



## Dense linear algebra : direct methods

P. Amestoy, P. Berger, M. Daydé, F.-H. Rouet  
(INPT-ENSEEIH)  
and  
J.-Y. L'Excellent (INRIA/LIP-ENS Lyon)

2020-2021

### Linear Algebra Basics

Gaussian Elimination and LU factorization  
LU Factorization with partial pivoting  
Symmetric matrices  
Cholesky Factorization

1 / 38

2 / 38

## System of linear equations ?

### Example:

$$\begin{array}{rrcrcl} 2 & x_1 & - & 1 & x_2 & + & 3 & x_3 & = & 13 \\ -4 & x_1 & + & 6 & x_2 & - & 5 & x_3 & = & -28 \\ 6 & x_1 & + & 13 & x_2 & + & 16 & x_3 & = & 37 \end{array}$$

can be written under the form:

$$\mathbf{Ax} = \mathbf{b},$$

with  $\mathbf{A} = \begin{pmatrix} 2 & -1 & 3 \\ -4 & 6 & -5 \\ 6 & 13 & 16 \end{pmatrix}$ ,  $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$ , and  $\mathbf{b} = \begin{pmatrix} 13 \\ -28 \\ 37 \end{pmatrix}$

### Linear Algebra Basics

Gaussian Elimination and LU factorization  
LU Factorization with partial pivoting  
Symmetric matrices  
Cholesky Factorization

Example:

$$2x_1 - x_2 + 3x_3 = 13 \quad (1)$$

$$-4x_1 + 6x_2 - 5x_3 = -28 \quad (2)$$

$$6x_1 + 13x_2 + 16x_3 = 37 \quad (3)$$

With  $2 * (1) + (2) \rightarrow (2)$  and  $-3*(1) + (3) \rightarrow (3)$  we obtain:

$$2x_1 - x_2 + 3x_3 = 13 \quad (4)$$

$$0x_1 + 4x_2 + x_3 = -2 \quad (5)$$

$$0x_1 + 16x_2 + 7x_3 = -2 \quad (6)$$

Thus  $x_1$  is eliminated from (5) and (6). With  $-4*(5) + (6) \rightarrow (6)$ :

$$2x_1 - x_2 + 3x_3 = 13$$

$$0x_1 + 4x_2 + x_3 = -2$$

$$0x_1 + 0x_2 + 3x_3 = 6$$

The linear system is then solved by backward ( $x_3 \rightarrow x_2 \rightarrow x_1$ ) substitution:  $x_3 = \frac{6}{3} = 2$ ,  $x_2 = \frac{1}{4}(-2 - x_3) = -1$ , and finally  $x_1 = \frac{1}{2}(13 - 3x_3 + x_2) = 3$

- Find **L** unit lower triangular and **U** upper triangular such that:  
 $\mathbf{A} = \mathbf{L} \times \mathbf{U}$

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 3 \\ -4 & 6 & -5 \\ 6 & 13 & 16 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 3 & 4 & 1 \end{bmatrix} \times \begin{bmatrix} 2 & -1 & 3 \\ 0 & 4 & 1 \\ 0 & 0 & 3 \end{bmatrix}$$

- Procedure to solve  $\mathbf{Ax} = \mathbf{b}$ 
  - $\mathbf{A} = \mathbf{LU}$
  - Solve  $\mathbf{Ly} = \mathbf{b}$  (forward elimination, down)
  - Solve  $\mathbf{Ux} = \mathbf{y}$  (backward substitution, up)

$$\mathbf{Ax} = (\mathbf{LU})\mathbf{x} = \mathbf{L}(\mathbf{Ux}) = \mathbf{Ly} = \mathbf{b}$$

5 / 38

6 / 38

## From Gaussian Elimination to LU Factorization

## From Gaussian Elimination to LU Factorization

Typical Gaussian elimination at step  $k$ :

$$\begin{pmatrix} a_{11}^{(1)} & \dots & \dots & \dots & \dots & \dots & a_{1n}^{(1)} \\ 0 & \ddots & & & & & \vdots \\ \vdots & & a_{k-1,k-1}^{(k-1)} & \dots & \dots & \dots & a_{k-1,n}^{(k-1)} \\ \vdots & & 0 & a_{kk}^{(k)} & a_{k,k+1}^{(k)} & \dots & a_{kn}^{(k)} \\ \vdots & & \vdots & a_{k+1,k}^{(k)} & a_{k+1,k+1}^{(k)} & \dots & a_{k+1,n}^{(k)} \\ \vdots & & \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & \dots & 0 & a_{nk}^{(k)} & a_{n,k+1}^{(k)} & \dots & a_{nn}^{(k)} \end{pmatrix} \begin{pmatrix} b_1^{(1)} \\ \vdots \\ b_{k-1}^{(k-1)} \\ b_k^{(k)} \\ b_{k+1}^{(k)} \\ \vdots \\ b_n^{(k)} \end{pmatrix}$$

