Numerical Methods I

MATH-GA 2010.001/CSCI-GA 2420.001

Benjamin Peherstorfer
Courant Institute, NYU

Based on slides by G. Stadler and A. Donev

Today

Last time

- Linear least square problems
- Geometric perspective on the normal equations

Today

- Orthogonalization with Gram-Schmidt
- QR decomposition

Announcements

- Homework 3 has been posted, due Mon, Oct 24 before class

Recap: Least-squares problems

Choosing the least square error, this results in

$$\min_{\boldsymbol{x}} \|A\boldsymbol{x} - \boldsymbol{b}\|^2,$$

where $\boldsymbol{x} = (x_1, \ldots, x_n)^T$, $\boldsymbol{b} = (b_1, \ldots, b_m)^T$, and $a_{ij} = a_j(t_i)$.

In the following, we study the overdetermined case, i.e., $m \geq n$ and $\text{rank}(A) = n$

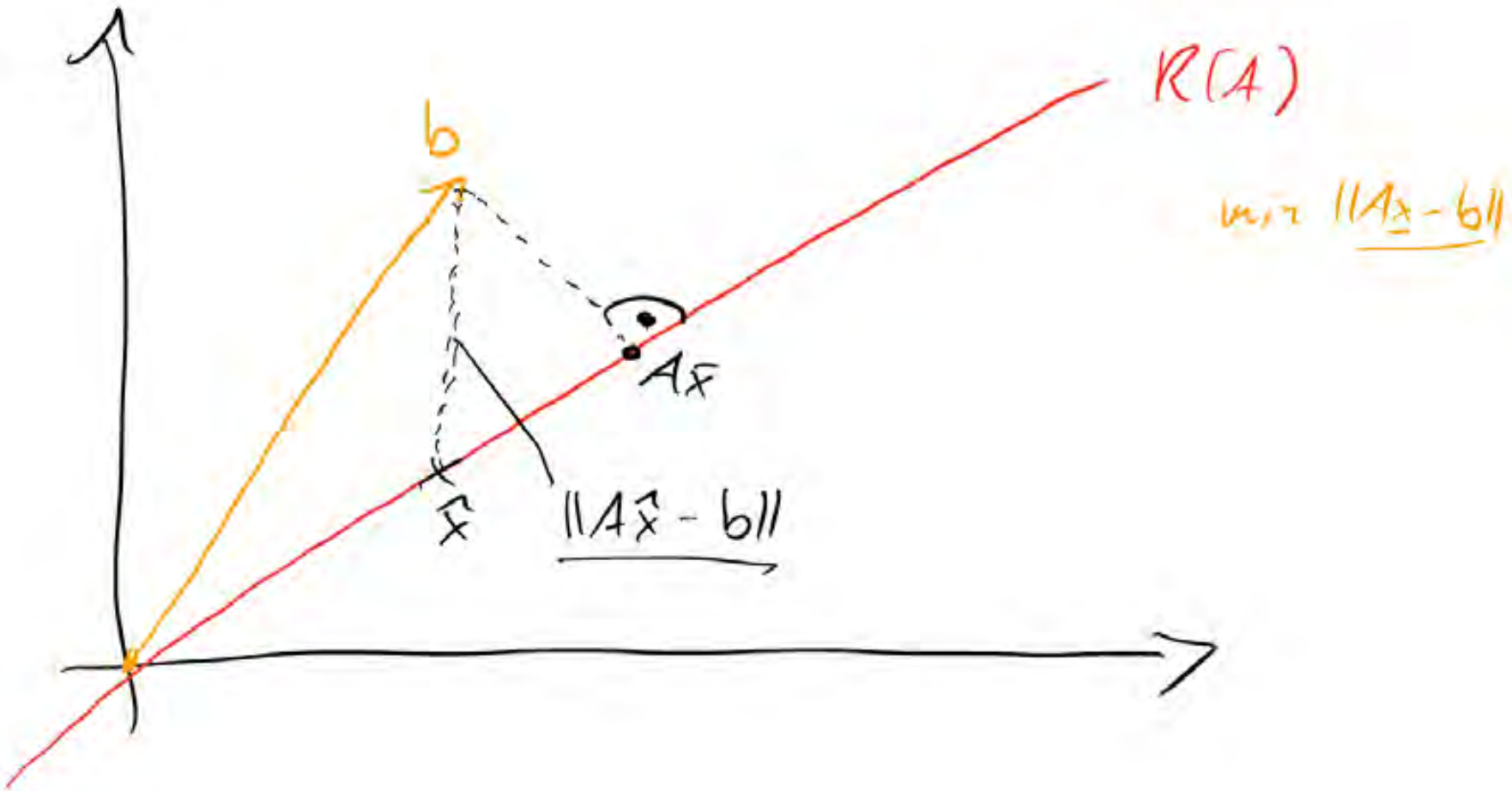
Solving the normal equations

$$A^T A \bar{\boldsymbol{x}} = A^T \boldsymbol{b}$$

requires:

