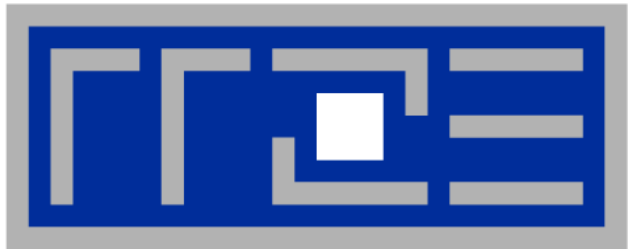

Elementary Numerical Mathematics

Supporting material for

Gaussian Elimination
Matrix factorization
Computational Complexity

Winter Term 2019/20

Gerhard Wellein

HPC Services - Regionales Rechenzentrum Erlangen (RRZE)

Department for Computer Science

Gaussian Elimination

Generating upper triangular form of A

a^0_{11}	a^0_{12}	…	a^0_{1n}
a^0_{21}	a^0_{22}	…	a^0_{2n}
…	…	…	…
a^0_{n1}	a^0_{n2}	…	a^0_{nn}

$A^{(0)} \rightarrow A^{(1)}$

a^0_{11}	a^0_{12}	…	a^0_{1n}
0	a^1_{22}	…	a^1_{2n}
0	…	…	…
0	a^1_{n2}	…	a^1_{nn}

$A^{(1)} \rightarrow A^{(2)}$

a^0_{11}	a^0_{12}	…	a^0_{1n}
0	a^1_{22}	…	a^1_{2n}
0	**0**	…	…
0	**0**	…	a^2_{nn}

In the k-th Gaussian elimination step zeros are introduced below the diagonal element in the k-th column leading to a upper triangular form of A after n-1 steps:

a^0_{11}	a^0_{12}	…	a^0_{1n}
0	a^1_{22}	…	a^1_{2n}
0	**0**	…	…
0	**0**	**0**	a^{n-1}_{nn}

Gaussian Elimination

Why generating upper triangular form of A?

a_{11}	a_{12}	…	a_{1n}
a_{21}	a_{22}	…	a_{2n}
…	…	…	…
a_{n1}	a_{n2}	…	a_{nn}

x_1
x_2
…
x_n

=

b_1
b_2
…
b_n

Gaussian elimination

u_{11}	u_{12}	…	u_{1n}
0	u_{22}	…	u_{2n}
0	**0**	…	…
0	**0**	**0**	u_{nn}

x_1
x_2
…
x_n

=

$\underline{b}_1$
$\underline{b}_2$
…
$\underline{b}_n$

Solve successively – starting with x_n (backward substitution)

…

$\mathbf{x_{n-1}} = (1/\mathbf{u_{n-1,n-1}})\ (\underline{b}_{n-1} - u_{n-1,n}\ \mathbf{x_n})$

$\mathbf{x_n} = \underline{b}_n\ /\ \mathbf{u_{n,n}}$

Upper triangular form of A

Backward-substitution: U x= y

```
Input:  U := matrix(n,n);       y := vector(n)
Output: x := vector(n);         % solution of U x =y

x(n) = y(n) / U(n,n)
for i= n-1 to 1 do
  x(i) = y(i);
  for j=i+1 to n do
       x(i) = x(i) – U(i,j) * x(j);
  end do
  x(i) = x(i) / U(i,i);
end do
```

Operation count:

- Divide: 1 + (n-1) = n
- Mult / Add / Sub:
  $\Sigma(i=1,n-1)\ 2*(n-i) =$

  $2* ((n-1)*n – (n-1)*n/2) = n*(n-1)$

Example: n=3

$$u_{1,1}\ x_1 + u_{1,2}\ x_2 + u_{1,3}\ x_3 = y_1$$
$$u_{2,2}\ x_2 + u_{2,3}\ x_3 = y_2$$
$$u_{3,3}\ x_3 = y_3$$

1) $x_3 = y_3 / u_{3,3}$
2) $x_2 = (y_2 - u_{2,3}\ x_3\ ) / u_{2,2}$
3) $x_1 = (y_1 - u_{1,2}\ x_2 - u_{1,3}\ x_3\ ) / u_{1,1}$