$\mathbf{A} = \mathbf{A}^{(1)}$ ,  $\mathbf{b} = \mathbf{b}^{(1)}$ ,  $\mathbf{A}^{(1)}\mathbf{x} = \mathbf{b}^{(1)}$ :

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \quad \begin{array}{l} 2 \leftarrow 2 - 1 \times a_{21}/a_{11} \\ 3 \leftarrow 3 - 1 \times a_{31}/a_{11} \end{array}$$

$\mathbf{A}^{(2)}\mathbf{x} = \mathbf{b}^{(2)}$

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} \\ 0 & a_{32}^{(2)} & a_{33}^{(2)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2^{(2)} \\ b_3^{(2)} \end{pmatrix} \quad \begin{array}{l} b_2^{(2)} = b_2 - a_{21}b_1/a_{11} \\ a_{32}^{(2)} = a_{32} - a_{31}a_{12}/a_{11} \end{array}$$

Finally  $3 \leftarrow 3 - 2 \times a_{32}/a_{22}$  gives  $\mathbf{A}^{(3)}\mathbf{x} = \mathbf{b}^{(3)}$

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} \\ 0 & 0 & a_{33}^{(3)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2^{(2)} \\ b_3^{(3)} \end{pmatrix} \quad a_{33}^{(3)} = a_{33}^{(2)} - a_{32}^{(2)}a_{23}^{(2)}/a_{22}^{(2)}$$

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} a_{kj}^{(k)}, \text{ for } i > k$$

$$(\text{and } a_{ij}^{(k+1)} = a_{ij}^{(k)} \text{ for } i \leq k)$$

7 / 38

8 / 38

$$\begin{cases} a_{ij}^{(k+1)} = a_{ij}^{(k)} - \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} a_{kj}^{(k)}, & \text{for } i > k \\ a_{ij}^{(k+1)} = a_{ij}^{(k)}, & \text{for } i \leq k \end{cases}$$

- One step of Gaussian elimination can be written:  
 $\mathbf{A}^{(k+1)} = \mathbf{L}^{(k)} \mathbf{A}^{(k)}$  (and  $b^{(k+1)} = \mathbf{L}^{(k)} b^{(k)}$ ), with

$$\mathbf{L}^{(k)} = \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & -l_{k+1,k} & \ddots \\ & & & \ddots & 1 \end{pmatrix} \text{ and } l_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}.$$

- After  $n-1$  steps,  $\mathbf{A}^{(n)} = \mathbf{U} = \mathbf{L}^{(n-1)} \dots \mathbf{L}^{(1)} \mathbf{A}$  gives  $\boxed{\mathbf{A} = \mathbf{LU}}$ ,  
 with  $\mathbf{L} = [\mathbf{L}^{(1)}]^{-1} \dots [\mathbf{L}^{(n-1)}]^{-1} =$

$$\begin{pmatrix} 1 & & & \\ l_{21} & 1 & & \\ \vdots & & \ddots & \\ l_{n1} & & & 1 \end{pmatrix} \dots \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & l_{n,n-1} & 1 \end{pmatrix} = \begin{pmatrix} 1 & & 0 \\ l_{21} & 1 & \\ \vdots & & \ddots \\ l_{n1} & \dots & l_{n,n-1} & 1 \end{pmatrix}$$

- Avoid building the zeros under the diagonal
- Before

```

L(n,n)=1
do k=1, n-1
  L(k,k) = 1
  do i=k+1, n
    L(i,k) = A(i,k)/A(k,k)
    do j=k, n
      A(i,j) = A(i,j) - L(i,k) * A(k,j)
    
```

- After

```

L(n,n)=1
do k=1, n-1
  L(k,k) = 1
  do i=k+1, n
    L(i,k) = A(i,k)/A(k,k)
    do j=k+1, n
      A(i,j) = A(i,j) - L(i,k) * A(k,j)
    
```