- computing $A^T A$ (which is $O(mn^2)$)
- condition number of $A^T A$ is square of condition number of A; (problematic for the Choleski factorization)

Recap: Least-squares problems



Recap: Linear least-squares problems

Now for the least-squares problem $\|\boldsymbol{Ax} - \boldsymbol{b}\|_2$. The relative condition number κ in the Euclidean norm is bounded by

- With respect to perturbations in $\boldsymbol{b}$:

$$\kappa \leq \frac{\kappa_2(A)}{\cos(\theta)}$$

- With respect to perturbations in $\boldsymbol{A}$:

$$\kappa \leq \kappa_2(A) + \kappa_2(A)^2 \tan(\theta)$$

Small residual problems, small angle θ $\cos(\theta) \approx 1$, $\tan(\theta) \approx 0$: behavior similar to linear system.

Large residual problems, large angle θ $\cos(\theta) \ll 1$, $\tan(\theta) \approx 1$: behavior very different from linear system because $\kappa_2(A)^2$ shows up

The QR decomposition

Recall that projecting $\boldsymbol{b}$ onto the column span (range) of $\boldsymbol{A}$ was the key step $\rightsquigarrow$ let's try to find a numerical method that computes an orthonormal basis $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$ of the rank-n column span of $\boldsymbol{A}$

$$\boldsymbol{A} = \begin{bmatrix} | & & | \\ \boldsymbol{a}_1 & \ldots & \boldsymbol{a}_n \\ | & & | \end{bmatrix} \in \mathbb{R}^{m \times n}, \qquad m \geq n$$

$$\Downarrow$$

$$\underbrace{\begin{bmatrix} | & & | \\ \boldsymbol{a}_1 & \ldots & \boldsymbol{a}_n \\ | & & | \end{bmatrix}}_{\boldsymbol{A}} = \underbrace{\begin{bmatrix} | & & | \\ \boldsymbol{q}_1 & \ldots & \boldsymbol{q}_n \\ | & & | \end{bmatrix}}_{\boldsymbol{Q}} \underbrace{\begin{bmatrix} r_{11} & r_{12} & \ldots & r_{1n} \\ & r_{22} & & \vdots \\ & & \ddots & \\ & & & r_{nn} \end{bmatrix}}_{\boldsymbol{R}}$$

with an invertible matrix $\boldsymbol{R}$ so that

$$\operatorname{span}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_k) = \operatorname{span}(\boldsymbol{q}_1, \ldots, \boldsymbol{q}_k), \qquad k = 1, \ldots, n$$

$$\underbrace{\begin{bmatrix} | & & | \\ \boldsymbol{a}_1 & \dots & \boldsymbol{a}_n \\ | & & | \end{bmatrix}}_{\boldsymbol{A}} = \underbrace{\begin{bmatrix} | & & | \\ \boldsymbol{q}_1 & \dots & \boldsymbol{q}_n \\ | & & | \end{bmatrix}}_{\boldsymbol{Q}} \underbrace{\begin{bmatrix} r_{11} & r_{12} & \dots & r_{1n} \\ & r_{22} & & \dots \\ & & \ddots & \\ & & & r_{nn} \end{bmatrix}}_{\boldsymbol{R}}$$

$\Downarrow$ leads to system of equations $\Downarrow$

$$\begin{aligned}
\boldsymbol{a}_1 &= r_{11}\boldsymbol{q}_1 \\
\boldsymbol{a}_2 &= r_{12}\boldsymbol{q}_1 + r_{22}\boldsymbol{q}_2 \\
\boldsymbol{a}_3 &= r_{13}\boldsymbol{q}_1 + r_{23}\boldsymbol{q}_2 + r_{33}\boldsymbol{q}_3 \\
&\vdots \\
\boldsymbol{a}_n &= r_{1n}\boldsymbol{q}_1 + r_{2n}\boldsymbol{q}_2 + \cdots + r_{nn}\boldsymbol{q}_n
\end{aligned}$$

This motivates a process for computing the basis $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$

- At step j, we have $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_{j-1}$ that span $\text{span}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{j-1})$
- We want to find $\boldsymbol{q}_j$ orthonormal to $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_{j-1}$ so that $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_j$ spans $\text{span}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_j)$
- Thus, set

$$\boldsymbol{v}_j = \boldsymbol{a}_j - (\boldsymbol{q}_1^T \boldsymbol{a}_j)\boldsymbol{q}_1 - (\boldsymbol{q}_2^T \boldsymbol{a}_j)\boldsymbol{q}_2 - \cdots - (\boldsymbol{q}_{j-1}^T \boldsymbol{a}_j)\boldsymbol{q}_{j-1}$$

  and normalize

$$\boldsymbol{q}_j = \frac{\boldsymbol{v}_j}{\|\boldsymbol{v}_j\|_2}$$

Notice that at step j, the quantities $\boldsymbol{q}_1^T \boldsymbol{a}_j, \boldsymbol{q}_2^T \boldsymbol{a}_j, \ldots, \boldsymbol{q}_{j-1}^T \boldsymbol{a}_j$ are the values $r_{j,1}, \ldots, r_{j,j-1}$ and r_{jj} is responsible for the normalization and set to

$$r_{jj} = \|a_j - \sum_{i=1}^{j-1} r_{ij} q_i\|_2$$

This process is the *classical Gram-Schmidt* procedure to compute the QR factorization; however, this process is numerically unstable!

