that every vector in this basis is taken from S.

Let $V = \text{span}(S)$. Then S is a basis for V if and only if S is linearly independent.

(i) Input v_1, v_2, v_3, v_4 as column vectors in MATLAB.

(ii) Define the matrix $\boldsymbol{G} = \begin{pmatrix} g_1 & g_2 & g_3 & g_4 \end{pmatrix}$:

```
>> G = [g1 g2 g3 g4]
G =  1   1  -1   0
     1  -1  -3   1
     1   2   0   1
     1   3   1  -1
     1   0  -2  -1
```

(iii) Find the reduced row-echelon form of $\boldsymbol{G}$:

```
>> rref(G)
ans =  1   0  -2   0
       0   1   1   0
       0   0   0   1
       0   0   0   0
       0   0   0   0
```

redundant vector

The 1st, 2nd and 4th columns are pivot, while the 3rd column is non-pivot. We conclude that

(i) $\{g_1, g_2, g_4\}$ is a basis for V.

(ii) $\dim(V) = 3$.

Moreover, by observing the entries of the 3rd column, $g_3 = -2g_1 + g_2$.

not {(1,0,0,0,0) … }

adjoint(A)

isAlways(cond1 == cond2)

↳ return matrix consistently of ones if true

Solving Linear system with MATLAB

Ax = 0

>> rref([A 0])

>> null(A) (give basis for nullspace of A)

Ax = b (consistent)

>> rref([A b])

>> A \ b (give a particular solution)

>> linsolve(A, b) (give a particular solution)

inv(A) * b

Rank : rank(A)

Transpose : v'

the dimension of the row space of $\boldsymbol{A}$ = the number of nonzero rows of $\boldsymbol{R}$
= the number of pivot points of $\boldsymbol{R}$
= the number of pivot columns of $\boldsymbol{R}$
= the dimension of the column space of $\boldsymbol{A}$.

Nullspace : null(A,'r')

rational

w/o r is also ok

Assume that the variables are x_1, x_2, x_3, x_4, x_5. Since the 2^{nd} and the 5^{th} columns of the reduced row-echelon form are non-pivot, set $x_2 = s$ and $x_5 = t$ as arbitrary parameters. We get

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} -4s \\ s \\ 0 \\ -4t \\ t \end{pmatrix} = s\begin{pmatrix} -4 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} + t\begin{pmatrix} 0 \\ 0 \\ 0 \\ -4 \\ 1 \end{pmatrix}.$$

Then the nullspace of $\boldsymbol{A}$ has a basis

$$\left\{\begin{pmatrix} -4 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \\ -4 \\ 1 \end{pmatrix}\right\}.$$

MATLAB can provide a basis for the nullspace of $\boldsymbol{A}$ using the same way by `null(A,'r')`.*

```
>> null(A, 'r')
ans =  -4    0
        1    0
        0    0
        0   -4
        0    1
```

The columns of the answer form a basis for the nullspace of $\boldsymbol{A}$.

Dot product : ① dot(u,v) → regardless if either one is row/column vector

② If both $\boldsymbol{u}$ and $\boldsymbol{v}$ are defined as row vectors, then $\boldsymbol{u} \cdot \boldsymbol{v} = \boldsymbol{u}\boldsymbol{v}^{\mathrm{T}}$. = u * v'
If both $\boldsymbol{u}$ and $\boldsymbol{v}$ are defined as column vectors, then $\boldsymbol{u} \cdot \boldsymbol{v} = \boldsymbol{u}^{\mathrm{T}}\boldsymbol{v}$. = u' * v

Norm(length): ① norm(u)
② sqrt(dot(u,u))

Note that the norm of a vector is usually irrational, and the output is in floating-point. We can use `sym` to define a vector as *symbolic* object, and use `norm` to get the exact value of the norm. For example,

```
>> u = sym([1 2 3 4 5])
u =  [1, 2, 3, 4, 5]
>> norm(u)
ans =  55^(1/2)
```

Check orthogonal set : C*C' ⇒ diagonal matrix

5.1.1. *Orthogonal Set.* A set of vectors $S = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$ in $\mathbb{R}^n$ is said to be an *orthogonal* set if

$$\boldsymbol{v}_i \cdot \boldsymbol{v}_j = 0 \quad \text{for all } i \neq j.$$

View each vector as a row vector and consider $\boldsymbol{A} = \begin{pmatrix} \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{v}_k \end{pmatrix}$. Then $\boldsymbol{A}^{\mathrm{T}} = \begin{pmatrix} \boldsymbol{v}_1^{\mathrm{T}} & \cdots & \boldsymbol{v}_k^{\mathrm{T}} \end{pmatrix}$ and

$$\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}} = \begin{pmatrix} \boldsymbol{v}_1\boldsymbol{v}_1^{\mathrm{T}} & \cdots & \boldsymbol{v}_1\boldsymbol{v}_k^{\mathrm{T}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{v}_k\boldsymbol{v}_1^{\mathrm{T}} & \cdots & \boldsymbol{v}_k\boldsymbol{v}_k^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{v}_1 \cdot \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_1 \cdot \boldsymbol{v}_k \\ \vdots & \ddots & \vdots \\ \boldsymbol{v}_k \cdot \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_k \cdot \boldsymbol{v}_k \end{pmatrix}.$$

Hence, $S = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$ is orthogonal if and only if $\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}$ is a diagonal matrix.