- Overwrite matrix  $\mathbf{A}$ : we store  $a_{ij}^{(k)}$ ,  $k = 2, \dots, n$  in  $A(i, j)$

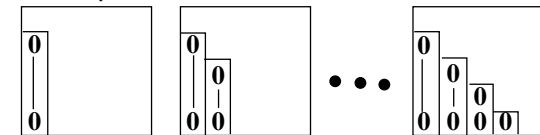
- In the end,  $\mathbf{A} = \mathbf{A}^{(n)} = \mathbf{U}$

```

do k=1, n-1
  L(k,k) = 1
  do i=k+1, n
    L(i,k) = A(i,k)/A(k,k)
    do j=k, n ! (better than: do j=1,n)
      A(i,j) = A(i,j) - L(i,k) * A(k,j)
    end do
  enddo
enddo
L(n,n)=1

```

- Matrix  $\mathbf{A}$  at each step:



10 / 38

- Use lower triangle of array  $\mathbf{A}$  to store  $\mathbf{L}_{i,k}$  multipliers

- Before:

```

L(n,n)=1
do k=1, n-1
  L(k,k) = 1
  do i=k+1, n
    L(i,k) = A(i,k)/A(k,k)
    do j=k+1, n
      A(i,j) = A(i,j) - L(i,k) * A(k,j)
    
```

- After (diagonal 1 of  $\mathbf{L}$  is not stored):

```

do k=1, n-1
  do i=k+1, n
    A(i,k) = A(i,k)/A(k,k)
    do j=k+1, n
      A(i,j) = A(i,j) - A(i,k) * A(k,j)
    
```

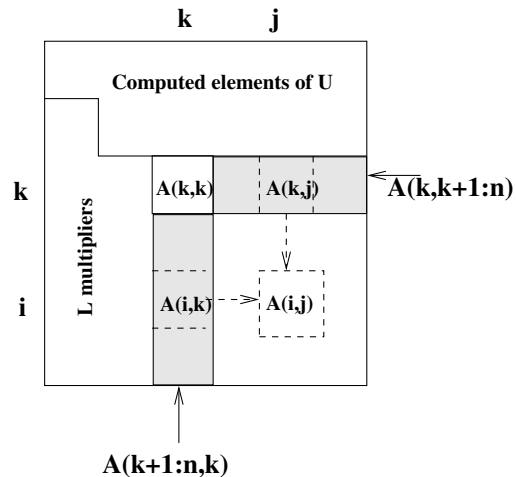
- ▶ More compact array syntax (Matlab, scilab):

```

do k=1, n-1
  A(k+1:n, k) = A(k+1:n, k) / A(k, k)
  A(k+1:n, k+1:n) = A(k+1:n, k+1:n)
    - A(k+1:n, k) * A(k, k+1:n)
end do

```

- ▶ corresponds to a rank-1 update:

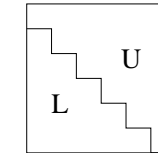


## What we have computed

- ▶ we have stored the **L** and **U** factors in **A**:

- ▶  $A_{i,j}, i > j$  corresponds to  $l_{ij}$
- ▶  $A_{i,j}, i \leq j$  corresponds to  $u_{ij}$
- ▶ with  $l_{ii} = 1, i = 1, n$

- ▶ Finally,



after factorization:  $\mathbf{A} = \mathbf{L} + \mathbf{U} - \mathbf{I}$

13 / 38

14 / 38

## LU factorization : summary

- ▶ Step by step columns of **A** are set to zero and **A** is updated  $\mathbf{L}^{(n-1)} \dots \mathbf{L}^{(1)} \mathbf{A} = \mathbf{U}$  leading to  $\mathbf{A} = \mathbf{L}\mathbf{U}$  where  $\mathbf{L} = [\mathbf{L}^{(1)}]^{-1} \dots [\mathbf{L}^{(n-1)}]^{-1}$
- ▶ At each step  $\mathbf{A}(k, k)$  is referred to as the **pivot**
  - zero entries in column of **A** can be replaced by entries in **L**
  - row entries of **U** are stored in corresponding locations of **A**

### Algorithm 1 LU factorization

```

for k = 1, n - 1 do
  if |A(k, k)| too small then
    exit (small pivots are not allowed)
  end if
  A(k+1:n, k) = A(k+1:n, k) / A(k, k)
  A(k+1:n, k+1:n) = A(k+1:n, k+1:n) - A(k+1:n, k) * A(k, k+1:n)
end for

```

When  $|\mathbf{A}(k, k)|$  is too small, one could consider other pivots: **numerical pivoting strategies** will be introduced later.

