LU factorization

LU factorisation, consists in looking for two matrices L lower triangular, and U upper triangular, both non-singular, such that

$$LU = A \tag{1}$$

If we find these matrices, the original system $A\underline{x} = \underline{b}$ splits into two triangular systems easy to solve:

$$A\underline{x} = \underline{b} \;\rightarrow\; L(U\underline{x}) = \underline{b} \;\rightarrow\; \begin{cases} L\underline{y} = \underline{b} & \text{solved forward} \\ U\underline{x} = \underline{y} & \text{solved backward} \end{cases}$$

Mathematically equivalent to GEM: matrix U is the same, $\widetilde{b} = L^{-1}b$.

LU: unicity of the factors

The factorization $LU = A$ is unique?

$$\underbrace{\begin{bmatrix} l_{11} & 0 & \cdots & 0 \\ l_{21} & l_{22} & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ 0 & u_{22} & \cdots & u_{2n} \\ \vdots & \vdots & & \vdots \\ 0 & & \cdots & u_{nn} \end{bmatrix}}_{U} = \underbrace{\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}}_{A}$$

The unknowns are the coefficients l_{ij} of L, which are $1 + 2 + \cdot + n = \dfrac{n(n+1)}{2}$, and the coefficients u_{ij} of U, also $\dfrac{n(n+1)}{2}$, for a total of $n^2 + n$ unknowns.
We only have n^2 equations (as many as the number of coefficients of A), so we need to fix n unknowns. Usually, the diagonal coefficients of L are set equal to 1: $l_{ii} = 1$. If you do so...

How to compute L and U

Let us introduce Atomic Lower Triangular matrix $L^{(k)}$ defined as

$$L^{(k)} = \begin{bmatrix} 1 & & & & 0 \\ & \ddots & & & \\ & & 1 & & \\ & & -l_{k+1,k} & \ddots & \\ & & \vdots & & \\ 0 & & \underbrace{-l_{n,k}}_{\text{k-th column}} & & 1 \end{bmatrix}$$

The basic version of GEM can be rewritten in terms of matrix multiplications as:

$$A^{(1)} = A, \quad A^{(2)} = L^{(1)}A, \quad \ldots, \quad A^{(n)} = L^{(n-1)}L^{(n-2)} \cdots L^{(1)}A = U,$$

where U is upper triangular.

We observe that

$$L^{(k)^{-1}} = \begin{bmatrix} 1 & & & & 0 \\ & \ddots & & & \\ & & 1 & & \\ & & l_{k+1,k} & \ddots & \\ & & \vdots & & \\ 0 & & \underbrace{l_{n,k}}_{\text{k-th column}} & & 1 \end{bmatrix}, \text{ is still lower triangular}$$

Let us define L as:

$$L := \left(L^{(n-1)} \dots L^{(1)}\right)^{-1} = L^{(1)^{-1}} \dots L^{(n)^{-1}}, \text{ that is still lower triangular}$$

and contains the coefficients $l_{i,k}$ computed in GEM:

$$L = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ l_{2,1} & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ l_{n,1} & l_{n,2} & \cdots & 1 \end{bmatrix}$$

Recalling that $L^{(n-1)} L^{(n-2)} \dots L^{(1)} A = U$, we obtain $A = LU$.

GEM vs LU (pseudocodes)

GEM

Input: $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$
for $k = 1, \ldots, n-1$
  for $i = k+1, \ldots, n$
    $l_{i,k} = a_{i,k}/a_{k,k}$
    $a_{i,k:n} = a_{i,k:n} - l_{i,k} a_{k,k:n}$
    $b_i = b_i - l_{i,k} b_k$
  end
end
set $U = A$, then solve $Ux = b$ with back substitution

- we can store the $l_{i,k}$in a matrix:

$$L = \begin{bmatrix} ? & ? & \cdots & ? \\ l_{2,1} & ? & \cdots & ? \\ \vdots & \vdots & & \vdots \\ l_{n,1} & l_{n,2} & \cdots & ? \end{bmatrix}$$

that can be completed as a lower triangular matrix

$$L = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ l_{2,1} & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ l_{n,1} & l_{n,2} & \cdots & 1 \end{bmatrix}$$

