Experiment:4
QR Factorization

1. **QR factorization using Gram–Schmidt process**
   Implement this algorithm as MATLAB function script.

Algorithm
// Input: Matrix V
// Output: Matrix Q and R.

- Define dimension of matrix(say $m \times n$)
- if rank of V is not equal to n
  Factorization does not exists
  return to main script
  end if
- Set, $v_1^{\perp} = v_1$ and $u_1 = \frac{v_1^{\perp}}{||v_1^{\perp}||}$ [comment: Note that v_k, u_k are column vectors of V,Q]
- for k from 2 to n
  Set, $v_k^{\perp} = v_k - \sum_{j=1}^{k-1} < u_j, v_k > u_j$
  $u_k = \frac{v_k^{\perp}}{||v_k^{\perp}||}$
  end for
- for i from 1 to n
  for j from i to n
  $R_{ij} =< u_i, v_j >$
  end for
  end for

2. Perform the Gram-Schmidt process to find QR factorization for the column vectors of the matrix $\begin{bmatrix} 1 & 0 & 1 \\ 7 & 7 & 8 \\ 1 & 2 & 1 \\ 7 & 7 & 6 \end{bmatrix}$.

3. Solve the following system of equation by decomposing coefficient matrix in QR,

$$2x_1 + 4x_2 + 3x_3 + 5x_4 + 6x_5 = 37$$
$$4x_1 + 8x_2 + 7x_3 + 5x_4 + 2x_5 = 74$$
$$-2x_1 - 4x_2 + 3x_3 + 4x_4 - 5x_5 = 20$$
$$x_1 + 2x_2 + 2x_3 - x_4 + 2x_5 = 26$$
$$5x_1 - 10x_2 + 4x_3 + 6x_4 + 4x_5 = 24$$

Hint: Once you get QR factorization of A, rewrite system of linear equation as,

$$AX = b$$

or,

$$QRX = b$$

or,

$$Q'QRX = Q'b$$

or,

$$RX = b1$$

where $Q'Q = I$ and $b1 = Q'b$. As R is upper triangular so apply back substitution on $RX = b1$ to get the solution of $AX = b$.

Algorithm for Back substitution

// Input: Matrix R,b1

// Output: solution x_i.

- Define dimension of matrix(say $n \times n$)
- if r_{nn} is equal to zero
  Method fails
  return to main script
  end if
- Set, $x_n = \frac{b1_n}{r_{nn}}$
- for i from n-1 to 1
  Set, $x_i = [b1_i - \sum_{j=i+1}^{n} r_{ij}x_j]/r_{ii}$
  end for

Experiment:5
Eigen values and Eigen vectors

1. **Power Method**

Like the Jacobi and Gauss-Seidel methods, the power method for approximating eigenvalues is iterative. First we assume that the matrix A has a dominant eigenvalue with corresponding dominant eigenvectors. Then we choose an initial approximation x_0 of one of the dominant eigenvectors of A. This initial approximation must be a nonzero vector in R^n

. Finally we form the sequence given by,

$$x_1 = Ax_0$$

$$x_2 = Ax_1 = A(Ax_0) = A^2x_0$$

$$x_3 = Ax_2 = A(A^2x_0) = A^3x_0$$

.

.

.

$$x_k = Ax_{k-1} = A(A^{k-1}x_0) = A^kx_0$$

For large powers of k, and by properly scaling this sequence, we will see that we obtain a good approximation of the dominant eigenvector of A.

And the corresponding eigenvalue is approximated as, if x is an eigenvector of a matrix A, then its corresponding eigenvalue is given by $\lambda = \frac{(Ax)'x}{x'x}$, this quotient is called the *Rayleigh* quotient.

Power method with Scaling: In practice it is best to "scale down" each approximation before proceeding to the next iteration. One way to accomplish this scaling is to determine the component of that has the largest absolute value and multiply the vector by the reciprocal of this component. The resulting vector will then have components whose absolute values are less than or equal to 1.

In that case, note that the scaling factors used to obtain the eigenvectors, converges to the dominant eigenvalue.

Sufficient condition for convergence of the power method is that the matrix A be diagonalizable (i.e. A has n linearly independent eigenvector).

To learn more about Power method see the link link

Algorithm

// Input: Matrix A, tolerance value tol, initial guess x0, maximum number of iterations N.
// Output: dominant eigenvalue e and eigenvector v.

- Define dimension of matrix A(say $n \times n$)
- Find the smallest integer i with $1 \leq i \leq n$ and $|x_i| = ||x0||_\infty$
- Set $x0 = x0/x_i$
- while k less or equal to N
- Set $y = Ax0$
- Find the smallest integer p with $1 \leq p \leq n$ and $|y_p| = ||y||_\infty$
- Set $\mu = y_p$
- if μ is equal to 0
  A has the eigenvalue 0, select a new vector x0
  return to main script
  end if
- Set $x1 = \frac{y}{\mu}$ and $error = ||x1 - x0||_\infty$
- if $error < tol$
  $e = \mu$ and $v = x1$
  return to main script
  end if
- Set $k = k + 1$ and $x0 = x1$
- end while
- print 'The maximum number of iterations exceeded';

