Computer-Assisted Problem-Solving / Numerical Methods

Direct Methods for $A\vec{x} = \vec{b}$

version: February 27th, 2020

Legend: Method, Theory, Example, Advanced, Appendix

Theory

Linear Algebra: Solving $A\vec{x} = \vec{b}$

Solving $A\vec{x} = \vec{b}$ occurs very often,
e.g. when solving PDEs or systems of ODEs.

Largest and most important subfield of numerical mathematics.

Subdivision:
- direct methods: Gaussian elimination, LU, ..
- iterative methods: Jacobi, Gauss-Seidel, SOR, CG, Preconditioners, MILU, ...

In practice:
- matrices $A = a_{ij}$ with $i, j = 1, ..., N$ very large,
  e.g. with $N = 10^5$ of 10^6,
  for this $10^{12} * 8$ bytes $= 8.10^3$ Gb needed!
- sparse matrices, i.e. most A_{ij} are zero
- band structure with "large gaps"
- store only $a_{ij} \neq 0$ in memory

  e.g. RAR(1:NNZ), non-zero's (reals)
  IAR(1:NNZ), JAR(1:NNZ), coordinates i, j
  NNZ number of non-zero's

Advanced

Effect of Disturbances

While solving $Ax = b$:

- little disturbance in b may strongly affect x

- similar in case of disturbance(s) in A

Example:

$$\begin{pmatrix} 0.9999 & -1.0001 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1+\epsilon \end{pmatrix} \Longrightarrow \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0.5 + 5000.5\epsilon \\ -0.5 + 4999.5\epsilon \end{pmatrix}$$

Almost singular: intersection of parallel lines

Example:

$$\begin{pmatrix} 7 & 10 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 0.7 \end{pmatrix} \Longrightarrow \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0.1 \end{pmatrix}$$

$$\begin{pmatrix} 7 & 10 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1.01 \\ 0.69 \end{pmatrix} \Longrightarrow \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -0.17 \\ 0.22 \end{pmatrix}$$

To study sensitivity:

check condition number of matrix: $k(A) = \|A\| \|A^{-1}\|$

(see Appendix A)

Method

Solving $Lx = b$ and $Ux = b$

Consider $L\vec{x} = \vec{b}$, L lower-triangular matrix

$$\begin{pmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

Remark: $l_{ii} \neq 0$, otherwise singular

Solve via forward substitution

$$\begin{aligned} x_1 &= b_1/l_{11} \\ x_2 &= (b_2 - l_{21}x_1)/l_{22} \\ x_3 &= (b_3 - l_{31}x_1 - l_{32}x_2)/l_{33} \end{aligned}$$

Algorithm:

$x_1 = b_1/l_{11}$

for $i = 2 \cdots n$

$$x_i = \frac{1}{l_{ii}}(b_i - \sum_{j=1}^{i-1} l_{ij}x_j)$$

next i

Required number of operations:

- **divisions** n
- **multiplications** $1 + 2 + \cdots n - 1 = n(n-1)/2$
- **additions/subtractions** $n(n-1)/2$

Total: n^2 **operations ("flops")**

Solving $U\vec{x} = \vec{b}$, U upper-triangular matrix

$$\begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

via backwards substitution

Remark: $u_{ii} \neq 0$**, otherwise singular**

Algorithm:

$x_n = b_n / u_{nn}$

for $i = n - 1 \cdots 1$

$$x_i = \frac{1}{u_{ii}}(b_i - \sum_{j=i+1}^{n} u_{ij} x_j)$$

next i

Required number of operations also n^2

Method

Gaussian Elimination

Gaussian Elimination Method (GEM):
row reduction of (general) matrix A_{ij}

Example: forward row reduction

$$\begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ a_{21}^{(1)} & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} \\ \vdots & \vdots & & \vdots \\ a_{n1}^{(1)} & a_{n2}^{(1)} & \cdots & a_{nn}^{(1)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1^{(1)} \\ b_2^{(1)} \\ \vdots \\ b_n^{(1)} \end{pmatrix}$$

With the multipliers:

$$m_{i1} = \frac{a_{i1}^{(1)}}{a_{11}^{(1)}}, \quad i = 2 \cdots n, \quad a_{11}^{(1)} \neq 0$$

one obtains via

$$\begin{aligned} a_{ij}^{(2)} &= a_{ij}^{(1)} - m_{i1} a_{1j}^{(1)} \quad i, j = 2 \cdots n \\ b_i^{(2)} &= b_i^{(1)} - m_{i1} b_1^{(1)} \qquad i = 2 \cdots n \end{aligned}$$

the system

$$\begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ \vdots & \vdots & & \vdots \\ 0 & a_{n2}^{(2)} & \cdots & a_{nn}^{(2)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_n^{(2)} \end{pmatrix}$$

First row is not changed!