## Existence and uniqueness of LU decomposition

### Theorem 1

$\mathbf{A} \in \mathbb{R}^{n \times n}$  has an LU factorization (where **L** is unit lower triangular and **U** is upper triangular) if  $\det(\mathbf{A}(1:k, 1:k)) \neq 0$  for all  $k \in \{1 \dots n-1\}$ . If the LU factorization exists, then it is unique and  $\det(\mathbf{A}) = u_{11} \dots u_{nn}$ .

### Theorem 2

For each nonsingular matrix **A**, there exists a permutation matrix **P** such that **PA** possesses an LU factorization  $\mathbf{PA} = \mathbf{L}\mathbf{U}$ .

### Definition 1

$\mathbf{A} \in \mathbb{R}^{n \times n}$  is **strictly diagonally dominant** iff  $|a_{ii}| > \sum_{j=1, j \neq i}^n |a_{ij}|$  for all  $i = 1, \dots, n$

### Theorem 3

If  $\mathbf{A}^T$  is strictly diagonally dominant then **A** is non-singular and **A** has an LU factorization and  $l_{ij} < 1$

15 / 38

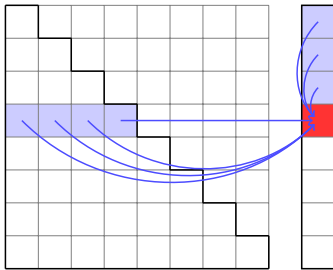
16 / 38

**Algorithm 2** LL (sans report)

```

x = b
for j = 1, n do
  for i = 1, j - 1 do
    x_j = x_j - l_ij x_i
  end for
  x_j = x_j / l_jj
end for

```

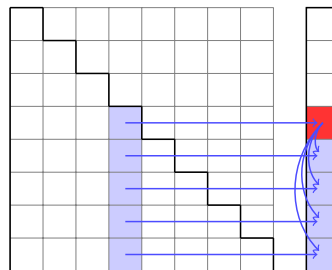


**Algorithm 3** RL (avec report)

```

x = b
for j = 1, n do
  x_j = x_j / l_jj
  for i = j + 1, n do
    x_i = x_i - l_ij x_j
  end for
end for

```



17 / 38

**Exercise 1** (Blocked  $\mathbf{LU}$  and Schur decomposition)

Let  $\mathbf{A}$  be a non singular matrix of order  $n$  for which  $\exists \mathbf{P}$  permutation matrix such that  $\mathbf{PA}$  can be factored without pivoting and consider the block form

$$\mathbf{PA} = \begin{pmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} \\ \mathbf{A}_{2,1} & \mathbf{A}_{2,2} \end{pmatrix}. \text{ We define the so called Schur matrix } \mathbf{S} \text{ as}$$

$$\mathbf{S} = \mathbf{A}_{2,2} - \mathbf{A}_{2,1} (\mathbf{A}_{1,1})^{-1} \mathbf{A}_{1,2}$$

1. Explain how to adapt the  $\mathbf{LU}$  factorisation algorithm to obtain the following decomposition of  $\mathbf{PA}$ . 
$$\mathbf{PA} = \begin{pmatrix} \mathbf{L}_{1,1} & 0 \\ \mathbf{L}_{2,1} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{U}_{1,1} & \mathbf{U}_{1,2} \\ 0 & \mathbf{S} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{1,1} & 0 \\ \mathbf{L}_{2,1} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{I} & 0 \\ 0 & \mathbf{S} \end{pmatrix} \begin{pmatrix} \mathbf{U}_{1,1} & \mathbf{U}_{1,2} \\ 0 & \mathbf{I} \end{pmatrix}$$
2. Prove that  $\det(\mathbf{A}) = \det(\mathbf{P})\det(\mathbf{A}_{1,1})\det(\mathbf{S})$
3. We assume that we know how to compute  $\mathbf{Y}$  such that  $\mathbf{Y} = \mathbf{S}^{-1}\mathbf{Z}$ . Describe how to use previous incomplete blocked factorization to solve  $\mathbf{AX} = \mathbf{B}$ .