Instead of directly computing

$$\boldsymbol{v}_j = \boldsymbol{a}_j - (\boldsymbol{q}_1^T \boldsymbol{a}_j)\boldsymbol{q}_1 - (\boldsymbol{q}_2^T \boldsymbol{a}_j)\boldsymbol{q}_2 - \cdots - (\boldsymbol{q}_{j-1}^T \boldsymbol{a}_j)\boldsymbol{q}_{j-1}$$

based on $\boldsymbol{a}_j$, the *modified* Gram-Schmidt procedure computes $\boldsymbol{v}_j$ iteratively

$$\begin{aligned}
\boldsymbol{v}_j^{(1)} &= \boldsymbol{a}_j, \\
\boldsymbol{v}_j^{(2)} &= \boldsymbol{v}_j^{(1)} - \boldsymbol{q}_1 \boldsymbol{q}_1^T \boldsymbol{v}_j^{(1)}, && \text{"subtract from } \boldsymbol{v}_j^{(1)} \text{ what is already in } \boldsymbol{q}_1\text{"} \\
\boldsymbol{v}_j^{(3)} &= \boldsymbol{v}_j^{(2)} - \boldsymbol{q}_2 \boldsymbol{q}_2^T \boldsymbol{v}_j^{(2)}, && \text{"subtract from } \boldsymbol{v}_j^{(2)} \text{ what is already in } \boldsymbol{q}_2\text{"} \\
&\vdots \\
\boldsymbol{v}_j &= \boldsymbol{v}_j^{(j)} = \boldsymbol{v}_j^{(j-1)} - \boldsymbol{q}_{j-1} \boldsymbol{q}_{j-1}^T \boldsymbol{v}_j^{(j-1)}
\end{aligned}$$

Computing a QR factorization with the modified Gram-Schmidt procedure is stabler than with the classical Gram-Schmidt procedure. However, even the modified Gram-Schmidt procedure can lead to vectors $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$ that are far from orthogonal if the condition number of $\boldsymbol{A}$ is large (see, Golub et al., Matrix Computations, Section 5.2.9)

Let's recall what the Gram-Schmidt procedure is doing: It is applying a succession of triangular matrices R_k on the right of A so that the resulting matrix

$$\boldsymbol{A}\underbrace{\boldsymbol{R}_1\boldsymbol{R}_2\ldots\boldsymbol{R}_n}_{\boldsymbol{R}^{-1}} = \boldsymbol{Q}$$

has orthonormal columns and $\boldsymbol{R}$ is upper-triangular.

Instead, we could try to find orthonormal matrices ($\boldsymbol{X}^T\boldsymbol{X} = \boldsymbol{X}\boldsymbol{X}^T = \boldsymbol{I}$) so that

$$\underbrace{\boldsymbol{Q}_n\ldots\boldsymbol{Q}_2\boldsymbol{Q}_1}_{\boldsymbol{Q}}\boldsymbol{A} = \boldsymbol{R}$$

is upper-triangular. The product $\boldsymbol{Q}_n\ldots\boldsymbol{Q}_2\boldsymbol{Q}_1 = \boldsymbol{Q}^T$ is orthonormal too and thus $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$ a QR factorization of $\boldsymbol{A}$.

The Householder method judiciously finds the matrices $\boldsymbol{Q}_1, \boldsymbol{Q}_2, \ldots, \boldsymbol{Q}_n$ via so-called Householder reflectors $\rightsquigarrow$ board. The Householder method is backward stable.

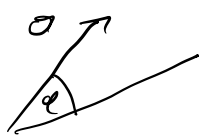
Transform

$$A \rightsquigarrow Q_1 A \rightsquigarrow Q_2 Q_1 A \rightsquigarrow \dots$$

$k_2(Q) = 1$, $|\det(Q)| = 1$

In $\mathbb{R}^2$; two orthogonal transformations

rotation : $\det = 1$

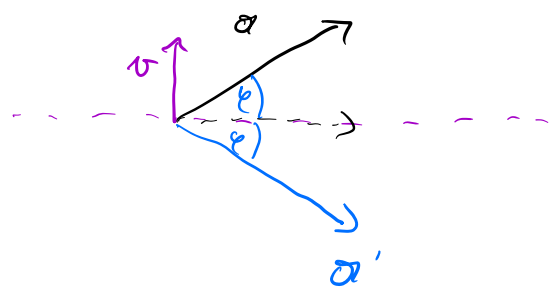


$$\begin{bmatrix} \cos(\varphi) & \sin(\varphi) \\ -\sin(\varphi) & \cos(\varphi) \end{bmatrix}$$

