Homework 2: Solving Eigenvalue

A. Rayleigh method

As one of the extensions of the Power method, the algorithm of Rayleigh method is based on the Power method. Here is the algorithm and pseudocode of the Power method.

```
Sub Power()

    Set eSheet = ThisWorkbook.Worksheets("Eigen")
    Set eValep = eSheet.Range("B1")           'Convergence Criteria
    Set eMatA = eSheet.Range("B2:K11")         'Matrix (Eigen values calculated)
    Set eVecX0 = eSheet.Range("M2:M11")        'Initial Vector
    Set eVecXk = eSheet.Range("D13:D22")       'Iteration Vector X(k)
    Set eVecXkp = eSheet.Range("F13:F22")      'Iteration Vector X(k+1)
    Set eValK = eSheet.Range("B13")            'Iteration count
    Set eValJk = eSheet.Range("B14")           'Element No. of maximum in X(k)
    Set eValRk = eSheet.Range("B15")           'Value of maximum element in X(k)
    Set eValrkp = eSheet.Range("B16")          'Value of maximum element in X(k+1)

    eValK.Value = 0                                            'Counter Initalize
    eValJk.Value = findAbsMax(eVecX0)                          'Get Elem No. of Maximun in X0
    eValrkp.Value = eVecX0(eValJk.Value, 1)                    'Get its value
    Call copyMatDivByValue(eVecX0, eVecXk, eValrkp.Value)      'Normalized X0 -> X(k)
    Do
        eValRk.Value = eValrkp.Value                           'r(k+1) -> r(k)
        Call matVecMult(eMatA, eVecXk, eVecXkp)                'A*X(k) -> X(k+1)
        eValJk.Value = findAbsMax(eVecXkp)                     'Get Elem No. of Maximun in X(k+1)
        eValrkp.Value = eVecXkp(eValJk.Value, 1).Value
        eValK.Value = eValK.Value + 1                          'Counter Increment
        Call copyMatDivByValue(eVecXkp, eVecXk, eValrkp.Value) 'Normalized X(k+1) -> X(k)
        diff = Abs(eValrkp.Value - eValRk.Value)               'Check Convergence
    Loop While diff > Abs(eValep.Value * eValRk.Value)

End Sub
```

Propose there are n independent eigenvector V and eigenvalue λ in matrix A

Set $|\lambda_1|>|\lambda_2|>\ldots>|\lambda_n|$

Loop

$X(k) = X(0)/\mathrm{Max}[X(0)]$

$X(k+1) = A * X(k)$

$R(k) = \max[X(k+1)] \,/\, \max[X(k)]$

$X(k+1) = X(k+1)/\max[X(k+1)]$

$k = k+1$

Until $|\mathrm{R(k+1)} - \mathrm{R(k)}| < \varepsilon * |\mathrm{R(k)}|$

We obtain the eigenvector $V(i)$ and eigenvalue $X(i)$

The difference between the Power method and the Rayleigh method is that the Rayleigh method replaces the maximum value of $X(i)$ with its norm value.

$$R(X) = \frac{X^T A X}{X^T X}$$

$$\mathrm{Norm[X(i)]} = ||\,\mathrm{X(i)}\,||$$

Here is the algorithm and pseudocode of the Rayleigh method.

```
Sub Rayleigh()

    Set eSheet = ThisWorkbook.Worksheets("Eigen")
    Set eValep = eSheet.Range("B1")        'Convergence Criteria
    Set eMatA = eSheet.Range("B2:K11")      'Matrix (Eigen values calculated)|
    Set eVecX0 = eSheet.Range("M2:M11")     'Initial Vector
    Set eVecXk = eSheet.Range("D13:D22")    'Iteration Vector X(k)
    Set eVecXkp = eSheet.Range("F13:F22")   'Iteration Vector X(k+1)
    Set eValK = eSheet.Range("B13")         'Iteration count
    Set eValJk = eSheet.Range("B14")        'Element No. of maximum in X(k)
    Set eValRk = eSheet.Range("B15")        'Value of Rayleigh in X(k)
    Set eValrkp = eSheet.Range("B16")       'Value of Rayleigh in X(k+1)

    'write codes here

    eValK.Value = 0                                             'Counter Initalize
    eValrkp.Value = GetEuclidNorm(eVecX0)                       'GetEucridNorm ||X0|| of X0
    Call copyMatDivByValue(eVecX0, eVecXk, eValrkp.Value)       'Normalized X0 -> X(k)

    Do

       eValRk.Value = eValrkp.Value                             'r(k+1) -> r(k)
       Call matVecMult(eMatA, eVecXk, eVecXkp)                  'A*X(k) -> X(k+1)
       eValrkp.Value = GetInnerProduct(eVecXk, eVecXkp)         '(X(k),X(K+1)) -> r(k+1)
       eValK.Value = eValK.Value + 1                            'Counter Increment
       Call copyMatDivByValue(eVecXkp, eVecXk, GetEuclidNorm(eVecXkp)) 'Normalized X(k+1) -> X(k)
       diff = Abs(eValrkp.Value - eValRk.Value)                 'Check Convergence

    Loop While diff > Abs(eValep.Value * eValRk.Value)

End Sub
```

