Iterative methdos

Zecheng Zhang

April 19, 2023

1 Eigenvalue problem

We will consider symmetric matrix $A \in \mathbb{R}^{m\times m}$. We define the Rayleigh Quotient,

$$r(x) = \frac{x^t Ax}{x^t x}. \tag{1}$$

Note that if x is an eigenvector of A, $r(x) = \lambda$ is its eigenvalue.

One way to understand this formula is: given x, what is the scale α which acts almost like an eigenvalue of x in the sense that $Ax - \alpha x$ is minimized? This is a least square problem, but x is the matrix α is the unknown vector, and Ax is the right-hand side b vector. We can see that $\alpha = r(x)$ if we consider the normal equation.

Take the derivative of $r(x)$ with respect to all component x_j of x, we can easily derive that,

$$\nabla r(x) = \frac{2}{x^t x}(Ax - r(x)x). \tag{2}$$

We can see that when x is the eigenvector, the gradient vanishes. Conversely, if the gradient is trivial with $x \neq 0$, x is an eigenvector with eigenvalue $r(x)$.

Theorem 1.1. Let q_j be an eigenvector of A, we have

$$r(x) - q_j = \mathcal{O}(\|x - q_j\|^2), \tag{3}$$

as $x \to q_j$.

The Power iteration is expected to return an eigenvector corresponding to the largest eigenvalues.

Algorithm 1: Power Iteration

Set v_0 with $\|v_0\| = 1$.
for $k = 1$ *to* ... **do**
    $w = Av^k$
    $v^k = w/\|w\|$
    $\lambda^k = (v^k)^T Av^k$

Theorem 1.2. Suppose $|\lambda_1| > |\lambda_2| \geq ... \geq |\lambda_m| \geq 0$ and $q_1^T v^0 \neq 0$. Then the algorithm satisfies,

$$\|v^k - q_1\| = \mathcal{O}(\left|\frac{\lambda_2}{\lambda_1}\right|^k), \tag{4}$$

$$|\lambda^k - \lambda_1| = \mathcal{O}(\left|\frac{\lambda_2}{\lambda_1}\right|^{2k}), \tag{5}$$

as $k \to \infty$

Remark 1. Power iteration has some limitations.

1. It can only find the largest eigenvectors corresponding to the largest eigenvalues.
2. The convergence is linear, i.e., the algorithm reduces the error by a factor $|\frac{\lambda_2}{\lambda_1})$ in every iteration.
3. The quality of the convergence depends on the quotient. If there is no huge eigen-gap, the convergence is slow.

1.1 Inverse Iteration

Let μ be a number which is not an eigenvalue of A, the eigenvectors of $(A-\mu I)^{-1}$ are the same as the eigenvectors of A, and the coresponding eigenvalues are $(\lambda_j - \mu)^{-1}$, where $\{\lambda_j\}$ are the eigenvalues of A.

This motivates us to design an algorithm to identify λ_j and the corresponding eigenvectors of A. Suppose we know any estimate of λ_j and denote it as μ. $(\mu-\lambda_j)^{-1}$ will be very large. According to the Remark, the power iteration can identify q_j, which are the eigenvectors of $(A-\mu I)^{-1}$ (also the eigenvectors of A). This idea is called the inverse iteration.

Algorithm 2: Inverse iteration

$v^0 =$ some vectors with norm 1
for $k = 1$ *to* ... **do**
    Solve $(A-\mu I)w = v^{k-1}$ for w
    $v^k = w/\|w\|$
    $\lambda^k = (v^k)^T A v^k$.

Rayleigh quotient is one method to estimate eigenvalues from an eigenvector estimation. Inverse iteration is an estimate of the eigenvector from the eigenvalues.

Algorithm 3: RQ iteration

$v^0 =$ some vectors with norm 1
$\lambda^0 = v^0 A v^0 =$ coresponding Rayleigh quotient.
for $k = 1$ *to* ... **do**
    Solve $(A-\lambda^{k-1} I)w = v^{k-1}$ for w
    $v^k = w/\|w\|$
    $\lambda^k = (v^k)^T A v^k$.

Without proof, the Rayleigh Quotient iteration has cubic convergence.

2 Reduction to Hessenberg form

Schur factorization returns $A = QTQ^*$, where T is a triangular matrix, i.e., we would like to apply unitary similarity transformation to introduce zeros below the diagonal. The natural first idea is to use the Householder.

The first Householder reflector Q_1^* multiplied on the left of A would introduce zeros below the diagonal in the first column, and the Householder reflector will change all rows of A. This is good up to now; however, if we complete the process of multiplying Q_1 on the right, all zeros previously introduced are destroyed. We will verify this in class.

The good idea in step 1 is to choose a unitary matrix Q_1^* that will leave the first row unchanged. It will change the second row to the last row and introduce zeros below the second entry in the first column. It can be verified that the right multiplication by Q_1 will not change the zeros introduced by Q_1^*. After repeating this process for $m-2$ times, the resulting matrix is in the Hessenberg form, denoted as H.