Gaussian Elimination

Structure of augmented matrix after k steps: $\tilde{A}^{(k)}$

$\mathbf{a^0_{11}}$	a^0_{12}	a^0_{13}	a^0_{14}	..	..	..	a^0_{1n}	$\mathbf{a^0_{1n+1}}$
0	$\mathbf{a^1_{22}}$	a^1_{23}	a^1_{24}	..	..	..	a^1_{2n}	$\mathbf{a^1_{2n+1}}$
0	0	$\mathbf{a^2_{33}}$	a^2_{34}	..	..	..	a^2_{3n}	$\mathbf{a^2_{3n+1}}$
0	0	0	..	..	..	..	..	..
0	0	0	0	$\mathbf{a^{k-1}_{kk}}$	..	..	a^{k-1}_{kn}	$\mathbf{a^{k-1}_{k+1n+1}}$
0	0	0	0	0	a^k_{k+1k+1}	..	a^k_{k+1n}	$\mathbf{a^k_{k+1n+1}}$
..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..
0	0	0	0	0	a^k_{nk+1}	..	a^k_{nn}	$\mathbf{a^k_{nn+1}}$

Augmented matrix

Backward-substitution with augmented matrix

```
Input:  A := matrix(n,n+1)   % augmented matrix;
Output: x := vector(n);      % solution of A x = b



x(n) = A(n,n+1) / A(n,n)

for i= n-1 to 1 do
  x(i) = A(i,n+1);

  for j=i+1 to n do
       x(i) = x(i) - A(i,j) * x(j);
  end do

  x(i) = x(i) / A(i,i);

end do
```

Example:
Gaussian elimination in matrix formulation – step 1

Original problem Ax = b:

$$\begin{pmatrix} 1 & 2 & -1 \\ 3 & 8 & -2 \\ -2 & -2 & 6 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 11 \\ -3 \end{pmatrix}$$

$$(E_2 - 3\,E_1) \rightarrow E_2$$
$$(E_3 + 2\,E_1) \rightarrow E_3$$

$$\begin{pmatrix} 1 & 0 & 0 \\ -3 & 1 & 0 \\ +2 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 & -1 \\ 3 & 8 & -2 \\ -2 & -2 & 6 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -3 & 1 & 0 \\ +2 & 0 & 1 \end{pmatrix} \begin{pmatrix} 3 \\ 11 \\ -3 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 2 & -1 \\ 0 & 2 & 1 \\ 0 & 2 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \\ 3 \end{pmatrix}$$

Example:
Gaussian elimination in matrix formulation – step 2

Original problem Ax = b:

$$\begin{pmatrix} 1 & 2 & -1 \\ 3 & 8 & -2 \\ -2 & -2 & 6 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 11 \\ -3 \end{pmatrix}$$

$(E_3 - 1\,E_2) \rightarrow E_3$

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 & -1 \\ 0 & 2 & 1 \\ 0 & 2 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} 3 \\ 2 \\ 3 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 2 & -1 \\ 0 & 2 & 1 \\ 0 & 0 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix}$$

Total transformation matrix applied to original problem Ax = b:

$$\bar{L} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ -3 & 1 & 0 \\ +2 & 0 & 1 \end{pmatrix}$$

The original problem $Ax = b \rightarrow \bar{L}Ax = \bar{L}b \rightarrow Ux = \bar{L}\,b$ with $U = \bar{L}A$

Gaussian Elimination

Matrix formulation – $L^{(k)}$

1	0	0	…	0
$-m_{21}$	1	0	…	0
$-m_{31}$	0	1	…	0
…	…	…	…	…
$-m_{n1}$	0	0	…	1

a^0_{11}	a^0_{12}	…	…	a^0_{1n}
a^0_{21}	a^0_{22}	…	…	a^0_{2n}
a^0_{31}	a^0_{32}	…	…	a^0_{3n}
…	…	…	…	…
a^0_{n1}	a^0_{n2}	…	…	a^0_{nn}

a^0_{11}	a^0_{12}	…	…	a^0_{1n}
0	a^1_{22}	…	…	a^1_{2n}
0	a^1_{32}	…	…	a^1_{3n}
…	…	…	…	…
0	a^1_{n2}	…	…	a^1_{nn}

