Introduction to high performance scientific computing
Numerical linear algebra

MAM5 - INUM, Polytech Nice Sophia

Stéphane Lanteri
Stephane.Lanteri@inria.fr

Nachos project-team
Inria Sophia Antipolis - Méditerranée research center, France

December 28, 2016

Physical phenomenon

$\Downarrow$

Mathematical (continuous) model (system of PDEs)

$\Downarrow$

Formulation of a discrete model (discretization in space, time integration, etc.)
and numerical analysis (stability, convergence, etc.)

$\Downarrow$

Numerical algorithms and computer implementation

- The operations $Ax = b$ and $Ax = \lambda x$ are central numerical kernels in the solution of numerous scientific computing problems
  - Numerical solution of systems of PDEs:
    finite difference, finite element and finite volume methods,
    boundary element methods,
    structured (cartesian, body-fitted) grids,
    unstructured grids.
  - Relevant topics:
    sparse versus dense systems,
    symmetric versus non-symmetric systems,
    definite versus indefinite systems,
    arithmetic types (real versus complex),
    etc.

References

- Numerical linear algebra

  `Numerical linear algebra`
  Grégoire Allaire and Sidi Mahmoud Kaber

  `Numerical linear algebra`
  Grégoire Allaire and Sidi Mahmoud Kaber
  Texts in Applied Mathematics, Vo. 55
  Springer, 2008

  `Computer solution of large linear systems`
  Gérard Meurant
  Studies in Mathematics and its Applications, Vol. 28
  North Holland, 1999

  `Iterative methods for sparse linear systems`
  Yousef Saad
  SIAM, 2003

Outline

- $\mathbb{K} = \mathbb{R}$ or $\mathbb{C}$

Definition 1.1

A matrix A is a rectangular array $(a_{i,j})_{1 \leq i \leq n, 1 \leq j \leq p}$ where $a_{i,j} \in \mathbb{K}$ is the entry in row i and column j,

$$A = \begin{pmatrix} a_{1,1} & \cdots & a_{1,p} \\ \vdots & & \vdots \\ a_{n,1} & \cdots & a_{n,p} \end{pmatrix}.$$

The set of all matrices of size $n \times p$ (n rows and p columns) is denoted $\mathcal{M}_{n,p}(\mathbb{K})$.

Definition 1.2

Let A and B be two matrices in $\mathcal{M}_{n,p}(\mathbb{K})$.

The sum $A + B$ is the matrix in $\mathcal{M}_{n,p}(\mathbb{K})$ defined by,

$$A + B = (a_{i,j} + b_{i,j})_{1 \leq i \leq n, 1 \leq j \leq p}.$$

- $\mathbb{K} = \mathbb{R}$ or $\mathbb{C}$

Definition 1.1

A matrix A is a rectangular array $(a_{i,j})_{1 \leq i \leq n, 1 \leq j \leq p}$ where $a_{i,j} \in \mathbb{K}$ is the entry in row i and column j,

$$A = \begin{pmatrix} a_{1,1} & \cdots & a_{1,p} \\ \vdots & & \vdots \\ a_{n,1} & \cdots & a_{n,p} \end{pmatrix}.$$

The set of all matrices of size $n \times p$ (n rows and p columns) is denoted $\mathcal{M}_{n,p}(\mathbb{K})$.

Definition 1.2

Let A and B be two matrices in $\mathcal{M}_{n,p}(\mathbb{K})$.

The sum $A + B$ is the matrix in $\mathcal{M}_{n,p}(\mathbb{K})$ defined by,

$$A + B = (a_{i,j} + b_{i,j})_{1 \leq i \leq n, 1 \leq j \leq p}.$$

Definition 1.3

Let A be a matrix in $\mathcal{M}_{n,p}(\mathbb{K})$ and $\lambda \in \mathbb{R}$.

The scalar multiplication of A by λ is the matrix in $\mathcal{M}_{n,p}(\mathbb{K})$ defined by,

$$\lambda A = (\lambda a_{i,j})_{1 \leq i \leq n, 1 \leq j \leq p}.$$

Definition 1.4

Let A and B be two matrices respectively in $\mathcal{M}_{n,p}(\mathbb{K})$ and $\mathcal{M}_{p,q}(\mathbb{K})$.

The product AB is the matrix in $\mathcal{M}_{n,q}(\mathbb{K})$ defined by,

$$AB = \left(\sum_{k=1}^{p} a_{i,k} b_{k,j}\right)_{1 \leq i \leq n, 1 \leq j \leq q}.$$

- Associativity: $(MN)P = M(NP)$.
- In general, $AB \neq BA$.

Definition 1.3

Let A be a matrix in $\mathcal{M}_{n,p}(\mathbb{K})$ and $\lambda \in \mathbb{R}$.

The scalar multiplication of A by λ is the matrix in $\mathcal{M}_{n,p}(\mathbb{K})$ defined by,

$$\lambda A = (\lambda a_{i,j})_{1 \leq i \leq n, 1 \leq j \leq p}.$$

Definition 1.4

Let A and B be two matrices respectively in $\mathcal{M}_{n,p}(\mathbb{K})$ and $\mathcal{M}_{p,q}(\mathbb{K})$.

The product AB is the matrix in $\mathcal{M}_{n,q}(\mathbb{K})$ defined by,

$$AB = (\sum_{k=1}^{p} a_{i,k} b_{k,j})_{1 \leq i \leq n, 1 \leq j \leq q}.$$

- Associativity: $(MN)P = M(NP)$.
- In general, $AB \neq BA$.

Definition 1.3

Let A be a matrix in $\mathcal{M}_{n,p}(\mathbb{K})$ and $\lambda \in \mathbb{R}$.

The scalar multiplication of A by λ is the matrix in $\mathcal{M}_{n,p}(\mathbb{K})$ defined by,

$$\lambda A = (\lambda a_{i,j})_{1 \leq i \leq n, 1 \leq j \leq p}.$$

Definition 1.4

Let A and B be two matrices respectively in $\mathcal{M}_{n,p}(\mathbb{K})$ and $\mathcal{M}_{p,q}(\mathbb{K})$.

The product AB is the matrix in $\mathcal{M}_{n,q}(\mathbb{K})$ defined by,

$$AB = (\sum_{k=1}^{p} a_{i,k} b_{k,j})_{1 \leq i \leq n, 1 \leq j \leq q}.$$

- Associativity: $(MN)P = M(NP)$.
- In general, $AB \neq BA$.

Definition 1.5

Let A be a matrix in $\mathcal{M}_{n,p}(\mathbb{K})$.

The transpose matrix A^t is the matrix in $\mathcal{M}_{p,n}(\mathbb{K})$ defined by,

$$A^t = (a_{j,i})_{1\leq j\leq p, 1\leq i\leq n} = \begin{pmatrix} a_{1,1} & \cdots & a_{n,1} \\ \vdots & & \vdots \\ a_{1,p} & \cdots & a_{n,p} \end{pmatrix}.$$

If $A = A^t$ (which can happen only if A is a square matrix, i.e. if $n = p$), then A is said to be symmetric.

Definition 1.6

A square matrix $A \in \mathcal{M}_n(\mathbb{K})$ is said to be invertible (or nonsingular) if there exists a matrix $B \in \mathcal{M}_n(\mathbb{K})$ such that $AB = BA = I_n$ where $(I_n)_{1\leq i,j\leq n} = \delta_{i,j}$ is the identity matrix of $A \in \mathcal{M}_n(\mathbb{K})$.

This matrix B is denoted by A^{-1} and is called the inverse matrix of A.

Definition 1.5

Let A be a matrix in $\mathcal{M}_{n,p}(\mathbb{K})$.

The transpose matrix A^t is the matrix in $\mathcal{M}_{p,n}(\mathbb{K})$ defined by,

$$A^t = (a_{j,i})_{1\leq j\leq p, 1\leq i\leq n} = \begin{pmatrix} a_{1,1} & \cdots & a_{n,1} \\ \vdots & & \vdots \\ a_{1,p} & \cdots & a_{n,p} \end{pmatrix}.$$

If $A = A^t$ (which can happen only if A is a square matrix, i.e. if $n = p$), then A is said to be symmetric.

Definition 1.6

A square matrix $A \in \mathcal{M}_n(\mathbb{K})$ is said to be invertible (or nonsingular) if there exists a matrix $B \in \mathcal{M}_n(\mathbb{K})$ such that $AB = BA = I_n$ where $(I_n)_{1\leq i,j\leq n} = \delta_{i,j}$ is the identity matrix of $A \in \mathcal{M}_n(\mathbb{K})$.

This matrix B is denoted by A^{-1} and is called the inverse matrix of A.

Definition and properties of matrices

- A noninvertible matrix is said to be singular.
- The kernel, or null space, of a matrix $A \in \mathcal{M}_{n,p}(\mathbb{K})$ is the set of vectors $x \in \mathbb{K}^p$ such that $Ax = 0$, and is denoted by $\texttt{Ker}\ A$.
- The image, or range, of A is the set of vectors $y \in \mathbb{K}^n$ such that $y = Ax$, with $x \in \mathbb{K}^p$, and is denoted by $\texttt{Im}\ A$.
- The dimension of the linear space $\texttt{Im}\ A$ is called the rank of A, and is denoted by $\texttt{rk}\ A$.

Lemma 1.7

For any matrix $A \in \mathcal{M}_n(\mathbb{K})$ the following statements are equivalent:

1. A is invertible,
2. $\texttt{Ker}\ A = 0$,
3. $\texttt{Im}\ A = \mathbb{K}^n$,
4. there exists $B \in \mathcal{M}_n(\mathbb{K})$ such that $AB = I_n$,
5. there exists $B \in \mathcal{M}_n(\mathbb{K})$ such that $BA = I_n$.

In the last two cases the matrix B is precisely equal to A^{-1}.

Definition and properties of matrices

- A noninvertible matrix is said to be singular.
- The kernel, or null space, of a matrix $A \in \mathcal{M}_{n,p}(\mathbb{K})$ is the set of vectors $x \in \mathbb{K}^p$ such that $Ax = 0$, and is denoted by $\texttt{Ker}\ A$.
- The image, or range, of A is the set of vectors $y \in \mathbb{K}^n$ such that $y = Ax$, with $x \in \mathbb{K}^p$, and is denoted by $\texttt{Im}\ A$.
- The dimension of the linear space $\texttt{Im}\ A$ is called the rank of A, and is denoted by $\texttt{rk}\ A$.

Lemma 1.7

For any matrix $A \in \mathcal{M}_n(\mathbb{K})$ the following statements are equivalent:

1. A is invertible,
2. $\texttt{Ker}\ A = 0$,
3. $\texttt{Im}\ A = \mathbb{K}^n$,
4. there exists $B \in \mathcal{M}_n(\mathbb{K})$ such that $AB = I_n$,
5. there exists $B \in \mathcal{M}_n(\mathbb{K})$ such that $BA = I_n$.

In the last two cases the matrix B is precisely equal to A^{-1}.

Lemma 1.8

Let A and B be two invertible matrices in $\mathcal{M}_n(\mathbb{K})$. Then,

$$(AB)^{-1} = B^{-1}A^{-1}.$$

Definition 1.9

A permutation of order n is a one-to-one mapping from the set $\{1, 2, \cdots, n\}$ into itself. We denote by $\mathcal{S}_n$ the set of all permutations of order n. The signature of a permutation σ is the number $\varepsilon(\sigma)$, equal to $+1$ or -1 defined by,

$$\varepsilon(\sigma) = (-1)^{p(\sigma)} \quad \text{with} \quad p(\sigma) = \sum_{1 \leq i \leq j \leq n} \mathtt{Inv}_\sigma(i,j),$$

where the number $\mathtt{Inv}_\sigma(i,j)$ indicates whether the order between i and j is inverted or not by the permutation σ, and is defined for $i \leq j$ by,

$$\mathtt{Inv}_\sigma(i,j) = \begin{cases} 0 & \text{if} \quad \sigma(i) \leq \sigma(j), \\ 1 & \text{if} \quad \sigma(i) > \sigma(j). \end{cases}$$

Lemma 1.8

Let A and B be two invertible matrices in $\mathcal{M}_n(\mathbb{K})$. Then,

$$(AB)^{-1} = B^{-1}A^{-1}.$$

Definition 1.9

A permutation of order n is a one-to-one mapping from the set $\{1, 2, \cdots, n\}$ into itself. We denote by $\mathcal{S}_n$ the set of all permutations of order n. The signature of a permutation σ is the number $\varepsilon(\sigma)$, equal to $+1$ or -1 defined by,

$$\varepsilon(\sigma) = (-1)^{p(\sigma)} \quad \text{with} \quad p(\sigma) = \sum_{1 \leq i \leq j \leq n} \mathtt{Inv}_\sigma(i,j),$$

where the number $\mathtt{Inv}_\sigma(i,j)$ indicates whether the order between i and j is inverted or not by the permutation σ, and is defined for $i \leq j$ by,

$$\mathtt{Inv}_\sigma(i,j) = \begin{cases} 0 & \text{if} \quad \sigma(i) \leq \sigma(j), \\ 1 & \text{if} \quad \sigma(i) > \sigma(j). \end{cases}$$

Definition 1.10

The determinant of a square matrix $A \in \mathcal{M}_n(\mathbb{K})$ is,

$$\det(A) = \sum_{\sigma \in \mathcal{S}_n} \varepsilon(\sigma) \prod_{i=1}^{n} a_{i,\sigma(i)}.$$

Lemma 1.11

Let A and B be two square matrices in $\mathcal{M}_n(\mathbb{K})$. Then,

1. $\det(AB) = \det(A)\det(B) = \det(BA)$,
2. $\det(A^t) = \det(A)$,
3. A is invertible if and only if $\det(A) \neq 0$.

Definition 1.10

The determinant of a square matrix $A \in \mathcal{M}_n(\mathbb{K})$ is,

$$\det(A) = \sum_{\sigma \in \mathcal{S}_n} \varepsilon(\sigma) \prod_{i=1}^{n} a_{i,\sigma(i)}.$$

Lemma 1.11

Let A and B be two square matrices in $\mathcal{M}_n(\mathbb{K})$. Then,

1. $\det(AB) = \det(A)\det(B) = \det(BA)$,
2. $\det(A^t) = \det(A)$,
3. A is invertible if and only if $\det(A) \neq 0$.

Definition 1.12

A square matrix $A \in \mathcal{M}_n(\mathbb{K})$ is said to be diagonal if its entries satisfy $a_{i,j} = 0$ for $i \neq j$. A diagonal matrix is often denoted by $A = \texttt{diag}(a_{1,1}, \cdots, a_{n,n})$.

Definition 1.13

Let T be a matrix in $\mathcal{M}_{n,p}(\mathbb{K})$. It is said to be an upper triangular matrix if $t_{i,j} = 0$ for all indices (i, j) such that $i > j$. It is said to be a lower triangular matrix if $t_{i,j} = 0$ for all indices (i, j) such that $i < j$.

Lemma 1.14

Let T be a lower triangular matrix (respectively, upper triangular) in $\mathcal{M}_n(\mathbb{K})$. Its inverse (when it exists) is also a lower triangular matrix (respectively, upper triangular matrix) with diagonal entries equal to the inverse of the diagonal entries of T.

Let $T^{'}$ be another lower triangular matrix (respectively, upper triangular) in $\mathcal{M}_n(\mathbb{K})$. The product $TT^{'}$ is also a lower triangular matrix (respectively, upper triangular) with diagonal entries equal to the product of the diagonal entries of T and $T^{'}$.

Definition 1.12

A square matrix $A \in \mathcal{M}_n(\mathbb{K})$ is said to be diagonal if its entries satisfy $a_{i,j} = 0$ for $i \neq j$. A diagonal matrix is often denoted by $A = \texttt{diag}(a_{1,1}, \cdots, a_{n,n})$.

Definition 1.13

Let T be a matrix in $\mathcal{M}_{n,p}(\mathbb{K})$. It is said to be an upper triangular matrix if $t_{i,j} = 0$ for all indices (i,j) such that $i > j$. It is said to be a lower triangular matrix if $t_{i,j} = 0$ for all indices (i,j) such that $i < j$.

