Exercises on Eigenproblems and Iterative Methods for LES

20/12/2023 17:20

NLA_exos_2

Master in Foundations of Data Science — 2023-2024

NUMERICAL LINEAR ALGEBRA

Exercises from previous exams on *algorithms for eigenvalues and eigenproblems* and on *iterative methods for linear equation solving.*

1. Let $A \in \mathbb{C}^{n\times n}$ be a matrix whose eigenvalues satisfy the strict inequalities
2.1

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|.$$

(1) Which iterative algorithm would you apply to compute the eigenvalue λ_1 of largest absolute value? Give an estimate for its rate of convergence.
(2) How would you proceed to compute the eigenvalue λ_n of smallest absolute value? Similarly as before, give an estimate for the rate of convergence of such an iterative method.

2. Consider the matrix
2.2

$$A = \begin{pmatrix} 6.45 & -4.17 \\ 1.79 & -0.76 \end{pmatrix}.$$

(1) For an starting vector $x_0 \in \mathbb{R}^2$, compute the corresponding *iterative scheme* for the power method, aimed to approximate the largest eigenvalue and its corresponding eigenvector.
(2) Choose the starting vector $x_0 = (1,0)^T$. Knowing that the largest eigenvalue of A is about 10 times bigger than the other one, how many iterations of this scheme are necessary to compute 10 decimal digits of the coordinates of the solution? And how many if we want to compute 30 decimal digits?

3. The *Schur decomposition* of a matrix $A \in \mathbb{C}^{n\times n}$ is its factorization as
2.3

$$A = UTU^T$$

where Q is a unitary matrix and R is upper triangular.
(1) Given A, which algorithm would you apply to compute its Schur decomposition?
(2) How can you compute the eigenvalues and the eigenvectors of A from its Schur decomposition?

4. The *real Schur form* of a matrix $A \in \mathbb{R}^{n\times n}$ is its factorization as

$$A = QRQ^T$$

where Q is an orthogonal matrix and R is block upper triangular, with diagonal blocks of size of 1×1 and 2×2.
(1) Given A, which algorithm would you apply to compute its real Schur decomposition?
(2) How can you compute the eigenvalues and the eigenvectors of A using its real Schur form?

4. A matrix $H = (h_{i,j})_{i,j} \in \mathbb{R}^{n\times n}$ is *upper Hessenberg* if all its coefficients below the lower secondary diagonal are zero, that is, if $h_{i,j} = 0$ whenever $i \geq j+2$.
(1) Explain how a matrix $A \in \mathbb{R}^{n\times n}$ can be reduced to upper Hessenberg form via an orthogonal similarity, that is, how to compute an orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ such that QAQ^T is upper Hessenberg.
(2) Show that the QR iteration preserves the Hessenberg form.

5. Let
2.5

$$A = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \quad\text{and}\quad b = \begin{pmatrix} 1 \\ -2 \end{pmatrix}$$

(1) For this matrix and vector, compute the iterative scheme given by (a) the Jacobi method, (b) the Gauss-Seidel method, and (c) the SOR(ω) method for a parameter $\omega \in \mathbb{R}$.
(2) Using the criterium based on the spectral radius, check if you can guarantee if these Jacobi and Gauss-Seidel schemes converge for any choice of initial vector $x_0 \in \mathbb{R}^2$.
(3) Choosing the starting vector $x_0 = (0,0)^T$, how many iterations of each of these schemes are necessary to compute 30 decimal digits of the solution? And how many if we want to compute 100 decimal digits?

---

1. $A \in \mathbb{C}^{n\times n}$, eigenvalues satisfy $|\lambda_1| > \dots > |\lambda_n|$

a) iterative algorithm to compute the eigenvalue λ_1 of largest absolute value, estimate of its rate of convergence

power method: start $x_0 \in \mathbb{C}^n$ / $\|x_0\| = 1$

$$x_{k+1} = \frac{Ax_k}{\|Ax_k\|} = \frac{A^k x_0}{\|A^k x_0\|}$$

relative error: $\frac{\|x - x_k\|}{\|x\|} \leq c\left(\frac{|\lambda_2|}{|\lambda_1|}\right)^k$

precision: $-\log_\beta \frac{\|x - x_k\|}{\|x\|} \geq k \log \frac{|\lambda_1|}{|\lambda_2|} + o(1)$

b) eigenvalue λ_n of smallest absolute value, estimate of the rate of convergence