Propose there are n independent eigenvector V and eigenvalue λ in matrix A

Loop

$X(k) = X(0)/\mathrm{Norm}[X(0)]$

$X(k+1) = A * X(k)$

$R(k+1) = X(k)^T * X(k+1)$

$X(k+1) = X(k+1)/\mathrm{Norm}[X(k+1)]$

$k = k+1$

Until $|\mathrm{R(k+1) - R(k)}| < \varepsilon * |\mathrm{R(k)}|$

We obtain the eigenvector $V(i)$ and eigenvalue $X(i)$

B. Inverse iteration method

As the stated above, the power method use the maximum value of $X(i)$ to calculate the eigenvector $V(i)$ and eigenvalue $X(i)$. However, the inverse iteration method tries to use its non-zero minimum value instead of the maximum one.

So we can also use the algorithm of the power method but with $(A-\sigma I)^{-1}$ not A and what we get is $1/(\lambda - \sigma)$ not λ. As for the calculation of $(A-\sigma I)^{-1}$, the LU Decomposition method will be used.

According to calculation, we get $2.34 \times 10^{-7}$with $\sigma = 0$, 0.99 with $\sigma = 1$ and 2.56 with $\sigma = 2$.

Here is the algorithm and pseudocode of the Inverse iteration method.

```
Sub Rayleigh_Inverse()

    Call LUDecomposition

    Set eSheet = ThisWorkbook.Worksheets("Eigen")
    Set eValep = eSheet.Range("B1")          'Convergence Criteria
    Set eVecX0 = eSheet.Range("M29:M38")       'Initial Vector
    Set eVecXk = eSheet.Range("D52:D61")      'Iteration Vector X(k)
    Set eVecXkp = eSheet.Range("F52:F61")     'Iteration Vector X(k+1)
    Set eValK = eSheet.Range("B51")           'Iteration count
    Set eValRk = eSheet.Range("B53")         'Eucrid Norm of X(k)
    Set eValrkp = eSheet.Range("B54")        'Eucrid Norm of X(k+1)

    eValK.Value = 0                                           'counter initalize

    'write codes here

    eValrkp.Value = GetEuclidNorm(eVecX0)                             'GetEucridNorm ||X0|| of X0
    Call copyMatDivByValue(eVecX0, eVecXk, eValrkp.Value)              'Normalized X0 -> X(k)

    Do

       eValRk.Value = eValrkp.Value                                    'r(k+1) -> r(k)
       Call ForwardSubstitution
       Call BackwardSubstitution                                       'A with LU Decomposition
       eValrkp.Value = GetInnerProduct(eVecXk, eVecXkp)                '(X(k),X(K+1)) -> r(k+1)
       eValK.Value = eValK.Value + 1                                   'Counter Increment
       Call copyMatDivByValue(eVecXkp, eVecXk, GetEuclidNorm(eVecXkp))  'Normalized X(k+1) -> X(k)
       diff = Abs(eValrkp.Value - eValRk.Value)                        'Check Convergence

    Loop While diff > Abs(eValep.Value * eValRk.Value)

End Sub
```