Blocked factorization and null space

Number of floating-point operations (flops)

**Exercise 2** (Null space)

We suppose that after  $n - r$  steps of  $\mathbf{LU}$  factorization we have

$$\mathbf{PA} = \begin{pmatrix} \mathbf{L}_{1,1} & 0 \\ \mathbf{L}_{2,1} & \mathbf{I}_r \end{pmatrix} \begin{pmatrix} \mathbf{U}_{1,1} & \mathbf{U}_{1,2} \\ 0 & \mathbf{S}_r \end{pmatrix} \text{ where } \mathbf{S}_r \text{ is the Schur}$$

complement matrix of order  $r$ . We also suppose that  $\mathbf{S}_r = 0$  (in practice one could also assume that  $\|\mathbf{S}_r\| \leq \varepsilon \|\mathbf{A}\|$  for some matrix norm). Finally we assume that  $\det(\mathbf{U}_{11}) \neq 0$ .

Prove that the dimension of the null-space is  $r$  and compute a basis of the null space.

- In forward elimination ( $\mathbf{Ly} = \mathbf{b}$ ), computing the  $k^{th}$  unknown

$$y_k = b_k - \sum_{j=1}^{k-1} L_{kj} y_j$$

leads to  $(k-1)$  multiplications and  $(k-1)$  additions, for  $1 \leq k \leq n$   
 $n^2 - n$  flops overall

- Idem for  $\mathbf{Ux} = \mathbf{y}$  and at worst  $n$  divisions ( $U_{kk} \neq 1$ ).

- Number of flops during factorization:

- $n - k$  divisions
- $(n - k)^2$  multiplications,  $(n - k)^2$  additions
- $k = 1, 2, \dots, n - 1$
- total:  $\approx \frac{2 \times n^3}{3}$

(Strassen's algorithm can reduce this to  $\Theta(n^{\log_2 7}) \simeq \Theta(n^{2.8})$ )

## Exercise 3 (How to compute $\mathbf{x}$ such that $\mathbf{x} = (\mathbf{A}^2)^{-1} \mathbf{b}$ )

Let  $\mathbf{A}$  be a non singular matrix of order  $n$  (i.e. it exists  $\mathbf{L}$ ,  $\mathbf{U}$  and  $\mathbf{P}$  such that  $\mathbf{PA} = \mathbf{LU}$  (note that  $\mathbf{A}^2$  is also non singular).

1. Compare the computational complexity of solving  $\mathbf{A}^2 \mathbf{x} = \mathbf{b}$  with the following two algorithms:
  - 1.1 Compute  $\mathbf{B} = \mathbf{A}^2$ , factor  $\mathbf{B}$  and solve  $\mathbf{Bx} = \mathbf{b}$
  - 1.2 Factor  $\mathbf{A}$  and use the factored form to solve  $\mathbf{A}^2 \mathbf{x} = \mathbf{b}$
2. Explain why computing directly  $\mathbf{C} = (\mathbf{A}^2)^{-1}$  and performing  $\mathbf{x} = \mathbf{C}\mathbf{b}$  is not a method of choice.

## Linear Algebra Basics

Gaussian Elimination and LU factorization

LU Factorization with partial pivoting

Symmetric matrices

Cholesky Factorization

21 / 38

$$\text{Consider } A = \begin{bmatrix} \varepsilon & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{1}{\varepsilon} & 1 \end{bmatrix} \times \begin{bmatrix} \varepsilon & 1 \\ 0 & 1 - \frac{1}{\varepsilon} \end{bmatrix}$$

$$\kappa_2(A) = \frac{\lambda_{\max}}{\lambda_{\min}} = \frac{1 + \varepsilon + \sqrt{5 + \varepsilon^2 - 2\varepsilon}}{-1 - \varepsilon + \sqrt{5 + \varepsilon^2 - 2\varepsilon}} \simeq 2.6$$

If one solves:

$$\begin{bmatrix} \varepsilon & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 + \varepsilon \\ 2 \end{bmatrix}$$

Exact solution  $\mathbf{x}^* = (1, 1)$ .

22 / 38

$$A = \begin{bmatrix} \varepsilon & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{1}{\varepsilon} & 1 \end{bmatrix} \times \begin{bmatrix} \varepsilon & 1 \\ 0 & 1 - \frac{1}{\varepsilon} \end{bmatrix}$$

$\varepsilon$	$\frac{\ \mathbf{x}^* - \mathbf{x}\ }{\ \mathbf{x}^*\ }$	$\kappa_2(A)$
$10^{-3}$	$6 \times 10^{-16}$	2.621
$10^{-6}$	$2 \times 10^{-11}$	2.618
$10^{-9}$	$9 \times 10^{-8}$	2.618
$10^{-12}$	$9 \times 10^{-5}$	2.618
$10^{-15}$	$7 \times 10^{-2}$	2.618

Table: Relative error as a function of  $\varepsilon$ .