$B = A^{-1} \Rightarrow$ eigenvalues of B: $|\lambda_n^{-1}| > \dots > |\lambda_1^{-1}|$

apply power method to B: $x_{k+1} = \frac{Bx_k}{\|Bx_k\|} = \frac{B^k x_0}{\|B^k x_0\|} = \frac{A^{-k} x_0}{\|A^{-k} x_0\|}$

$$-\log_\beta \frac{\|x - x_k\|}{\|x\|} \geq k \log \frac{|\lambda_{n-1}|}{|\lambda_n|} + o(1)$$

$m_k = y_k^T B y_k$ and $m_k \to \lambda_n^{-1}$

$$-\log_\beta \left|\frac{\lambda_n^{-1} - m_k}{\lambda_n}\right| \geq k \log \frac{|\lambda_{n-1}|}{|\lambda_n|} + o(1)$$

$Av = \lambda v$

$A^{-1}(Av) = A^{-1}(\lambda v)$

$\Rightarrow A^{-1}v = \lambda^{-1} v$

power method to A^{-1}: obtain (u_k, v_k)

approximate eigenpair of A^{-1} such that $u_k \to \lambda^{-1}_n$

(u_k^{-1}, v_k) = approximate eigenpair of A such that $u_k^{-1} \to \lambda_n$

$\|v_n - y_k\| \leq c\left(\frac{|\lambda_n|}{|\lambda_{n-1}|}\right)^k$

A singular?

one eigenvector is zero → v_n generator of the ker (A)

2. $A = \begin{bmatrix} 6.45 & -4.17 \\ 1.79 & -0.76 \end{bmatrix}$

a) starting vector $x_0 \in \mathbb{R}^2$, iterative scheme for the power method, approximate the largest eigenvalue and its corresponding vector

x_0: $x_{k+1} = \frac{Ax_k}{\|Ax_k\|} = \frac{A^k x_0}{\|A^k x_0\|}$

iteration 1 approximations: $\tilde{x} = x_{i+1}$

$\tilde{\lambda} = \tilde{x}^* A \tilde{x}$

b) starting vector $x_0 = (1,0)^T$, largest eigenvalue of A 10 times bigger than the other one, iterations necessary to compute 10 decimal digits

$\left|\frac{\lambda_1}{\lambda_2}\right| = 10$ $(\mathbb{R}^2)$

$k \log_{10}(10) \geq 10$ → number of wanted decimals

$\underbrace{10}_{\text{base}}$, $(10) = \left|\frac{\lambda_1}{\lambda_2}\right|$

$k \cdot 1 \geq 10$ → need at least 10 iterations

30 decimal digits: $k \cdot 1 \geq 30$ → 30 iterations

$c = 0$ → minimum amount of needed iterations

3. Schur decomposition, matrix $A \in \mathbb{C}^{n\times n}$, factorization as $A = UTU^T$, Q is unitary, R is upper triangular

a) algorithm to compute its Schur decomposition

Hessenberg Reduction using Householder Transformations

b) compute the eigenvalues and eigenvectors of A from its Schur decomposition

eigenvalues are the diagonal elements of the upper triangular matrix T

eigenvectors are the columns of U

4. real Schur form of a matrix $A \in \mathbb{R}^{n\times n}$ and its factorization as $A = QRQ^T$, Q is an orthogonal matrix, R is a block upper triangular matrix with diagonal blocks of size 1x1 and 2x2

a) algorithm to compute its Schur decomposition

QR iteration

b) compute the eigenvalues and eigenvectors of A from its Schur decomposition

eigenvalues are the diagonal elements of the block upper triangular matrix R

eigenvectors can be computed from the blocks of R

* 1x1 block - corresponding eigenvector is the corresponding column of Q
* 2x2 block - corresponding eigenvectors can be computed by solving a system of linear equations
  eigenvectors are then the columns of the matrix Q

