Technische Universität Braunschweig

Institute for Numerical Analysis

Ramp Up Mathematics — Numerical Analysis Ramp Up for Data Science

Matthias Bollhöfer, SS 2024

Contents I

- **Basics**
  - Finite Precision Representation
  - Round off Errors
  - Error Propagation
- **Error Analysis**
  - Introduction
  - Conditioning
  - Forward and Backward Analysis
- **Linear Systems**
  - Matrix Norms
  - Condition when Solving Linear Systems
  - The *LU* Decomposition
  - The Cholesky Decomposition
  - The CG Method
- **Nonlinear Systems**
  - Newton's Method

Contents II

- **Linear Least Squares**
  - Linear Least Squares and Normal Equations
  - The *QR* Decomposition
  - Stable Computation of the *QR* Decomposition
- **The Singular Value Decomposition**
  - Basic Properties
  - The Key Property
- **Fourier Transformation**
  - Fourier Series and Discrete Fourier Transformation
  - Fast Fourier Transformation

Least Squares

- systems of equations $\vec{F}(\vec{x}) \approx 0$, where the number of equations m differs from the number of unknowns n
- surrogate problem: minimize $\|\vec{F}(\vec{x})\|$, *balancing problem*
- goal: determine $x_1, \ldots, x_n$ which are only implicitly available via a function f depending on $\vec{x}$ and possibly further parameters $\vec{z}$, i.e., $\vec{y} = f(\vec{z}, x_1, \ldots, x_n)$

Example 4.1 (Experiment for determining the acceleration due to gravity g)

- *equation $h = g\frac{t^2}{2}$, where h is the height of fall, t refers to the time, g is sought.*
- *$h = 44.12m$, 10 experiments, gauge t [sec].*

i	1	2	3	4	5	6	7	8	9	10
t	2.998	2.993	3.001	3.001	3.000	2.999	2.999	2.995	2.998	3.000

- *each measurement i yields a formula $h = g \cdot \frac{t_i^2}{2}$ for estimating g.*
- *all measurements are error prone, averaging is necessary*
- *one way is to minimize squares of the errors $r_i = \frac{t_i^2}{2} \cdot g - h$:*

$$\sum_{i=1}^{10} r_i^2 \stackrel{!}{=} \min .$$

Least Squares

- suppose we have m evaluations of f (e.g. by data or experiments)

$$b_i \approx f_i(x_1, \ldots, x_n) := f(z_i, x_1, \ldots, x_n), \ i = 1, \ldots, m,$$

  in order to determine $x_1, \ldots, x_n$
- we require the minimization of

$$\sum_{i=1}^{m} \left(f_i(x_1, \ldots, x_n) - b_i\right)^2 \qquad (LS)$$

- this approach is referred to as *method of least squares*.
- we use the squared sum since this expression is differentiable if f is differentiable
- we set

$$\vec{b} := \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix}, \ \vec{x} := \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \ \vec{F}(\vec{x}) := \begin{bmatrix} f_1(x_1, \ldots, x_n) \\ \vdots \\ f_m(x_1, \ldots, x_n) \end{bmatrix}.$$

- using vector notation we can rewrite (LS) as

$$\|\vec{F}(\vec{x}) - \vec{b}\|_2 \stackrel{!}{=} \min$$

Linear Least Squares

- an important special cases are *linear least squares*, when we have $\vec{F}(\vec{x}) = A\vec{x}$, $A \in \mathbb{R}^{m,n}$
- in this case we obtain

$$\|A\vec{x} - \vec{b}\|_2 \stackrel{!}{=} \min \qquad (LLS)$$

- For minimizing problem (LLS) we have the following necessary and sufficient condition

$$A^\top A\vec{x} = A^\top \vec{b}.$$

  These equations are called *normal equations*.

Theorem 4.1 (Linear Least Squares and Normal Equations)

There exists a solution $\vec{x}$ of the linear least squares problem $\|A\vec{x} - \vec{b}\|_2 \stackrel{!}{=} \min$. Every solution $\vec{x}$ also satisfies $A^\top A\vec{x} = A^\top \vec{b}$ and vice versa.

Normal Equations

Example 4.2 (Experiment for determining g (ctd.))

use linear least squares (LLS), to obtain an estimate for g. To do so, set

$$A := \qquad , \ \vec{b} := \qquad \Rightarrow \vec{r} =$$

goal:

$$\|\vec{r}\|_2 = \|A \cdot g - \vec{b}\|_2 \stackrel{!}{=} \min$$

here: normal equations $A^\top A \cdot g = A^\top \vec{b}$ are just scalar

$$202.0692g = 1983.2841$$

with solution $g \approx 9.8149$.