Lemma 1.14

Let T be a lower triangular matrix (respectively, upper triangular) in $\mathcal{M}_n(\mathbb{K})$. Its inverse (when it exists) is also a lower triangular matrix (respectively, upper triangular matrix) with diagonal entries equal to the inverse of the diagonal entries of T.

Let $T^{'}$ be another lower triangular matrix (respectively, upper triangular) in $\mathcal{M}_n(\mathbb{K})$. The product $TT^{'}$ is also a lower triangular matrix (respectively, upper triangular) with diagonal entries equal to the product of the diagonal entries of T and $T^{'}$.

Definition 1.12

A square matrix $A \in \mathcal{M}_n(\mathbb{K})$ is said to be diagonal if its entries satisfy $a_{i,j} = 0$ for $i \neq j$. A diagonal matrix is often denoted by $A = \texttt{diag}(a_{1,1}, \cdots, a_{n,n})$.

Definition 1.13

Let T be a matrix in $\mathcal{M}_{n,p}(\mathbb{K})$. It is said to be an upper triangular matrix if $t_{i,j} = 0$ for all indices (i, j) such that $i > j$. It is said to be a lower triangular matrix if $t_{i,j} = 0$ for all indices (i, j) such that $i < j$.

Lemma 1.14

Let T be a lower triangular matrix (respectively, upper triangular) in $\mathcal{M}_n(\mathbb{K})$. Its inverse (when it exists) is also a lower triangular matrix (respectively, upper triangular matrix) with diagonal entries equal to the inverse of the diagonal entries of T.

Let $T^{'}$ be another lower triangular matrix (respectively, upper triangular) in $\mathcal{M}_n(\mathbb{K})$. The product $TT^{'}$ is also a lower triangular matrix (respectively, upper triangular) with diagonal entries equal to the product of the diagonal entries of T and $T^{'}$.

Definition 1.15

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix. The matrix $A^\star \in \mathcal{M}_n(\mathbb{C})$ defined by,

$$A^\star = \overline{A}^t = (\overline{a}_{j,i})_{1 \leq i,j \leq n},$$

is the adjoint matrix of A.

Definition 1.16

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix. Then,

1. A is self-adjoint or Hermitian if $A = A^\star$,
2. A is unitary if $A^{-1} = A^\star$,
3. A is normal if $AA^\star = A^\star A$.

Definition 1.17

Let $A \in \mathcal{M}_n(\mathbb{R})$ be a real square matrix. Then,

1. A is symmetric or self-adjoint if $A = A^t$,
2. A is orthogonal or unitary if $A^{-1} = A^t$,
3. A is normal if $AA^t = A^t A$.

Definition 1.15

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix. The matrix $A^\star \in \mathcal{M}_n(\mathbb{C})$ defined by,

$$A^\star = \overline{A}^t = (\overline{a}_{j,i})_{1 \leq i,j \leq n},$$

is the adjoint matrix of A.

Definition 1.16

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix. Then,

1. A is self-adjoint or Hermitian if $A = A^\star$,
2. A is unitary if $A^{-1} = A^\star$,
3. A is normal if $AA^\star = A^\star A$.

Definition 1.17

Let $A \in \mathcal{M}_n(\mathbb{R})$ be a real square matrix. Then,

1. A is symmetric or self-adjoint if $A = A^t$,
2. A is orthogonal or unitary if $A^{-1} = A^t$,
3. A is normal if $AA^t = A^t A$.

Definition 1.15

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix. The matrix $A^\star \in \mathcal{M}_n(\mathbb{C})$ defined by,

$$A^\star = \overline{A}^t = (\overline{a}_{j,i})_{1 \leq i,j \leq n},$$

is the adjoint matrix of A.

Definition 1.16

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix. Then,

1. A is self-adjoint or Hermitian if $A = A^\star$,
2. A is unitary if $A^{-1} = A^\star$,
3. A is normal if $AA^\star = A^\star A$.

Definition 1.17

Let $A \in \mathcal{M}_n(\mathbb{R})$ be a real square matrix. Then,

1. A is symmetric or self-adjoint if $A = A^t$,
2. A is orthogonal or unitary if $A^{-1} = A^t$,
3. A is normal if $AA^t = A^t A$.

- We consider the vector space $\mathbb{K}^d$ with the scalar product $< x, y >= \sum_{i=1}^{d} x_i y_i$

  if $\mathbb{K} = \mathbb{R}$, or the Hermitian product $< x, y >= \sum_{i=1}^{d} x_i \overline{y}_i$ if $\mathbb{K} = \mathbb{C}$

- The Gram-Schmidt orthonormalization process constructs an orthonormal family out of a family of linearly independent vectors in $\mathbb{K}^d$

Theorem 1.18

Let $(x_1, \cdots, x_n)$ be a linearly independent family in $\mathbb{K}^d$. There exists an orthonormal family $(y_1, \cdots, y_n)$ such that $\mathrm{span}(y_1, \cdots, y_p) = \mathrm{span}(x_1, \cdots, x_p)$ for any index p in the range $1 \leq p \leq n$.

If $\mathbb{K} = \mathbb{R}$, this family is unique up to a change of sign of each vector y_p.

If $\mathbb{K} = \mathbb{C}$, this family is unique up to a multiplicative factor of unit modulus for each vector y_p.

- We consider the vector space $\mathbb{K}^d$ with the scalar product $< x, y >= \sum_{i=1}^{d} x_i y_i$

  if $\mathbb{K} = \mathbb{R}$, or the Hermitian product $< x, y >= \sum_{i=1}^{d} x_i \overline{y}_i$ if $\mathbb{K} = \mathbb{C}$

- The Gram-Schmidt orthonormalization process constructs an orthonormal family out of a family of linearly independent vectors in $\mathbb{K}^d$

Theorem 1.18

Let $(x_1, \cdots, x_n)$ be a linearly independent family in $\mathbb{K}^d$. There exists an orthonormal family $(y_1, \cdots, y_n)$ such that $\mathtt{span}(y_1, \cdots, y_p) = \mathtt{span}(x_1, \cdots, x_p)$ for any index p in the range $1 \leq p \leq n$.

If $\mathbb{K} = \mathbb{R}$, this family is unique up to a change of sign of each vector y_p.

If $\mathbb{K} = \mathbb{C}$, this family is unique up to a multiplicative factor of unit modulus for each vector y_p.

Definition 1.19

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix. The characteristic polynomial of A is the polynomial $P_A(\lambda)$ defined on $\mathbb{C}$ by,

$$P_A(\lambda) = \det(A - \lambda I_n)^a.$$

It is a polynomial of degree equal to n and has thus n roots in $\mathbb{C}$ which are called the eigenvalues of A. The algebraic multiplicity of an eigenvalue is its multiplicity as a root of $P_A(\lambda)$.

A nonzero vector $x \in \mathbb{C}^n$ such that $Ax = \lambda x$ is called an eigenvector of A associated to the eigenvalue λ.

[a] p1-r1

Definition 1.20

We call the maximum of the moduli of the eigenvalues of a matrix $A \in \mathcal{M}_n(\mathbb{C})$ the spectral radius of A, and we denote it by $\varrho(A)$.

Definition 1.19

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix. The characteristic polynomial of A is the polynomial $P_A(\lambda)$ defined on $\mathbb{C}$ by,

$$P_A(\lambda) = \det(A - \lambda I_n)^a.$$

It is a polynomial of degree equal to n and has thus n roots in $\mathbb{C}$ which are called the eigenvalues of A. The algebraic multiplicity of an eigenvalue is its multiplicity as a root of $P_A(\lambda)$.

A nonzero vector $x \in \mathbb{C}^n$ such that $Ax = \lambda x$ is called an eigenvector of A associated to the eigenvalue λ.

[a] p1-r1

Definition 1.20

We call the maximum of the moduli of the eigenvalues of a matrix $A \in \mathcal{M}_n(\mathbb{C})$ the spectral radius of A, and we denote it by $\varrho(A)$.

Definition 1.21

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix. Let λ be an eigenvalue of A. The vector subspace defined by,

$$E_\lambda = \texttt{Ker}\ (A - \lambda I_n),$$

is the eigenspace associated with the eigenvalue λ.

We call the vector subspace defined by,

$$F_\lambda = \bigcup_{1 \leq k \leq k_0} \texttt{Ker}\ (A - \lambda I_n)^k,$$

the generalized eigenspace associated with λ.

Definition 1.22

Let $P(x) = \sum_{i=0}^{d} a_i x^i$ be a polynomial on $\mathbb{C}$ and $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix.

The corresponding matrix polynomial is defined by $P(A) = \sum_{i=0}^{d} a_i A^i$.

Definition 1.21

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix. Let λ be an eigenvalue of A. The vector subspace defined by,

$$E_\lambda = \texttt{Ker}\ (A - \lambda I_n),$$

is the eigenspace associated with the eigenvalue λ.

We call the vector subspace defined by,

$$F_\lambda = \bigcup_{1 \leq k \leq k_0} \texttt{Ker}\ (A - \lambda I_n)^k,$$

the generalized eigenspace associated with λ.

Definition 1.22

Let $P(x) = \sum_{i=0}^{d} a_i x^i$ be a polynomial on $\mathbb{C}$ and $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix.

The corresponding matrix polynomial is defined by $P(A) = \sum_{i=0}^{d} a_i A^i$.

Lemma 1.23

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix. If $Ax = \lambda x$ with $x \neq 0$ then $P(A)x = P(\lambda)x$ for all polynomials $P(x)$ that is, if λ is an eigenvalue of A then $P(\lambda)$ is an eigenvalue of $P(A)$.

Theorem 1.24

(Spectral decomposition)
Consider a matrix $A \in \mathcal{M}_n(\mathbb{C})$ that has p distinct eigenvalues $(\lambda_1, \cdots, \lambda_p)$, with $1 \leq p \leq n$, of algebraic multiplicity $n_1, \cdots, n_p$, with $1 \leq n_i \leq n$ and $\sum_{i=1}^{p} n_i = n$.

Then its generalized eigenspaces satisfy,

$$\mathbb{C}^n = \bigoplus_{i=1}^{p} F_{\lambda_i}, \qquad F_{\lambda_i} = \mathtt{Ker}\,(A - \lambda I_n)^{n_i} \quad \text{and} \quad \mathtt{dim}\; F_{\lambda_i} = n_i.$$

Lemma 1.23

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix. If $Ax = \lambda x$ with $x \neq 0$ then $P(A)x = P(\lambda)x$ for all polynomials $P(x)$ that is, if λ is an eigenvalue of A then $P(\lambda)$ is an eigenvalue of $P(A)$.

Theorem 1.24

(Spectral decomposition)
Consider a matrix $A \in \mathcal{M}_n(\mathbb{C})$ that has p distinct eigenvalues $(\lambda_1, \cdots, \lambda_p)$, with $1 \leq p \leq n$, of algebraic multiplicity $n_1, \cdots, n_p$, with $1 \leq n_i \leq n$ and $\sum_{i=1}^{p} n_i = n$.

Then its generalized eigenspaces satisfy,

$$\mathbb{C}^n = \bigoplus_{i=1}^{p} F_{\lambda_i}, \qquad F_{\lambda_i} = \texttt{Ker}\ (A - \lambda I_n)^{n_i} \ \text{ and } \ \texttt{dim}\ F_{\lambda_i} = n_i.$$

Definition 1.25

A complex square matrix $A \in \mathcal{M}_n(\mathbb{C})$ can be reduced to triangular form (respectively, diagonal form) if there exists a nonsingular matrix P and a triangular matrix T (respectively, a diagonal matrix D) such that,

$$A = PTP^{-1} \quad \text{(respectively,} \quad A = PDP^{-1}\text{)}.$$

- If A can be reduced to triangular or diagonal form, then the eigenvalues of A, repeated with their algebraic multiplicities, appear on the diagonal of T or D.
- When A can be diagonalized, the column vectors of P are eigenvectors of A.

Definition 1.25

A complex square matrix $A \in \mathcal{M}_n(\mathbb{C})$ can be reduced to triangular form (respectively, diagonal form) if there exists a nonsingular matrix P and a triangular matrix T (respectively, a diagonal matrix D) such that,

$$A = PTP^{-1} \quad \text{(respectively, } A = PDP^{-1}\text{)}.$$

- If A can be reduced to triangular or diagonal form, then the eigenvalues of A, repeated with their algebraic multiplicities, appear on the diagonal of T or D.
- When A can be diagonalized, the column vectors of P are eigenvectors of A.

Proposition 1.26

Any matrix $A \in \mathcal{M}_n(\mathbb{C})$ can be reduced to triangular form [a].

[a] p1-d1

Theorem 1.27

(Schur factorization)
For any complex square matrix $A \in \mathcal{M}_n(\mathbb{C})$ there exists a unitary matrix U (i.e. $U^{-1} = U^\star$) such that $U^{-1}AU$ is triangular[a].

[a] p1-d2

Proposition 1.26

Any matrix $A \in \mathcal{M}_n(\mathbb{C})$ can be reduced to triangular form [a].

[a] p1-d1

Theorem 1.27

(Schur factorization)
For any complex square matrix $A \in \mathcal{M}_n(\mathbb{C})$ there exists a unitary matrix U (i.e. $U^{-1} = U^\star$) such that $U^{-1}AU$ is triangular[a].

[a] p1-d2

Proposition 1.28

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix with distinct eigenvalues $(\lambda_1, \cdots, \lambda_p)$, $1 \leq p \leq n$. The matrix A is diagonalizable if and only if,

$$\mathbb{C}^n = \bigoplus_{i=1}^{p} E_{\lambda_i},$$

or equivalently, if and only if $F_{\lambda_i} = E_{\lambda_i}$ for any $1 \leq i \leq p$[a].

[a] p1-d3

Theorem 1.29

(Diagonalization)
A matrix $A \in \mathcal{M}_n(\mathbb{C})$ is normal (i.e. $AA^\star = A^\star A$) if and only if there exists a unitary matrix U such that,

$$A = U \, \mathtt{diag}(\lambda_1, \cdots, \lambda_n) \, U^{-1},$$

where $(\lambda_1, \cdots, \lambda_n)$ are the eigenvalues of A[a].

[a] p1-d4

Proposition 1.28

Let $A \in \mathcal{M}_n(\mathbb{C})$ be a complex square matrix with distinct eigenvalues $(\lambda_1, \cdots, \lambda_p)$, $1 \leq p \leq n$. The matrix A is diagonalizable if and only if,

$$\mathbb{C}^n = \bigoplus_{i=1}^{p} E_{\lambda_i},$$

or equivalently, if and only if $F_{\lambda_i} = E_{\lambda_i}$ for any $1 \leq i \leq p$[a].

[a] p1-d3

Theorem 1.29

(Diagonalization)
A matrix $A \in \mathcal{M}_n(\mathbb{C})$ is normal (i.e. $AA^\star = A^\star A$) if and only if there exists a unitary matrix U such that,

$$A = U \, \mathtt{diag}(\lambda_1, \cdots, \lambda_n) \, U^{-1},$$

where $(\lambda_1, \cdots, \lambda_n)$ are the eigenvalues of A[a].

[a] p1-d4

Theorem 1.30

A matrix $A \in \mathcal{M}_n(\mathbb{C})$ is self-adjoint (or Hermitian, i.e. $A^\star = A$) if and only if it is diagonalizable in an orthonormal basis with real eigenvalues. In other words, there exists a unitary matrix U such that,

$$A = U\ \mathtt{diag}(\lambda_1, \cdots, \lambda_n)\ U^{-1} \quad \text{with} \quad \lambda_i \in \mathbb{R}^a.$$

[a] p1-d5

Corollary 1.31

A matrix $A \in \mathcal{M}_n(\mathbb{R})$ is real symmetric (i.e. $A^t = A$) if and only there exists a unitary matrix Q (also called orthogonal, i.e. $Q^{-1} = Q^t$) and real eigenvalues $(\lambda_1, \cdots, \lambda_n)$ such that,

$$A = Q\ \mathtt{diag}(\lambda_1, \cdots, \lambda_n)\ Q^{-1} \quad \text{with} \quad \lambda_i \in \mathbb{R}.$$

- A self-adjoint matrix $A \in \mathcal{M}_n(\mathbb{C})$ is said to be positive definite if all its eigenvalues are strictly positive. It is said to be nonnegative definite (or positive semidefinite) if all its eigenvalues are are nonnegative.