5.1.2. *Orthonormal Set.* A set of vectors $S = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$ in $\mathbb{R}^n$ is said to be an *orthonormal* set if

$$\boldsymbol{v}_i \cdot \boldsymbol{v}_j = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases}$$

View each vector as a row vector and consider $\boldsymbol{A} = \begin{pmatrix} \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{v}_k \end{pmatrix}$. Then $S = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$ is orthonormal if and only if $\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}} = \boldsymbol{I}_k$, the identity matrix of order k.

Check orthonormal set : C*C' ⇒ I_k (identity matrix w/ order k)

Deriving the orthonormal basis: orth(E)

5.2. **Orthogonal and Orthonormal Basis.** If $V = \text{span}(S)$, and S is an orthonormal set, then S is linearly independent, and S is called an *orthonormal basis* for V.

In MATLAB, `orth` can be used to get an *orthonormal basis* for the column space of a matrix.

Suppose $V = \text{span}(S)$, where $S = \{(1,1,1,1), (1,1,0,0), (0,1,1,0)\}$.

(i) Define matrix $\boldsymbol{E} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$ whose columns are the vectors in the spanning set.

```
>> E = [1 1 0; 1 1 1; 1 0 1; 1 0 0];
```

or

```
>> E = [1 1 1 1; 1 1 0 0; 0 1 1 0]';
```

(ii) Use `orth` to get an orthonormal basis for the column space of $\boldsymbol{E}$, i.e., for V.

```
>> orth(E)     v1         v2          v3
ans =   -0.4835     0.7071     -0.1273
        -0.6635     0.0000      0.5565
        -0.4835    -0.7071     -0.1273
        -0.3035    -0.0000     -0.8111
```

no. of columns of the resulting matrix = dim(V)

orthonormal basis = {v1, v2, v3}

⇒ To get the exact form : sym

(i) Define matrix $\boldsymbol{E}$ whose columns are the vectors in S as an symbolic object.

```
>> E = sym([1 1 0; 1 1 1; 1 0 1; 1 0 0]);
E =  [1, 1, 0]
     [1, 1, 1]
     [1, 0, 1]
     [1, 0, 0]
```

(ii) Use `orth` to get an orthonormal basis for the column space of $\boldsymbol{E}$, i.e., for V.

```
>> orth(E)
ans =  [1/2, 1/2, -1/2]
       [1/2, 1/2, 1/2]
       [1/2, -1/2, 1/2]
       [1/2, -1/2, -1/2]
```

The columns of the resulting matrix form an orthonormal basis

$$\{(\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{2}), (\tfrac{1}{2}, \tfrac{1}{2}, -\tfrac{1}{2}, -\tfrac{1}{2}), (-\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{2}, -\tfrac{1}{2})\}$$

for the vector space V.

```
>> orth(E, 'skipnormalization')
ans =  [1, 1/2, -1/2]
       [1, 1/2, 1/2]
       [1, -1/2, 1/2]
       [1, -1/2, -1/2]
```

characteristic polynomial: charpoly(matrix)

solving the equation (=0) : solve(ans)

eigenvalues : ① solve(charpoly(A, lambda))
② roots(charpoly(A))
③ eig(A)

eigenspace (nullspace): null(λI-A, 'r')

1.1. **Characteristic Polynomial.** The *characteristic polynomial* of $\boldsymbol{A}$ is the polynomial given by

$$p_{\boldsymbol{A}}(\lambda) = \det(\lambda \boldsymbol{I}_n - \boldsymbol{A}). \tag{1}$$

Note that the degree of the characteristic polynomial is n and its leading coefficient is 1.

MATLAB provides several ways to find the characteristic polynomial of $\boldsymbol{A}$.