$A^{(0)} \rightarrow A^{(1)}$

Use $m_{jk} := a^{(k-1)}_{jk} / a^{(k-1)}_{kk}$ (e.g. $m_{21} = a^0_{21} / a^0_{11}$)

Define: $L^{(1)}, L^{(2)}, \ldots, L^{(n-1)}$

1	0	0	…	0
0	1	0	…	0
0	$-m_{32}$	1	…	…
0	…	…	…	…
0	$-m_{n2}$	0	…	1

1	0	…	0	0
0	1	…	0	0
…	…	…	…	0
…	…	…	1	…
0	0	…	$-m_{nn-1}$	1

Gaussian Elimination

Matrix formulation – $L^{(k)-1}$

1	0	0	…	0
m_{21}	1	0	…	0
m_{31}	0	1	…	0
…	…	…	…	…
m_{n1}	0	0	…	1

1	0	0	…	0
$-m_{21}$	1	0	…	0
$-m_{31}$	0	1	…	0
…	…	…	…	…
$-m_{n1}$	0	0	…	1

$L^{(1)}$

1	0	…	…	0
0	1	…	…	0
0	0	…	…	0
…	…	…	…	…
0	0	…	…	1

$L^{(1)-1} L^{(1)} \rightarrow I$

Define: $L^{(1)-1}$, $L^{(2)-1}$, …, $L^{(n-1)-1}$

(Remember $m_{jk} := a^{(k-1)}_{jk} / a^{(k-1)}_{kk}$)

1	0	0	…	0
0	1	0	…	0
0	m_{32}	1	…	…
0	…	…	…	…
0	m_{n2}	0	…	1

1	0	…	0	0
0	1	…	0	0
…	…	…	…	0
…	…	…	1	…
0	0	…	m_{nn-1}	1

Gaussian Elimination

Matrix formulation – $L = L^{(1)-1} L^{(2)-1} \dots L^{(n-1)-1}$

1	0	0	…	0
m_{21}	1	0	…	0
m_{31}	0	1	…	0
…	…	…	…	…
m_{n1}	0	0	…	1

1	0	0	…	0
0	1	0	…	0
0	m_{32}	1	…	0
…	…	…	…	…
0	m_{n2}	0	…	1

1	0	0	…	0
m_{21}	1	0	…	0
m_{31}	m_{32}	1	…	0
…	…	…	…	…
m_{n1}	m_{n2}	0	…	1

1	0	0	…	0	0
$m_{2,1}$	1	0	…	0	0
$m_{3,1}$	$m_{3,2}$	1	…	0	0
…	…	…	…	…	…
$m_{n-1,1}$	$m_{n-1,2}$	$m_{n-1,3}$	…	1	…
$m_{n,1}$	$m_{n,2}$	$m_{n,3}$	…	$m_{n,n-1}$	1

$$L = L^{(1)-1} L^{(2)-1} \dots L^{(n-1)-1}$$

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

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

Gaussian Elimination

LU factorization of A – An in place algorithm

```
Input: A := matrix(n,n);   % contains original values of A
Output: A ;                % a_ij = l_ij if j < i; a_ij = u_ij if j >= i

for k= 1 to n-1 do
  for i = k+1 to n do
    A(i,k) = A(i,k) / A(k,k);                % l_ik=m_ik

    for j=k+1 to n do
       A(i,j) = A(i,j) - A(i,k)*A(k,j); % (E_i - l_ik*E_k) → E_i
    end do

  end do
end do
```

a^0_{11}	a^0_{12}	…	a^0_{1n-1}	a^0_{1n}
m_{21}	a^1_{22}	…	a^1_{2n-1}	a^1_{2n}
m_{31}	m_{32}	…	a^2_{3n-1}	a^2_{3n}
…	…	…	…	…
m_{N1}	m_{N2}	…	m_{N-1N}	a^{n-1}_{nn}