$\Rightarrow$ Givens rotations

$\Rightarrow$ text books

reflection $\det = -1$



projection

$$a \mapsto \left(I - \frac{vv^T}{v^Tv}\right) a$$

reflection

$$a \mapsto \left(I - 2\frac{vv^T}{v^Tv}\right) a$$

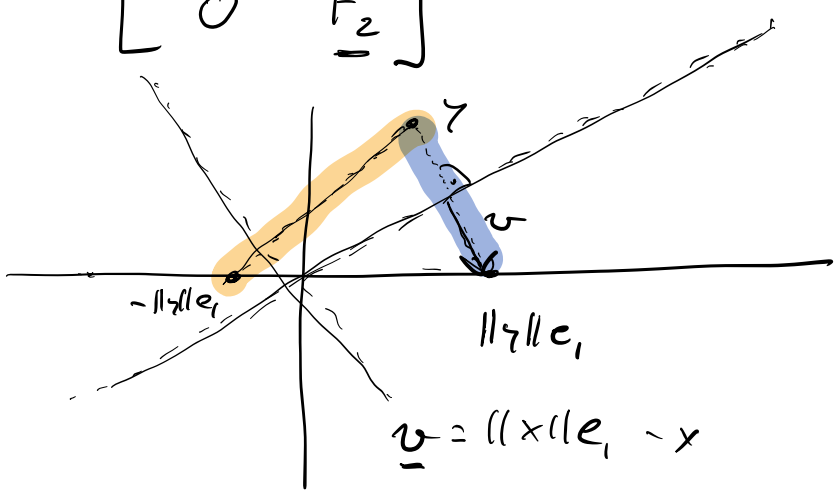
$\Rightarrow$ Householder reflections

$$\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{bmatrix} \xrightarrow[Q_1]{k=1} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow[Q_2]{k=2} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & x \\ 0 & 0 & x \end{bmatrix}$$

$$y = \begin{bmatrix} x \\ x \\ x \\ x \end{bmatrix} \xrightarrow[\underline{F}]{} \begin{bmatrix} x \\ 0 \\ 0 \\ 0 \end{bmatrix} = x\, e_1$$

find

$$Q_2 = \begin{bmatrix} I_{k-1} & 0 \\ 0 & \underline{F}_2 \end{bmatrix}$$



$$\underline{v} = \|x\| e_1 - x$$

$$F_2 = \left(I - 2\frac{v\, v^T}{v^T v}\right)$$

Costs of Householder reflection

$$\underline{2mn^2 - \frac{2}{3}n^3} \quad \text{FLOPs}$$

to get R

Additionally get Q $\quad O(mn)$

$\Rightarrow \quad O(mn^2)$

$$A = QR = \begin{array}{c} \\ n \end{array} \overset{n}{\left[\quad\right]} \left[\quad\right]$$

$$\underline{Rx} = \underline{Q^T b}$$

The QR factorization

All these three algorithms (classical Gram-Schmidt, modified Gram-Schmidt, Householder triangularization) have roughly the FLOPs of $2mn^2$ for an $m \times n$ matrix

Why would we ever want to use (modified) Gram-Schmidt instead of Householder triangularization?

The QR factorization

All these three algorithms (classical Gram-Schmidt, modified Gram-Schmidt, Householder triangularization) have roughly the FLOPs of $2mn^2$ for an $m \times n$ matrix

Why would we ever want to use (modified) Gram-Schmidt instead of Householder triangularization? Gram-Schmidt can be easier to parallelize, for example (Recall that best algorithm depends also on what hardware we want to implement it on.)

Every matrix $A \in \mathbb{R}^{m \times n}$ with $m \geq n$ has a QR factorization. It is unique if we require the diagonal elements of R to be positive.

If $m > n$ and $\boldsymbol{Q} \in \mathbb{R}^{m \times n}$, then we speak of a reduced QR factorization. Otherwise, we have $\boldsymbol{Q} \in \mathbb{R}^{m \times m}$ and we speak of a full QR factorization.

```
>> A = randn(10, 10); [Q, R] = qr(A);
>> size(Q)
ans =
    10    10
>> size(R)
ans =
    10    10
```

```
>> A = randn(10, 4); [Q, R] = qr(A)
>> size(Q)
ans =
    10    10
>> size(R)
ans =
    10     4
>>
>> [Q, R] = qr(A, 0); % reduced QR
>> size(Q)
ans =
    10     4
>> size(R)
ans =
     4     4
```

Back to our least-squares problem

One would like to avoid the multiplication $A^T A$ and use a suitable factorization of A that avoids solving the normal equation directly:

$$A = QR = [Q_1, Q_2] \begin{bmatrix} R_1 \\ \hline 0 \end{bmatrix} = Q_1 R_1,$$

where $Q \in \mathbb{R}^{m \times m}$ is an orthonormal matrix ($QQ^T = I$), and $R \in \mathbb{R}^{m \times n}$ consists of an upper triangular matrix and a block of zeros.