(a) `charpoly(A)` gives a *vector* with $n+1$ components which are the coefficients of the characteristic polynomial in *descending* order. For example,

```
>> charpoly(A)
ans =  1    4    -10    -40    45    108    -108
```

The output means that the characteristic polynomial of $\boldsymbol{A}$ is (in variable λ)

$$p_{\boldsymbol{A}}(\lambda) = \lambda^6 + 4\lambda^5 - 10\lambda^4 - 40\lambda^3 + 45\lambda^2 + 108\lambda - 108.$$

(b) `charpoly(A, lambda)` gives the characteristic polynomial in variable `lambda`. In order to use λ as the variable for the characteristic polynomial of $\boldsymbol{A}$, we shall use `syms` to declare it as a symbolic object:

```
>> syms lambda;
```

Then type

```
>> charpoly(A, lambda)
ans =  lambda^6 + 4*lambda^5 - 10*lambda^4 - 40 * lambda^3 + 45*lambda^2
+ 108*lambda - 108
```

(c) We can also use the definition (1) to find the characteristic polynomial. Recall that `eye(n)` generates the identity matrix of order n. In this example, $\boldsymbol{A}$ has order 6. So we use

```
>> syms lambda;
>> det(lambda*eye(6) - A)
ans =  (lambda - 1) * (lambda - 2)^2 * (lambda + 3)^3
```

Hence, the characteristic polynomial of $\boldsymbol{A}$ is

$$p_{\boldsymbol{A}}(\lambda) = (\lambda-1)(\lambda-2)^2(\lambda+3)^3.$$

```
>> det(lambda*eye(6) - A); Or charpoly(A, lambda);
>> solve(ans)
ans =  -3
       -3
       -3
       1
       2
       2
```

`roots` computes numerically the roots of the polynomial whose coefficients are the components of the input vector in ascending order.

In particular, if the characteristic polynomial is generated as a vector using the method (c) in Section 1.1, then `roots` can be used to compute the eigenvalues:

```
>> roots(charpoly(A))
ans =  -3.0000 + 0.0000i
       -3.0000 + 0.0000i
       -3.0000 + 0.0000i
       2.0000 + 0.0000i
       2.0000 - 0.0000i
       1.0000 + 0.0000i
```

$\lambda = 2$:

```
>> 2*eye(6) - A;
>> null(ans, 'r')
ans =  -1    -1
        0     0
        0     0
        1     0
        0     0
        0     1
```

Then the eigenspace E_2 of $\boldsymbol{A}$ associated to eigenvalue 2 has a basis

$$\{(-1,0,0,1,0,0), (-1,0,0,0,0,1)\},$$

```
>> V1 = null(-3*eye(6) - A, 'r');
>> V2 = null(2*eye(6) - A, 'r');
>> V3 = null(1*eye(6) - A, 'r');
```

Then

```
>> P = [V1 V2 V3]
```

Recurrence Relation

Example. A sequence (b_n) is defined by

$$b_0 = 0, \quad b_1 = 1, \quad \text{and} \quad b_{n+1} = 3b_{n-1} + 2b_n, \quad n \geq 1.$$

Let $\boldsymbol{y}_n = \begin{pmatrix} b_n \\ b_{n+1} \end{pmatrix}$ and $\boldsymbol{B} = \begin{pmatrix} 0 & 1 \\ 3 & 2 \end{pmatrix}$. Then $\boldsymbol{y}_0 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ and $\boldsymbol{y}_n = \boldsymbol{B}^n \boldsymbol{y}_0$.

(i) Define $\boldsymbol{y}_0$ and input $\boldsymbol{B}$ in MATLAB as symbolic object:

```
>> y0 = [0; 1];
>> B = sym([0 1; 3 2]);
```

(ii) Find $\boldsymbol{P}$ and $\boldsymbol{D}$ such that $\boldsymbol{P}^{-1}\boldsymbol{B}\boldsymbol{P} = \boldsymbol{D}$. In this example, $\boldsymbol{B}$ is diagonalizable.

```
>> [P D] = eig(B)  ⟶ to obtain P&D directly
P =  [-1,   1/3]
     [1,      1]
D =  [-1,     0]
     [0,      3]
```

(iii) $\boldsymbol{y}_n = \boldsymbol{B}^n \boldsymbol{y}_0 = \boldsymbol{P}\boldsymbol{D}^n\boldsymbol{P}^{-1}\boldsymbol{y}_0$:

```
>> y(n) = P * D^n * inv(P) * y0
y(n) =  3^n/4 - (-1)^n/4
        (-1)^n/4 + (3*3^n)/4
```

(iv) By definition, b_n is the 1[st] component of $\boldsymbol{y}_n$: $b_n = \frac{1}{4}[3^n - (-1)^n]$.