- the loops on the left replace $\underline{b}$ by $L^{-1}\underline{b}$ (see the "equivalent forward substitution" algoritms): $\underline{b} \leftarrow L^{-1}\underline{b}$
- similarly $A \leftarrow L^{-1}A$, which is called U and is an upper triangular matrix:

$$L^{-1}A = U \quad \Rightarrow \quad A = LU$$

GEM vs LU (pseudocodes)

GEM

Input: $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$
for $k = 1, \ldots, n-1$
  for $i = k+1, \ldots, n$
    $l_{i,k} = a_{i,k}/a_{k,k}$
    $a_{i,k:n} = a_{i,k:n} - l_{i,k} a_{k,k:n}$
    $b_i = b_i - l_{i,k} b_k$
  end
end
set $U = A$, then solve $Ux = b$ with backward substitution

LU

Input: $A \in \mathbb{R}^{n\times n}$
$L = I_n \in \mathbb{R}^{n\times n}$
for $k = 1, \ldots, n-1$
  for $i = k+1, \ldots, n$
    $l_{i,k} = a_{i,k}/a_{k,k}$
    $a_{i,k:n} = a_{i,k:n} - l_{i,k} a_{k,k:n}$
  end
end
Set $U = A$ and output: U and L

Homework

Write the LU algorithm in MATLAB

Permutation matrices

$P \in \mathbb{R}^{n\times n}$ is a permutation matrix if it has only one entry 1 on each row and each column, while the remaining entries are all 0. P produces permutation of rows when multiplying on the left and of columns when multiplying on the right. For example:

$$P = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad A = \begin{bmatrix} \text{---} r_1 \text{---} \\ \text{---} r_2 \text{---} \\ \text{---} r_3 \text{---} \\ \text{---} r_4 \text{---} \end{bmatrix} = \begin{bmatrix} | & | & | & | \\ c_1 & c_2 & c_3 & c_4 \\ | & | & | & | \end{bmatrix}$$

$$PA = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \text{---} r_1 \text{---} \\ \text{---} r_2 \text{---} \\ \text{---} r_3 \text{---} \\ \text{---} r_4 \text{---} \end{bmatrix} = \begin{bmatrix} \text{---} r_3 \text{---} \\ \text{---} r_1 \text{---} \\ \text{---} r_2 \text{---} \\ \text{---} r_4 \text{---} \end{bmatrix}$$

$$AP = \begin{bmatrix} | & | & | & | \\ c_1 & c_2 & c_3 & c_4 \\ | & | & | & | \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} | & | & | & | \\ c_2 & c_3 & c_1 & c_4 \\ | & | & | & | \end{bmatrix}$$

Remark 1

The product of permutation matrices is still a permutation matrix

GEM: possible troubles and remedy

The condition $det(A) \neq 0$ is not sufficient to guarantee that the elimination procedure will be successful. For example $A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

To avoid this the remedy is the "pivoting" algorithm:

• first step: before eliminating the first column, look for the coefficient of the column biggest in absolute value, the so-called "pivot"; if r is the row where the pivot is found, exchange the first and the r^{th} row.

• second step: before eliminating the second column, look for the coefficient of the column biggest in absolute value, starting from the second row; if r is the row where the pivot is found, exchange the second and the r^{th} row.

⋮

• step j: before eliminating the column j, look for the pivot in this column, from the diagonal coefficient down to the last row. If the pivot is found in the row r, exchange the rows j and r.

This is the pivoting procedure on the rows, which amounts to multiply at the left the matrix A by a permutation matrix P. (An analogous procedure can be applied on the columns, or globally).

Lemma 1

Let $A \in \mathbb{R}^{n\times n}$ be a non singular matrix. Then, at each step of GEM, the "pivot" is not null.

Remark 2

If A is non singular, Lemma 1 ensures that GEM with pivoting can be successfully completed.