Theorem 1.30

A matrix $A \in \mathcal{M}_n(\mathbb{C})$ is self-adjoint (or Hermitian, i.e. $A^\star = A$) if and only if it is diagonalizable in an orthonormal basis with real eigenvalues. In other words, there exists a unitary matrix U such that,

$$A = U\ \mathtt{diag}(\lambda_1, \cdots, \lambda_n)\ U^{-1} \quad \text{with} \quad \lambda_i \in \mathbb{R}^a.$$

[a] p1-d5

Corollary 1.31

A matrix $A \in \mathcal{M}_n(\mathbb{R})$ is real symmetric (i.e. $A^t = A$) if and only there exists a unitary matrix Q (also called orthogonal, i.e. $Q^{-1} = Q^t$) and real eigenvalues $(\lambda_1, \cdots, \lambda_n)$ such that,

$$A = Q\ \mathtt{diag}(\lambda_1, \cdots, \lambda_n)\ Q^{-1} \quad \text{with} \quad \lambda_i \in \mathbb{R}.$$

- A self-adjoint matrix $A \in \mathcal{M}_n(\mathbb{C})$ is said to be positive definite if all its eigenvalues are strictly positive. It is said to be nonnegative definite (or positive semidefinite) if all its eigenvalues are are nonnegative.

Theorem 1.30

A matrix $A \in \mathcal{M}_n(\mathbb{C})$ is self-adjoint (or Hermitian, i.e. $A^\star = A$) if and only if it is diagonalizable in an orthonormal basis with real eigenvalues. In other words, there exists a unitary matrix U such that,

$$A = U\ \mathtt{diag}(\lambda_1, \cdots, \lambda_n)\ U^{-1} \quad \text{with} \quad \lambda_i \in \mathbb{R}^a.$$

[a] p1-d5

Corollary 1.31

A matrix $A \in \mathcal{M}_n(\mathbb{R})$ is real symmetric (i.e. $A^t = A$) if and only there exists a unitary matrix Q (also called orthogonal, i.e. $Q^{-1} = Q^t$) and real eigenvalues $(\lambda_1, \cdots, \lambda_n)$ such that,

$$A = Q\ \mathtt{diag}(\lambda_1, \cdots, \lambda_n)\ Q^{-1} \quad \text{with} \quad \lambda_i \in \mathbb{R}.$$

- A self-adjoint matrix $A \in \mathcal{M}_n(\mathbb{C})$ is said to be positive definite if all its eigenvalues are strictly positive. It is said to be nonnegative definite (or positive semidefinite) if all its eigenvalues are are nonnegative.

Lemma 1.32

For any matrix $A \in \mathcal{M}_{m,n}(\mathbb{C})$, the matrix $A^\star A$ is Hermitian and has real, nonnegative eigenvalues[a].

[a] p1-d6

Lemma 1.33

Let $A \in \mathcal{M}_{m,n}(\mathbb{C})$ and $B \in \mathcal{M}_{n,m}(\mathbb{C})$. The nonzero eigenvalues of the matrices AB and BA are the same[a].

[a] p1-d7

Definition 1.34

The singular values of a matrix $A \in \mathcal{M}_{m,n}(\mathbb{C})$ are the nonnegative square roots of the n eigenvalues of $A^\star A$.

Proposition 1.35

The singular values of a normal matrix are the moduli of its eigenvalues[a].

[a] p1-d8

Lemma 1.32

For any matrix $A \in \mathcal{M}_{m,n}(\mathbb{C})$, the matrix $A^\star A$ is Hermitian and has real, nonnegative eigenvalues[a].

[a] p1-d6

Lemma 1.33

Let $A \in \mathcal{M}_{m,n}(\mathbb{C})$ and $B \in \mathcal{M}_{n,m}(\mathbb{C})$. The nonzero eigenvalues of the matrices AB and BA are the same[a].

[a] p1-d7

Definition 1.34

The singular values of a matrix $A \in \mathcal{M}_{m,n}(\mathbb{C})$ are the nonnegative square roots of the n eigenvalues of $A^\star A$.

Proposition 1.35

The singular values of a normal matrix are the moduli of its eigenvalues[a].

[a] p1-d8

Lemma 1.32

For any matrix $A \in \mathcal{M}_{m,n}(\mathbb{C})$, the matrix $A^\star A$ is Hermitian and has real, nonnegative eigenvalues[a].

[a] p1-d6

Lemma 1.33

Let $A \in \mathcal{M}_{m,n}(\mathbb{C})$ and $B \in \mathcal{M}_{n,m}(\mathbb{C})$. The nonzero eigenvalues of the matrices AB and BA are the same[a].

[a] p1-d7

Definition 1.34

The singular values of a matrix $A \in \mathcal{M}_{m,n}(\mathbb{C})$ are the nonnegative square roots of the n eigenvalues of $A^\star A$.

Proposition 1.35

The singular values of a normal matrix are the moduli of its eigenvalues[a].

[a] p1-d8

Lemma 1.32

For any matrix $A \in \mathcal{M}_{m,n}(\mathbb{C})$, the matrix $A^\star A$ is Hermitian and has real, nonnegative eigenvalues[a].

[a] p1-d6

Lemma 1.33

Let $A \in \mathcal{M}_{m,n}(\mathbb{C})$ and $B \in \mathcal{M}_{n,m}(\mathbb{C})$. The nonzero eigenvalues of the matrices AB and BA are the same[a].

[a] p1-d7

Definition 1.34

The singular values of a matrix $A \in \mathcal{M}_{m,n}(\mathbb{C})$ are the nonnegative square roots of the n eigenvalues of $A^\star A$.

Proposition 1.35

The singular values of a normal matrix are the moduli of its eigenvalues[a].

[a] p1-d8

Theorem 1.36

(SVD factorization)
Let $A \in \mathcal{M}_{m,n}(\mathbb{C})$ be a matrix having r positive singular values. There exist two unitary matrices $U \in \mathcal{M}_n(\mathbb{C})$ and $V \in \mathcal{M}_m(\mathbb{C})$, and a diagonal matrix $\tilde{\Sigma} \in \mathcal{M}_{m,n}(\mathbb{R})$, such that,

$$A = V\tilde{\Sigma}U^\star \quad \text{and} \quad \tilde{\Sigma} = \begin{pmatrix} \Sigma & 0 \\ 0 & 0 \end{pmatrix},$$

where $\Sigma = \mathtt{diag}(\mu_1, \cdots, \mu_r)$ and $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_r > 0$ are the positive singular values of A[a].

[a] p1-d9

Definition 1.37

Let $A = V\tilde{\Sigma}U^\star$ be the SVD factorization of some matrix $A \in \mathcal{M}_{m,n}(\mathbb{C})$ having r nonzero singular values. We call the matrix $A^\dagger \in \mathcal{M}_{n,m}(\mathbb{C})$ defined by $A^\dagger = U\tilde{\Sigma}^\dagger V^\star$ with,

$$\tilde{\Sigma}^\dagger = \begin{pmatrix} \Sigma^{-1} & 0 \\ 0 & 0 \end{pmatrix} \in \mathcal{M}_{n,m}(\mathbb{R}),$$

the pseudoinverse matrix of A.

Theorem 1.36

(SVD factorization)
Let $A \in \mathcal{M}_{m,n}(\mathbb{C})$ be a matrix having r positive singular values. There exist two unitary matrices $U \in \mathcal{M}_n(\mathbb{C})$ and $V \in \mathcal{M}_m(\mathbb{C})$, and a diagonal matrix $\tilde{\Sigma} \in \mathcal{M}_{m,n}(\mathbb{R})$, such that,

$$A = V\tilde{\Sigma}U^\star \quad \text{and} \quad \tilde{\Sigma} = \begin{pmatrix} \Sigma & 0 \\ 0 & 0 \end{pmatrix},$$

where $\Sigma = \mathtt{diag}(\mu_1, \cdots, \mu_r)$ and $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_r > 0$ are the positive singular values of A[a].

[a] p1-d9

Definition 1.37

Let $A = V\tilde{\Sigma}U^\star$ be the SVD factorization of some matrix $A \in \mathcal{M}_{m,n}(\mathbb{C})$ having r nonzero singular values. We call the matrix $A^\dagger \in \mathcal{M}_{n,m}(\mathbb{C})$ defined by $A^\dagger = U\tilde{\Sigma}^\dagger V^\star$ with,

$$\tilde{\Sigma}^\dagger = \begin{pmatrix} \Sigma^{-1} & 0 \\ 0 & 0 \end{pmatrix} \in \mathcal{M}_{n,m}(\mathbb{R}),$$

the pseudoinverse matrix of A.

Definition 1.38

We call a mapping denoted by $\| . \|$, from $\mathbb{K}^n$ into $\mathbb{R}^+$, satisfying the following properties, a norm on $\mathbb{K}^n$,

1. $\forall x \in \mathbb{K}^n$, $\| x \| = 0 \Longrightarrow x = 0$,
2. $\forall x \in \mathbb{K}^n$, $\forall \lambda \in \mathbb{K}$, $\| \lambda x \| = |\lambda| \, \| x \|$,
3. $\forall x \in \mathbb{K}^n$, $\forall y \in \mathbb{K}^n$, $\| x + y \| \leq \| x \| + \| y \|$.

- If $\forall x \in \mathbb{K}^n$ is endowed with a scalar (or Hermitian) product $< ., . >$, then the mapping $x \longmapsto < x, x >^{\frac{1}{2}}$ defines a norm on $\mathbb{K}^n$.
- The most common norms on $\mathbb{K}^n$ are (x_i denotes the coordinates of a vector x in the canonical basis of $\mathbb{K}^n$):
  - the Euclidean norm, $\| x \|_2 = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{\frac{1}{2}}$;
  - the l^p norm, $\| x \|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}}$;
  - the l^∞ norm, $\| x \|_\infty = \max_{1 \leq i \leq n} |x_i|$.

Definition 1.38

We call a mapping denoted by $\| . \|$, from $\mathbb{K}^n$ into $\mathbb{R}^+$, satisfying the following properties, a norm on $\mathbb{K}^n$,

1. $\forall x \in \mathbb{K}^n, \| x \| = 0 \Longrightarrow x = 0,$
2. $\forall x \in \mathbb{K}^n, \forall \lambda \in \mathbb{K}, \| \lambda x \| = |\lambda| \| x \|,$
3. $\forall x \in \mathbb{K}^n, \forall y \in \mathbb{K}^n, \| x + y \| \leq \| x \| + \| y \|.$

- If $\forall x \in \mathbb{K}^n$ is endowed with a scalar (or Hermitian) product $< ., . >$, then the mapping $x \longmapsto < x, x >^{\frac{1}{2}}$ defines a norm on $\mathbb{K}^n$.
- The most common norms on $\mathbb{K}^n$ are (x_i denotes the coordinates of a vector x in the canonical basis of $\mathbb{K}^n$):
  - the Euclidean norm, $\| x \|_2 = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{\frac{1}{2}}$;
  - the l^p norm, $\| x \|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}}$;
  - the l^∞ norm, $\| x \|_\infty = \max_{1 \leq i \leq n} |x_i|$.

Theorem 1.39

If E is a vector space of finite dimension then all norms are equivalent on E.
That is, for all pairs of norms $\| . \|$ and $\| . \|'$ there exists two constants c and C such that $0 < c \leq C$ and for all $x \in E$,

$$c \| x \| \leq \| x \|' \leq C \| x \| .$$

Definition 1.40

A norm $\| . \|$ defined on $\mathcal{M}_n(\mathbb{K})$ (i.e. the vector space of square matrices of size n with entries in $\mathbb{K}$) is a matrix norm if for all matrices A and B in $\mathcal{M}_n(\mathbb{K})$,

$$\| AB \| \leq \| A \| \| B \| .$$

Definition 1.41

Let $\| . \|$ be a vector norm on $\mathbb{K}^n$. It induces a matrix norm defined by,

$$\| A \| = \sup_{x \in \mathbb{K}^n ,\ x \neq 0} \frac{\| Ax \|}{\| x \|},$$

which is said to be subordinate to this vector norm.

Theorem 1.39

If E is a vector space of finite dimension then all norms are equivalent on E.
That is, for all pairs of norms $\| . \|$ and $\| . \|'$ there exists two constants c and C such that $0 < c \leq C$ and for all $x \in E$,

$$c \| x \| \leq \| x \|' \leq C \| x \| .$$

Definition 1.40

A norm $\| . \|$ defined on $\mathcal{M}_n(\mathbb{K})$ (i.e. the vector space of square matrices of size n with entries in $\mathbb{K}$) is a matrix norm if for all matrices A and B in $\mathcal{M}_n(\mathbb{K})$,

$$\| AB \| \leq \| A \| \| B \| .$$

Definition 1.41

Let $\| . \|$ be a vector norm on $\mathbb{K}^n$. It induces a matrix norm defined by,

$$\| A \| = \sup_{x \in \mathbb{K}^n \, , \, x \neq 0} \frac{\| Ax \|}{\| x \|},$$

which is said to be subordinate to this vector norm.

Theorem 1.39

If E is a vector space of finite dimension then all norms are equivalent on E.
That is, for all pairs of norms $\| . \|$ and $\| . \|'$ there exists two constants c and C such that $0 < c \leq C$ and for all $x \in E$,

$$c \| x \| \leq \| x \|' \leq C \| x \| .$$

Definition 1.40

A norm $\| . \|$ defined on $\mathcal{M}_n(\mathbb{K})$ (i.e. the vector space of square matrices of size n with entries in $\mathbb{K}$) is a matrix norm if for all matrices A and B in $\mathcal{M}_n(\mathbb{K})$,

$$\| AB \| \leq \| A \| \| B \| .$$

Definition 1.41

Let $\| . \|$ be a vector norm on $\mathbb{K}^n$. It induces a matrix norm defined by,

$$\| A \| = \sup_{x \in \mathbb{K}^n \ , \ x \neq 0} \frac{\| Ax \|}{\| x \|},$$

which is said to be subordinate to this vector norm.

Proposition 1.42

Let $\| . \|$ be a subordinate matrix norm on $\mathcal{M}_n(\mathbb{K})$.

1. For all matrices A in $\mathcal{M}_n(\mathbb{K})$, the norm $\| A \|$ is also defined by,

$$\| A \| = \sup_{x \in \mathbb{K}^n \ , \ \|x\|=1} \| Ax \| = \sup_{x \in \mathbb{K}^n \ , \ \|x\| \leq 1} \| Ax \| \ .$$

2. There exists $x_A \in \mathbb{K}^n, x_a \neq 0$ such that,

$$\| A \| = \frac{\| Ax_A \|}{\| x_A \|},$$

   and sup can be replaced by max in the definition of $\| A \|$.

3. The identity matrix satisfies $\| I_n \| = 1$.

4. A subordinate norm is indeed a matrix norm: for all matrices A and B in $\mathcal{M}_n(\mathbb{K})$ we have,

$$\| AB \| \leq \| A \| \| B \| \ .$$

Proposition 1.43

We consider matrices in $\mathcal{M}_n(\mathbb{K})$.

1. The matrix norm $\| A \|_1$, subordinate to the l^1 norm on $\mathbb{K}^n$, satisfies,
$$\| A \|_1 = \max_{1 \leq j \leq n} \left(\sum_{i=1}^{n} |a_{i,j}| \right).$$
2. The matrix norm $\| A \|_\infty$, subordinate to the l^∞ norm on $\mathbb{K}^n$, satisfies,
$$\| A \|_\infty = \max_{1 \leq i \leq n} \left(\sum_{j=1}^{n} |a_{i,j}| \right).$$
3. Let $\| A \|_2$ be the marix norm subordinate to the l^2 norm on $\mathbb{K}^n$. We have,
$$\| A \|_2 = \| A^\star \|_2 = \text{the largest singular value of } A^a.$$

[a] p1-d10

Lemma 1.44

Let U be a unitary matrix (i.e. $U^\star = U^{-1}$). We have,

$$\| UA \|_2 = \| AU \|_2 = \| A \|_2 .$$

Consequently, if A is a normal matrix, then $\| A \|_2 = \varrho(A)$[a].