- Even if  $A$  is well conditioned, Gaussian elimination may introduce errors

$$A = \begin{bmatrix} \varepsilon & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{1}{\varepsilon} & 1 \end{bmatrix} \times \begin{bmatrix} \varepsilon & 1 \\ 0 & 1 - \frac{1}{\varepsilon} \end{bmatrix}$$

$\varepsilon$	$\frac{\ x^* - x\ }{\ x^*\ }$	$\kappa_2(A)$
$10^{-3}$	$6 \times 10^{-16}$	2.621
$10^{-6}$	$2 \times 10^{-11}$	2.618
$10^{-9}$	$9 \times 10^{-8}$	2.618
$10^{-12}$	$9 \times 10^{-5}$	2.618
$10^{-15}$	$7 \times 10^{-2}$	2.618

Table: Relative error as a function of  $\varepsilon$ .

- ▶ Even if  $A$  is well conditioned, Gaussian elimination may introduce errors
- ▶ Explanation: pivot  $\varepsilon$  is too small and leads to a large element growth (**growth factor**) in  $L$  and  $U$ :  $\frac{1}{\varepsilon}$  in  $L$  leads to a loss of information/accuracy in  $1 - \frac{1}{\varepsilon}$

25 / 38

## Partial Pivoting

- ▶ Partial pivoting: choose at each step the largest element of the column as the pivot
- avoids large elements in factors matrix (**growth factor**)
- ▶ Then ( $P$ : permutation),  $PA = LU$ ,  $Ly = Pb$ ,  $Ux = y$
- ▶  $LU$  with partial pivoting is practically *backward stable*

$$\frac{\|Ax - b\|}{\|A\| \times \|x\| + \|b\|} \approx \varepsilon \quad (1)$$

$$\frac{\|x - x^*\|}{\|x^*\|} \approx \varepsilon \times \kappa(A) \quad (2)$$

- (1) small backward error (and small residual) independently of the conditioning
- (2) accuracy depends on conditioning  
if  $\varepsilon \approx 10^{-q}$  et  $\kappa(A) \approx 10^p$  then  $x$  has approximatively  $(q - p)$  correct digits

27 / 38

$$A = \begin{bmatrix} \varepsilon & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{1}{\varepsilon} & 1 \end{bmatrix} \times \begin{bmatrix} \varepsilon & 1 \\ 0 & 1 - \frac{1}{\varepsilon} \end{bmatrix}$$

- ▶ Let us try to exchange rows 1 and 2 of  $A$  and  $b$ :

$$\begin{bmatrix} 1 & 1 \\ \varepsilon & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 + \varepsilon \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 \\ \varepsilon & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \varepsilon & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 1 - \varepsilon \end{bmatrix}$$

- Multipliers are bounded:  $\forall i = k + 1 : n, \frac{|a_{i,k}^{(k)}|}{|a_{k,k}^{(k)}|} \leq 1$
- terms of original matrix remain significant in  $LU$  factors
- perfect accuracy obtained !

26 / 38

## LU factorization with partial pivoting

Next algorithm computes  $L$  and  $U$  such that  $PA = LU$ , and computes  $Pb$ .

### Algorithm 4 LU factorization with partial pivoting

```

for  $k = 1, n - 1$  do
  Pivot search : Find index  $i$  of largest entry in  $A(k : n, k)$ 
  if  $|A(i, k)| \leq \varepsilon \|A\|$  then
    exit since  $A$  is numerically singular
  end if
  Swap rows  $i$  and  $k$  of  $A$  and  $b$ 
   $A(k+1:n, k) = A(k+1:n, k) / A(k, k)$ 
   $A(k+1:n, k+1:n) = A(k+1:n, k+1:n) - A(k+1:n, k) * A(k, k+1:n)$ 
end for

```

To solve  $Ax = b$  one should thus perform  $LU = Pb$ . Permutation  $P$  is either applied on  $b$  during Algorithm 4 or needs to be saved to be applied later.