Gaussian Elimination

Permutation matrix – P_{13}

0	0	**1**	0	0
0	1	0	0	0
1	0	**0**	0	0
0	0	0	1	0
0	0	0	0	1

a^0_{11}	a^0_{12}	a^0_{13}	a^0_{14}	a^0_{15}
a^0_{21}	a^0_{22}	a^0_{23}	a^0_{24}	a^0_{25}
a^0_{31}	a^0_{32}	a^0_{33}	a^0_{34}	a^0_{35}
a^0_{41}	a^0_{42}	a^0_{43}	a^0_{44}	a^0_{45}
a^0_{51}	a^0_{52}	a^0_{53}	a^0_{54}	a^0_{55}

PA

a^0_{31}	a^0_{32}	a^0_{33}	a^0_{34}	a^0_{35}
a^0_{21}	a^0_{22}	a^0_{23}	a^0_{24}	a^0_{25}
a^0_{11}	a^0_{12}	a^0_{13}	a^0_{14}	a^0_{15}
a^0_{41}	a^0_{42}	a^0_{43}	a^0_{44}	a^0_{45}
a^0_{51}	a^0_{52}	a^0_{53}	a^0_{54}	a^0_{55}

If $\{k_1,\ldots,k_n\}$ is a permutation of $\{1,\ldots,n\}$ than the permutation matrix P is defined as:
If $(j=k_i)$ then $(P)_{ji} = 1$ else $(P)_{ij}=0$.

Row permutation

$$P\,A^{(0)} = \begin{pmatrix} a^0_{k_1 1} & a^0_{k_1 2} & \cdots & a^0_{k_1 n} \\ a^0_{k_2 1} & a^0_{k_2 2} & \cdots & a^0_{k_2 n} \\ \cdots & \cdots & \cdots & \cdots \\ a^0_{k_n 1} & a^0_{k_n 2} & \cdots & a^0_{k_n n} \end{pmatrix}$$

Cholesky factorization

LL^T factorization of A – An in place algorithm

$l_{1,1}$	$a_{1,2}$	…	$a_{1,n-1}$	$a_{1,n}$
$l_{2,1}$	$l_{2,2}$	…	$a_{2,n-1}$	$a_{2,n}$
$l_{3,1}$	$l_{3,2}$	…	$a_{3,n-1}$	$a_{3,n}$
…	…	…	…	…
$l_{n,1}$	$l_{n,2}$	…	$l_{n-1,n}$	$l_{n,n}$

```
Input: A := matrix(n,n);
Output:A ;

for k= 1 to n do
  for j = 1 to k-1 do
    A(k,k) = A(k,k) – A(k,j)*A(k,j);
  end do
  A(k,k)=sqrt(A(k,k));        → l_k,k

  for i=k+1 to n do
     for j=1 to k-1
       A(i,k) = A(i,k) – A(i,j)*A(k,j);
     end do
     A(i,k)=A(i,k)/A(k,k);    → l_i,k
  end do
end do
```

Refresh: The determinant

- **The following statements are equivalent for any `n x n` matrix `A`**
  - $\det(A) \neq 0$
  - The matrix A is nonsingular: that is A^{-1} exists.
  - The system $Ax=b$ has a unique solution for any vector b.
  - $Ax = 0 \Leftrightarrow x=0$
  - Gaussian elimination can be performed with row interchanges on the system $Ax=b$ for any vector b.
- **Features of the determinant (assuming `n x n` matrices)**
  - If B is obtained from A by the operation
    - $(E_i) \leftrightarrow (E_j)$ with $i \neq j \rightarrow \det(B) = -\det(A)$
    - $(E_i) \rightarrow (\lambda E_i) \rightarrow \det(B) = \lambda \det(A)$
    - $(E_i + \lambda E_j) \rightarrow (E_i)$ with $i \neq j \rightarrow \det(B) = \det(A)$
  - $\det(AB) = \det(A) * \det(B)$
  - $\det(A^t) = \det(A)$
  - If A^{-1} exists: $\det(A^{-1}) = \det(A)^{-1}$
  - If A is upper or lower triangular or diagonal: $\det(A) = \prod a_{i,i}$

Computational Complexity – basic linear operations

- **Scalar product**
  - number of operations 2n → Complexity: $O(n)$

```
s=0
do i=1,n
   s=s + a(i) * b(i)
enddo
```