Normal Equations

recall: a symmetric matrix $M \in \mathbb{R}^{n,n}$ is called positive definite, whenever

$$\vec{x}^\top M\vec{x} > 0, \text{ for all } \vec{x} \in \mathbb{R}^n \setminus \{\vec{0}\}.$$

- If $m \geqslant n$ and $\operatorname{rank} A = n$, then $M = A^\top A$ is positive definite since

$$\vec{x}^\top M\vec{x} = \vec{x}^\top A^\top A\vec{x} = (A\vec{x})^\top (A\vec{x}) = \|A\vec{x}\|_2^2$$

- consequence
  1. $\vec{x}^\top M\vec{x} = \|A\vec{x}\|_2^2 \geqslant 0$
  2. $\vec{x}^\top M\vec{x} = \|A\vec{x}\|_2^2 = 0 \Rightarrow A\vec{x} = \vec{0}$
  3. since we have $\operatorname{rank} A = n$, $A\vec{x} = \vec{0}$ is only possible if $\vec{x} = \vec{0}$

Algorithm 4.1 (Blue Print Linear Least Squares via Normal Equations)

Input: $A \in \mathbb{R}^{m,n}$ *where* $\operatorname{rank} A = n$, $b \in \mathbb{R}^m$

Output: *Compute solution* $\vec{x} \in \mathbb{R}^n$ *of problem (LLS)*

1: $\vec{d} \leftarrow A^\top \vec{b}$, $M \leftarrow A^\top A$

2: *compute Cholesky decomposition* $M = GG^\top$ *of* M

3: *solve* $G\vec{y} = \vec{d}$ *and* $G^T\vec{x} = \vec{y}$ *by forward and back substation.*

computational costs of this approach are $\mathcal{O}((m+n)n^2)$

Condition Number

recall: $\kappa(M) = \|M^{-1}\| \cdot \|M\|$

Definition 4.1 (condition number for non-square matrices)

Let $A \in \mathbb{R}^{m,n}$ such that $\operatorname{rank} A = n$. *We define $\kappa(A)$ via*

$$\kappa(A) := \frac{\max_{\|\vec{x}\|=1} \|A\vec{x}\|}{\min_{\|\vec{x}\|=1} \|A\vec{x}\|}$$

- for square nonsingular matrices A this definition reduces to $\kappa(A) = \|A^{-1}\| \cdot \|A\|$
- one can show that for $A \in \mathbb{R}^{m,n}$ such that $\operatorname{rank} A = n$ we have $\kappa_2(A^\top A) = \kappa_2(A)^2$, i.e. the condition number is squared!

Condition Number

Example 4.3 (matrix with two linear independent columns)

$$A = \begin{bmatrix} 1 & 1 \\ 4 \cdot 10^{-8} & 0 \\ 0 & 4 \cdot 10^{-8} \end{bmatrix}$$

MATLAB:

```
>> A=[1 1; 4e-8 0; 0 4e-8]
>> cond(A)
   ans =
        3.5355e+07
```

Cholesky decomposition for normal equations

Example 4.4

consider the linear least squares problem, where we have

$$A = \begin{bmatrix} 1 & 1 \\ 4 \cdot 10^{-8} & 0 \\ 0 & 4 \cdot 10^{-8} \end{bmatrix}, \ \vec{b} = \begin{bmatrix} 2 \\ 4 \cdot 10^{-8} \\ 4 \cdot 10^{-8} \end{bmatrix}, \ \vec{x} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \ \kappa_2(A) = 3.5 \cdot 10^7.$$

using 15-digits arithmetics, $M = A^\top A = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ *were already singular,*

$$\tilde{\vec{x}} = \begin{bmatrix} 1.0769 \cdot 10^0 \\ 9.2308 \cdot 10^{-1} \end{bmatrix}$$

and therefore

$$\frac{\|\tilde{\vec{x}} - \vec{x}\|_2}{\|\vec{x}\|_2} \approx 7.6696 \cdot 10^{-2} \lessapprox 2 \cdot \underbrace{\epsilon}_{2.2204 \cdot 10^{-16}} \underbrace{\kappa_2(A)^2}_{1.3087 \cdot 10^{15}} = 5.8116 \cdot 10^{-1}.$$

QR Decomposition

- normal equations can lead to poor results
- alternative: *QR* decomposition
- idea of the *QR* decomposition: similar to *LU* decomposition, decompose A as $A = QR$, where Q is orthogonal ($Q^\top Q = I = QQ^\top$) and R is upper triangular
- Using *QR* decomposition it follows that

$$\|A\vec{x} - \vec{b}\|_2^2 \quad =$$

$$=$$

- advantage: since R is upper triangular we conclude

$$R = \begin{bmatrix} R_1 \\ 0 \end{bmatrix}, Q^\top = \begin{bmatrix} Q_1^\top \\ Q_2^\top \end{bmatrix} \Rightarrow$$

- term is minimal, if and only if $R_1\vec{x} = Q_1^\top \vec{b}$
- note that, since Q is orthogonal, we have that $\kappa_2(A) = \kappa_2(R)$, condition number remains the same!