The pivoting procedure corresponds then to solve, instead of the original system $A\underline{x} = \underline{b}$, the system

$$PA\underline{x} = P\underline{b} \tag{2}$$

GEM pseudocode with pivoting

Input as before: $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$

for $k = 1, \ldots, n-1$

  select $j \geq k$ that maximise $|a_{jk}|$

  $a_{j,k:n} \longleftrightarrow a_{k,k:n}$

  $b_j \longleftrightarrow b_k$

  for $i = k+1, \ldots, n$

    $l_{i,k} = a_{i,k}/a_{k,k}$

    $a_{i,k:n} = a_{i,k:n} - l_{i,k} a_{k,k:n}$

    $b_i = b_i - l_{i,k} b_k$

  end

end

define $U = A$, $\widetilde{b} = b$, then solve $Ux = \widetilde{b}$ with backward substitution

remark: we do not need to define U and $\widetilde{b}$, it is just to be consistent with the notation of the previous slides

LU-continued

GEM and LU have the same computational cost: $\frac{2}{3}n^3$

In GEM, the coefficients l_{ik} are discarded after application to the right-hand side b, while in the LU factorisation they are stored in the matrix L.
If we have to solve a single linear system, GEM is preferable (less memory storage).
If we have to solve many systems with the same matrix and different right-hand sides LU is preferable (the heavy cost is payed only once).

Pivoting is applied also to the LU factorization to ensure that the factorisation is succesful

$$PA = LU \quad \Longrightarrow \quad PA\underline{x} = P\underline{b} \;\rightarrow\; L(U\underline{x}) = P\underline{b} \;\rightarrow\; \begin{cases} L\underline{y} = P\underline{b} \\ U\underline{x} = \underline{y} \end{cases}$$

The Matlab function that computes L and U is $lu(.,.)$.

How to compute L, U and P

GEM with pivoting can be rewritten in terms of matrix multiplications as:

$$A^{(1)} = A, \quad A^{(2)} = L^{(1)}P^{(1)}A, \quad \dots,$$
$$A^{(n)} = L^{(n-1)}P^{(n-1)}L^{(n-2)}\dots L^{(1)}P^{(1)}A = U,$$

where U is upper triangular.

Given two generic matrices M, N, it holds $MN \neq NM$, but if $P^{(j)}$ is a permutation matrix that switches i-th and j-th rows, then, for $i \geq j > k$:

$$P^{(j)}L^{(k)} = \widetilde{L}^{(k)}P^{(j)},$$

where $\widetilde{L}^{(k)}$ is obtained from $L^{(k)}$ by switching $L^{(k)}_{i,k}$ and $L^{(k)}_{j,k}$.

$\widetilde{L}^{(k)}$ is still atomic lower triangular, thus $\widetilde{L}^{(k)^{-1}}$ is obtained from $\widetilde{L}^{(k)}$ by changing signs of the off-diagonal coefficients.

$$
\begin{aligned}
U =& L^{(n-1)} P^{(n-1)} L^{(n-2)} P^{(n-2)} L^{(n-3)} \dots L^{(1)} P^{(1)} A \\
=& L^{(n-1)} \widetilde{L}^{(n-2)} P^{(n-1)} P^{(n-2)} L^{(n-3)} P^{(n-3)} \dots L^{(1)} P^{(1)} A \\
=& L^{(n-1)} \widetilde{L}^{(n-2)} \widetilde{\widetilde{L}}^{(n-3)} P^{(n-1)} P^{(n-2)} P^{(n-3)} \dots L^{(1)} P^{(1)} A \\
=& \left(L^{(n-1)} \widetilde{L}^{(n-2)} \widetilde{\widetilde{L}}^{(n-3)} \widetilde{\widetilde{\widetilde{L}}}^{(n-4)} \dots \right) P^{(n-1)} P^{(n-2)} P^{(n-3)} \dots P^{(1)} A
\end{aligned}
$$

Defining

$$
\begin{aligned}
L =& \left(L^{(n-1)} \widetilde{L}^{(n-2)} \widetilde{\widetilde{L}}^{(n-3)} \widetilde{\widetilde{\widetilde{L}}}^{(n-4)} \dots \right)^{-1} \\
P =& P^{(n-1)} P^{(n-2)} P^{(n-3)} \dots P^{(1)}
\end{aligned}
$$

we get

$$
LU = PA.
$$

LU with pivoting (pseudocode)