- **(Dense) Matrix Vector Multiply**
  - number of operations $2n^2$ → Complexity $O(n^2)$

```
do j=1,n
   do i=1,n
      y(j) = y(j) + M(j,i) * x(i)
   enddo
enddo
```

Hardware costs of FP operations

- **1 FP-Multiply & FP-Add hardware unit per processor core –**
  - Each hardware unit: 2 (4) 64-Bit FP operations/cycle with SSE (AVX)
- **Dedicated hardware may be available, but slow operations:** **divide, square root**
- **No dedicated HW → expensive:** **exp, pow, sin/cos/tan …**
- **"64-Bit FP costs" in cycles per Operation for Intel Ivy Bridge (AVX) architecture (Data in L1 cache)**

Operation	y=a+y (y=a*y)	y=a/y	y=sqrt(y)	y=sin(y)
Latency	**3 (5)**	**29**	**29**	**85**
Throughput [cy/instr.]	1 (1)	28	28	85
Cycles/Operation	**0.25**	**7**	**7**	**22-85**

```
DO I=1,N
      A(I) = A(I-1) * s
ENDDO
```

```
DO I=1,N
      A(I) = A(I) * s
ENDDO
```

Upper triangular form of A

Backward-substitution: U x= y

```
Input:  U := matrix(n,n);      y := vector(n)
Output: x := vector(n);        % solution of U x =y

x(n) = y(n) / U(n,n)
for i= n-1 to 1 do
  x(i) = y(i);
  for j=i+1 to n do
       x(i) = x(i) – U(i,j) * x(j);
  end do
  x(i) = x(i) / U(i,i);
end do
```

Operation count:

- Divide: 1 + (n-1) = n
- Mult / Add / Sub:
  $\Sigma(i=1,n-1)\ 2*(n-i) =$

  $2* ((n-1)*n – (n-1)*n/2) = n*(n-1)$

Example: n=3

$$u_{1,1}\ x_1 + u_{1,2}\ x_2 + u_{1,3}\ x_3 = y_1$$
$$u_{2,2}\ x_2 + u_{2,3}\ x_3 = y_2$$
$$u_{3,3}\ x_3 = y_3$$

1) $x_3 = y_3 / u_{3,3}$
2) $x_2 = (y_2 - u_{2,3}\ x_3\ ) / u_{2,2}$
3) $x_1 = (y_1 - u_{1,2}\ x_2 - u_{1,3}\ x_3\ ) / u_{1,1}$

Gaussian Elimination

LU factorization of A – An in place algorithm

```
Input: A := matrix(n,n);   % contains original values of A
Output: A ;                % a_ij = l_ij if j < i; a_ij = u_ij if j >= i

for k= 1 to n-1 do
  for i = k+1 to n do
    A(i,k) = A(i,k) / A(k,k);                % l_ik=m_ik

    for j=k+1 to n do
       A(i,j) = A(i,j) - A(i,k)*A(k,j); % (E_i - l_ik*E_k) → E_i
    end do

  end do
end do
```

a^0_{11}	a^0_{12}	…	a^0_{1n-1}	a^0_{1n}
m_{21}	a^1_{22}	…	a^1_{2n-1}	a^1_{2n}
m_{31}	m_{32}	…	a^2_{3n-1}	a^2_{3n}
…	…	…	…	…
m_{N1}	m_{N2}	…	m_{N-1N}	a^{n-1}_{nn}

Computational Complexity: Memory footprint

- For algorithms involving dense matrices available main memory may also become limiting factor

- $8 \times n^2$ Bytes are required to store $n \times n$ matrix A (assuming 64-Bit)

Required for Gaussian elimination

Assuming 1 TFlop/s (10^{12} Flop/s)

n	FP operations	Time to solution	Memory footprint
10^3	$^2/_3 \times 10^9$	$^2/_3 \times 10^{-3}\ s$	$8\ MBytes$
10^6	$^2/_3 \times 10^{18}$	$7.7\ days$	**8 TBytes**

- Our Emmy compute cluster (560 nodes / 11,000+ cores / 35 Tbytes / 200 Tflop/s):

$$n = 1.9 \times 10^6 \rightarrow Time: \sim 7\ hrs$$