2. Find the dominant eigen value and corresponding eigen vector of following matrix.

(a). $\begin{bmatrix} 1 & 2 & 0 \\ -2 & 1 & 2 \\ 1 & 3 & 1 \end{bmatrix}$

(b). $\begin{bmatrix} 5 & -2 & -\frac{1}{2} & \frac{3}{2} \\ -2 & 5 & \frac{3}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{3}{2} & 5 & -2 \\ \frac{3}{2} & -\frac{1}{2} & -2 & 5 \end{bmatrix}$, $x0 = \begin{bmatrix} 1 & 1 & 0 & -3 \end{bmatrix}$

Inverse Power Method

The Inverse Power method is a modification of the Power method that gives faster convergence. It is used to determine the eigenvalue of A that is closest to a specified number q. Suppose the matrix A has eigenvalues $\lambda_1, ..., \lambda_n$ with linearly independent eigenvectors $v^{(1)}, ..., v^{(n)}$. The eigenvalues of $(A - qI)^{-1}$, where $q \neq \lambda_i$, for i = 1, 2, . . . , n, are

$$\frac{1}{\lambda_1 - q}, \frac{1}{\lambda_2 - q}, ..., \frac{1}{\lambda_n - q}$$

with these same eigenvectors $v^{(1)}, v^{(2)}, ..., v^{(n)}$.Applying the Power method to $(A-qI)^{-1}$, we get the dominant eigenvalue and eigenvector of $(A - qI)^{-1}$ (say μ and v). Hence the eigenvalue of A closest to q is given by $\frac{1}{\mu} + q$ and the corresponding eigenvector is same as v.

3. Solve question 3(b) using Inverse Power Method by taking different q's from 0 to 10(what you see, when you take q=0).

Experiment:6
Numerical Method to Solve ODE

1. **Runge-Kutta Method of Order Four**
   Suppose we have system of ode as,

$$\frac{dx}{dt} = f_1(t, x, y) \tag{1}$$
$$\frac{dy}{dt} = f_2(t, x, y) \tag{2}$$

where f_1 and f_2 are any functions(linear or non-linear) having continuous partial derivatives up to order the two, with initial condition $x(t_0) = x_0$ and $y(t_0) = y_0$, then RK method allows us to calculate the value of x and y at $t_1 = t_0 + h$ for some step size h. Formula for fourth order RK method is,

$$k_1 = hf_1(t_0, x_0, y_0) \tag{3}$$
$$s_1 = hf_2(t_0, x_0, y_0) \tag{4}$$
$$k_2 = hf_1\left(t_0 + \frac{h}{2}, x_0 + \frac{k_1}{2}, y_0 + \frac{s_1}{2}\right) \tag{5}$$
$$s_2 = hf_2\left(t_0 + \frac{h}{2}, x_0 + \frac{k_1}{2}, y_0 + \frac{s_1}{2}\right) \tag{6}$$
$$k_3 = hf_1\left(t_0 + \frac{h}{2}, x_0 + \frac{k_2}{2}, y_0 + \frac{s_2}{2}\right) \tag{7}$$
$$s_3 = hf_2\left(t_0 + \frac{h}{2}, x_0 + \frac{k_2}{2}, y_0 + \frac{s_2}{2}\right) \tag{8}$$
$$k_4 = hf_1(t_0 + h, x_0 + k_3, y_0 + s_3) \tag{9}$$
$$s_4 = hf_2(t_0 + h, x_0 + k_3, y_0 + s_3) \tag{10}$$

then,

$$x(t_0 + h) = x_1 = x_0 + (k_1 + 2k_2 + 2k_3 + k_4)/6 \tag{11}$$
$$y(t_0 + h) = y_1 = y_0 + (s_1 + 2s_2 + 2s_3 + s_4)/6 \tag{12}$$

similarly as $x_2, y_2, x_3, y_3...$
Note:The method can be extended to solve the system of any number of first order ODE's and also be used as to solve a single first order ODE.

Algorithm

// Input: Functions f_1 and f_2, initial guess t_0, x_0, y_0 ,number of steps n, final time t_n.

// Output: Vectors x and y.

(a) Set $h = \frac{t_n - t_0}{n}$.

(b) Declare dynamic vectors $x = [x_0]$ and $y = [y_0]$.

(c) Set $k = 1$.

(d) while $k \leq n$.

(e) Set $k_1, s_1, k_2, s_2, k_3, s_3, k_4, s_4$ accordingly equation (3) to (10).

(f) Set $x_1 = x_0 + (k_1 + 2k_2 + 2k_3 + k_4)/6$ and $y_1 = y_0 + (s_1 + 2s_2 + 2s_3 + s_4)/6$.

(g) Store x_1, y_1 in x and y.

(h) Set $t_0 = t_0 + h, x_0 = x_1, y_0 = y_1, k = k + 1$

(i) end while.

Implement the above algorithm as MATLAB function script and by using it solve the following question.

2. Consider the following second order ODE,

$$Li'' + Ri' + \frac{1}{C}i = 0 \tag{13}$$

associated with initial conditions $i(0) = 0$ and $i'(0) = 1$ with $L = 1H$,$R = 0.1\Omega$.

(a) Convert the given ODE into system of first order ODE by taking $i' = v$.

(b) For which choice of C, the given system is stable.

(c) Find the solution of the above system of ODE's using above algorithm in the interval $t \in [0, 200]$ for, $n = 500, 1000$ and $C = 1$.

(d) Draw the graph of i and v against t.

(e) Solve the above system using ode45 solver and compare both solutions.

(f) Draw the phase portrait of the above system using quiver command.