[a] p1-d11

Proposition 1.45

Let $\| . \|$ be a matrix norm on $\mathcal{M}_n(\mathbb{C})$. It satisfies,

$$\varrho(A) \leq \| A \| .$$

Conversely, for any matrix $A \in \mathcal{M}_n(\mathbb{C})$ and for any real number $\varepsilon > 0$, there exists a subordinate norm $\| . \|$ (which depends on A and ε) such that,

$$\| A \| \leq \varrho(A) + \varepsilon.$$

Lemma 1.44

Let U be a unitary matrix (i.e. $U^\star = U^{-1}$). We have,

$$\| UA \|_2 = \| AU \|_2 = \| A \|_2 .$$

Consequently, if A is a normal matrix, then $\| A \|_2 = \varrho(A)$[a].

[a] p1-d11

Proposition 1.45

Let $\| . \|$ be a matrix norm on $\mathcal{M}_n(\mathbb{C})$. It satisfies,

$$\varrho(A) \leq \| A \| .$$

Conversely, for any matrix $A \in \mathcal{M}_n(\mathbb{C})$ and for any real number $\varepsilon > 0$, there exists a subordinate norm $\| . \|$ (which depends on A and ε) such that,

$$\| A \| \leq \varrho(A) + \varepsilon.$$

Definition 1.46

A sequence of matrices $(A_i)_{i\geq 1}$ in $\mathcal{M}_n(\mathbb{C})$ converges to a limit A if for a matrix norm $\| \, . \, \|$ we have,

$$\lim_{i \longmapsto +\infty} \| A_i - A \| = 0,$$

and we write $A = \lim_{i \longmapsto +\infty} A_i$.

Lemma 1.47

Let A be a matrix in $\mathcal{M}_n(\mathbb{C})$. The following four conditions are equivalent:

1. $\lim_{i \longmapsto +\infty} A^i = 0$,
2. $\lim_{i \longmapsto +\infty} A^i x = 0$ for all vectors $x \in \mathbb{C}^n$,
3. $\varrho(A) < 1$,
4. there exists at least one subordinate matrix norm such that $\| A \| < 1$[a].

[a] p1-d12

Definition 1.46

A sequence of matrices $(A_i)_{i\geq 1}$ in $\mathcal{M}_n(\mathbb{C})$ converges to a limit A if for a matrix norm $\| \, . \, \|$ we have,

$$\lim_{i \longmapsto +\infty} \| A_i - A \| = 0,$$

and we write $A = \lim_{i \longmapsto +\infty} A_i$.

Lemma 1.47

Let A be a matrix in $\mathcal{M}_n(\mathbb{C})$. The following four conditions are equivalent:

1. $\lim_{i \longmapsto +\infty} A^i = 0$,
2. $\lim_{i \longmapsto +\infty} A^i x = 0$ for all vectors $x \in \mathbb{C}^n$,
3. $\varrho(A) < 1$,
4. there exists at least one subordinate matrix norm such that $\| A \| < 1$[a].

[a] p1-d12

Theorem 1.48

Consider a power series on $\mathbb{C}$ of positive radius of convergence R,

$$\left| \sum_{i=0}^{+\infty} a_i z^i \right| < +\infty, \quad \forall z \in \mathbb{C} \text{ such that } |z| < R.$$

For any matrix $A \in \mathcal{M}_n(\mathbb{C})$ such that $\varrho(A) < R$, the series $(a_i A^i)_{i \geq 0}$ is convergent i.e. $\sum_{i=0}^{+\infty} a_i A^i$ is well defined in $\mathcal{M}_n(\mathbb{C})$.

Proposition 1.49

Let A be a matrix in $\mathcal{M}_n(\mathbb{C})$ with spectral radius $\varrho(A) < 1$. The matrix $(I - A)$ is nonsingular and its inverse is given by,

$$(I_n - A)^{-1} = \sum_{i=0}^{+\infty} A^{i\,a}.$$

[a] p1-d13

Theorem 1.48

Consider a power series on $\mathbb{C}$ of positive radius of convergence R,

$$\left| \sum_{i=0}^{+\infty} a_i z^i \right| < +\infty, \quad \forall z \in \mathbb{C} \text{ such that } |z| < R.$$

For any matrix $A \in \mathcal{M}_n(\mathbb{C})$ such that $\varrho(A) < R$, the series $(a_i A^i)_{i \geq 0}$ is convergent i.e. $\sum_{i=0}^{+\infty} a_i A^i$ is well defined in $\mathcal{M}_n(\mathbb{C})$.

Proposition 1.49

Let A be a matrix in $\mathcal{M}_n(\mathbb{C})$ with spectral radius $\varrho(A) < 1$. The matrix $(I - A)$ is nonsingular and its inverse is given by,

$$(I_n - A)^{-1} = \sum_{i=0}^{+\infty} A^{i\,a}.$$

[a] p1-d13

Outline

Linear systems

- We call the problem that consists in finding the (possibly multiple) solution $x \in \mathbb{K}^p$, if any, of the following algebraic equation,

$$Ax = b,$$

  a linear system.
- The matrix $A \in \mathcal{M}_{n,p}(\mathbb{K})$, called the system matrix, and the vector $b \in \mathbb{K}^n$, called the right-hand side, are the data of the problem.
- The vector $x \in \mathbb{K}^p$ is the unknown vector.
- Important particular case: square linear system, i.e. $n = p$.
- $n < p$ and $n > p$ respectively correspond to underdetermined and overdetermined systems.

Theorem 2.1

If the matrix A is nonsingular, then there exists a unique solution of the linear system $Ax = b$.

If A is singular, then one of the following alternatives holds:

- the right-hand side b belongs to the range of A and there exists an infinity of solutions that differ one from the other by addition of an element of the kernel of A,
- the right-hand side b does not belong to the range of A and there are no solutions.

- Particular trivial cases:
  - A is diagonal,
  - A is unitary i.e. $A^{-1} = A^{\star}$.

Theorem 2.1

If the matrix A is nonsingular, then there exists a unique solution of the linear system $Ax = b$.

If A is singular, then one of the following alternatives holds:

- the right-hand side b belongs to the range of A and there exists an infinity of solutions that differ one from the other by addition of an element of the kernel of A,
- the right-hand side b does not belong to the range of A and there are no solutions.

- Particular trivial cases:
  - A is diagonal,
  - A is unitary i.e. $A^{-1} = A^{\star}$.

Linear systems

- If A is a triangular matrix then the solution can be computed by the so-called forward substitution algorithm[1].

  Data: A, b. Output: $x = A^{-1}$.

  For $i = 1 \nearrow n$
  $s = 0$
  For $j = 1 \nearrow i - 1$
  $s = s + A_{i,j} x_j$
  End j
  $x_i = (b_i - s)/A_{i,i}$
  End i

- $1 + 2 + \cdots + n - 1 = n(n-1)/2$ multiplications.
- n divisions.
- Total: $n^2/2$ operations.
- In general, solving a linear system does not require computing the inverse matrix A^{-1} because it is too expensive.

[1] p2-r1

Linear systems

- If A is a triangular matrix then the solution can be computed by the so-called forward substitution algorithm[1].

  Data: A, b. Output: $x = A^{-1}$.

  For $i = 1 \nearrow n$
  $s = 0$
  For $j = 1 \nearrow i - 1$
  $s = s + A_{i,j} x_j$
  End j
  $x_i = (b_i - s)/A_{i,i}$
  End i

- $1 + 2 + \cdots + n - 1 = n(n-1)/2$ multiplications.
- n divisions.
- Total: $n^2/2$ operations.
- In general, solving a linear system does not require computing the inverse matrix A^{-1} because it is too expensive.

[1] p2-r1

Linear systems

- If A is a triangular matrix then the solution can be computed by the so-called forward substitution algorithm[1].

  Data: A, b. Output: $x = A^{-1}$.

  For $i = 1 \nearrow n$
  $s = 0$
  For $j = 1 \nearrow i - 1$
  $s = s + A_{i,j} x_j$
  End j
  $x_i = (b_i - s)/A_{i,i}$
  End i

- $1 + 2 + \cdots + n - 1 = n(n-1)/2$ multiplications.
- n divisions.
- Total: $n^2/2$ operations.
- In general, solving a linear system does not require computing the inverse matrix A^{-1} because it is too expensive.

[1] p2-r1

Linear systems

- Important criteria for the numerical solution of linear systems:
  - efficiency i.e. algorithms have to be fast (minimizing the number of performed operations) and spare memory storage,
  - stability (propagation of rounding errors).

- To quantify the rounding error phenomenon, we introduce the notion of matrix conditioning[2].
- It helps to measure the sensitivity of the solution x to the perturbation of the data A and b.
- Let $\varepsilon \geq 0$ be a small parameter of data perturbation,

$$A_\varepsilon = A + \varepsilon B \quad \text{and} \quad b_\varepsilon = b + \varepsilon\gamma$$

  with $B \in \mathcal{M}_n(\mathbb{K})$ and $\gamma \in \mathbb{K}^n$
- The system $A_\varepsilon x_\varepsilon = b_\varepsilon$ is assumed to be nonsingular

[2] p2-r2

Linear systems

- Important criteria for the numerical solution of linear systems:
  - efficiency i.e. algorithms have to be fast (minimizing the number of performed operations) and spare memory storage,
  - stability (propagation of rounding errors).
- To quantify the rounding error phenomenon, we introduce the notion of matrix conditioning[2].
- It helps to measure the sensitivity of the solution x to the perturbation of the data A and b.
- Let $\varepsilon \geq 0$ be a small parameter of data perturbation,

$$A_\varepsilon = A + \varepsilon B \quad \text{and} \quad b_\varepsilon = b + \varepsilon\gamma$$

  with $B \in \mathcal{M}_n(\mathbb{K})$ and $\gamma \in \mathbb{K}^n$
- The system $A_\varepsilon x_\varepsilon = b_\varepsilon$ is assumed to be nonsingular

[2] p2-r2

Definition 2.2

The condition number of a matrix $A \in \mathcal{M}_n(\mathbb{K})$, relative to a subordinate matrix norm $\| \, . \, \|$, is the quantity defined by,

$$\mathrm{cond}(A) = \| \, A \, \| \| \, A^{-1} \, \| \, .$$

- $\mathrm{cond}(A) \geq 1$ since $1 = \| \, I \, \| = \| \, AA^{-1} \, \| \leq \| \, A \, \| \| \, A^{-1} \, \|$.

Proposition 2.3

Let A be a nonsingular matrix in $\mathcal{M}_n(\mathbb{K})$ and $b \neq 0$ in $\mathbb{K}^n$.

1. If x and $x + \delta x$ are respectively the solutions of the systems,

$$Ax = b \quad \text{and} \quad A(x + \delta x) = b + \delta b,$$

we have,

$$\frac{\| \delta x \|}{\| x \|} \leq \text{cond}(A) \frac{\| \delta b \|}{\| b \|}.$$

2. If x and $x + \delta x$ are respectively the solutions of the systems,

$$Ax = b \quad \text{and} \quad (A + \delta A)(x + \delta x) = b,$$

we have,

$$\frac{\| \delta x \|}{\| x + \delta x \|} \leq \text{cond}(A) \frac{\| \delta A \|}{\| A \|}^{a}.$$

[a] p2-d1

Proposition 2.4

Let A be a nonsingular matrix in $\mathcal{M}_n(\mathbb{C})$.

1. $\mathrm{cond}(A) = \mathrm{cond}(A^{-1}), \quad \mathrm{cond}(\alpha A) = \mathrm{cond}(A) \quad \forall \alpha \neq 0.$
2. For any matrix A,

$$\mathrm{cond}_2(A) = \frac{\mu_{max}(A)}{\mu_{min}(A)},$$

   where $\mu_{min}(A)$ and $\mu_{max}(A)$ respectively denote the smallest and the largest singular values of A.
3. For a normal matrix A (i.e. $AA^\star = A^\star A$),

$$\mathrm{cond}_2(A) = \frac{|\lambda_{max}(A)|}{|\lambda_{min}(A)|} = \varrho(A)\varrho(A^{-1}),$$

   where $\lambda_{min}(A)$ and $\lambda_{max}(A)$ respectively denote the smallest and the largest eigenvalues of A.
4. For any unitary matrix U, $\quad \mathrm{cond}_2(U) = 1.$
5. For any unitary matrix U, $\quad \mathrm{cond}_2(AU) = \mathrm{cond}_2(UA) = \mathrm{cond}_2(A).$

- A matrix A is said to be *well conditioned* if, for a given norm, $\mathtt{cond}(A) \approx 1$

Proposition 2.5

Conditionings $\mathtt{cond}_1$, $\mathtt{cond}_2$ and $\mathtt{cond}_\infty$ are equivalent,

$$\begin{array}{lcll} n^{-1}\mathtt{cond}_2(A) & \leq & \mathtt{cond}_1(A) & \leq & n\mathtt{cond}_2(A), \\ n^{-1}\mathtt{cond}_\infty(A) & \leq & \mathtt{cond}_2(A) & \leq & n\mathtt{cond}_\infty(A), \\ n^{-2}\mathtt{cond}_1(A) & \leq & \mathtt{cond}_\infty(A) & \leq & n^2\mathtt{cond}_1(A). \end{array}$$

Outline

Direct methods

This section discusses the solution of $Ax = b$ where $A \in \mathcal{M}_n(\mathbb{R})$ is a square matrix and $b \in \mathbb{R}^n$ (here $K = \mathbb{R}$ to simplify the presentation).

A direct method computes the solution x in a finite number of operations (in exact arithmetic).

Gaussian elimination: the goal is to find a nonsingular matrix M such that the product MA is an upper triangular matrix.

Solution of $Ax = b$ follows a three steps process:

1. elimination - computation of the matrix M,
2. right-hand side update - computation of the product Mb,
3. substitution - solution of the triangular system $Tx = Mb$.

Theorem 3.1

Let A be a square matrix (invertible or not). There exists at least one nonsingular matrix M such that the matrix $T = MA$ is upper triangular[a].

[a] p3-d1

Direct methods

This section discusses the solution of $Ax = b$ where $A \in \mathcal{M}_n(\mathbb{R})$ is a square matrix and $b \in \mathbb{R}^n$ (here $K = \mathbb{R}$ to simplify the presentation).

A direct method computes the solution x in a finite number of operations (in exact arithmetic).

Gaussian elimination: the goal is to find a nonsingular matrix M such that the product MA is an upper triangular matrix.

Solution of $Ax = b$ follows a three steps process:

1. elimination - computation of the matrix M,
2. right-hand side update - computation of the product Mb,
3. substitution - solution of the triangular system $Tx = Mb$.

Theorem 3.1

Let A be a square matrix (invertible or not). There exists at least one nonsingular matrix M such that the matrix $T = MA$ is upper triangular[a].

[a] p3-d1

Direct methods

This section discusses the solution of $Ax = b$ where $A \in \mathcal{M}_n(\mathbb{R})$ is a square matrix and $b \in \mathbb{R}^n$ (here $K = \mathbb{R}$ to simplify the presentation).

A direct method computes the solution x in a finite number of operations (in exact arithmetic).

Gaussian elimination: the goal is to find a nonsingular matrix M such that the product MA is an upper triangular matrix.

Solution of $Ax = b$ follows a three steps process:

1. elimination - computation of the matrix M,
2. right-hand side update - computation of the product Mb,
3. substitution - solution of the triangular system $Tx = Mb$.

Theorem 3.1

Let A be a square matrix (invertible or not). There exists at least one nonsingular matrix M such that the matrix $T = MA$ is upper triangular[a].

[a] p3-d1

LU decomposition method (Gauss method): the goal is to factorize A into a product of two triangular matrices, $A = LU$, where L is lower triangular and U is upper triangular.

This decomposition allows us to reduce the solution of the system $Ax = b$ to solving two triangular systems $Ly = b$ and $Ux = y$.

It turns out to be nothing else than Gaussian elimination in the case without pivoting.