Solving Linear Least Squares via QR Decomposition

Algorithm 4.2 (Blue Print Linear Least Squares via QR Decomposition)

Input: $A \in \mathbb{R}^{m,n}$ *where* $\operatorname{rank} A = n$, $b \in \mathbb{R}^m$

Output: *Compute the solution* $\vec{x} \in \mathbb{R}^n$ *of the (LLS) problem*

1: *Compute QR decomposition* $A = QR$, *where* $R = \begin{bmatrix} R_1 \\ 0 \end{bmatrix}$, $Q^\top = \begin{bmatrix} Q_1^\top \\ Q_2^\top \end{bmatrix}$

2: $\vec{d} \leftarrow Q_1^\top \vec{b}$

3: *solve* $R_1 \vec{x} = \vec{d}$ *by back substitution*

Example 4.5 ((LLS) using MATLAB)

$$A = \begin{bmatrix} 1 & 1 \\ 4 \cdot 10^{-8} & 0 \\ 0 & 4 \cdot 10^{-8} \end{bmatrix}, \quad \vec{b} = \begin{bmatrix} 2 \\ 4 \cdot 10^{-8} \\ 4 \cdot 10^{-8} \end{bmatrix}, \quad \vec{x}_{exact} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

```
>> A=[1 1;4e-8 0;0 4e-8]; b=[2;4e-8;4e-8];
>> [Q,R]=qr(A); R1=R(1:2,1:2); Q1=Q(:,1:2);
>> d=Q1'*b
>> x=R1\d
```

Properties of the QR Decomposition

Theorem 4.2

Let $A = QR$ be (the) QR decomposition of a matrix $A = [\vec{a}^{(1)}, \ldots, \vec{a}^{(n)}] \in \mathbb{R}^{m,n}$ such that $\operatorname{rank} A = n$ *and let $Q = [\vec{q}^{(1)}, \ldots, \vec{q}^{(n)}]$. Then we have:*

1. $$\operatorname{span}\{\vec{a}^{(1)}, \ldots, \vec{a}^{(k)}\} = \operatorname{span}\{\vec{q}^{(1)}, \ldots, \vec{q}^{(k)}\},\ k = 1, \ldots, n,$$

   i.e., the columns of Q form an orthonormal basis of the same space that is already spanned by the columns of A.

2. *For $Q_1 = Q(1:m, 1:n)$, $Q_2 = Q(1:m, n+1:m)$ we can further state:*

   $$\begin{aligned} \operatorname{Image}(A) &= \operatorname{Image}(Q_1), \\ \operatorname{Image}(A)^{\perp} &= \operatorname{Image}(Q_2), \end{aligned}$$

   i.e., while the leading n columns of Q form an orthonormal basis of the column space of A, the remaining columns span its orthogonal complement.

3. *we can further reduce the decomposition to $A = Q_1 R_1$, where $R_1 = R(1:n, 1:n)$ (slim QR decomposition, only $Q_1^T Q_1 = I$ holds!).*

Householder Transformations

- use Householder transformations for computing a QR decomposition
- basic idea: similar to the LU decomposition multiply A step by step using tailored *orthogonal* transformations H_k, which take A one by one to upper triangular form.

$$A \to H_1 A \to H_2 H_1 A \to H_3 H_2 H_1 A \to \cdots$$

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \to \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & * & * \\ 0 & * & * \end{bmatrix} \to \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \\ 0 & 0 & * \end{bmatrix} \to \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \\ 0 & 0 & 0 \end{bmatrix}$$

- each transformation requires in step k only the leading column of the remaining transformed matrix $\vec{x} = A(k : m, k)$, similar to the LU decomposition
- $\vec{y} = H\vec{x}$ uses a reflection mapping H w.r.t. a suitable hyperplane such that $\vec{y}$ is a multiple of the first unit vector

Householder Transformations

- each single transformation matrix H_i can be geometrically interpreted as *elementary reflection*

$$H = I - \frac{2}{\vec{v}^\top \vec{v}} \cdot \vec{v}\vec{v}^\top,$$

  where $\vec{v} \in \mathbb{R}^n \backslash \{0\}$ is called *reflector*
- geometrically, $\vec{y} = H \cdot \vec{x}$ is a reflection of $\vec{x}$ w.r.t. the hyperplane orthogonal to $\vec{v}$
- if $\vec{y}$ is a multiple of the first unit vector, then this reflector is called *Householder transformation*.