4. $H = (h_{i,j})_{i,j} \in \mathbb{R}^{n\times n}$ upper Hessenberg all coefficients below the lower secondary diagonal are zero → $h_{i,j} = 0$ whenever $i \geq j+2$

a) $A \in \mathbb{R}^{n\times n}$ reduced to upper Hessenberg form via an orthogonal similarity (how to compute an orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ such that QAQ^T is upper Hessenberg)

using the Householder transformation:

- initialize Q as an identity matrix, $Q = \mathbb{1}_n$
- apply householder transformations
  for each column $j = 1, 2, \dots, n-2$, construct a householder transformation H_j such that when applied to A, it zeroes out the subdiagonal entries below the $(j+1)^{th}$ entry in column j
  the Householder transformation is of the form $H_j = \mathbb{1}_n - 2v_jv_j^T$ where v_j is a chosen vector
- update A and Q
  update A and Q by applying H_j: $A \leftarrow H_jAH_j^T$ and $Q \leftarrow QH_j$
- repeat until A becomes upper Hessenberg via the orthogonal similarity QAQ^T (Q is orthogonal)

b) QR iteration preserves the Hessenberg form

orthogonal similarity: at each step of the QR iteration, the matrix A_k is factorized into $A_k = Q_kR_k$ where Q_k is orthogonal and R_k is upper triangular

similarity transformation: $A_{k+1} = R_kQ_k$ involves multiplying an upper triangular matrix R_k with an orthogonal matrix Q_k

since orthogonal matrices preserve orthogonality and upper triangular structure, A_{k+1} is also in upper Hessenberg form

iteration: as the QR iteration progresses through successive iterations, A_k is updated and A_{k+1} retains the upper Hessenberg form

5. $A = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$, $b = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$

a) matrix/vector for iterative scheme

Jacobi method: for $i = 1, \dots, n$ $x_{i+1,j} \leftarrow \frac{1}{a_{jj}}\left(b_j - \sum_{k\neq j} a_{j,k}\, x_{i,k}\right)$

Gauss-Seidel method: for $j = 1, \dots, n$ $x_{i+1,j} \leftarrow \frac{1}{a_{jj}}\left(b_j - \sum_{k<j} a_{j,k}\, x_{i+1,k} - \sum_{k>j} a_{j,k}\, x_{i,k}\right)$

SOR(ω) with $\omega \in \mathbb{R}$ method: $x_{i+1,j} = (1-\omega)x_i^{GS} + \omega x_{i+1}^{GS}$

b) criterion based on the spectral radius, check if you can guarentee if the Jacobi and Gauss-Seidel schemes converge for any choice of initial vector $x_0 \in \mathbb{R}^2$

$\rho(R) < 1$

Jacobi: $R = L + U = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ with $A = \overset{D}{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}\left(\overset{I}{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}} - \overset{L}{\begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}} - \overset{U}{\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}}\right)$

$\chi_R = \begin{vmatrix} -\lambda & 1 \\ 1 & -\lambda \end{vmatrix} = \lambda^2 - 1 \Rightarrow \lambda = \pm\sqrt{1} \Rightarrow \rho(R) = 1$

no guarenteed convergence

Gauss-Seidel: $R = (\mathbb{1}_n - L)^{-1}U = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}^{-1}\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix}$

$\chi_{GS} = \begin{vmatrix} -\lambda & 1 \\ 0 & -1-\lambda \end{vmatrix} = \lambda^2 + \lambda - 1 \Rightarrow \lambda = \frac{-1 \pm \sqrt{1 - 4 \cdot 1 \cdot -1}}{2} = \frac{-1 \pm \sqrt{5}}{2} \Rightarrow \rho(R) > 1$

no guarenteed convergence

c) starting vector $x_0 = (0,0)^T$, iterations necessary to compute 30 decimal digits, 100 decimal digits?