Transformation to equivalent system:
$A^{(1)}x = b^{(1)} \rightarrow A^{(2)}x = b^{(2)}$

Continue procedure (step k to $k+1$):

$$m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}, \quad i = k+1 \cdots n$$

$$\begin{aligned} a_{ij}^{(k+1)} &= a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)} \quad i,j = k+1 \cdots n \\ b_i^{(k+1)} &= b_i^{(k)} - m_{ik} b_k^{(k)} \qquad i = k+1 \cdots n \end{aligned}$$

Until finally $A^{(n)}x = b^{(n)}$**:**

$$\begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & a_{nn}^{(n)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_n^{(n)} \end{pmatrix}$$

Remark:
this only works if $a_{ii}^{(i)} \neq 0$ **for** $i = 1 \cdots n-1$**,**
because of the multipliers

$$m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}, \quad i = k+1 \cdots n$$

System $A^{(n)}x = b^{(n)}$ **has the form** $Ux = b$
$\Longrightarrow$ **can be solved simply (fast)**

Number of operations GEM: $\sim \mathcal{O}(\frac{2}{3}n^3)$ **(if** n **large)**
After that still n^2**, because of** $Ux = b$
Total: $\sim \mathcal{O}(\frac{2}{3}n^3)$ **operations**

Example for 10^{+9} operations/second

task	$n = 1,000$	$n = 10,000$	$n = 100,000$
row reduction	0.7 sec	11 min	185 hours
solving	0.001 sec	0.1 sec	10 sec

Theorem: if matrix A

1) diagonally dominant (per row or column)

or

2) symmetric and positively definite

$\Longrightarrow$ GEM safe

Definition diagonally dominant (per row):

$$|a_{ii}| \geq \sum_{j=1, j\neq i}^{n} |a_{ij}|, \quad i = 1 \cdots n$$

Definition symmetric: $a_{ij} = a_{ji},\ i, j = 1 \cdots n$

Definition positively definite: $(Ax, x) > 0\ \forall x \neq 0$

If at some point $a_{ii}^{(i)}\!=\!0$: GEM does not work

Remedy: apply a permutation, i.e.

change sequence $(1, \cdots, n)$ (pivoting)

Remark: during GEM matrix can become dense

$\Longrightarrow A^{(k)}$ have increasing number of non-zeros

(sparse structure disappears)

Method

LU Factorisation

Often one has to solve $Ax = b$ for several b's $\Longrightarrow$

multipliers m_{ij} of GEM used more often

Mostly done in other way:

$$L := \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ m_{21} & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ m_{n1} & m_{n2} & \cdots & \cdots & 1 \end{pmatrix}$$

$$U := \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn}^{(n)} \end{pmatrix}$$

Theorem: $A = LU$

Proof: expand $(k = 1 \cdots n)$

$$\begin{aligned} m_{ik} &= \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}, \quad i = k+1 \cdots n \\ a_{ij}^{(k+1)} &= a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)} \quad i, j = k+1 \cdots n \end{aligned}$$

Remark:

ones on diagonal $L \to$ Doolittle method

ones on diagonal $U \to$ Crout method

Example:

$$\begin{pmatrix} 1 & 2 & 1 \\ 2 & 2 & 3 \\ -1 & -3 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & \frac{1}{2} & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 & 1 \\ 0 & -2 & 1 \\ 0 & 0 & \frac{1}{2} \end{pmatrix}$$

Solve $Ax = b \Longleftrightarrow LUx = b$**:**

(1) first $Ly = b$

(2) then $Ux = y$

(both are triangular systems $\rightarrow$ **fast solution!)**

Theorem: if matrix A **diagonally dominant (per row or column)** $\Longrightarrow$ **LU-factorisation exists**