Propose there are n independent eigenvector V and eigenvalue λ in matrix A

Loop

$X(k) = X(0)/\mathrm{Norm}[X(0)]$

$X(k) = A * X(k+1)$ with LU decomposition

/ $Y(k+1) = L * X(k+1)$ /

/ $X(k) = U * Y(k+1)$ /

$X(k+1) = A * X(k)$

$R(k+1) = X(k)^T * X(k+1)$

$X(k+1) = X(k+1)/\mathrm{Norm}[X(k+1)]$

$k = k+1$

Until $|R(k+1) - R(k)| < \varepsilon * |R(k)|$

We obtain the eigenvector $V(i)$ and eigenvalue $X(i)$

The advantage of inverse iteration combined with σ over the Rayleigh method is the ability to converge to any desired eigenvalue. By choosing a σ close to a desired eigenvalue, inverse iteration can converge very quickly. As the stated above, the inverse iteration method can traversal the different σ to search the eigenvalue one by one.

C. Jacobi method

The final method is the Jacobi method, which can calculate all the eigenvector $V(i)$ and eigenvalue $X(i)$ simutanously. The basic idea of this method is to diagonal the matrix A with its congruence transformation and those elements located at the diagonal line is the eigenvalue $X(i)$ in need.

$$A(k+1) = Q^T(k) * A(k) * Q(k)$$

$$V = \{Q_1, Q_2 \ \dots \ Q_n\}$$

Here is the algorithm and pseudocode of the Inverse Jacobi method.

```
Sub EigenJacobi()

    Set eSheet = ThisWorkbook.Worksheets("Jacobi")
    Set eValep = eSheet.Range("B1")         'Convergence Criteria
    Set eMatA = eSheet.Range("B2:K11")       'Matrix (Eigen values calculated)
    Set eMatAk = eSheet.Range("B18:K27")    'Matrix (for iteration)
    Set eMatVk = eSheet.Range("B29:K38")    'Eigen Vector Matrix
    Set eValK = eSheet.Range("B13")         'Iteration count
    Set eValI = eSheet.Range("B14")         'Row of maximum element aij  i<j
    Set eValJ = eSheet.Range("B15")         'Column no of aij

    Call copyMat(eMatA, eMatAk)            'initalize A(k)
    Call GenerateUnitMatrix(eMatVk)         'initalize V(k)
    eValK.Value = 0                         'counter initalize

    'write codes here

    Do

       Call FindAbsMaxMat(eMatAk, posi, posj)                'Find Posi of Maximun in A
       eValJ.Value = posj                                    'Remeber in J
       eValI.Value = posi                                    'Remeber in I
       If (Abs(eMatAk(eValI, eValJ)) < eValep) Then Exit Do  'Check Convergence

       eValK.Value = eValK.Value + 1                         'Counter Increment
       tan2 = -2 * (eMatAk(eValI, eValJ) / (eMatAk(eValI, eValI) - eMatAk(eValJ, eValJ)))
       cos2 = 1 / Sqr(1 + tan2 * tan2)
       sin2 = cos2 * tan2
       cos1 = Sqr((1 + cos2) / 2)
       sin1 = sin2 / (2 * cos1)                               'Get Cos & Sin

       Call CalcQtAkQ(eMatAk, cos1, sin1, eValI, eValJ)      'QT * A * Q -> A
       Call CalcVQ(eMatVk, cos1, sin1, eValI, eValJ)         'QT * V -> V

    Loop

End Sub
```

Propose there are n independent eigenvector V and eigenvalue λ in matrix A

Loop

$$a(i,j) = Max[A(k,l) | k \neq l]$$

$$\tan 2\theta = -\frac{2a(i,j)}{a(i,i) - a(j,j)}$$

$$\cos 2\theta = \frac{1}{\sqrt{1 + (\tan 2\theta)^2}}$$

$$\sin 2\theta = \frac{\tan 2\theta}{\sqrt{1 + (\tan 2\theta)^2}}$$

$$\cos\theta = \sqrt{(1 + \cos 2\theta)/2}$$

$$\sin\theta = \cos\theta * \tan\theta$$

$$Q = \{I \mid [I(i,i) = \cos\theta][\, I(i,j) = \sin\theta][I(j,i) = -\sin\theta][I(j,j) = \cos\theta]\}$$

$$A(k+1) = Q^T(k) * A(k) * Q(k)$$

$\mathrm{V(k+1) = V(k)} * Q(k)$

Until $|\mathrm{a(i,j)}| < \varepsilon$

We obtain the eigenvector $V(i)$ and eigenvalue $X(i)$