28 / 38

### Exercise 4 (LU with pivoting)

1. Explain how to modify algorithm 4 to factor singular matrices in the form proposed at exercise 2 (so that using exercise 2 one could then also compute the null-space of  $\mathbf{A}$ )
2. Let us suppose then that in algorithm 4, we want at each step of **Pivot search** step to find the largest entry not only in the column ( $\mathbf{A}(k : n, k)$ ) but also ( $\mathbf{A}(k : n, k : n)$ ), so called **total pivoting**. Describe how algorithm 4 should be modified.
3. Compare algorithm proposed at questions 1 and 2.

### Linear Algebra Basics

Gaussian Elimination and LU factorization  
 LU Factorization with partial pivoting  
 Symmetric matrices  
 Cholesky Factorization

29 / 38

## Symmetric matrices

- Assumption:  $A$  has a LU factorization
- $A$  symmetric: only store lower or upper triangle
- $A = LU$  and  $A^T = A \Rightarrow LU = U^T L^T$ ,  
 thus  $LU(L^T)^{-1} = U^T \Rightarrow (U)(L^T)^{-1} = L^{-1}U^T = D$  diagonal  
 and  $U = DL^T$ ,  
 finally  $A = L(DL^T) = LDL^T$ , with  $D = \text{Diag}(U)$  and

- Example:

$$\begin{bmatrix} 4 & -8 & -4 \\ -8 & 18 & 14 \\ -4 & 14 & 25 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -1 & 3 & 1 \end{bmatrix} \times \begin{bmatrix} 4 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix} \times \begin{bmatrix} 1 & -2 & -1 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{bmatrix}$$

- Solution of  $Ax = b$  : with  $A = LDL^T$ :

1.  $Ly = b$  then  $Dz = y$  followed by
2.  $L^T x = z$

## Properties of $LDL^T$ Algorithm

We have shown that if  $\mathbf{A}$  symmetric and  $A = LU$  exists then  $\exists L$  and  $D (= \text{Diag}(U))$  such that  $A = LDL^T$ . **LU** Algorithm 1 thus already computes all we need:  $L$  and  $D$ .

### Proposition 1 ( $LDL^T$ Algorithm)

Given a symmetric matrix  $A$  for which an LU factorisation exists, the LU algorithm 1 can be adapted to compute  $LDL^T$  factorization.

### Proposition 2 (Complexity of $LDL^T$ factorization)

If only the lower triangular part of the matrix (including diagonal) is used/updated and if only  $L$  and  $D$  matrices are stored, then the cost of  $LDL^T$  factorisation is  $\approx \frac{n^3}{3}$

31 / 38

30 / 38

32 / 38



- ▶ let  $l_k$  be column  $k$  of  $L$  and  $u_k$  be row  $k$  of  $U$  then in Algorithm 1,  
 $A(k+1:n, k+1:n) = A(k+1:n, k+1:n) - l_k * u_k^T$
- ▶ since  $l_k = u_k / u_{kk}$ , when  $U$  is not stored, one must temporarily save  $u_k$  to perform the update.

## Algorithm 5 LDL<sup>T</sup> factorization

```

for  $k = 1, n-1$  do
  if  $|A(k, k)|$  too small exit (small pivots)
   $\mathbf{v}_k = A(k+1:n, k)$  (corresponds to  $u_k$  in LU Algorithm)
   $A(k+1:n, k) = A(k+1:n, k) / A(k, k)$ 
  for  $j = k+1, n$  do
     $A(j:n, j) = A(j:n, j) - A(j:n, k) * \mathbf{v}_k(j)$ 
  end for
end for
    
```

```

for  $k = 1, n-1$  do
  if  $|A(k, k)|$  too small exit (small pivots)
   $\mathbf{v}_k = A(k+1:n, k)$  (corresponds to  $u_k$  in LU Algorithm)
   $A(k+1:n, k) = A(k+1:n, k) / A(k, k)$ 
  for  $j = k+1, n$  do
     $A(j:n, j) = A(j:n, j) - A(j:n, k) * \mathbf{v}_k(j)$ 
  end for
end for
    
```

▶  $\text{flops}(LDL^T) = 2 \sum_{k=1}^{n-1} \left( \sum_{i=1}^{n-k} i \right) = 2 \sum_{k=1}^{n-1} \left( \frac{(n-k)(n-k+1)}{2} \right)$

$\text{flops}(LDL^T) \approx \sum_{k=1}^{n-1} (n-k)^2$  (thus  $\frac{1}{2} \text{flops}(\mathbf{LU})$ )