Remark:

use $A = LU$ **to simply calculate determinant:**

$det(A) = det(L)det(U) = a_{11}^{(1)} a_{22}^{(2)} \cdots a_{nn}^{(n)}$

Number of operations factorisation $A = LU$

step k	subtract.	multipl.	divisions
1	$(n-1)^2$	$(n-1)^2$	$n-1$
2	$(n-2)^2$	$(n-2)^2$	$n-2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-1$	1	1	1
total	$\frac{n(n-1)(2n-1)}{6}$	$\frac{n(n-1)(2n-1)}{6}$	$\frac{n(n-1)}{2}$

Number of operations adjustments r.h.s. b

step k	**subtract.**	**multipl.**
1	$(n-1)$	$(n-1)$
2	$(n-2)$	$(n-2)$
$\vdots$	$\vdots$	$\vdots$
$n-1$	1	1
total	$\frac{n(n-1)}{2}$	$\frac{n(n-1)}{2}$

Required operations to solve $A^{(n)}x = Ux = y$: n^2
Total for solving $Ax = b$: $\sim \mathcal{O}(\frac{2}{3}n^3)$
(for large systems)

LU-factorisation most expensive part
Solving for different b's relatively cheap

Remark:
with right-hand terms
$b_1 = (1, 0, .., 0)$, $b_2 = (0, 1, 0, .., 0)$, $\cdots$, $b_n = (0, .., 0, 1)$ $\Longrightarrow$
inverse matrix A^{-1}
This requires $\sim \mathcal{O}(\frac{2}{3}n^3) + n * 2n^2 \sim \mathcal{O}(\frac{8}{3}n^3)$ flops

But ... why would one determine A^{-1} ?

Advanced

Cholesky Factorisation

GEM/LU faster for special matrices A

Theorem:
A **symmetric and positive definite** $\Longrightarrow A = LL^T$,
with L lower triangular,
positive (real) diagonal

Pattern in case of 3x3 matrices:

$$A = \begin{pmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{pmatrix} \begin{pmatrix} l_{11} & l_{21} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{pmatrix}$$

This is called Cholesky Factorisation

Operations: $\sim \mathcal{O}(\frac{1}{3}n^3)$ **flops (2x faster as LU)**

Example: $$\begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2\sqrt{3}} & 0 \\ \frac{1}{3} & \frac{1}{2\sqrt{3}} & \frac{1}{6\sqrt{5}} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2\sqrt{3}} & 0 \\ \frac{1}{3} & \frac{1}{2\sqrt{3}} & \frac{1}{6\sqrt{5}} \end{pmatrix}^T$$

Construction of L: use $A = LL^T$

- **row 1 L * column 1 L^T** $\Longrightarrow l_{11}^2 = a_{11}$
  A **pos.def.** $\Longrightarrow a_{11} > 0 \Longrightarrow l_{11} = \sqrt{a_{11}}$
- **row 2 L * columns 1 and 2 L^T** $\Longrightarrow$
  $l_{21}l_{11} = a_{21}$ **and** $l_{21}^2 + l_{22}^2 = a_{22} \Longrightarrow$
  $l_{21} = a_{21}/l_{11} = a_{21}/\sqrt{a_{11}}$ **and** $l_{22} = \sqrt{a_{22} - l_{21}^2}$

- **general**

$$
\begin{aligned}
l_{ij} &= (a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk})/l_{jj}, \quad j = 1, \cdots, i-1 \\
l_{ii} &= (a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2)^{1/2}, \qquad i = 2, \cdots, n
\end{aligned}
$$

A **pos.def.** $\Longrightarrow$ **roots no problem (real)**

Alternative: $A = \tilde{L} D \tilde{L}^T$**, with**

- D **diagonal matrix**

- $\tilde{L}$ **lower triangular matrix, 1 on diagonal**

$\Longrightarrow$ **no roots required**

Example:

$$
\begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{3} & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{12} & 0 \\ 0 & 0 & \frac{1}{180} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{3} & 1 & 1 \end{pmatrix}^T
$$

Solution after Cholesky factorisation:

$Ax = b \Longleftrightarrow \tilde{L} D \tilde{L}^T x = b$

(1) first $\tilde{L} z = b$

(2) then $Dy = z$

(3) finally $\tilde{L}^T x = y$

Where:

(1) and (3) are triangular systems (fast)

(2) is diagonal division (very simple)

Method

Tri-Diagonal Matrices (TDMA)

Tri-Diagonal Matrix Algorithm

Solve $Ax=f$, with $A=a_{ij}$ tri-diagonal:

$a_{ij}=0$ **for** $|i-j|>1$

$$A=\begin{pmatrix} a_1 & c_1 & 0 & & 0 \\ b_2 & a_2 & c_2 & & 0 \\ 0 & b_3 & a_3 & c_3 & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & & & b_n & a_n \end{pmatrix}$$

LU factorisation for such a matrix:

$$A=LU=\begin{pmatrix} \alpha_1 & 0 & 0 & & 0 \\ b_2 & \alpha_2 & 0 & & 0 \\ 0 & b_3 & \alpha_3 & & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & & & b_n & \alpha_n \end{pmatrix}\begin{pmatrix} 1 & \gamma_1 & 0 & & 0 \\ 0 & 1 & \gamma_2 & & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ \vdots & & & 1 & \gamma_{n-1} \\ 0 & & & 0 & 1 \end{pmatrix}$$

Thus

$$a_1=\alpha_1 \quad \alpha_1\gamma_1=c_1$$

$$a_i=\alpha_i+b_i\gamma_{i-1} \quad i=2\cdots n$$

$$\alpha_i\gamma_i=c_i \quad i=2\cdots n-1$$

Hence

$$\alpha_1=a_1 \quad \gamma_1=c_1/\alpha_1$$

$$\alpha_i=a_i-b_i\gamma_{i-1} \quad \gamma_i=c_i/\alpha_i \quad i=2\cdots n-1$$

$$\alpha_n=a_n-b_n\gamma_{n-1}$$

LU factorisation is now available

Then solve $Ly=f$ and $Ux=y \implies Ax=f$

Example:

$$\begin{pmatrix} 2 & 1 & 0 & 0 \\ 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ 0 & 0 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 2 & 0 & 0 & 0 \\ 1 & \frac{7}{2} & 0 & 0 \\ 0 & 1 & \frac{26}{7} & 0 \\ 0 & 0 & 1 & \frac{45}{26} \end{pmatrix} \begin{pmatrix} 1 & \frac{1}{2} & 0 & 0 \\ 0 & 1 & \frac{2}{7} & 0 \\ 0 & 0 & 1 & \frac{7}{26} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Operations for LU:

$n-1$ **divisions,** $n-1$ **multipl.,** $n-1$ **subtract.**

$\Longrightarrow$ **total for LU** $\sim \mathcal{O}(3n)$

Solve $Ly = f$**:**

$$\begin{aligned} y_1 &= f_1/\alpha_1 \\ y_i &= (f_i - b_i y_{i-1})/\alpha_i \quad i = 2 \cdots n \end{aligned}$$

Solve $Ux = y$**:**

$$\begin{aligned} x_n &= y_n \\ x_i &= y_i - \gamma_i x_{i+1} \quad i = n-1 \cdots 1 \end{aligned}$$

Operations for solution:

n **divisions,** $2(n-1)$ **multipl.,** $2(n-1)$ **subtract.**

$\Longrightarrow$ **total for solving** $\sim \mathcal{O}(5n)$

Solution costs similar to LU **construction:**

both $\sim \mathcal{O}(n)$

General band matrices:

p **lower and** q **upper bands (with** $p, q \ll n$**)**

LU $\sim \mathcal{O}(2npq)$ **flops**

Solving $\sim \mathcal{O}(2np + 2nq)$

Example with 10^{+9} operations/second

Tri-Diagonal Matrix Algorithm (TDMA)

task	$n = 1,000$	$n = 10,000$	$n = 100,000$
LU	**3** μ**sec**	**30** μ**sec**	**300** μ**sec**
solve	**5** μ**sec**	**50** μ**sec**	**500** μ**sec**

Complete LU + solution

task	$n = 1,000$	$n = 10,000$	$n = 100,000$
row reduction	**0.7 sec**	**11 min**	**185 hours**
solve	**0.001 sec**	**0.1 sec**	**10 sec**

Use matrix structure:
large reduction in computer time!