Let $\boldsymbol{M} = \begin{pmatrix} 0.97 & 0.01 & 0.02 \\ 0.01 & 0.97 & 0.02 \\ 0.02 & 0.02 & 0.96 \end{pmatrix}$ and $\boldsymbol{x}_n = \begin{pmatrix} a_n \\ b_n \\ c_n \end{pmatrix}$. Then $\boldsymbol{x}_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ and $\boldsymbol{M}\boldsymbol{x}_n = \boldsymbol{x}_{n+1}$. Using the same argument as in Example 6.1.1 of the textbook,

$$\boldsymbol{x}_n = \boldsymbol{M}^n \boldsymbol{x}_0.$$

Input $\boldsymbol{M}$ and $\boldsymbol{x}_0$ in MATLAB:

```
>> M = [0.97 0.01 0.02; 0.01 0.97 0.02; 0.02 0.02 0.96];
>> x0 = [1; 0; 0];
```

We can answer the following questions:

(i) Compute the present market share of the three brands of soft drink.

This is four months after brands B and C are introduced. So we are asked for $\boldsymbol{x}_4 = \boldsymbol{M}^4\boldsymbol{x}_0$:

```
>> x4 = M^4 * x0
x4 =  0.8881
      0.0388
      0.0731
```

(ii) Compute the market shares of the three brands of soft drink one year after brand B and C are introduced.

We are asked for $\boldsymbol{x}_{12} = \boldsymbol{M}^{12}\boldsymbol{x}_0$:

```
>> x12 = M^12 * x0
x12 =  0.7190
       0.1063
       0.1747
```

(iii) Diagonalize $\boldsymbol{M}$ and get a formula for $\boldsymbol{x}_n$:

```
>> M1 = sym(M);
>> [P D] = eig(M1)
P =  [1,  -1/2,  -1]
     [1,  -1/2,   1]
     [1,     1,   0]

D =  [1,     0,       0]
     [0,  47/50,      0]
     [0,     0,   24/25]
```

Then $\boldsymbol{x}_n = \boldsymbol{M}^n\boldsymbol{x}_0 = \boldsymbol{P}\boldsymbol{D}^n\boldsymbol{P}^{-1}\boldsymbol{x}_0$:

```
>> syms n
>> x(n) = P * D^n * inv(P) * x0
```

```
x(n) =  (24/25)^n/2 + (47/50)^n/6 + 1/3
        (47/50)^n/6 - (24/25)^n/2 + 1/3
        1/3 - (47/50)^n/3
```

We conclude that

$$\boldsymbol{x}_n = \begin{pmatrix} a_n \\ b_n \\ c_n \end{pmatrix} = \begin{pmatrix} \frac{1}{2}(0.96)^n + \frac{1}{6}(0.94)^n + \frac{1}{3} \\ \frac{1}{6}(0.94)^n - \frac{1}{2}(0.96)^n + \frac{1}{3} \\ -\frac{1}{3}(0.94)^n + \frac{1}{3} \end{pmatrix}.$$

(iv) Use part (iii) to estimate the market shares in the long run if the trend continues. Will the market shares stabilize in the long run?

Using basic concepts in limits, when n is large ($n \to \infty$), 0.96^n and 0.94^n both tend to 0 ($0.96^n \to 0$ and $0.94^n \to 0$); and we have

$$\boldsymbol{x}_{\text{long run}} \approx \begin{pmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{pmatrix}.$$

Formally,

$$\lim_{n\to\infty} \boldsymbol{x}_n = \begin{pmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{pmatrix} \qquad \text{or} \qquad n \to \infty \Rightarrow \boldsymbol{x}_n \to \begin{pmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{pmatrix}.$$

In MATLAB, we may use

```
>>  limit(x(n), n, inf)
ans =  1/3
       1/3
       1/3
```

2.2.2. *Recursive Sequences.* A sequence (a_n) may be defined recursively by

$$a_0 = a, \quad a_1 = b, \quad \text{and} \quad a_{n+1} = \alpha a_{n-1} + \beta a_n, \quad n \geq 1.$$

One may use matrix to find a general formula for a_n.

Note that

$$\begin{pmatrix} a_{n+1} \\ a_{n+2} \end{pmatrix} = \begin{pmatrix} a_{n+1} \\ \alpha a_n + \beta a_{n+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ \alpha & \beta \end{pmatrix} \begin{pmatrix} a_n \\ a_{n+1} \end{pmatrix}$$

Let $\boldsymbol{x}_n = \begin{pmatrix} a_n \\ a_{n+1} \end{pmatrix}$ and $\boldsymbol{A} = \begin{pmatrix} 0 & 1 \\ \alpha & \beta \end{pmatrix}$. Then $\boldsymbol{x}_0 = \begin{pmatrix} a \\ b \end{pmatrix}$ and $\boldsymbol{x}_{n+1} = \boldsymbol{A}\boldsymbol{x}_n$. Consequently,

$$\boldsymbol{x}_n = \boldsymbol{A}^n \boldsymbol{x}_0.$$

It reduces to the problem of finding powers of $\boldsymbol{A}$.