$LDL^T \approx \frac{n^3}{3}$  floating point operations

33 / 38

34 / 38

## Symmetric matrices and pivoting

- ▶ Diagonal pivoting preserves symmetry but is insufficient for stability
- ▶ In general one looks for a permutation  $P$  such that:

$$PAP^T = LDL^T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ x & 1 & 0 & 0 \\ x & 0 & 1 & 0 \\ x & x & x & 1 \end{bmatrix} \times \begin{bmatrix} x & 0 & 0 & 0 \\ 0 & x & x & 0 \\ 0 & x & x & 0 \\ 0 & 0 & 0 & x \end{bmatrix} \times \begin{bmatrix} 1 & x & x & x \\ 0 & 1 & 0 & x \\ 0 & 0 & 1 & x \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- ▶  $D$ : matrix of diagonal  $1 \times 1$  and  $2 \times 2$  blocks
- Examples of  $2 \times 2$  pivots:  $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ ,  $\begin{bmatrix} \varepsilon_1 & 1 \\ 1 & \varepsilon_2 \end{bmatrix}$
- ▶ Pivot choice more complex: 2 columns at each step Let  
 $PAP^T = \begin{bmatrix} E & C^T \\ C & B \end{bmatrix}$ . If  $E$  is a  $2 \times 2$  pivot, form  $E^{-1}$  to get:  
 $PAP^T = \begin{bmatrix} I & 0 \\ CE^{-1} & I \end{bmatrix} \begin{bmatrix} E & 0 \\ 0 & B - CE^{-1}C^T \end{bmatrix} \begin{bmatrix} I & E^{-1}C^T \\ 0 & I \end{bmatrix}$

## Linear Algebra Basics

Gaussian Elimination and LU factorization  
 LU Factorization with partial pivoting  
 Symmetric matrices  
 Cholesky Factorization

35 / 38

36 / 38

- ▶ **A** positive definite if  $\mathbf{x}^T \mathbf{A} \mathbf{x} > 0 \quad \forall \mathbf{x} \neq 0$
- ▶ **A** symmetric positive definite  $\Rightarrow$  Cholesky factorization  
 $\mathbf{A} = \mathbf{L} \mathbf{L}^T$  with  $\mathbf{L}$  lower triangular,  $\mathbf{L}$  is unique
- ▶ By identification :

$$\begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{21} & a_{22} & a_{32} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \times \begin{bmatrix} l_{11} & l_{21} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{bmatrix}$$

- It follows:
- $$\begin{array}{lll} a_{11} = l_{11}^2 & a_{21} = l_{21} \times l_{11} & a_{31} = l_{31} \times l_{11} \\ a_{22} = l_{21}^2 + l_{22}^2 & a_{32} = l_{31} \times l_{21} + l_{32} \times l_{22} & a_{33} = l_{31}^2 + l_{32}^2 + l_{33}^2 \end{array}$$

Thus:

$$\begin{array}{lll} l_{11} = \sqrt{a_{11}} & l_{21} = a_{21}/l_{11} & l_{31} = a_{31}/l_{11} \\ a_{22}^{(1)} = a_{22} - l_{21}^2 & a_{32}^{(1)} = a_{32} - l_{31} \times l_{21} & a_{33}^{(1)} = a_{33} - l_{31}^2 \\ l_{22} = \sqrt{a_{22}^{(1)}} & l_{32} = a_{32}^{(1)}/l_{22} & a_{33}^{(2)} = a_{33}^{(1)} - l_{32}^2 \\ l_{33} = \sqrt{a_{33}^{(2)}} & & \end{array}$$

## Cholesky Factorization

```

do k=1, n
  A(k,k)=sqrt(A(k,k))
  A(k+1:n,k) = A(k+1:n,k)/A(k,k)
  do j=k+1, n
    A(j:n,j) = A(j:n,j) - A(j:n,k) A(j,k)
  end do
end do

```

- ▶ Cholesky is backward stable (without pivoting)
- ▶ Factorization:  $\approx \frac{n^3}{3}$  flops
- ▶ Similar to LU, but only on the lower triangle. **LU** factorization:

$$\begin{aligned} A(k+1:n, k) &= A(k+1:n, k) / A(k, k) \\ A(k+1:n, k+1:n) &= A(k+1:n, k+1:n) - & \\ & \quad A(k+1:n, k) * A(k, k+1:n) \end{aligned}$$