M. SEMBRA - 202[illegible]

IX. Computing eigenvalues and eigenvectors

Let A $n\times n$-matrix and consider the algorithm:

QR iteration

$T_0 \leftarrow U_0^* A U_0$ for a given U_0 $n\times n$ unitary

for $k = 1, 2, \dots$

$T_{k-1} = Q_k \cdot R_k$ QR factorization

$T_k \leftarrow R_k \cdot Q_k$

Since
$$T_k = R_k \cdot Q_k = Q_k^* \cdot \overbrace{Q_k \cdot R_k}^{T_{k-1}} \cdot Q_k$$
we have that

$$A = (Q_0 \cdots Q_k) \cdot T_k \cdot (Q_0 \cdots Q_k)^*$$

hence T_k is unitarily similar to A.

In most situations,

$$T_k \xrightarrow[k\to\infty]{} T \quad \text{upper triangular (Schur form)}$$

This is less obvious!

To motivate this method and understand its convergence, we first study two other eigenvalue iterations: the power method and the orthogonal iteration.

IX.1 The power method [D, § 4.4.1], [GvL, § 7.3.1]

It is the iteration:

Power method

Given $x_0 \in \mathbb{C}^n$ with $\|x_0\|_2 = 1$:

can use any other norm

For $k = 0, 1, 2, \ldots$

$$y_{k+1} \leftarrow A \cdot x_k$$

$$x_{k+1} \leftarrow \frac{y_{k+1}}{\|y_{k+1}\|_2}$$

When stopping the iteration at $k = K$ set

$$\tilde{\lambda} \leftarrow x_{K+1}^* \cdot A \cdot x_{K+1}$$

Suppose that A is <u>diagonalizable</u>

$$A = \underbrace{S}_{(s_1 \cdots s_n)} \cdot \underbrace{\Lambda}_{\mathrm{diag}(\lambda_1, \ldots, \lambda_n)} \cdot S^{-1}$$

s_i: <u>unit eigenvectors</u>

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

Write

$$x_0 = \alpha_1 s_1 + + \alpha_n s_n$$

with $\alpha_1 \neq 0$

linear combination of the eigenvectors

Then

$$A^k x_0 = \alpha_1 \lambda_1^k s_1 + \cdots + \alpha_n \lambda_n^k s_n$$

$$= \alpha_1 \lambda_1^k \left(s_1 + \sum_{j=2}^{n} \frac{\alpha_j}{\alpha_1} \left(\frac{\lambda_j}{\lambda_1} \right)^k s_j \right)$$

This implies that for $x_k = \dfrac{A^k x_0}{\|A^k x_0\|_2}$

we have that

$$\|x_k - s_1\|_2 \leq O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right)$$

It can also be shown that for $\tilde{\lambda}_k = x_k^* \cdot A \cdot x_k$

$$|\tilde{\lambda}_k - \lambda_1| \leq O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right)$$

The number of correct digits in base b of this approximation:

$$-\log_b \|x_k - s_1\|_2 , -\log_b |\tilde{\lambda}_k - \lambda_1| \geq \log_b\left(\left|\lambda_1/\lambda_2\right|\right) \cdot k + \text{constant}$$

Minor drawback: the assumption $\boxed{\alpha_1 \neq 0}$.
If x_0 is random, this holds with very high probability.
Even if this is not the case, round offs during the iteration will ensure a nonzero component in the direction of s_1. Many times we have a reasonable guess for x_0.

In the PageRank algorithm: x_0 is the score vector of the previous web.

Major drawback: it converges to ~~a pair~~ eigenvalue/eigenvector only if there is dominant one, and the convergence is linear with ratio depending on $|\lambda_1|/|\lambda_2|$. This is not the case in many situations:

- orthogonal matrices (all eigenvalues have absolute value 1)
- real matrices with complex (non real) eigenvalues. (the eigenvalues come in pairs of conjugate complex numbers)

In the PageRank algorithm, $\lambda_1 = 1$ is the dominant eigenvalue and $\frac{|\lambda_2|}{|\lambda_1|} \leq 0.85$.

Advantage: the implementation only uses matrix-vector multiplications

In PageRank: the matrix-vector multiplication reduces to multiplication with the link matrix, which is sparse.

IX.2 Orthogonal iteration [D, § 4.4.3], [GvL, § 7.3.2]

This is a generalization of the power method, that can be used to compute higher dimensional invariant subspaces. Let $1 \leq r \leq n$.