Efficient storage of band matrix A:

$$A_{i,j} = \begin{pmatrix} a_1 & c_1 & 0 & & 0 \\ b_2 & a_2 & c_2 & & 0 \\ 0 & b_3 & a_3 & c_3 & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & & & b_n & a_n \end{pmatrix} \longrightarrow A(k,l) = \begin{pmatrix} 0 & a_1 & c_1 \\ b_2 & a_2 & c_2 \\ b_3 & a_3 & c_3 \\ \vdots & \vdots & \vdots \\ b_n & a_n & 0 \end{pmatrix}$$

$i, j = 1 \ldots n$ $\qquad$ $k = 1 \ldots n$, $l = 1, 2, 3$

Appendix

A1: Condition Number

For sensitivity $Ax = b$: check condition number

Definition: condition number of matrix A

$$k(A) = \|A\|\|A^{-1}\|$$

Options for Matrix Norm:

$$\|A\|_1 = \max_{j=1,\dots,n} \sum_{i=1}^{m} |a_{ij}| \quad \textbf{column-norm}$$

$$\|A\|_\infty = \max_{i=1,\dots,m} \sum_{j=1}^{n} |a_{ij}| \quad \textbf{row-norm}$$

$$\|A\|_F = \sqrt{\sum_{i,j=1}^{m,n} |a_{ij}|^2} \quad \textbf{Frobenius-norm}$$

$$\|A\|_2 = \sqrt{r_\sigma(A^* A)} \quad \textbf{2-norm}$$

$$\textbf{with } r_\sigma(A) := \max_{\lambda \in \sigma(A)} |\lambda| \quad \textbf{spectral radius}$$

Matrix norm often in terms of Vector Norm

$$\|A\|_v = \sup_{x \neq 0} \frac{\|Ax\|_v}{\|x\|_v}, \quad v = 1, 2, \infty$$

Vector norms:

$$\|x\|_1 = \sum_{i=1}^{n} |x_i| \qquad \|x\|_\infty = \max_{i=1,\dots,n} |x_i| \qquad \|x\|_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2}$$

Remark: norms used for iterative methods:

- for convergence/stability analyses

- as part of the method (algorithm)

Appendix

A2: Measure Disturbances

Disturbance r in rhs b:

$$\begin{aligned} Ax &= b \\ A\tilde{x} &= b + r \end{aligned}$$

causes disturbance $e := \tilde{x} - x$ in solution

Theorem: $$\frac{1}{k(A)} \frac{\|r\|}{\|b\|} \leq \frac{\|e\|}{\|x\|} \leq k(A) \frac{\|r\|}{\|b\|}$$

Proof:

With $e := \tilde{x} - x \Longrightarrow Ae = r \Longrightarrow e = A^{-1} r$

Estimations:

$$\begin{aligned} r = Ae &\Longrightarrow \|r\| \leq \|A\| \|e\| \\ e = A^{-1} r &\Longrightarrow \|e\| \leq \|A^{-1}\| \|r\| \end{aligned}$$

This gives

$$\frac{\|r\|}{\|A\| \|x\|} \leq \frac{\|e\|}{\|x\|} \leq \frac{\|A^{-1}\| \|r\|}{\|x\|}$$

More estimations:

$$b = Ax \Longrightarrow \|b\| \leq \|A\| \|x\| \Longrightarrow \frac{1}{\|x\|} \leq \frac{\|A\|}{\|b\|}$$

$$x = A^{-1} b \Longrightarrow \|x\| \leq \|A^{-1}\| \|b\| \Longrightarrow \frac{1}{\|A^{-1}\| \|b\|} \leq \frac{1}{\|x\|}$$

Final result:

$$\frac{1}{\|A\| \|A^{-1}\|} \frac{\|r\|}{\|b\|} \leq \frac{\|e\|}{\|x\|} \leq \|A\| \|A^{-1}\| \frac{\|r\|}{\|b\|}$$

Remark: $k(A) \geq 1$, **since**
$1 = \|I\| = \|AA^{-1}\| \leq \|A\|\|A^{-1}\| = k(A)$

If $k(A) \approx 1 \implies$ **small disturbance** r **in rhs** b
gives comparable disturbance in solution x

Example:

$$A = \begin{pmatrix} 7 & 10 \\ 5 & 7 \end{pmatrix} \implies A^{-1} = \begin{pmatrix} -7 & 10 \\ 5 & -7 \end{pmatrix}$$

$k(A)_1 = k(A)_\infty = 17 * 17 = 289$
$k(A)_2 = 223$

Condition numbers large $\implies$
large influence of disturbances

Remark:
$$\begin{pmatrix} 1 & 0 \\ 0 & 10^{-10} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 10^{-10} \end{pmatrix}$$
can be solved safely, although $k(A) = 10^{10}$
Reason: norms of full vectors/matrices taken for $k(A)$, **instead of individual elements**

Scaling of problem:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

gives condition number $k(A) = 1$