The matrices defined by,

$$\Delta^k = \begin{pmatrix} a_{1,1} & \cdots & a_{1,k} \\ \vdots & & \vdots \\ a_{k,1} & \cdots & a_{k,k} \end{pmatrix},$$

are called the diagonal submatrices of order k of $A \in \mathcal{M}_n(\mathbb{R})$.

Theorem 3.2

Let $A = (a_{i,j})_{1 \leq i,j \leq n}$ be a square matrix such that all its diagonal submatrices of order k are nonsingular. There exists a unique pair of matrices (L, U), with U upper triangular and L lower triangular with a unit diagonal (i.e. $l_{i,i} = 1$), such that $A = LU$[a].

[a] p3-d2

LU decomposition method (Gauss method): the goal is to factorize A into a product of two triangular matrices, $A = LU$, where L is lower triangular and U is upper triangular.

This decomposition allows us to reduce the solution of the system $Ax = b$ to solving two triangular systems $Ly = b$ and $Ux = y$.

It turns out to be nothing else than Gaussian elimination in the case without pivoting.

The matrices defined by,

$$\Delta^k = \begin{pmatrix} a_{1,1} & \cdots & a_{1,k} \\ \vdots & & \vdots \\ a_{k,1} & \cdots & a_{k,k} \end{pmatrix},$$

are called the diagonal submatrices of order k of $A \in \mathcal{M}_n(\mathbb{R})$.

Theorem 3.2

Let $A = (a_{i,j})_{1 \leq i,j \leq n}$ be a square matrix such that all its diagonal submatrices of order k are nonsingular. There exists a unique pair of matrices (L, U), with U upper triangular and L lower triangular with a unit diagonal (i.e. $l_{i,i} = 1$), such that $A = LU$[a].

[a] p3-d2

- The matrix A is scanned column by column
- At the kth step, column k is changed such that the entries below the diagonal vanish by performing linear combinations of the kth row with every row from the $(k+1)$th to the nth
- At the kth step, the first k rows and the first $k-1$ columns are no longer modified $\Rightarrow$ use this space to store the corresponding nonzero entries of L^k

Data: A. Output: A containing U and L (but its diagonal).

For $k = 1 \nearrow n-1$
  For $i = k+1 \nearrow n$
    $a_{i,k} = \dfrac{a_{i,k}}{a_{k,k}}$
    For $j = k+1 \nearrow n$
      $a_{i,j} = a_{i,j} - a_{i,k} a_{k,j}$
    End For j
  End For i
End For k

- The matrix A is scanned column by column
- At the kth step, column k is changed such that the entries below the diagonal vanish by performing linear combinations of the kth row with every row from the $(k+1)$th to the nth
- At the kth step, the first k rows and the first $k-1$ columns are no longer modified $\Rightarrow$ use this space to store the corresponding nonzero entries of L^k

Data: A. Output: A containing U and L (but its diagonal).

For $k = 1 \nearrow n-1$
  For $i = k+1 \nearrow n$
    $a_{i,k} = \dfrac{a_{i,k}}{a_{k,k}}$
    For $j = k+1 \nearrow n$
      $a_{i,j} = a_{i,j} - a_{i,k} a_{k,j}$
    End For j
  End For i
End For k

- Operation count (multiplications and divisions only)
  - LU factorization:

$$N_{\text{op}}(n) = \sum_{k=1}^{n-1} \sum_{i=k+1}^{n} \left(1 + \sum_{j=k+1}^{n} 1 \right),$$

    which to first order yields,

$$N_{\text{op}}(n) \approx \frac{n^3}{3}.$$

  - Back substitution (on a triangular system):

$$N_{\text{op}}(n) = \sum_{j=1}^{n} j \approx \frac{n^2}{2}.$$

  - Solution of a linear system $Ax = b$: an LU factorization of A is followed by two substitutions, $Ly = b$ and $Ux = y$:

$$N_{\text{op}}(n) \approx \frac{n^3}{3} \quad \text{(for } n \text{ large)}.$$

- Computing $\det(A)$: product of the diagonal entries of U
- Computing A^{-1}: the columns of the inverse are the solutions of the n systems $Ax_i = e_i$ where $(e_i)_{1 \leq i \leq n}$ is the canonical basis of $\mathbb{R}^n$

- Operation count (multiplications and divisions only)
  - LU factorization:

$$N_{\text{op}}(n) = \sum_{k=1}^{n-1} \sum_{i=k+1}^{n} \left(1 + \sum_{j=k+1}^{n} 1 \right),$$

    which to first order yields,

$$N_{\text{op}}(n) \approx \frac{n^3}{3}.$$

  - Back substitution (on a triangular system):

$$N_{\text{op}}(n) = \sum_{j=1}^{n} j \approx \frac{n^2}{2}.$$

  - Solution of a linear system $Ax = b$: an LU factorization of A is followed by two substitutions, $Ly = b$ and $Ux = y$:

$$N_{\text{op}}(n) \approx \frac{n^3}{3} \quad \text{(for } n \text{ large)}.$$

- Computing $\det(A)$: product of the diagonal entries of U
- Computing A^{-1}: the columns of the inverse are the solutions of the n systems $Ax_i = e_i$ where $(e_i)_{1 \leq i \leq n}$ is the canonical basis of $\mathbb{R}^n$

- The Cholesky method applies to real symmetric positive definite matrices.

Theorem 3.3

Let A be a real symmetric positive definite matrix. There exists a unique real lower triangular matrix B, having positive diagonal entries, such that $A = BB^{\star}$[a].

[a] p3-d3

- The Cholesky method applies to real symmetric positive definite matrices.

Theorem 3.3

Let A be a real symmetric positive definite matrix. There exists a unique real lower triangular matrix B, having positive diagonal entries, such that $A = BB^{\star}$[a].

[a] p3-d3

Data: A. Output: A containing B in its lower triangular part.

For $j = 1 \nearrow n$
  For $k = 1 \nearrow j-1$
    $a_{j,j} = a_{j,j} - (a_{j,k})^2$
  End k
  $a_{j,j} = \sqrt{a_{j,j}}$
  For $i = j+1 \nearrow n$
    For $k = 1 \nearrow j-1$
      $a_{i,j} = a_{i,j} - a_{j,k} a_{i,k}$
    End k
    $a_{i,j} = \dfrac{a_{i,j}}{a_{j,j}}$
  End i
End j

- Operation count (multiplications and divisions only)
- Only n square roots $\Rightarrow$ not taken into account
  - Cholesky factorization:

$$N_{\text{op}}(n) = \sum_{j=1}^{n} \left((j-1) + \sum_{i=j+1}^{n} j \right),$$

    which to first order yields,

$$N_{\text{op}}(n) \approx \frac{n^3}{6}.$$

  - Substitutions (a forward and a backward substitutions are performed on the triangular systems associated with B and $B^\star$): $N_{\text{op}}(n) \approx n^2$.
  - Solution of a linear system $Ax = b$,

$$N_{\text{op}}(n) \approx \frac{n^3}{6} \quad \text{(for } n \text{ large)}.$$

  - The Cholesky method is approximately twice as fast as the Gauss method for a positive definite symmetric matrix

The goal of the QR factorization is to reduce the solution of a linear system to that of a triangular one.

However, the original matrix A is not factorized as the product of two triangular matrices but as the product of an upper triangular matrix R and an orthogonal (unitary) matrix Q (recall that $Q^{-1} = Q^\star$).

Solution of $Ax = b$ follows a three steps process:

1. factorization - find an orthogonal matrix Q such that $Q^\star A = R$ is upper triangular,
2. right-hand side update - computation of the product $Q^\star b$,
3. substitution - solution of the triangular system $Rx = Q^\star b$.

Theorem 3.4

Let A be a real nonsingular square matrix. There exists a unique pair (Q, R), where Q is an orthogonal matrix and R is an upper triangular matrix, whose diagonal entries are positive, satisfying $A = QR$[a].

[a] p3-d4

The goal of the QR factorization is to reduce the solution of a linear system to that of a triangular one.

However, the original matrix A is not factorized as the product of two triangular matrices but as the product of an upper triangular matrix R and an orthogonal (unitary) matrix Q (recall that $Q^{-1} = Q^\star$).

Solution of $Ax = b$ follows a three steps process:

1. factorization - find an orthogonal matrix Q such that $Q^\star A = R$ is upper triangular,
2. right-hand side update - computation of the product $Q^\star b$,
3. substitution - solution of the triangular system $Rx = Q^\star b$.

Theorem 3.4

Let A be a real nonsingular square matrix. There exists a unique pair (Q, R), where Q is an orthogonal matrix and R is an upper triangular matrix, whose diagonal entries are positive, satisfying $A = QR$[a].

[a] p3-d4

The goal of the QR factorization is to reduce the solution of a linear system to that of a triangular one.

However, the original matrix A is not factorized as the product of two triangular matrices but as the product of an upper triangular matrix R and an orthogonal (unitary) matrix Q (recall that $Q^{-1} = Q^\star$).

Solution of $Ax = b$ follows a three steps process:

1. factorization - find an orthogonal matrix Q such that $Q^\star A = R$ is upper triangular,
2. right-hand side update - computation of the product $Q^\star b$,
3. substitution - solution of the triangular system $Rx = Q^\star b$.

Theorem 3.4

Let A be a real nonsingular square matrix. There exists a unique pair (Q, R), where Q is an orthogonal matrix and R is an upper triangular matrix, whose diagonal entries are positive, satisfying $A = QR$[a].

[a] p3-d4

- Operation count (multiplications and divisions only)
- Only n square roots $\Rightarrow$ not taken into account
  - Gram-Schmidt factorizationn

$$N_{\text{op}}(n) = \sum_{i=1}^{n} \left((i-1)2n + (n+1) \right),$$

    which to first order yields,

$$N_{\text{op}}(n) \approx n^3.$$

  - Updating the right-hand side: computation of the matrix-vector product $Q^\star b$ requires $N_{\text{op}}(n) \approx n^2$.
  - Substitution: solution of the triangular system associated with R requires $N_{\text{op}}(n) \approx \dfrac{n^2}{2}$.
  - Solution of a linear system $Ax = b$,

$$N_{\text{op}}(n) \approx n^3 \quad (\text{for } n \text{ large}).$$

  - The Gram-Schmidt algorithm for the QR method is thus three times slower that the Gauss method (and is thus rarely used in practice)

Definition 3.5

A matrix $A \in \mathcal{M}_n(\mathbb{K})$ satisfying $a_{i,j} = 0$ for $|i-j| > p$ with $p \in \mathbb{N}$ is said to be a band matrix of bandwitdth $2p+1$.

Proposition 3.6

The LU factorization preserves the band structure of matrices, that is the L and U factors have same bandwitdth as the original matrix A[a].

[a] p3-d5

Definition 3.5

A matrix $A \in \mathcal{M}_n(\mathbb{K})$ satisfying $a_{i,j} = 0$ for $|i-j| > p$ with $p \in \mathbb{N}$ is said to be a band matrix of bandwitdth $2p+1$.

Proposition 3.6

The LU factorization preserves the band structure of matrices, that is the L and U factors have same bandwitdth as the original matrix A[a].

[a] p3-d5

- Symmetric tridiagonal matrix

$$T = \begin{pmatrix} a_1 & -b_2 & & & \\ -b_2 & a_2 & -b_3 & & \\ & \ddots & \ddots & \ddots & \\ & & -b_{n-1} & a_{n-1} & -b_n \\ & & & -b_n & a_n \end{pmatrix}.$$

  - We assume $b_i \neq 0, \ \forall i$
  - The minus sign in front of the b_i's is a technical convenience

- The Cholesky factorization is $T = LD_L^{-1}L^t$ with,

$$L = \begin{pmatrix} \delta_1 & & & \\ -b_2 & \delta_2 & & \\ & \ddots & \ddots & \\ & & -b_{n-1} & \delta_{n-1} \\ & & & -b_n & \delta_n \end{pmatrix} \quad \text{and} \quad D_L = \mathtt{diag}(\delta_1, \delta_2, \cdots, \delta_{n-1}, \delta_n),$$

$$\delta_1 = a_1 \quad \text{and} \quad \delta_i = a_i - \frac{b_i^2}{\delta_{i-1}} \quad \text{for} \quad i = 2, \cdots, n.$$

- An alternative factorization with an upper triangular matrix, is also easily obtained, that is $T = UD_U^{-1}U^t$ with,

$$U = \begin{pmatrix} d_1 & -b_2 & & \\ & d_2 & -b_3 & \\ & \ddots & \ddots & \\ & & d_{n-1} & -b_n \\ & & & d_n \end{pmatrix} \quad \text{and} \quad D_U = \mathtt{diag}(d_1, d_2, \cdots, d_{n-1}, d_n),$$

$$d_n = a_n \quad \text{and} \quad d_i = a_i - \frac{b_{i+1}^2}{d_{i+1}} \quad \text{for} \quad i = n-1, \cdots, 1.$$

- The Cholesky factorization is $T = LD_L^{-1}L^t$ with,

$$L = \begin{pmatrix} \delta_1 & & & \\ -b_2 & \delta_2 & & \\ & \ddots & \ddots & \\ & & -b_{n-1} & \delta_{n-1} \\ & & & -b_n & \delta_n \end{pmatrix} \quad \text{and} \quad D_L = \mathtt{diag}(\delta_1, \delta_2, \cdots, \delta_{n-1}, \delta_n),$$

$$\delta_1 = a_1 \quad \text{and} \quad \delta_i = a_i - \frac{b_i^2}{\delta_{i-1}} \quad \text{for} \quad i = 2, \cdots, n.$$

- An alternative factorization with an upper triangular matrix, is also easily obtained, that is $T = UD_U^{-1}U^t$ with,

$$U = \begin{pmatrix} d_1 & -b_2 & & \\ & d_2 & -b_3 & \\ & \ddots & \ddots & \\ & & d_{n-1} & -b_n \\ & & & d_n \end{pmatrix} \quad \text{and} \quad D_U = \mathtt{diag}(d_1, d_2, \cdots, d_{n-1}, d_n),$$

$$d_n = a_n \quad \text{and} \quad d_i = a_i - \frac{b_{i+1}^2}{d_{i+1}} \quad \text{for} \quad i = n-1, \cdots, 1.$$

Theorem 3.7

(G. Meurant, 1992)
The inverse of T is caracterized as,

$$\begin{cases} (T^{-1})_{i,j} &= b_{i+1} \cdots b_j \left(\dfrac{d_{j+1} \cdots d_n}{\delta_i \cdots \delta_n} \right), \quad \forall i, \quad \forall j > i, \\ (T^{-1})_{i,i} &= \left(\dfrac{d_{i+1} \cdots d_n}{\delta_i \cdots \delta_n} \right), \quad \forall i. \end{cases}$$

where terms that have indices greater that n must be taken equal to 1.

Theorem 3.8

Let $A \in \mathcal{M}_n(\mathbb{K})$ be a (row or column) digonally dominant matrix. Then there exists triangular matrices L (unit lower triangular) and U (upper triangular) such that $A = LU$.

Suppose that A is row diagonally dominant,

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

then $a_{1,1} \neq 0$, otherwise all the elements in the first row are 0 and A is singular.

For any other row we have,

$$\begin{aligned} a_{i,j}^{(2)} &= a_{i,j} - \frac{a_{i,1}a_{1,j}}{a_{1,1}}, \quad 2 \leq i \leq n, \quad 2 \leq j \leq n, \\ a_{i,1}^{(2)} &= 0, \quad 2 \leq i \leq n, \end{aligned}$$

$$\sum_{j, j\neq i}^{n} |a_{i,j}^{(2)}| = \sum_{j, j\neq i, j\neq 1}^{n} |a_{i,j}^{(2)}| \leq \sum_{j, j\neq i, j\neq 1}^{n} |a_{i,j}| + \left|\frac{a_{i,1}}{a_{1,1}}\right| \sum_{j, j\neq i, j\neq 1}^{n} |a_{1,j}|.$$

But,

$$|a_{1,1}| \geq |a_{1,i}| + \sum_{j,j\neq i,j\neq 1}^{n} |a_{1,j}|.$$

Therefore,

$$\begin{aligned}
\sum_{j,j\neq i}^{n} |a_{i,j}^{(2)}| &\leq \sum_{j,j\neq i,j\neq 1}^{n} |a_{i,j}| + \left|\frac{a_{i,1}}{a_{1,1}}\right| (|a_{1,1}| - |a_{1,i}|) \\
&\leq \sum_{j,j\neq i}^{n} |a_{i,j}| - \frac{|a_{i,1}a_{1,i}|}{|a_{1,1}|} \\
&\leq |a_{i,i}| - \frac{|a_{i,1}a_{1,i}|}{|a_{1,1}|} \quad \left(= \left||a_{i,i}| - \frac{|a_{i,1}a_{1,i}|}{|a_{1,1}|}\right|\right) \\
&\leq \left|a_{i,i} - \frac{a_{i,1}a_{1,i}}{a_{1,1}}\right| = |a_{i,i}^{(2)}|.
\end{aligned}$$

- The reduced matrix is also diagonally dominant.
- All the pivots are non-zero.