Orthogonal iteration

Given Q_0 $n \times r$ unitary matrix

for $k = 0, 1, 2, \dots$

$Y_{k+1} \leftarrow A \cdot Q_k$

$Q_{k+1} R_{k+1} = Y_{k+1}$ QR factorization

Suppose again A diagonalizable

$$A = S \cdot \underbrace{\Lambda}_{\operatorname{diag}(\lambda_1, \dots, \lambda_n)} \cdot S^{-1}$$

with

$$|\lambda_1| \geq \dots \geq |\lambda_r| > |\lambda_{r+1}| \geq \dots \geq |\lambda_n|$$

We have that

$$\underbrace{\operatorname{span}(Q_k)}_{\text{linear space generated by the columns of } Q_k} \overset{Y_k = Q_k \cdot R_k}{=} \operatorname{span}(Y_k) = \dots = \operatorname{span}(A^k Q_0)$$

Hence Q_k gives ~~an~~ orthonormal basis of $\operatorname{span}(A^k Q_0)$

prevents under/overflow and collapsing dimensions:

for $k \to \infty$ this should converge to an orthonormal basis of the invariant subspace generated by the dominant eigenvectors $s_1, \dots, s_r$:

$$\operatorname{span}(Q_k) = \operatorname{span}(A^k Q_0) \xrightarrow[k \to \infty]{} \operatorname{span}(s_1, \dots, s_r)$$

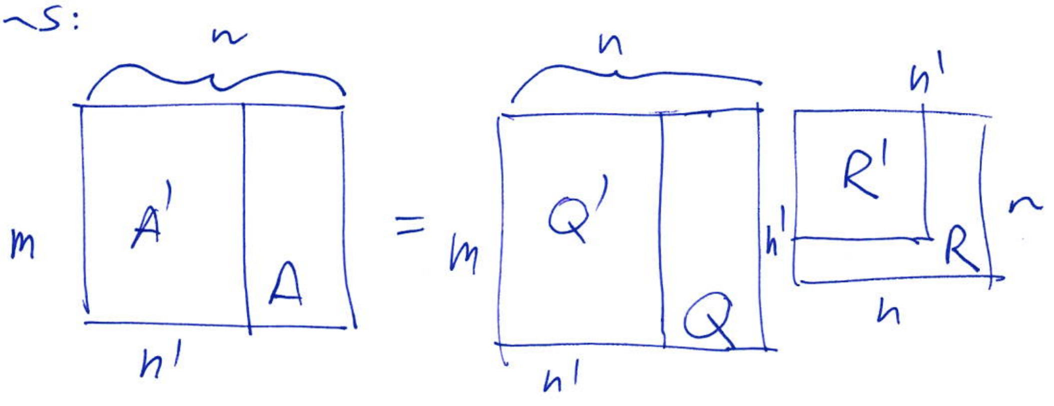
For $r=1$ it coincides with the power method, because

$$x_{k+1} \cdot \|A y_{k+1}\|_2 = y_{k+1}$$

is the QR factorization of the vector y_{k+1}.
In this case $r=1$, $Q_0, Q_1, Q_2, \dots \in \mathbb{C}^{n\times 1}$ is the sequence of vectors produced by the power method.

Also for $1 \leq s < r$ the first s columns of Q_k coincide with the orthogonal iteration of rank s starting with the first s columns of Q_0:

the QR factorization is compatible with restricting columns:



$A' = Q'R'$ is the QR factorization of A.

We can then set $r=n$. , which would run the orthogonal iteration for all r (and initial matrix given by the first r columns of Q_0)

To study the convergence of the orthogonal iteration, suppose

$$|\lambda_1| > \cdots > |\lambda_n|$$

Then for a "typical" Q_0 we have that for $k \to \infty$

$$Q_k \to Q \qquad n\times n \text{ unitary}$$

$$Q_k^* \cdot A \cdot Q_k \to T \qquad \underline{\text{upper triangular}}$$

s.t. $A = Q \cdot T \cdot Q^*$ (Schur decomposition)

see [D, Theorem 4.8]

IX.3 The QR iteration [D, §4.4.4] [GvL, §7.3.3]

Set

$$T_k := Q_k^* \cdot A \cdot Q_k \qquad k = 0, 1, 2, \ldots$$

The QR iteration arises when considering how to compute T_k directly from its predecessor T_{k-1}.