How can the QR factorization be used to solve the least-squares problem?

$$\min_{\boldsymbol{x}} \|A\boldsymbol{x} - \boldsymbol{b}\|^2 = \min_{\boldsymbol{x}} \|Q^T(A\boldsymbol{x} - \boldsymbol{b})\|^2 \qquad = \min_{\boldsymbol{x}} \left\| \begin{bmatrix} \boldsymbol{b}_1 - R_1\boldsymbol{x} \\ \boldsymbol{b}_2 \end{bmatrix} \right\|^2,$$

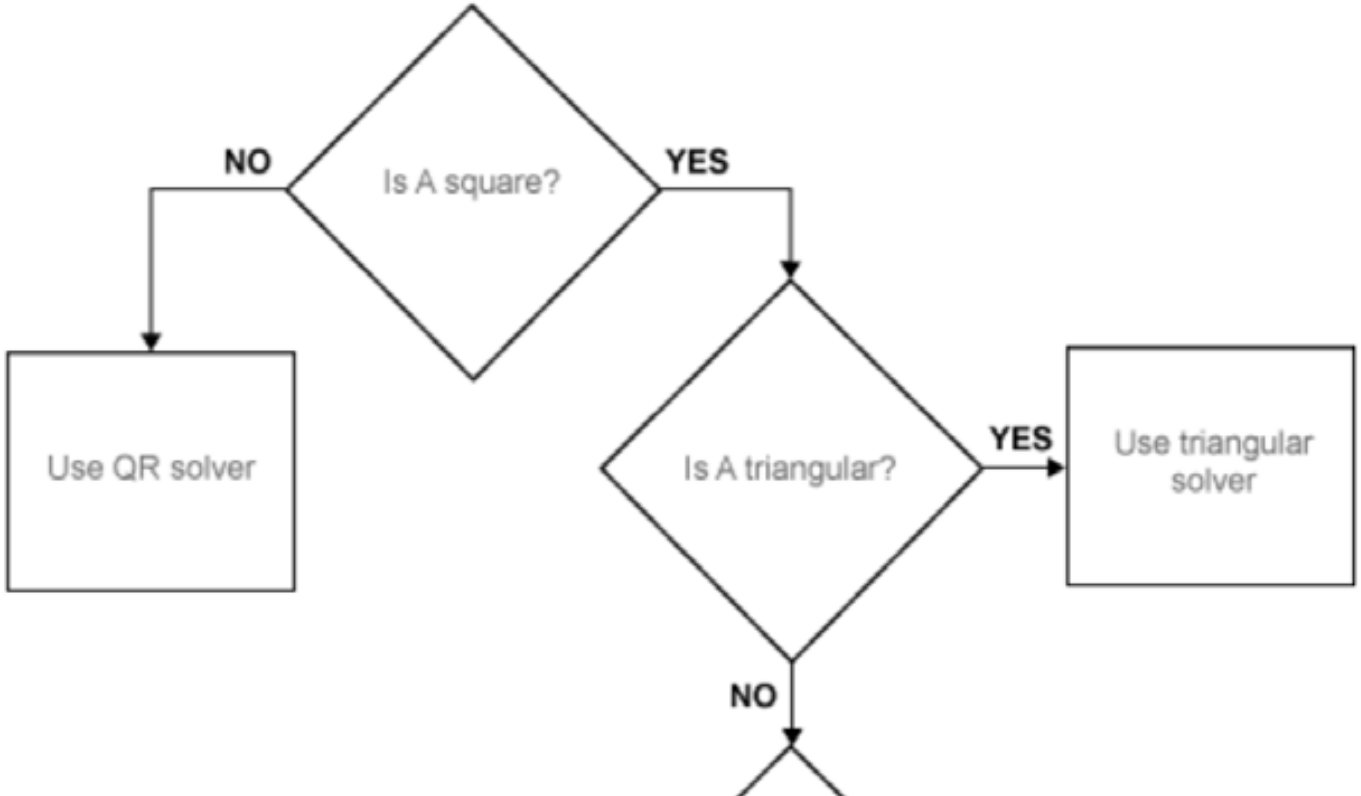
$$= \min_{\boldsymbol{x}} \|\boldsymbol{b}_1 - R_1\boldsymbol{x}\|^2 + \|\boldsymbol{b}_2\|^2$$

where $Q^T \boldsymbol{b} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{bmatrix}$.

Thus, the least squares solution is $\boldsymbol{x} = R^{-1}\boldsymbol{b}_1$ and the residual is $\|\boldsymbol{b}_2\|$.

Stability of solving least-squares problem with Householder triangularization

Solving a least-squares problem with $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, $m \geq n$ and $\text{rank}(\boldsymbol{A}) = n$ via QR factorization computed with Householder triangularization is backward stable.