Definition 3.9

A matrix $A \in \mathcal{M}_n(\mathbb{K})$ which can be written as $A = \sigma I_n - B$ with $\sigma > 0$, $B \geq 0$ (i.e. $(B_{i,j})_{1 \leq i,j \leq n} \geq 0$) abd such that $\varrho(B) \leq \sigma$ is said to be a M-matrix.

Recall that $\varrho(B) = \max\limits_{1 \leq i \leq n} |\lambda_i(B)|$.

Theorem 3.10

A matrix $A \in \mathcal{M}_n(\mathbb{K})$ is a nonsingular M-matrix if and only if A is nonsingular with $a_{i,j} \leq 0$ for $i \neq j$, and $A^{-1} \geq 0$.

Theorem 3.11

Let A be M-matrix. Then there exists triangular matrices L (unit lower triangular) and U (upper triangular) such that $A = LU$.

Definition 3.9

A matrix $A \in \mathcal{M}_n(\mathbb{K})$ which can be written as $A = \sigma I_n - B$ with $\sigma > 0$, $B \geq 0$ (i.e. $(B_{i,j})_{1 \leq i,j \leq n} \geq 0$) abd such that $\varrho(B) \leq \sigma$ is said to be a M-matrix.

Recall that $\varrho(B) = \max\limits_{1 \leq i \leq n} |\lambda_i(B)|$.

Theorem 3.10

A matrix $A \in \mathcal{M}_n(\mathbb{K})$ is a nonsingular M-matrix if and only if A is nonsingular with $a_{i,j} \leq 0$ for $i \neq j$, and $A^{-1} \geq 0$.

Theorem 3.11

Let A be M-matrix. Then there exists triangular matrices L (unit lower triangular) and U (upper triangular) such that $A = LU$.

Definition 3.9

A matrix $A \in \mathcal{M}_n(\mathbb{K})$ which can be written as $A = \sigma I_n - B$ with $\sigma > 0$, $B \geq 0$ (i.e. $(B_{i,j})_{1 \leq i,j \leq n} \geq 0$) abd such that $\varrho(B) \leq \sigma$ is said to be a M-matrix.

Recall that $\varrho(B) = \max\limits_{1 \leq i \leq n} |\lambda_i(B)|$.

Theorem 3.10

A matrix $A \in \mathcal{M}_n(\mathbb{K})$ is a nonsingular M-matrix if and only if A is nonsingular with $a_{i,j} \leq 0$ for $i \neq j$, and $A^{-1} \geq 0$.

Theorem 3.11

Let A be M-matrix. Then there exists triangular matrices L (unit lower triangular) and U (upper triangular) such that $A = LU$.

Outline

Relaxation methods

This section discusses the solution of,

$$Ax = b,$$

where $A \in \mathcal{M}_n(\mathbb{R})$ is a square matrix and $b \in \mathbb{R}^n$ (here $K = \mathbb{R}$ to simplify the presentation), by means of iterative methods.

A method for solving the linear system $Ax = b$ is called iterative if it is a numerical method computing a sequence of approximate solutions x_k that converges to the exact solution x as the number of iterations k goes to $+\infty$.

Relaxation methods: x_{k+1} is a function of x_k only and not of the previous iterates.

Definition 4.1

Lest A be a nonsingular matrix. A pair of matrices (M, N) with M nonsingular (and easily invertible in practice) satisfying,

$$A = M - N,$$

is called a splitting (or regular decomposition) of A.
An iterative method based on the splitting (M, N) is defined by,

$$x_0 \text{ given in } \mathbb{R}^n, \quad Mx_{k+1} = Nx_k + b, \quad \forall k \geq 1.$$

Relaxation methods

This section discusses the solution of,

$$Ax = b,$$

where $A \in \mathcal{M}_n(\mathbb{R})$ is a square matrix and $b \in \mathbb{R}^n$ (here $K = \mathbb{R}$ to simplify the presentation), by means of iterative methods.

A method for solving the linear system $Ax = b$ is called iterative if it is a numerical method computing a sequence of approximate solutions x_k that converges to the exact solution x as the number of iterations k goes to $+\infty$.

Relaxation methods: x_{k+1} is a function of x_k only and not of the previous iterates.

Definition 4.1

Lest A be a nonsingular matrix. A pair of matrices (M, N) with M nonsingular (and easily invertible in practice) satisfying,

$$A = M - N,$$

is called a splitting (or regular decomposition) of A.
An iterative method based on the splitting (M, N) is defined by,

$$x_0 \text{ given in } \mathbb{R}^n, \qquad Mx_{k+1} = Nx_k + b, \quad \forall k \geq 1.$$

Relaxation methods

The task of solving the linear system $Ax = b$ is replaced by a sequence of several linear systems $M\tilde{x} = \tilde{b}$ to be solved.

Therefore, M has to be much easier to invert than A.

Definition 4.2

An iterative method is said to converge if for any choice of the initial vector $x_0 \in \mathbb{R}^n$, the sequence of approximate solutions x_k converges to the exact solution x.

Definition 4.3

We call the vector $r_k = b - Ax_k$ (respectively, $e_k = x_k - x$) residual (respectively, error) at the kth iteration.

Obviously, an iterative method converges if and only if e_k converges to 0, which is equivalent to $r_k = Ae_k$ converging to 0.

Convergence is detected on the residual in practice.

The matrix $M^{-1}N$ is called an iteration matrix or amplification matrix of the iterative method.

Relaxation methods

The task of solving the linear system $Ax = b$ is replaced by a sequence of several linear systems $M\tilde{x} = \tilde{b}$ to be solved.

Therefore, M has to be much easier to invert than A.

Definition 4.2

An iterative method is said to converge if for any choice of the initial vector $x_0 \in \mathbb{R}^n$, the sequence of approximate solutions x_k converges to the exact solution x.

Definition 4.3

We call the vector $r_k = b - Ax_k$ (respectively, $e_k = x_k - x$) residual (respectively, error) at the kth iteration.

Obviously, an iterative method converges if and only if e_k converges to 0, which is equivalent to $r_k = Ae_k$ converging to 0.

Convergence is detected on the residual in practice.

The matrix $M^{-1}N$ is called an iteration matrix or amplification matrix of the iterative method.

Relaxation methods

The task of solving the linear system $Ax = b$ is replaced by a sequence of several linear systems $M\tilde{x} = \tilde{b}$ to be solved.

Therefore, M has to be much easier to invert than A.

Definition 4.2

An iterative method is said to converge if for any choice of the initial vector $x_0 \in \mathbb{R}^n$, the sequence of approximate solutions x_k converges to the exact solution x.

Definition 4.3

We call the vector $r_k = b - Ax_k$ (respectively, $e_k = x_k - x$) residual (respectively, error) at the kth iteration.

Obviously, an iterative method converges if and only if e_k converges to 0, which is equivalent to $r_k = Ae_k$ converging to 0.

Convergence is detected on the residual in practice.

The matrix $M^{-1}N$ is called an iteration matrix or amplification matrix of the iterative method.

Relaxation methods

The task of solving the linear system $Ax = b$ is replaced by a sequence of several linear systems $M\tilde{x} = \tilde{b}$ to be solved.

Therefore, M has to be much easier to invert than A.

Definition 4.2

An iterative method is said to converge if for any choice of the initial vector $x_0 \in \mathbb{R}^n$, the sequence of approximate solutions x_k converges to the exact solution x.

Definition 4.3

We call the vector $r_k = b - Ax_k$ (respectively, $e_k = x_k - x$) residual (respectively, error) at the kth iteration.

Obviously, an iterative method converges if and only if e_k converges to 0, which is equivalent to $r_k = Ae_k$ converging to 0.

Convergence is detected on the residual in practice.

The matrix $M^{-1}N$ is called an iteration matrix or amplification matrix of the iterative method.

Theorem 4.4

The iterative method defined by 4.1 converges if and only if he spectral radius of $M^{-1}N$ satisfies,

$$\varrho(M^{-1}N) < 1^a.$$

[a] p4-d1

pause

Theorem 4.5

Let A be an Hermitian positive definite matrix. Consider a splitting of $A = M - N$ with M nonsingular. Then the matrix $(M^\star + N)$ is Hermitian.

Furthermore, if $(M^\star + N)$ is also positive definite, we have,

$$\varrho(M^{-1}N) < 1^a.$$

[a] p4-d2

Relaxation methods

Iterative methods are often used with sparse matrices (i.e. matrices that have relatively few nonzero entries).

Such matrices often result from the discretization of PDEs by finite difference, finite element or finite volume methods.

Specific storage formats are used for such matrices.

Iterative methods for solving linear systems may require a large number of iterations to converge.

Thus, one might think that that the accumulation of rounding errors during the iterations completely destroys the convergence of these methods on computers (or even worse, makes them converge to wrong solutions).

Relaxation methods

Iterative methods are often used with sparse matrices (i.e. matrices that have relatively few nonzero entries).

Such matrices often result from the discretization of PDEs by finite difference, finite element or finite volume methods.

Specific storage formats are used for such matrices.

Iterative methods for solving linear systems may require a large number of iterations to converge.

Thus, one might think that that the accumulation of rounding errors during the iterations completely destroys the convergence of these methods on computers (or even worse, makes them converge to wrong solutions).

Theorem 4.6

Consider a splitting of $A = M - N$ with A and M nonsingular. Let $b \in \mathbb{R}^n$ be the right-hand side, and let $x \in \mathbb{R}^n$ be the solution of $Ax = b$.
We assume that at each step k the iterative method is tainted by an error $\epsilon_k \in \mathbb{R}^n$, meaning that x_{k+1} is not exactly given by,

$$x_{k+1} = M^{-1}Nx_k + M^{-1}b,$$

but rather by,

$$x_{k+1} = M^{-1}Nx_k + M^{-1}b + \epsilon_k.$$

We also assume that $\varrho(M^{-1}N) < 1$ and that there exist a vector norm and a positive constant ε such that for all $k \geq 0$,

$$\| \epsilon_k \| \leq \varepsilon.$$

Then, there exists a constant K, which depends on $M^{-1}N$ but not on ε, such that,

$$\lim_{x \to +\infty} \sup \| x_k - x \| \leq K\varepsilon^a$$

[a] p4-d3

Definition 4.7

The Jacobi method is the iterative method defined by the splitting,

$$M = D \quad , \quad N = D - A.$$

The iteration matrix of this method is denoted by,

$$\mathcal{J} = M^{-1}N = I_n - D^{-1}A.$$

- The Jacobi method is well defined if the diagonal part of A is nonsingular
- If A is Hermitian, the Jacobi method converges if A and $2D - A$ are positive definite

Definition 4.7

The Jacobi method is the iterative method defined by the splitting,

$$M = D \quad , \quad N = D - A.$$

The iteration matrix of this method is denoted by,

$$\mathcal{J} = M^{-1}N = I_n - D^{-1}A.$$

- The Jacobi method is well defined if the diagonal part of A is nonsingular
- If A is Hermitian, the Jacobi method converges if A and $2D - A$ are positive definite

Relaxation methods

For any matrix $A = (a_{i,j})_{1 \leq i,j \leq n}$ consider the decomposition $A = D - E - F$, where D is the diagonal part, $-E$ the lower triangular part and $-F$ the upper triangular part of A.

Definition 4.8

The Gauss-Seidel method is the iterative method defined by the splitting,

$$M = D - E \quad , \quad N = F.$$

The iteration matrix of this method is denoted by,

$$\mathcal{G} = M^{-1}N = (D - E)^{-1}F.$$

- The Gauss-Seidel method is well defined if the matrix $D - E$ is nonsingular, which is equivalent to asking that D be nonsingular
- The matrix $D - E$ is easy to invert since it is triangular
- If A is Hermitian and positive definite, then $M^{\star} + N = D$ is also Hermitian and positive definite, so the Gauss-Seidel method converges

Relaxation methods

For any matrix $A = (a_{i,j})_{1 \leq i,j \leq n}$ consider the decomposition $A = D - E - F$, where D is the diagonal part, $-E$ the lower triangular part and $-F$ the upper triangular part of A.

Definition 4.8

The Gauss-Seidel method is the iterative method defined by the splitting,

$$M = D - E \quad , \quad N = F.$$

The iteration matrix of this method is denoted by,

$$\mathcal{G} = M^{-1}N = (D - E)^{-1}F.$$

- The Gauss-Seidel method is well defined if the matrix $D - E$ is nonsingular, which is equivalent to asking that D be nonsingular
- The matrix $D - E$ is easy to invert since it is triangular
- If A is Hermitian and positive definite, then $M^{\star} + N = D$ is also Hermitian and positive definite, so the Gauss-Seidel method converges

Relaxation methods

For any matrix $A = (a_{i,j})_{1 \leq i,j \leq n}$ consider the decomposition $A = D - E - F$, where D is the diagonal part, $-E$ the lower triangular part and $-F$ the upper triangular part of A.

Definition 4.8

The Gauss-Seidel method is the iterative method defined by the splitting,

$$M = D - E \quad , \quad N = F.$$

The iteration matrix of this method is denoted by,

$$\mathcal{G} = M^{-1}N = (D - E)^{-1}F.$$

- The Gauss-Seidel method is well defined if the matrix $D - E$ is nonsingular, which is equivalent to asking that D be nonsingular
- The matrix $D - E$ is easy to invert since it is triangular
- If A is Hermitian and positive definite, then $M^{\star} + N = D$ is also Hermitian and positive definite, so the Gauss-Seidel method converges

Definition 4.9

Let $\omega \in \mathbb{R}^+$. The iterative method defined by the splitting,

$$M = \frac{D}{\omega} - E \quad , \quad N = \frac{1-\omega}{\omega} D + F.$$

is called relaxation method for the parameter ω.

The iteration matrix of this method is denoted by,

$$\mathcal{G}_\omega = M^{-1}N = \left(\frac{D}{\omega} - E\right)^{-1} \left(\frac{1-\omega}{\omega} D + F\right).$$

- The relaxation method is well defined if D is nonsingular
- If $\omega = 1$, we recover the Gauss-Seidel method
- If $\omega < 1$, we talk about an under-relaxation method
- If $\omega > 1$, we talk about an over-relaxation method
- The idea is to look for an optimal ω that produces the the smallest spectral radius $\varrho(\mathcal{G}_\omega)$ possible
- A relaxation method for the Jacobi method also exists

Definition 4.9

Let $\omega \in \mathbb{R}^+$. The iterative method defined by the splitting,

$$M = \frac{D}{\omega} - E \quad , \quad N = \frac{1-\omega}{\omega} D + F.$$

is called relaxation method for the parameter ω.

The iteration matrix of this method is denoted by,

$$\mathcal{G}_\omega = M^{-1}N = \left(\frac{D}{\omega} - E\right)^{-1} \left(\frac{1-\omega}{\omega} D + F\right).$$

- The relaxation method is well defined if D is nonsingular
- If $\omega = 1$, we recover the Gauss-Seidel method
- If $\omega < 1$, we talk about an under-relaxation method
- If $\omega > 1$, we talk about an over-relaxation method
- The idea is to look for an optimal ω that produces the the smallest spectral radius $\varrho(\mathcal{G}_\omega)$ possible
- A relaxation method for the Jacobi method also exists

Theorem 4.10

Let A be a Hermitian positive definite matrix.