We have

by definition of Q_k

$$T_{k-1} = Q_{k-1}^* \cdot A\, Q_{k-1} \overset{\downarrow}{=} (Q_{k-1}^* \cdot Q_k) \cdot R_k \qquad (*)$$

$$T_k = Q_k^* \cdot A \cdot Q_k \overset{\downarrow}{=} (Q_k^* \cdot A \cdot Q_{k-1})\, Q_{k-1}^*\, Q_k = R_k \cdot (Q_{k-1}^* \cdot Q_k)$$

$$\boxed{Q_{k-1} \cdot A = R_k \cdot Q_k}$$

We have that $Q_{k-1}^* \cdot Q_k$ are unitary and $(*)$ is the QR factorization of T_{k-1}

Drawbacks:

* A single QR iteration costs $\mathcal{O}(n^3)$ flops.
* Convergence (when it exists) is only linear.

We will next study how to overcome these practical difficulties.

[D, § 4.4.6], [GvL, § 7.4.1, 7.4.2 & 7.4.3]

IX.5 The real Schur form and the Hessenberg reduction

Matrices from applications have real entries. Hence we concentrate on the real version of the QR iteration: for $A \in \mathbb{R}^{n\times n}$ choose $Q_0 \in \mathbb{R}^{n\times n}$ orthogonal and set:

$$H_0 \leftarrow Q_0^T \cdot A \cdot Q_0$$

for $k = 1, 2, \dots$

$$H_{k-1} = Q_k \cdot R_k \qquad \text{(QR factorization)}$$

$$H_k \leftarrow R_k \cdot Q_k$$

If A has complex eigenvalues then H_k cannot converge to a triangular matrix. In this case we ~~must~~ content ourselves with convergence to the real Schur form:

$$A = \underset{\text{orthogonal}}{Q} \cdot \underset{\text{block upper triangular}}{T} \cdot Q^T$$

with $T = \begin{pmatrix} T_{11} & T_{12} & \cdots & T_{1n} \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & T_{n-1,n} \\ 0 & \cdots & 0 & T_{nn} \end{pmatrix}$ and T_{ii}: 1×1, or 2×2 with complex conjugate eigenvalues

To implement the QR iteration, we have to choose Q_0 carefully s.t.

$$H_0 = Q_0^T \cdot A \cdot Q_0 \quad \text{is } \underline{\text{Hessenberg}}$$

$(h_{ij} = 0 \text{ for } i > j+1)$

In this way, the complexity of each iteration is Lowered to $\mathcal{O}(n^2)$ flops.

This <u>Hessenberg reduction</u> is a variation of the QR factorization, and can be done with a sequence of <u>Householder</u> reflections.

We illustrate it in the 5×5 case:

1) Choose $P_1 = \begin{pmatrix} 1 & \\ & \tilde{P}_1 \end{pmatrix}$ s.t.

$\tilde{P}_1$ ← 4×5 Householder

$$P_1 A = \begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \end{pmatrix} \text{ and } A_1 = P_1 A P_1^T = \begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \end{pmatrix}$$

P_1 leaves the 1st row of $P_1 \cdot A$ unchanged

P_1^T leaves the 1st column of $(P_1 A) \cdot P_1^T$ unchanged, including the 0's.

2) Choose $P_2 = \begin{pmatrix} \mathbb{1}_2 & \\ & \tilde{P}_2 \end{pmatrix}$ s.t.

$\tilde{P}_2$ ← 3×4 Householder

$$P_2 A_1 = \begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \end{pmatrix} \text{ and } A_2 = P_2 A_1 P_2^T = \begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \end{pmatrix}$$

P_2 leaves the 1st and 2nd rows of $P_2 \cdot A_1$ unchanged

P_2^T leaves the 1st and 2nd columns of $(P_2 A_1) \cdot P_2^T$ unchanged

3) Choose $P_3 = \begin{pmatrix} \mathbb{1}_3 & \\ & \tilde{P}_3 \end{pmatrix}$ s.t.

2×2 Householder

$$P_3 \cdot A_2 = \begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{pmatrix} \quad \text{and} \quad P_3 \cdot A_2 \cdot P_3^T = \begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{pmatrix}$$

In general

$$(P_{n-2} \cdots P_1) \cdot A \cdot P_1^T \cdots P_{n-2}^T = H_0 \quad \underline{\text{Hessenberg}}$$

$$\shortparallel$$

$$Q_0^T \cdot A \cdot Q_0$$

with $Q_0 = P_1 \cdots P_{n-2}$ $n \times n$-orthogonal

(P_i is symmetric).