Eigen decomposition

Eigen decomposition

- For a square matrix $\boldsymbol{A} \in \mathbb{C}^{n \times n}$, there exists at least one λ such that

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x} \implies (\boldsymbol{A} - \lambda \boldsymbol{I})\boldsymbol{x} = 0$$

- Putting the eigenvectors $\boldsymbol{x}_j$ as columns in a matrix $\boldsymbol{X}$, and the eigenvalues λ_j on the diagonal of a diagonal matrix $\boldsymbol{\Lambda}$, we get

$$\boldsymbol{A}\boldsymbol{X} = \boldsymbol{X}\boldsymbol{\Lambda}$$

- A matrix is non-defective or diagonalizable if there exist n linearly independent eigenvectors, which means that $\boldsymbol{X}$ is invertible

$$\begin{aligned} \boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X} &= \boldsymbol{\Lambda} \\ \boldsymbol{A} &= \boldsymbol{X}\boldsymbol{\Lambda}\boldsymbol{X}^{-1} \end{aligned}$$

- The transformation from $\boldsymbol{A}$ to $\boldsymbol{\Lambda} = \boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X}$ is called a similarity transformation and it preserves the eigenvalues.

- A matrix is unitarily diagonalizable if there exist n linearly independent orthogonal eigenvectors, i.e., if the matrix $\boldsymbol{X}$ can be chosen to be unitary (orthonormal), $\boldsymbol{X} = \boldsymbol{U}$, where $\underline{\boldsymbol{U}^{-1} = \boldsymbol{U}^H}$

$$\underline{\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^H}$$

  Note that unitary matrices generalize orthogonal matrices to the complex domain, so we use adjoints (conjugate transpose) instead of transpose throughout

- Theorem: A matrix is unitarily diagonlizable iff it is normal, i.e., it commutes with its adjoint:

$$\boldsymbol{A}^H\boldsymbol{A} = \boldsymbol{A}\boldsymbol{A}^H$$

- Theorem: Hermitian (symmetric) matrices, $\boldsymbol{A}^H = \boldsymbol{A}$, are unitarily diagonalizable and have *real* eigenvalues.

- The usual eigenvectors are more precisely called *right* eigenvectors. There are also *left* eigenvectors corresponding to a given eigenvalue λ

$$\underbrace{\boldsymbol{y}^H \boldsymbol{A} = \lambda \boldsymbol{y}^H}_{\boldsymbol{Y}^H \boldsymbol{A} = \boldsymbol{\Lambda} \boldsymbol{Y}^H} \implies \underbrace{\boldsymbol{A}^H \boldsymbol{y} = \bar{\lambda} \boldsymbol{y}}_{} \,,$$

  with conjugate $\bar{\lambda}$ of λ

- For a matrix that is diagonalizable, observe that

$$\boldsymbol{Y}^H = \boldsymbol{X}^{-1}$$

  and so the left eigenvectors provide no new information

- For unitarily diagonalizable matrices, $\boldsymbol{Y} = (\boldsymbol{X}^{-1})^H = (\boldsymbol{X}^H)^H = \boldsymbol{X} = \boldsymbol{U}$, so that the left and right eigenvectors coincide.

Numerically finding eigenvalues

For a matrix $A \in \mathbb{C}^{n \times n}$ (potentially real), we want to find $\lambda \in \mathbb{C}$ and $\boldsymbol{x} \neq 0$ such that

$$A\boldsymbol{x} = \lambda \boldsymbol{x}.$$

Most relevant problems:

- A symmetric (and large)
- A spd (and large)
- A stochastic matrix, i.e., all entries $0 \leq a_{ij} \leq 1$ are probabilities, and thus $\sum_j a_{ij} = 1$.

How hard are they to find numerically?

How hard are they to find numerically?

- This is a nonlinear problem.

How hard are they to find numerically?

- This is a nonlinear problem.
- How difficult is this?

How hard are they to find numerically?

- This is a nonlinear problem.

- How difficult is this? Eigenvalues are the roots of the characteristic polynomial. Also, any polynomial is the characteristic polynomial of a matrix $\rightsquigarrow$ For matrices larger than 4×4, eigenvalues cannot be computed in closed form (Abel's theorem).

How hard are they to find numerically?

- This is a nonlinear problem.

- How difficult is this? Eigenvalues are the roots of the characteristic polynomial. Also, any polynomial is the characteristic polynomial of a matrix $\rightsquigarrow$ For matrices larger than 4×4, eigenvalues cannot be computed in closed form (Abel's theorem).

- Must use an iterative algorithm

How hard are they to find numerically?

- This is a nonlinear problem.

- How difficult is this? Eigenvalues are the roots of the characteristic polynomial. Also, any polynomial is the characteristic polynomial of a matrix $\rightsquigarrow$ For matrices larger than 4×4, eigenvalues cannot be computed in closed form (Abel's theorem).

- Must use an iterative algorithm $\rightsquigarrow$ this is fundamentally different from what we have seen previously when solving systems of *linear* equations! These algorithms (LU, QR) give the *exact* solution in *exact* arithmetic in finite number of steps. We *cannot* expect something similar for computing eigenvalues!