Then for any $\omega \in]0, 2[$, the relaxation method converges[a].

[a] p4-d4

Theorem 4.11

For any matrix A, we always have,

$$\varrho(\mathcal{G}_\omega) \geq |1 - \omega|, \quad \forall \omega \neq 0.$$

Consequently, the relaxation method can converge only if $0 < \omega < 2$[a].

[a] p4-d5

Theorem 4.10

Let A be a Hermitian positive definite matrix.

Then for any $\omega \in]0, 2[$, the relaxation method converges[a].

[a] p4-d4

Theorem 4.11

For any matrix A, we always have,

$$\varrho(\mathcal{G}_\omega) \geq |1 - \omega|, \qquad \forall \omega \neq 0.$$

Consequently, the relaxation method can converge only if $0 < \omega < 2$[a].

[a] p4-d5

Theorem 4.12

Let A be a tridiagonal matrix. Then

$$\varrho(\mathcal{G}) = \varrho(\mathcal{J})^2,$$

so the Jacobi and Gauss-Seidel methods converge or diverge simultaneously, but Gauss-Seidel alway converges faster than Jacobi.

Theorem 4.12

Let A be a tridiagonal matrix. Then

$$\varrho(\mathcal{G}) = \varrho(\mathcal{J})^2,$$

so the Jacobi and Gauss-Seidel methods converge or diverge simultaneously, but Gauss-Seidel alway converges faster than Jacobi.

Theorem 4.13

Let A be a tridiagonal Hermitian positive definite matrix.

Then the Jacobi, Gauss-Seidel and relaxation methods converge.

Moreover, there exists a unique optimal parameter ω_{opt} in the sense that,

$$\varrho(\mathcal{G}_{\omega_{opt}}) = \min_{0<\omega<2} \varrho(\mathcal{G}_\omega),$$

where,

$$\omega_{opt} = \frac{2}{1+\sqrt{1-\varrho(\mathcal{J})^2}},$$

and,

$$\varrho(\mathcal{G}_{\omega_{opt}}) = \omega_{opt} - 1.$$

Thus, for a tridiagonal Hermitian positive definite matrix, we have $\omega_{opt} \geq 1$ and it is better to perform over-relaxation than under-relaxation.

The CG method actually is a direct method used as an iterative one.

For simplicity, we will restrict ourselves throughout this section, to real symmetric matrices (results extend easily to complex self-adjoint matrices).

Definition 4.14

The iterative method, known as the gradient method, is defined by the following regular decomposition,

$$M = \frac{1}{\alpha} I_n \quad , \quad N = \frac{1}{\alpha} I_n - A,$$

where α is a real nonzero parameter.

In other words, the gradient method consists in computing the sequence x_k defined by,

$$x_0 \text{ given in } \mathbb{R}^n, \quad x_{k+1} = x_k + \alpha(b - Ax_k), \quad \forall k \geq 1.$$

The CG method actually is a direct method used as an iterative one.

For simplicity, we will restrict ourselves throughout this section, to real symmetric matrices (results extend easily to complex self-adjoint matrices).

Definition 4.14

The iterative method, known as the gradient method, is defined by the following regular decomposition,

$$M = \frac{1}{\alpha} I_n \quad , \quad N = \frac{1}{\alpha} I_n - A,$$

where α is a real nonzero parameter.

In other words, the gradient method consists in computing the sequence x_k defined by,

$$x_0 \text{ given in } \mathbb{R}^n, \quad x_{k+1} = x_k + \alpha(b - Ax_k), \quad \forall k \geq 1.$$

Theorem 4.15

Let A be a matrix with eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

1. If $\lambda_1 \leq 0 \leq \lambda_n$, then the gradient method does not converge for any value of α.
2. If $0 < \lambda_1 \leq \cdots \leq \lambda_n$, then the gradient method converges if and only if $0 < \alpha < \frac{2}{\lambda_n}$. In this case, the optimal parameter α, which minimizes $\varrho(M^{-1}N)$ is,

$$\alpha_{opt} = \frac{2}{\lambda_n + \lambda_1} \quad \text{and} \quad \min_{\alpha} \varrho(M^{-1}N) = \frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1} = \frac{\mathrm{cond}_2(A) - 1}{\mathrm{cond}_2(A) + 1}.$$

[a] p4-d6

- A is a normal matrix ($AA^t = A^tA$)
- The conditioning of a normal invertible matrix A is $\mathrm{cond}_2(A) = \frac{\lambda_n}{\lambda_1}$ (with the assumption of the second point). Thus, for the optimal parameter α_{opt}, the better the matrix A is conditioned, the faster the gradient method converges.

Theorem 4.15

Let A be a matrix with eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

1. If $\lambda_1 \leq 0 \leq \lambda_n$, then the gradient method does not converge for any value of α.
2. If $0 < \lambda_1 \leq \cdots \leq \lambda_n$, then the gradient method converges if and only if $0 < \alpha < \frac{2}{\lambda_n}$. In this case, the optimal parameter α, which minimizes $\varrho(M^{-1}N)$ is,

$$\alpha_{opt} = \frac{2}{\lambda_n + \lambda_1} \quad \text{and} \quad \min_{\alpha} \varrho(M^{-1}N) = \frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1} = \frac{\mathrm{cond}_2(A) - 1}{\mathrm{cond}_2(A) + 1}.$$

[a] p4-d6

- A is a normal matrix ($AA^t = A^tA$)
- The conditioning of a normal invertible matrix A is $\mathrm{cond}_2(A) = \frac{\lambda_n}{\lambda_1}$ (with the assumption of the second point). Thus, for the optimal parameter α_{opt}, the better the matrix A is conditioned, the faster the gradient method converges.

Gradient algorithm

Data: A, b. Output: $\mathbf{x}$ (approximation of the solution of $Ax = b$).

Initialization
  Choose α
  Choose $\mathbf{x} \in \mathbb{R}^n$
  Compute $r = b - A\mathbf{x}$

While $\| r \|_2 > \varepsilon \| b \|_2$
  $\mathbf{x} = \mathbf{x} + \alpha r$
  $r = b - A\mathbf{x}$
End While

Definition 4.16

Let f be a function from $\mathbb{R}^n$ into $\mathbb{R}$. We call the vector of partial derivatives at the point x the gradient (or differential) of the function f at x, which we denote by,

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}(x), \cdots, \frac{\partial f}{\partial x_n}(x) \right)^t.$$

Definition 4.17

The iterative method for solving the linear system $Ax = b$, known as the gradient method with variable step size, is defined by,

$$x_0 \text{ given in } \mathbb{R}^n, \quad x_{k+1} = x_k + \alpha_k (b - Ax_k), \quad \forall k \geq 1.$$

where α_k is chosen as the minimizer of the function,

$$g(\alpha) = f(x_k - \alpha \nabla f(x_k)),$$

and where $f(x)$ is the quadratic functional $\frac{1}{2} < Ax, x > - < b, x >$.

Definition 4.16

Let f be a function from $\mathbb{R}^n$ into $\mathbb{R}$. We call the vector of partial derivatives at the point x the gradient (or differential) of the function f at x, which we denote by,

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}(x), \cdots, \frac{\partial f}{\partial x_n}(x) \right)^t.$$

Definition 4.17

The iterative method for solving the linear system $Ax = b$, known as the gradient method with variable step size, is defined by,

$$x_0 \text{ given in } \mathbb{R}^n, \qquad x_{k+1} = x_k + \alpha_k (b - Ax_k), \quad \forall k \geq 1.$$

where α_k is chosen as the minimizer of the function,

$$g(\alpha) = f(x_k - \alpha \nabla f(x_k)),$$

and where $f(x)$ is the quadratic functional $\frac{1}{2} < Ax, x > - < b, x >$.

Lemma 4.18

Let A be a positive definite symmetric matrix. For the gradient method with variable step size, there exists a unique optimal step size given by,

$$\alpha_k = \frac{\| b - Ax_k \|^2}{< A(b - Ax_k), (b - Ax_k) >}.$$

- When $b - Ax_k = 0$, the method has converged!

Lemma 4.18

Let A be a positive definite symmetric matrix. For the gradient method with variable step size, there exists a unique optimal step size given by,

$$\alpha_k = \frac{\| b - Ax_k \|^2}{< A(b - Ax_k), (b - Ax_k) >}.$$

- When $b - Ax_k = 0$, the method has converged!

Gradient algorithm with variable step size

Data: A, b. Output: $\mathbf{x}$ (approximation of the solution of $Ax = b$).

Initialization

Choose $\mathbf{x} \in \mathbb{R}^n$

Compute $r = b - A\mathbf{x}$

Compute $\alpha = \dfrac{\| r \|_2^2}{< Ar, r >}$

While $\| r \|_2 > \varepsilon \| b \|_2$

$\mathbf{x} = \mathbf{x} + \alpha r$

$r = b - A\mathbf{x}$

$\alpha = \dfrac{\| r \|_2^2}{< Ar, r >}$

End While

Definition 4.19

Let r be a vector in $\mathbb{R}^n$. We call the vector subspace of $\mathbb{R}^n$ spanned by the $k+1$ vectors $\{r, Ar, \cdots, A^k r\}$ the Krylov space associated with the vector r (and matrix A), denoted by $K_k(A, r)$ (or simply K_k).

The Krylov spaces $(K_k)_{k\geq 0}$ form by inclusion an increasing sequence of vector subspaces. Since $K_k \subset \mathbb{R}^n$, this sequence becomes stationary from a certain k.

Lemma 4.20

The sequence of Krylov spaces $(K_k)_{k\geq 0}$ is increasing,

$$K_k \subset K_{k+1}, \quad \forall k \geq 0.$$

Moreover, for all vectors $r_0 \neq 0$, there exists $k_0 \in \{0, 1, \cdots, n-1\}$ such that,

$$\begin{cases} \dim K_k = k+1 & \text{if } 0 \leq k \leq k_0, \\ \dim K_k = k_0 + 1 & \text{if } k \geq k_0. \end{cases}$$

The integer k_0 is called the Krylov critical dimension[a].

[a] p4-d7

Definition 4.19

Let r be a vector in $\mathbb{R}^n$. We call the vector subspace of $\mathbb{R}^n$ spanned by the $k+1$ vectors $\{r, Ar, \cdots, A^k r\}$ the Krylov space associated with the vector r (and matrix A), denoted by $K_k(A, r)$ (or simply K_k).

The Krylov spaces $(K_k)_{k\geq 0}$ form by inclusion an increasing sequence of vector subspaces. Since $K_k \subset \mathbb{R}^n$, this sequence becomes stationary from a certain k.

Lemma 4.20

The sequence of Krylov spaces $(K_k)_{k\geq 0}$ is increasing,

$$K_k \subset K_{k+1}, \quad \forall k \geq 0.$$

Moreover, for all vectors $r_0 \neq 0$, there exists $k_0 \in \{0, 1, \cdots, n-1\}$ such that,

$$\begin{cases} \dim K_k = k+1 & \text{if } 0 \leq k \leq k_0, \\ \dim K_k = k_0 + 1 & \text{if } k \geq k_0. \end{cases}$$

The integer k_0 is called the Krylov critical dimension[a].

[a] p4-d7

Definition 4.19

Let r be a vector in $\mathbb{R}^n$. We call the vector subspace of $\mathbb{R}^n$ spanned by the $k+1$ vectors $\{r, Ar, \cdots, A^k r\}$ the Krylov space associated with the vector r (and matrix A), denoted by $K_k(A, r)$ (or simply K_k).

The Krylov spaces $(K_k)_{k\geq 0}$ form by inclusion an increasing sequence of vector subspaces. Since $K_k \subset \mathbb{R}^n$, this sequence becomes stationary from a certain k.

Lemma 4.20

The sequence of Krylov spaces $(K_k)_{k\geq 0}$ is increasing,

$$K_k \subset K_{k+1}, \quad \forall k \geq 0.$$

Moreover, for all vectors $r_0 \neq 0$, there exists $k_0 \in \{0, 1, \cdots, n-1\}$ such that,

$$\begin{cases} \dim K_k = k+1 & \text{if } 0 \leq k \leq k_0, \\ \dim K_k = k_0 + 1 & \text{if } k \geq k_0. \end{cases}$$

The integer k_0 is called the Krylov critical dimension[a].

[a] p4-d7

Proposition 4.21

We consider the gradient method (with constant or variable step size),

$$x_0 \text{ given in } \mathbb{R}^n, \quad x_{k+1} = x_k + \alpha_k(b - Ax_k), \quad \forall k \geq 1.$$

The vector $r_k = b - Ax_k$, called the residual, satisfies the following properties:

1. r_k belongs to the Krylov space K_k corresponding to the initial residual r_0.
2. x_{k+1} belongs to the affine space $[x_0 + K_k]$ defined as the collection of vectors x such that $x - x_0$ belongs to the vector subspace K_k[a].

[a] p4-d8

Lemma 4.22

Let $(x_k)_{k\geq 0}$ be a sequence in $\mathbb{R}^n$. Let K_k be the Krylov space relative to the vector $r_0 = b - Ax_0$. If $x_{k+1} \in [x_0 + K_k]$, then $r_{k+1} = b - Ax_{k+1} \in K_{k+1}$[a].

[a] p4-d9

Proposition 4.21

We consider the gradient method (with constant or variable step size),

$$x_0 \text{ given in } \mathbb{R}^n, \quad x_{k+1} = x_k + \alpha_k(b - Ax_k), \quad \forall k \geq 1.$$

The vector $r_k = b - Ax_k$, called the residual, satisfies the following properties:

1. r_k belongs to the Krylov space K_k corresponding to the initial residual r_0.
2. x_{k+1} belongs to the affine space $[x_0 + K_k]$ defined as the collection of vectors x such that $x - x_0$ belongs to the vector subspace K_k[a].

[a] p4-d8

Lemma 4.22

Let $(x_k)_{k\geq 0}$ be a sequence in $\mathbb{R}^n$. Let K_k be the Krylov space relative to the vector $r_0 = b - Ax_0$. If $x_{k+1} \in [x_0 + K_k]$, then $r_{k+1} = b - Ax_{k+1} \in K_{k+1}$[a].

[a] p4-d9

We now assume that all the matrices considered in the sequel are symmetric positive definite.

To improve the gradient method, we forget, from now on, the induction relation that gives x_{k+1} in terms of x_k and we keep as the starting point only the relation $x_{k+1} \in [x_0 + K_k]$ where K_k is the Krylov space relative to the vector $r_0 = b - Ax_0$.

Of course, there exists an infinity of possible choices for x_{k+1} in the affine space $[x_0 + K_k]$.

To determine x_{k+1} in a unique fashion, we put forward two simple criteria:

1. 1st definition (orthogonalization principle).
   We choose $x_{k+1} \in [x_0 + K_k]$ such that $r_{k+1} \perp K_k$.
2. 2nd definition (minimization principle).
   We choose $x_{k+1} \in [x_0 + K_k]$ that minimizes in $[x_0 + K_k]$ the functional,

$$f(x) = \frac{1}{2} < Ax, x > - < b, x > .$$

We now assume that all the matrices considered in the sequel are symmetric positive definite.

To improve the gradient method, we forget, from now on, the induction relation that gives x_{k+1} in terms of x_k and we keep as the starting point only the relation $x_{k+1} \in [x_0 + K_k]$ where K_k is the Krylov space relative to the vector $r_0 = b - Ax_0$.

Of course, there exists an infinity of possible choices for x_{k+1} in the affine space $[x_0 + K_k]$.

To determine x_{k+1} in a unique fashion, we put forward two simple criteria:

1. 1st definition (orthogonalization principle).
   We choose $x_{k+1} \in [x_0 + K_k]$ such that $r_{k+1} \perp K_k$.
2. 2nd definition (minimization principle).
   We choose $x_{k+1} \in [x_0 + K_k]$ that minimizes in $[x_0 + K_k]$ the functional,

$$f(x) = \frac{1}{2} < Ax, x > - < b, x > .$$

Theorem 4.23

Let A be a symmetric positive definite matrix.
For the two previous definitions, there exists indeed a unique vector $x_{k+1} \in [x_0 + K_k]$.
Both definitions correspond to the same algorithm in the sense that they lead to the same value of x_{k+1}.
Furthermore, this algorithm converges to the solution of the linear system $Ax = b$ in at most n iterations.
We call this method, the Conjugate Gradient (CG) method[a].