The complexity of computing Q_0 is $\underline{5n^3 \text{ flops}}$.

Now we turn to <u>each</u> <u>QR step.</u>

<u>Important observation:</u> the Hessenberg form is preserved during the QR iteration: Let

$$H = Q \cdot R \qquad \text{and} \qquad H_+ = R \cdot Q$$

Hessenberg

Since R is upper triangular, the jth column of Q is a linear combination of the first j columns of H:

$$H = (h_1 \cdots h_n) \qquad Q = (q_1 \cdots q_n)$$

$$R = (r_{ij}) \qquad S = R^{-1} = (s_{ij})$$

Then

$$q_j = \sum_{k=1}^{j} s_{kj} h_k$$

and so Q is Hessenberg.

Similarly the jth row of H_+ is a linear combination of the first j rows of Q and so H_+ is Hessenberg.

The QR factorization of H is computed with $n-1$ Givens rotations

$$Q^T \cdot H = R \qquad \text{upper triangular}$$
$$\| $$
$$(G_{n-1}^T \cdots G_1^T) \cdot H$$

with $G_i =$ Givens $(i, i+1, \theta_i)$. Then

$$H_+ = R \cdot Q = R \cdot (G_1 \cdots G_{n-1})$$

This requires $6n^2 + O(n)$ flops.

IX.5 Deflation [GvL, § 7.5.1]

A Hessenberg matrix $H \in \mathbb{R}^{n\times n}$ is reduced if it has a zero subdiagonal entry. In this case

$$H = \begin{array}{cc} \begin{pmatrix} H_{11} & H_{12} \\ 0 & H_{22} \end{pmatrix} & \begin{array}{c} p \\ n-p \end{array} \\ \begin{array}{cc} p & n-p \end{array} & \end{array} \qquad 1 \leq p < n$$

and the problem decouples into two smaller problems H_{11} and H_{22} (deflation). Typically occurs when $p = n-1$ or $p = n-2$.

In practice, this is done when a subdiagonal entry is sufficiently small, e.g. if

$$|h_{p+1,p}| \leq c \cdot \varepsilon \cdot (|h_{p,p}| + |h_{p+1,p+1}|)$$

for a small constant c. machine epsilon

IX.6 The shifted QR iteration [GvL, § 7.5.2 and § 7.5.3]

For $\mu \in \mathbb{R}$ consider the iteration

$H \leftarrow Q_0^T \cdot A \cdot Q_0$ Hessenberg reduction

for $k = 1, 2, \ldots$

$H - \mu \mathbb{1}_n = Q \cdot R$ QR factorization

$H \leftarrow R \cdot Q + \mu \mathbb{1}_n$

Each H is Hessenberg and orthogonally similar to A:

$$\tilde{H} = R \cdot Q + \mu \mathbb{1}_n = Q^T (QR + \mu \mathbb{1}_n) Q = Q^T \cdot H \cdot Q$$

If we shift by an eigenvalue $\mu \in \lambda(A)$, deflation occurs in <u>one step</u>:

[GvL, Theorem 7.5.1]

$$\tilde{h}_{n,n-1} = 0 \quad \text{and} \quad \tilde{h}_{nn} = \lambda$$

<u>In practice</u>, we recognize that the QR iteration is converging when

$$h_{n,n-1} \text{ is small}$$

Then we use $\mu = h_{nn}$, which gives a <u>quadratically converging algorithm</u>: if

$$|h_{n,n-1}| \leq \eta \|H\| \quad \text{with } 0 \leq \eta \ll 1$$

then for $\mu = h_{n,n}$ we have that

$$|\tilde{h}_{n,n-1}| \leq \mathcal{O}(\eta^2 \|\tilde{H}\|)$$

the number of correct digits of $h_{nn} \approx \lambda \in \lambda(A)$ <u>doubles</u> at each step.

IX.7 The double shift strategy [GvL, § 7.5.4 & § 7.5.5]

The previous strategy ~~fails~~ when A has a complex (non real) eigenvalue $\lambda \in \mathbb{C} \setminus \mathbb{R}$.