Condition of finding eigenvalues of a matrix

The absolute condition number of determining a simple eigenvalue λ_0 of a matrix $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ with respect to the $\|\cdot\|_2$ is

$$\kappa_{\text{abs}} = \frac{1}{|\cos(\angle(\boldsymbol{x},\boldsymbol{y}))|}, \qquad \cos(\angle(\boldsymbol{x},\boldsymbol{y})) = \frac{|\langle \boldsymbol{x},\boldsymbol{y}\rangle|}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|}$$

and the relative condition number is

$$\kappa_{\text{rel}} = \frac{\|\boldsymbol{A}\|}{|\lambda_0 \cos(\angle(\boldsymbol{x},\boldsymbol{y}))|},$$

where $\boldsymbol{x}$ is an eigenvector of $\boldsymbol{A}$ for the eigenvalue λ_0 ($\boldsymbol{A}\boldsymbol{x} = \lambda_0\boldsymbol{x}$) and $\boldsymbol{y}$ an adjoint eigenvector ($\boldsymbol{A}^H\boldsymbol{y} = \bar{\lambda}_0\boldsymbol{y}$).

Sketch of proof $\rightsquigarrow$ board

(see also Deuflhard, Theorem 5.2)

$Ax = \lambda_0 x$, λ_0 is simple

$\ulcorner A \in \mathbb{C}^{n\times n}$, n eigenvalues , $d \le n$ are distinct

$$\det(A - \lambda I) = \prod_{i=1}^{n} (\lambda_i - \lambda) = \prod_{i=1}^{d} (\lambda_i' - \lambda)^{\mu(\lambda_i')}$$

$\mu(\lambda_i')$ ... algebraic multiplicity

Geometric multiplicity $\gamma(\lambda_i)$: # linearly indep eigenvectors associated with eigenvalue λ_i

$$1 \le \gamma(\lambda_i) \le \mu(\lambda_i) \le n$$

Simple means

$$\mu(\lambda_0) = 1$$

$\lrcorner$

We want to show

$$\lambda : A \longmapsto \lambda(A)$$

exists, cont. diff. in neighborhood of A

$$\lambda(A) = \lambda_0$$

$$\lambda'(A)\,C = \frac{\langle C x_0, y_0 \rangle}{\langle x_0, y_0 \rangle}$$

x_0 is eigenvector of A with λ_0

y_0 is eigenvector of A^H with $\bar{\lambda}_0$

$A + tC$ $\qquad$ t ... small

$C \in \mathbb{C}^{n \times n}$

Implicit function theorem

$$F(x, \lambda) = Ax - \lambda x$$

$$\frac{\partial}{\partial \lambda} F(x, \lambda) = -x$$

$$\frac{\partial}{\partial x} F(x, \lambda) = A - \lambda I$$

$$J_F(x, \lambda) = [x \mid A - \lambda I] \in \mathbb{C}^{n \times (n+1)}$$

" geo. multi $\leq$ alg. multi " λ_0 simple

$$\ker(A - \lambda_0 I) = \underline{\operatorname{span}\{x_0\}}$$

$$\Rightarrow \operatorname{rank}(A - \lambda_0 I) = \underline{n-1}$$

$$\boxed{x_0 \notin \operatorname{colspan} A - \lambda_0 I}$$

$$J_F = [x_0 \mid A - \lambda_0 I]$$

$$\operatorname{rank}(J_F) = \underline{\underline{n}}$$

$\Rightarrow$ full rank J_F

$\Rightarrow$ apply implicit function theorem

$$A + tC$$

$$\lambda : A + tC \mapsto \lambda(A + tC)$$

$$\lambda(0) = \lambda_0$$

λ is C^1 over small $(-\varepsilon, \varepsilon)$

$\underline{\lambda'(A)} : \underline{C} \mapsto \dfrac{\langle Cx_0, y_0 \rangle}{\langle x_0, y_0 \rangle}$

$$\|\lambda'(A)\| = \sup_{C \neq 0} \frac{|\langle Cx_0, y_0 \rangle / \langle x_0, y_0 \rangle|}{\|C\|}$$

$$= \dots = \frac{1}{|\cos(\measuredangle(x_0, y_0))|} = k_{abs}$$

rel.

$$k_{rel} = \frac{\|A\|}{\|\lambda\|} \frac{1}{|\cos(\measuredangle(x_0, y_0))|}$$

Interpretation

Perturbations of order δ in entries of matrix $\boldsymbol{A}$ induce changes of the order $\delta\lambda = \underline{\delta / \cos(\angle(\boldsymbol{x}_0, \boldsymbol{y}_0))}$

In particular, for normal matrices* ($\boldsymbol{A}\boldsymbol{A}^H = \boldsymbol{A}^H\boldsymbol{A}$), we have $\underline{\boldsymbol{x}_0 = \boldsymbol{y}_0}$ and thus $\angle(\boldsymbol{x}_0, \boldsymbol{y}_0) = 0$ and thus $\underline{\cos(\angle(\boldsymbol{x}_0, \boldsymbol{y}_0)) =} 1$, which means $\underline{\kappa_{\text{abs}} = 1}$, which can be considered well conditioned

Finding non-simple eigenvalues can have very high absolute condition number (but can still be done numerically). For a detailed treatment have a look at textbook by Golub et al. on Matrix Computations.