Algorithm 4: Reduction to Hessenberg

for $k = 1$ *to* $m-2$ **do**
- $x = A_{k+1:m,k}$
- $v_k = (sign(x_1))\|x\|_2 e_1 + x$
- $v_k = v_k/\|v_k\|$
- $A_{k+1:m,k:m} = A_{k+1:m,k:m} - 2v_k v_k^* A_{k+1:m,k:m}$
- $A_{1:m,k+1:m} = A_{1:m,k+1:m} - 2A_{1:m,k+1:m} v_k v_k^*$

When A is Hermitian, H is symmetric, then H is a tridiagonal matrix.

3 QR Algorithm

Algorithm 5: QR Algorithm

$A_1 = A$

for $k = 1$ *to* ... **do**
- $Q_k R_k = A_k$
- $A_{k+1} = R_k Q_k$.

The algorithm converges to the Schur form of the matrix A. Specifically, suppose A admits the Schur decomposition $A = UTU^T$, then A_k converges to T.

Remark 2. Some properties regarding the algorithm.

1. $A_{k+1} = R_k Q_k$, since $A_k = Q_k R_k$, $R_k = Q_k^t A_k$, this implies that $A_{k+1} = Q_k^t A_k Q_k$. That is, all A_k are unitarily similar to each other, i.e., eigenvalues of all A_k and A are the same. Since A^k converges to T, we have the eigenvalues of A.

2. Let us define $Q^{(k)} = Q_1 Q_2 ... Q_k$ and $R^{(k)} = R_k R_{k-1} ... R_1$, we have the following theorem.

   Property 3.0.1. (a) $A_{k+1} = (Q^{(k)})^t A Q^{(k)}$.
   (b) $A^k = Q^{(k)} R^{(k)}$.

   Proof. The property (a) is trivial to prove and let us the property (b). Let us prove by induction. $k = 1$ case is trivial. Suppose $A^{k-1} = Q^{(k-1)} R^{(k-1)}$ is true. By the property (a) and the algorithm definition, we have,

$$A_k = (Q^{(k-1)})^t A Q^{(k-1)} = Q_k R_k. \tag{6}$$

Multiplying both sides by $Q^{(k-1)}$, it follows that $AQ^{(k-1)} = Q^{(k-1)}Q_kR_k$. Substitute into the assumption,

$$A^k = AA^{k-1} = AQ^{(k-1)}R^{(k-1)} = Q^{(k-1)}Q_kR_kR^{(k-1)} = Q^{(k)}R^{(k)}. \tag{7}$$

□

The property provides us with one way to compute the QR factorization of matrix power. It can be shown that, this algorithm is stable.

We now intuitively explain the connection between QR and the Power iteration. It can be shown that columns of A^k are dominated by the "leading" eigenvector x_1 of A, i.e., $Ax_1 = \lambda_1 x_1$. Let us consider $A^k e_1 = Q^{(k)}R^{(k)}e_1 = cq_1$, where q_1 is the first column of $Q^{(k)}$ scaled by constant c. This implies that the leading eigenvector of A is related to q_1. Property (a) shows that $A_{k+1} = (Q^{(k)})^t AQ^{(k)}$ and A_{k+1} is the Schur form of A, this indicates that q_1 is the eigenvector of A and $A_{k+1}[1,1]$ is the corresponding eigenvalue.

3.1 Shifted QR

Algorithm 6: Shifted QR Algorithm

```
A_1 = A
for k = 1 to ... do
    Q_k R_k = A_k − s_k I
    A_{k+1} = R_k Q_k + s_k I.
```

If $s_k \sim \lambda_n$, then $A_{k+1}[m,m] \sim \lambda_m$. It can be shown that $(A - s_kI)(A - s_kI)...(A - s_kI) = Q^{(k)}R^{(k)}$.

3.2 Preprocessing

For QR and shifted QR, we need to run Householder to QR the matrix A_k in each iteration. The cost is m^3 for one QR, this is very costly. It is important to find a good initial condition to reduce the number of iterations.

As we have discussed before, $A_k[m,m]$ converges to λ_m. Motivated by the inverse iteration, the iterative algorithm will find it very fast if we choose s_k closed to λ_m. We can choose $s_k = A_k[m,m]$ or some number that is close to $A_k[m,m]$.

One method that works well is to reduce the matrix A to the Hessenberg form. Hessenberg form is different from the Schur form, but it is very close to the upper triangular form.