In this case we might perform two single shifts QR steps in succession, using

λ and $\bar{\lambda}$ as shifts:

$$H - \lambda \mathbb{1}_n = Q_1 R_1$$
$$H_1 \leftarrow R_1 Q_1 + \lambda \cdot \mathbb{1}_n$$
$$H_1 - \bar{\lambda} \mathbb{1}_n = Q_2 R_2$$
$$H_2 \leftarrow R_2 Q_2 + \bar{\lambda} \mathbb{1}_n$$

Set $\qquad Q = Q_1 Q_2, \qquad R = R_2 R_1, \qquad M = QR$

Manipulating these equation it can be shown that

$$M = (H - \lambda \mathbb{1}_n)(H - \bar{\lambda} \mathbb{1}_n)$$
$$= H^2 - \underbrace{(\lambda + \bar{\lambda})}_{2\mathrm{Re}(\lambda)} H + \overbrace{\lambda \cdot \bar{\lambda}}^{|\lambda|^2} \mathbb{1}_n$$

and so M is real. It can also be shown that

$$H_2 = Q^T \cdot H \cdot Q \in \mathbb{R}^{n \times n}$$

To avoid complex arithmetics, we would like to pass from H to H_2 directly and using only real numbers.

The <u>direct</u> strategy would be:

- compute $M = H^2 - 2\mathrm{Re}(\lambda) + |\lambda|^2 \mathbb{1}_n$
- ——— the QR factorization $M = QR$
- set $H_2 = Q^T H Q$

The first step requires $\mathcal{O}(n^3)$ and so it is <u>not</u> practical.

Fortunately, there is a strategy for computing this double shift in <u>$\mathcal{O}(n^2)$ flops</u>: the <u>implicit double shift</u>, or <u>Francis QR step</u>:

- Compute $M \cdot e_1$ (1st column of M)
- Determine a Householder matrix P_0 s.t.

$$P_0 \cdot M e_1 = \text{multiple of } e_1$$

- Compute Householder matrices $P_1, \dots, P_{n-2}$ s.t. if

$$Z = P_0 P_1 \cdots P_{n-2}$$

then $Z \cdot H \cdot Z$ is upper Hessenberg and the 1st columns of Q and Z are equal.

This is based on the $\boxed{\text{implicit QR thm}}$: if

$$Q^T A Q = H \quad \text{and} \quad Z^T A Z = G$$

are Hessenberg, H is unreduced and Q and Z have the same first column, then

$$G = D^{-1} H D \quad \text{with } D = \mathrm{diag}(\pm 1, \dots, \pm 1)$$

Hence if $Q^T H Q$ and $Z^T H Z$ are unreduced, they are essentially equal. Else we can decouple the problem into smaller unreduced subproblems.

In detail:

$$Me_1 = \begin{pmatrix} x \\ y \\ z \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad \text{with} \quad \begin{aligned} x &= h_{11}^2 + h_{12} h_{21} - \underbrace{s}_{2\,\mathrm{Re}(\lambda)} h_{11} + \underbrace{t}_{|\lambda|^2} \\ y &= h_{21}(h_{11} + h_{22} - s) \\ z &= h_{21} h_{32} \end{aligned}$$

The computation of Me_1 and P_0 takes $O(1)$ flops. A similarity with P_0 only changes rows and columns 1, 2 and 3:

$$P_0^T H P_0 = \begin{pmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{pmatrix}$$

The mission of $P_1, \dots, P_{n-2}$ is to restore this matrix to Hessenberg form:

$$\begin{pmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \otimes & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{pmatrix} \xrightarrow{P_1} \begin{pmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \otimes & \otimes & \times & \times & \times \\ 0 & \otimes & \otimes & 0 & \times & \times \end{pmatrix} \xrightarrow{P_2} \begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \end{pmatrix}$$

etc

We have that

$$P_0 \text{ and } \quad Z = P_0 P_1 \cdots P_{n-2}$$

have the same 1st column and so

$$Z e_1 = Q e_1$$

and so Z and Q coincide (up to signs).
To implement this strategy, we check when

$$h_{n-1,n-2} \text{ is small}$$

Then the eigenvalues of

$$\begin{pmatrix} h_{n-1,n-1} & h_{n-1,n} \\ h_{n,n-1} & h_{n,n} \end{pmatrix}$$

approximate eigenvalues of A. In this case we set

$$\begin{cases} s = h_{n-1,n-1} + h_{n,n} & \text{(trace)} \\ t = h_{n-1,n-1} \cdot h_{n,n} - h_{n-1,n} h_{n,n-1} & \text{(determinant)} \end{cases}$$

and gives <u>quadratic</u> convergence: on average, two QR iterations are needed per eigenvalue, and the overall process costs $O(n^3)$ flops