Input as before: $A \in \mathbb{R}^{n \times n}$
$L = I_n \in \mathbb{R}^{n \times n}$
$P = I_n \in \mathbb{R}^{n \times n}$
for $k = 1, \ldots, n-1$
    select $j \geq k$ that maximise $|a_{jk}|$
    $a_{j,k:n} \longleftrightarrow a_{k,k:n}$
    $p_{j,:} \longleftrightarrow p_{k,:}$
    if $k \geq 2$
        $l_{j,1:k-1} \longleftrightarrow l_{k,1:k-1}$
    end
    for $i = k+1, \ldots, n$
        $l_{i,k} = a_{i,k} / a_{k,k}$
        $a_{i,k:n} = a_{i,k:n} - l_{i,k} a_{k,k:n}$
    end
end
define $U = A$ and output: U, L and P

LU versus GEM

If one needs to compute the inverse of a matrix, LU is the cheapest way. Indeed, recalling the definition, the inverse of a matrix A is the matrix A^{-1} solution of

$$AA^{-1} = I$$

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & & \vdots \\ \underbrace{c_{n1}}_{\underline{c}^{(1)}} & \underbrace{c_{n2}}_{\underline{c}^{(2)}} & \cdots & \underbrace{c_{nn}}_{\underline{c}^{(n)}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ \underbrace{0}_{\underline{e}^1} & \underbrace{0}_{\underline{e}^2} & \cdots & \underbrace{1}_{\underline{e}^n} \end{bmatrix}$$

$$
\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}
\begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & & \vdots \\ \underbrace{c_{n1}}_{\underline{c}^{(1)}} & \underbrace{c_{n2}}_{\underline{c}^{(2)}} & \cdots & \underbrace{c_{nn}}_{\underline{c}^{(n)}} \end{bmatrix}
=
\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ \underbrace{0}_{\underline{e}^{1}} & \underbrace{0}_{\underline{e}^{2}} & \cdots & \underbrace{1}_{\underline{e}^{n}} \end{bmatrix}
$$

Hence, each column $\underline{c}^{(i)}$ of A^{-1} is the solution of

$$
A\underline{c}^{(i)} = \underline{e}^{(i)}, \qquad i = 1, 2, \cdots, n
$$

with $\underline{e}^{(i)} = (0, 0, \cdots, 1, \cdots, 0)$. The factorisation can be done once and for all at the cost of $O(2n^3/3)$ operations; for each column we have to solve 2 triangular systems ($2n^2$ operations) so that the total cost is of the order of $\frac{2}{3}n^3 + n \times 2n^2 = \frac{8}{3}n^3$.

In case of pivoting, we solve $PA\underline{c}^{(i)} = P\underline{e}^{(i)} = P_{:,i}$, $i = 1, 2, \cdots, n$.

Computation of the determinant

We can use the LU factorisation to compute the determinant of a matrix. Indeed, if $A = LU$, thanks to the Binet theorem we have

$$\det(A) = \det(L)\det(U) = \prod_{i=1}^{n} l_{ii} \prod_{i=1}^{n} u_{ii} = \prod_{i=1}^{n} u_{ii}$$

Thus the cost to compute the determinant is the same of the LU factorisation.
In the case of pivoting, $PA = LU$ and then

$$\det(A) = \frac{\det(L)\det(U)}{\det(P)} = \frac{\det(U)}{\det(P)}$$

It turns out that $\det(P) = (-1)^{\delta}$ where $\delta = \#$ of row exchanges in the LU factorisation.
Matlab function: det(·)

Cholesky factorization

If A is symmetric ($A = A^T$) and positive definite (positive eigenvalues) a variant of LU is due to Cholesky: there exists a non-singular lower triangular matrix L such that

$$LL^T = A$$

Costs: approximately $\sim \dfrac{n^3}{3}$ (half the cost of LU, using the symmetry of A).

Matlab function: chol(.,.)