*Equivalent: Have orthonormal eigenbasis of $\mathbb{C}$; diagonalizable by unitary matrix.

Bounding error in eigenvalue computation

Let $\boldsymbol{A} \in \mathbb{C}^{n \times n}$ be a Hermitian matrix and let $(\hat{\lambda}, \hat{\boldsymbol{x}})$ be a computed approximation of an eigenvalue/eigenvector pair $(\lambda, \boldsymbol{x})$ of $\boldsymbol{A}$. Defining the residual

$$\hat{\boldsymbol{r}} = \boldsymbol{A}\hat{\boldsymbol{x}} - \hat{\lambda}\hat{\boldsymbol{x}}\,, \qquad \hat{\boldsymbol{x}} \neq \boldsymbol{0}\,,$$

it then follows that

$$\min_{\lambda_i \in \sigma(\boldsymbol{A})} |\hat{\lambda} - \lambda_i| \leq \frac{\|\hat{\boldsymbol{r}}\|_2}{\|\hat{\boldsymbol{x}}\|_2}\,,$$

where $\sigma(\boldsymbol{A}) = \{\lambda | \lambda \text{ is an eigenvalue of } \boldsymbol{A}\}$ is the spectrum of $\boldsymbol{A}$.

Proof $\rightsquigarrow$ board

What is special about this bound?

Bounding error in eigenvalue computation

Let $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ be a Hermitian matrix and let $(\hat{\lambda}, \hat{\boldsymbol{x}})$ be a computed approximation of an eigenvalue/eigenvector pair $(\lambda, \boldsymbol{x})$ of $\boldsymbol{A}$. Defining the residual

$$\hat{\boldsymbol{r}} = \boldsymbol{A}\hat{\boldsymbol{x}} - \hat{\lambda}\hat{\boldsymbol{x}}\,, \qquad \hat{\boldsymbol{x}} \neq \boldsymbol{0}\,,$$

it then follows that

$$\min_{\lambda_i \in \sigma(\boldsymbol{A})} |\hat{\lambda} - \lambda_i| \leq \frac{\|\hat{\boldsymbol{r}}\|_2}{\|\hat{\boldsymbol{x}}\|_2}\,,$$

where $\sigma(\boldsymbol{A}) = \{\lambda | \lambda \text{ is an eigenvalue of } \boldsymbol{A}\}$ is the spectrum of $\boldsymbol{A}$.

Proof $\rightsquigarrow$ board

What is special about this bound?

- This is an *a posteriori* bound that bounds the error *after* we have computed the result
- We will see many more residual-based *a posteriori* bounds (broadly speaking: the residual is something we can compute, and if the problem is "well-behaved" then the norm of the residual is a reasonable bound of the norm of the error.)

exists orthogonal eigenbasis $\{v_i\}$

$$\boxed{\hat{x} = \sum_{i=1}^{n} \alpha_i v_i} \qquad \alpha_i = v_i^H \hat{x}$$

$$\hat{r} = A\hat{x} - \hat{\lambda}\hat{x} = A\left(\sum_{i=1}^{n} \alpha_i v_i\right) - \hat{\lambda} \sum_{i=1}^{n} \alpha_i v_i$$

$$= \sum \alpha_i \lambda_i v_i - \hat{\lambda} \sum \alpha_i v_i$$

$$= \sum \alpha_i (\lambda_i - \hat{\lambda}) v_i$$

$$\frac{\|\hat{r}\|^2}{\|\hat{x}\|^2} = \frac{\sum |\alpha_i|^2 (\lambda_i - \hat{\lambda})^2 \|v_i\|^2}{\sum |\alpha_i|^2 \|v_i\|^2}$$

$$\|v_i\|^2 = 1 \quad = \boxed{\sum_{i=1}^{n} \frac{|\alpha_i|^2}{\sum |\alpha_j|^2}} (\lambda_i - \hat{\lambda})^2$$

$$= \sum_{i=1}^{n} \beta_i (\lambda_i - \hat{\lambda})^2$$

$$\beta_i \geq 0 \quad , \quad \sum \beta_i = 1$$

$$\frac{\|r\|^2}{\|x\|^2} = \sum \beta_i (\lambda_i - \bar{\lambda})^2 \geq \underbrace{\sum \beta_i}_{=1} \min_j (\lambda_j - \bar{\lambda})^2$$

$$= \underline{\min_j (\lambda_j - \bar{\lambda})^2}$$

Condition of computing eigenvectors

- The condition of computing eigenvector $\boldsymbol{x}_i$ for an eigenvalue λ_i depends on the separation between the eigenvalues

$$\kappa = \frac{1}{\min_{i \neq j} |\lambda_i - \lambda_j|}$$

(Quarteroni et al., Section 5)

- Computing $\boldsymbol{x}_i$ can be ill-conditioned if some eigenvalue λ_j is "very close" to the eigenvalue λ_i

- This indicates that multiple eigenvalues require care. Even for Hermitian matrices eigen*vectors* can be hard to compute