$\rho(R) \geq 1$, $\lambda - x_{i+1} = \lambda^{i+1}(x - x_0)$

$A - D - \tilde{L} - \tilde{U} = D(\mathbb{1}_2 - U - L)$

Jacobi: splitting $A = D - (\tilde{L} + \tilde{U}) \to x_{k+1} = D^{-1}(\tilde{L} + \tilde{U})x_k + D^{-1}b$

$A = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ -1 & 0 \end{bmatrix} - \begin{bmatrix} 0 & -1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}\left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ -\frac{1}{2} & 0 \end{bmatrix} - \begin{bmatrix} 0 & -\frac{1}{2} \\ 0 & 0 \end{bmatrix}\right)$

$R_J = \begin{bmatrix} 0 & -\frac{1}{2} \\ -\frac{1}{2} & 0 \end{bmatrix}$, $C_J = \begin{bmatrix} \frac{1}{2} \\ 0 \end{bmatrix}$

iteration: $x_{k+1} = \begin{bmatrix} 0 & -\frac{1}{2} \\ -\frac{1}{2} & 0 \end{bmatrix} x_k + \begin{bmatrix} \frac{1}{2} \\ 0 \end{bmatrix}$

$$\begin{cases} x_{k+1,1} = -\frac{1}{2}x_{k,2} + \frac{1}{2} \\ x_{k+1,2} = -\frac{1}{2}x_{k,1} \end{cases}$$

Gauss-Seidel: $A = (D - \tilde{L}) - \tilde{U}$

$R_{GS} = (D - \tilde{L})^{-1}\tilde{U}$, $C_{GS} = (D - \tilde{L})^{-1}b$

$R_{GS} = \begin{bmatrix} 2 & 0 \\ 1 & 2 \end{bmatrix}^{-1}\begin{bmatrix} -1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{2} \\ 0 & \frac{1}{4} \end{bmatrix}$, $C_{GS} = \begin{bmatrix} 2 & 0 \\ 1 & 2 \end{bmatrix}^{-1}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ -\frac{1}{4} \end{bmatrix}$

iteration: $x_{k+1} = \begin{bmatrix} 0 & -\frac{1}{2} \\ 0 & \frac{1}{4} \end{bmatrix} x_k + \begin{bmatrix} \frac{1}{2} \\ -\frac{1}{4} \end{bmatrix}$

SOR(ω): splitting $A = M - K$

iteration: $x_{k+1} = M^{-1}Kx_k + M^{-1}b$

$x_{k+1}^{SOR} = \omega x_{k+1}^{GS} + (1-\omega)x_k$

$R_{SOR(\omega)} = (\mathbb{1}_2 - \omega L)^{-1}((1-\omega)\mathbb{1}_2 + \omega U) = \begin{bmatrix} 1 & 0 \\ \frac{\omega}{2} & 1 \end{bmatrix}\begin{bmatrix} 1-\omega & -\frac{\omega}{2} \\ 0 & 1-\omega \end{bmatrix} = \begin{bmatrix} 1-\omega & -\frac{\omega}{2} \\ \frac{\omega^2}{2} - \frac{\omega}{2} & \frac{\omega^2}{4} - \omega + 1 \end{bmatrix}$

$C_{SOR(\omega)} = \omega(\mathbb{1}_2 - \omega L)^{-1}D^{-1}b = \begin{bmatrix} \frac{\omega}{2} \\ -\frac{\omega^2}{4} \end{bmatrix}$

CONVERGENCE

$\rho(R_J) = \frac{1}{2}$, the iterator converges: $x_k \xrightarrow{k\to\infty} 0$

$\chi_J(t) = \det(R_J - t\mathbb{1}_2) = \det\begin{pmatrix} -t & -\frac{1}{2} \\ -\frac{1}{2} & -t \end{pmatrix} = t^2 - \frac{1}{4} = 0 \to t = \pm\frac{1}{2}$

$\chi_{GS}(t) = \det\begin{pmatrix} -t & -\frac{1}{2} \\ 0 & -t-\frac{1}{4} \end{pmatrix} = t\left(t + \frac{1}{4}\right) = 0 \to t = 0, t = -\frac{1}{4}$

$\rho(R_{GS}) = \frac{1}{4}$

ITERATIONS FOR 30 CORRECT DECIMAL DIGITS

initial vector: $x = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

number of correct digits of $\tilde{x}$: $-\log_{10} \frac{\|x - \tilde{x}\|}{\|x\|}$

need a bound for $\|x - x_k\| \leq c \cdot \rho(R_J)^k = c \cdot 2^{-k}$

for 30 digits: need $\|x - x_k\| \leq 10^{-30}$

acheived when $10^{-30} \leq 2^{-k}$ (up to c)

$-30 \log 10 \leq -k \log 2$

$\Rightarrow k \geq 30 \frac{\log 10}{\log 2} + c$