4 Iterative methods

In this section, let us consider matrix $A \in \mathbb{R}^{m\times m}$. The iterative methods has a structure $x_{n+1} = \phi(x_n)$, where x_n is the output of $n-$ step and ϕ is the algorithm. Broadly speaking, the idea of iterative methods is to:

1. Gradually refine the solution iteratively.

Fact, the cols of A^k are dominanted by the leading eigenvectors of A.

$$A x_1 = \lambda_1 x_1, \quad \lambda_1 \text{ is the largest eval of } A.$$

$$A^k = Q^{(k)} R^{(k)}$$

$$Q^{(k)} = Q_1 Q_2 \cdots Q_k$$

$$R^{(k)} = R_k R_{k-1} \cdots R_1 \rightarrow \text{upper triangular}$$

$$A^k e_1 = x_1 \quad \text{(eigen-vector of } A \text{ from the Power method)}$$

$$\overset{\text{P.P.(b)}}{=} Q^{(k)} R^{(k)} e_1 = Q^{(k)} \cdot \begin{pmatrix} c \\ 0 \\ \vdots \\ 0 \end{pmatrix} = c\, q_1$$

QR iteration

$\Rightarrow$ q_1 is also eigenvector of A.

QR iteration: simultaniously implement Power method for all eigenvectors.

$$A_k = [Q^{(k)}]^* A Q^{(k)}$$

If we take q_1 of $Q^{(k)}$

$$[Q^{(k)}]^* A_k Q^{(k)} = A_{11}$$

$$A_k q_1 = \lambda_1 q_1$$

shift QR algorithm

$A_1 = A$

for $k = 1, \ldots$

$Q_k R_k = A_k - s_k I$ → Householder $O(m^3)$

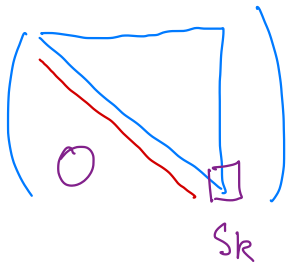
$A_{k+1} = R_k Q_k + s_k I.$

P.P. $(A - s_k I)(A - s_k I) \cdots (A - s_k I) = Q^{(k)} R^{(k)}$

P.P. $s_k \sim \lambda_m$, $A_k [m, m] \to \lambda_m$

Pre processing.

Reduction to the Hessenberg matrix



is closed to the Schur form of the A, & provides us with a good initial guess.

Preprocessing algorithm. (save computational cost).

$O(m^3)$ Perform Hessenberg reduction on A to H, $A_1 = H$

for $k = 1, \cdots$ $\quad s_k = H_{mm}$

$O(m^3)$ $\quad Q_k R_k = A_k - s_k I$

$A_{k+1} = R_k Q_k + s_k I$

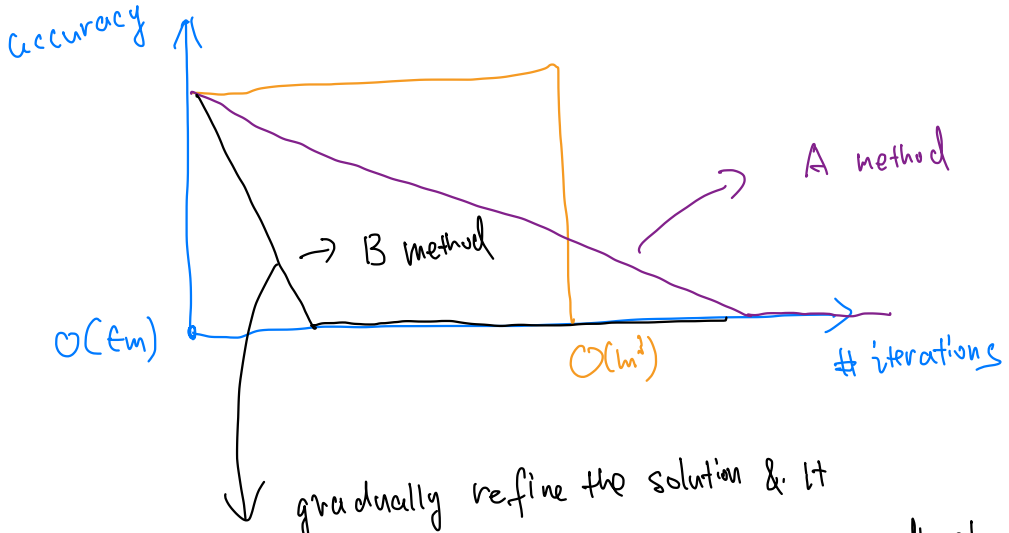
Cost $\quad O(m^3) + O(m^3) \cdot$ # iterations (small).

Iterative methods.

$x_{n+1} = \phi(x_n)$, $\quad x_{n+1} \to$ real solution, as $n \to \infty$

Example, Power method,

Household (not)



gradually refine the solution & it goes to $O(\varepsilon n)$ faster than the direct methods..

Krylov subspace methods.

$A \in \mathbb{R}^{n \times n}$, $K_n(A, b) = \{b, Ab, A^2b, \cdots A^{n-1}b\}$

We want to approximate the solution in $K_n(A, b)$.

Example. Power method: we only use $A^n b$ to represent the eigen-vector of A.