[a] p4-d10

- The CG method that we have devised as an iterative method is in fact a direct method since it converges in a finite number of iterations (precisely $k_0 + 1$ where k_0 is the critical Krylov dimension)
- Introducing $r = b - Ax$, we have,

$$h(r) = \frac{1}{2} < A^{-1}r, r >= f(x) + \frac{1}{2} < A^{-1}b, b >,$$

and the second definition is equivalent to finding $x_{k+1} \in [x_0 + K_k]$ such that its residual $r_{k+1} = b - Ax_{k+1}$ minimizes the functional $h(r)$ in K_{k+1}.

Theorem 4.23

Let A be a symmetric positive definite matrix.
For the two previous definitions, there exists indeed a unique vector $x_{k+1} \in [x_0 + K_k]$.
Both definitions correspond to the same algorithm in the sense that they lead to the same value of x_{k+1}.
Furthermore, this algorithm converges to the solution of the linear system $Ax = b$ in at most n iterations.
We call this method, the Conjugate Gradient (CG) method[a].

[a] p4-d10

- The CG method that we have devised as an iterative method is in fact a direct method since it converges in a finite number of iterations (precisely $k_0 + 1$ where k_0 is the critical Krylov dimension)
- Introducing $r = b - Ax$, we have,

$$h(r) = \frac{1}{2} < A^{-1}r, r >= f(x) + \frac{1}{2} < A^{-1}b, b >,$$

and the second definition is equivalent to finding $x_{k+1} \in [x_0 + K_k]$ such that its residual $r_{k+1} = b - Ax_{k+1}$ minimizes the functional $h(r)$ in K_{k+1}.

Theorem 4.23

Let A be a symmetric positive definite matrix.
For the two previous definitions, there exists indeed a unique vector $x_{k+1} \in [x_0 + K_k]$.
Both definitions correspond to the same algorithm in the sense that they lead to the same value of x_{k+1}.
Furthermore, this algorithm converges to the solution of the linear system $Ax = b$ in at most n iterations.
We call this method, the Conjugate Gradient (CG) method[a].

[a] p4-d10

- The CG method that we have devised as an iterative method is in fact a direct method since it converges in a finite number of iterations (precisely $k_0 + 1$ where k_0 is the critical Krylov dimension)
- Introducing $r = b - Ax$, we have,

$$h(r) = \frac{1}{2} < A^{-1}r, r >= f(x) + \frac{1}{2} < A^{-1}b, b >,$$

and the second definition is equivalent to finding $x_{k+1} \in [x_0 + K_k]$ such that its residual $r_{k+1} = b - Ax_{k+1}$ minimizes the functional $h(r)$ in K_{k+1}.

Proposition 4.24

Let A be a symmetric positive definite matrix.
Let $(x_k)_{0\leq k\leq n}$ be the sequence of approximate solutions obtained by the CG method.
Set,

$$r_k = b - Ax_k \quad \text{and} \quad d_k = x_{k+1} - x_k.$$

Then,

1. the the Krylov space K_k defined by $K_k = \{r_0, Ar_0, \cdots, A^k r_0\}$ satisfies,
$$K_k = \mathbf{span}\{r_0, \cdots, r_k\} = \mathbf{span}\{d_0, \cdots, d_k\}.$$
2. the sequence $(r_k)_{0\leq k\leq n-1}$ is orthogonal i.e.,
$$< r_k, r_l >= 0 \quad \text{for all} \quad 0 \leq l < k \leq n-1.$$
3. the sequence $(d_k)_{0\leq k\leq n-1}$ is conjugate with respect to A (or A-conjugate) i.e.,
$$< Ad_k, d_l >= 0 \quad \text{for all} \quad 0 \leq l < k \leq n-1.$$

Lemma 4.25

Let $(a_i)_{1\leq i\leq p}$ be a family of linearly independent vectors of $\mathbb{R}^n$, and let $(b_i)_{1\leq i\leq p}$ and $(c_i)_{1\leq i\leq p}$ be two orthogonal families for the same scalar product on $\mathbb{R}^n$ such that for all $1 \leq i \leq p$,

$$\mathbf{span}\{a_1, \cdots, a_i\} = \mathbf{span}\{b_1, \cdots, b_i\} = \mathbf{span}\{c_1, \cdots, c_i\}.$$

Then each vector b_i is parallel to c_i for $1 \leq i \leq p$.

Theorem 4.26

Let A be a symmetric positive definite matrix.
Let (x_k) be the sequence of approximate solutions of the CG method.
Let $(r_k = b - Ax_k)$ be the associated residual sequence.
Then there exists an A-conjugate sequence (p_k) such that,

$$(*) \quad p_0 = r_0 = b - Ax_0 \quad \text{and for} \quad 0 \leq k \leq k_0, \begin{cases} x_{k+1} &= x_k + \alpha_k p_k, \\ r_{k+1} &= r_k - \alpha_k A p_k, \\ p_{k+1} &= r_{k+1} + \beta_k p_k, \end{cases}$$

with,

$$\alpha_k = \frac{\| r_k \|^2}{< Ap_k, p_k >} \quad \text{and} \quad \beta_k = \frac{\| r_{k+1} \|^2}{\| r_k \|^2}.$$

Conversely, let (x_k, r_k, p_k) be three sequences defined by the induction relations $(*)$.
Then (x_k) is nothing but the sequence of approximate solutions of the CG method[a].

[a] p4-r1

CG algorithm

Data: A, b. Output: $\mathbf{x}$ (approximation of the solution of $Ax = b$).

Initialization

Choose $\mathbf{x} \in \mathbb{R}^n$

Compute $r = b - A\mathbf{x}$

Set $p = r$

Compute $\gamma = \| r \|^2$

While $\gamma > \varepsilon$

$y = Ap$

$\alpha = \dfrac{\gamma}{< y, p >}$

$\mathbf{x} = \mathbf{x} + \alpha p$

$r = r - \alpha y$

$\beta = \dfrac{\| r \|^2}{\gamma}$

$\gamma = \| r \|^2$

$p = r + \beta p$

End While

Outline

Proposition 5.1

Let A be a symmetric real and positive definite matrix.
Let x be the exact solution to the system $Ax = b$.
Let $(x_k)_k$ be the sequence of approximate solutions produced by the CG method.
We have,

$$\| x_k - x \|_2 \leq 2\sqrt{\mathrm{cond}_2(A)} \left(\frac{\sqrt{\mathrm{cond}_2(A)} - 1}{\sqrt{\mathrm{cond}_2(A)} + 1} \right)^k \| x_0 - x \|_2 \ [a].$$

[a] p5-d1

The preconditioned CG method

Definition 5.2

Let $Ax = b$ be the linear system to be solved. We call a matrix C that is easy to invert and such that $\mathtt{cond}_2(C^{-1}A)$ is smaller than $\mathtt{cond}_2(A)$ a preconditioning of A.
We call the equivalent system $C^{-1}Ax = C^{-1}b$ a preconditioned system.

- The goal of preconditioning is that the CG method converges faster for the preconditioned system than for the original one
- The price to pay is the requirement of inverting C
- In practice, it is not necessary to form the matrix $C^{-1}A$ and we merely successively multiply matrices A and C^{-1} by vectors
- Note that the preconditioned matrix $C^{-1}A$ has to be symmetric positive definite

Symmetric preconditioning: if C is a symmetric positive definite matrix then let $C = BB^t$ be the Cholesky decomposition of C and $Ax = b$ is replaced by $\tilde{A}\tilde{x} = \tilde{b}$ where $\tilde{A} = B^{-1}AB^{-t}$, $\tilde{b} = B^{-1}b$ and $\tilde{x} = B^t x$.

Definition 5.2

Let $Ax = b$ be the linear system to be solved. We call a matrix C that is easy to invert and such that $\mathrm{cond}_2(C^{-1}A)$ is smaller than $\mathrm{cond}_2(A)$ a preconditioning of A.
We call the equivalent system $C^{-1}Ax = C^{-1}b$ a preconditioned system.

- The goal of preconditioning is that the CG method converges faster for the preconditioned system than for the original one
- The price to pay is the requirement of inverting C
- In practice, it is not necessary to form the matrix $C^{-1}A$ and we merely successively multiply matrices A and C^{-1} by vectors
- Note that the preconditioned matrix $C^{-1}A$ has to be symmetric positive definite

Symmetric preconditioning: if C is a symmetric positive definite matrix then let $C = BB^t$ be the Cholesky decomposition of C and $Ax = b$ is replaced by $\tilde{A}\tilde{x} = \tilde{b}$ where $\tilde{A} = B^{-1}AB^{-t}$, $\tilde{b} = B^{-1}b$ and $\tilde{x} = B^t x$.

Definition 5.2

Let $Ax = b$ be the linear system to be solved. We call a matrix C that is easy to invert and such that $\mathrm{cond}_2(C^{-1}A)$ is smaller than $\mathrm{cond}_2(A)$ a preconditioning of A.
We call the equivalent system $C^{-1}Ax = C^{-1}b$ a preconditioned system.

- The goal of preconditioning is that the CG method converges faster for the preconditioned system than for the original one
- The price to pay is the requirement of inverting C
- In practice, it is not necessary to form the matrix $C^{-1}A$ and we merely successively multiply matrices A and C^{-1} by vectors
- Note that the preconditioned matrix $C^{-1}A$ has to be symmetric positive definite

Symmetric preconditioning: if C is a symmetric positive definite matrix then let $C = BB^t$ be the Cholesky decomposition of C and $Ax = b$ is replaced by $\tilde{A}\tilde{x} = \tilde{b}$ where $\tilde{A} = B^{-1}AB^{-t}$, $\tilde{b} = B^{-1}b$ and $\tilde{x} = B^t x$.

PCG algorithm

Data: A, b. Output: $\mathbf{x}$ (approximation of the solution of $Ax = b$).

Initialization

Choose $\mathbf{x} \in \mathbb{R}^n$
Compute $r = b - A\mathbf{x}$
Compute $z = C^{-1}r$
Set $p = z$
Compute $\gamma = \| r \|^2$

While $\gamma > \varepsilon$

$y = Ap$

$$\delta = < z, r >; \quad \alpha = \frac{\delta}{< y, p >}$$

$\mathbf{x} = \mathbf{x} + \alpha p$
$r = r - \alpha y$
$z = C^{-1}r$

$$\beta = \frac{< z, r >}{\delta}; \quad \gamma = \| r \|^2$$

$p = z + \beta p$

End While

Incomplete LU factorization preconditioners

Consider a general sparse matrix A whose elements are $a_{i,j}$ with $i, j = 1, \cdots, n$.

A general ILU factorization process computes a sparse lower triangular matrix L and a sparse upper triangular matrix U so that the residual matrix $R = LU - A$ satisfies certain consraints.

A general algorithm for building an ILU factorization can be derived by performing Gaussian elimination and dropping some elements in predetermined nondiagonal positions.

There exist different variants of ILU factorization preconditioners.

Incomplete LU factorization preconditioners

Consider a general sparse matrix A whose elements are $a_{i,j}$ with $i, j = 1, \cdots, n$.

A general ILU factorization process computes a sparse lower triangular matrix L and a sparse upper triangular matrix U so that the residual matrix $R = LU - A$ satisfies certain consraints.

A general algorithm for building an ILU factorization can be derived by performing Gaussian elimination and dropping some elements in predetermined nondiagonal positions.

There exist different variants of ILU factorization preconditioners.

Incomplete LU factorization preconditioners

Let $\mathcal{P}$ be a zero pattern such that $\mathcal{P} \subset \{(i,j)|i \neq j; 1 \leq i,j \leq n\}$

General static pattern ILU algorithm

For $(i,j) \in \mathcal{P}$ set $a_{i,j} = 0$

For $i = 2 \nearrow n$
  For $k = 1 \nearrow i-1$ and if $(i,k) \notin \mathcal{P}$
    $a_{i,k} = \dfrac{a_{i,k}}{a_{k,k}}$

    For $j = k+1 \nearrow n$ and if $(i,j) \notin \mathcal{P}$
      $a_{i,j} = a_{i,j} - a_{i,k} a_{k,j}$
    End For
  End For
End For

Incomplete LU factorization preconditioners

Let $\mathcal{P}$ be a zero pattern such that $\mathcal{P} \subset \{(i,j)|i \neq j; 1 \leq i,j \leq n\}$

General static pattern ILU algorithm

For $(i,j) \in \mathcal{P}$ set $a_{i,j} = 0$

For $i = 2 \nearrow n$
  For $k = 1 \nearrow i-1$ and if $(i,k) \notin \mathcal{P}$
    $a_{i,k} = \dfrac{a_{i,k}}{a_{k,k}}$

    For $j = k+1 \nearrow n$ and if $(i,j) \notin \mathcal{P}$
      $a_{i,j} = a_{i,j} - a_{i,k} a_{k,j}$
    End For
  End For
End For

Theorem 5.3

Let A be an M-matrix and $\mathcal{P}$ a given zero pattern.
Then the general static pattern ILU algorithm does not break down and produces an incomplete factorization,

$$A = LU - R,$$

which is a regular splitting of A.

Proposition 5.4

The general static pattern ILU algorithm produces factors L and U such that,

$$A = LU - R,$$

in which $-R$ is the matrix of the elements that are dropped during the incomplete elimination process.
When $(i, j) \in \mathcal{P}$, an entry $r_{i,j}$ of R is equal to the value of $-a_{i,j}$ obtained at the completion of the k loop of the general static pattern ILU algorithm.
Otherwise, $r_{i,j}$ is zero.

Theorem 5.3

Let A be an M-matrix and $\mathcal{P}$ a given zero pattern.
Then the general static pattern ILU algorithm does not break down and produces an incomplete factorization,

$$A = LU - R,$$

which is a regular splitting of A.

Proposition 5.4

The general static pattern ILU algorithm produces factors L and U such that,

$$A = LU - R,$$

in which $-R$ is the matrix of the elements that are dropped during the incomplete elimination process.
When $(i, j) \in \mathcal{P}$, an entry $r_{i,j}$ of R is equal to the value of $-a_{i,j}$ obtained at the completion of the k loop of the general static pattern ILU algorithm.
Otherwise, $r_{i,j}$ is zero.

The ILU factorization technique with no fill-in, denoted by ILU(0), takes the zero pattern $\mathcal{P}$ to be precisely the zero pattern of A,

$$\text{NZ}(A) \text{ is the set of pairs } (i,j), \quad 1 \leq i,j \leq n \text{ such that } a_{i,j} \neq 0.$$

ILU(0) algorithm

For $i = 2 \nearrow n$
  For $k = 1 \nearrow i-1$ and if $(i,k) \in \text{NZ}(A)$
    $a_{i,k} = \dfrac{a_{i,k}}{a_{k,k}}$
    For $j = k+1 \nearrow n$ and if $(i,j) \in \text{NZ}(A)$
      $a_{i,j} = a_{i,j} - a_{i,k} a_{k,j}$
    End For
  End For
End For

The ILU factorization technique with no fill-in, denoted by ILU(0), takes the zero pattern $\mathcal{P}$ to be precisely the zero pattern of A,

$$\text{NZ}(A) \text{ is the set of pairs } (i,j), \quad 1 \leq i,j \leq n \text{ such that } a_{i,j} \neq 0.$$

ILU(0) algorithm

For $i = 2 \nearrow n$
  For $k = 1 \nearrow i-1$ and if $(i,k) \in \text{NZ}(A)$
    $$a_{i,k} = \frac{a_{i,k}}{a_{k,k}}$$
    For $j = k+1 \nearrow n$ and if $(i,j) \in \text{NZ}(A)$
      $$a_{i,j} = a_{i,j} - a_{i,k} a_{k,j}$$
    End For
  End For
End For

The accuracy of the ILU(0) incomplete factorization may be insufficient to yield an adequate rate of convergence.

More accurate ILU factorizations that differ from ILU(0) by allowing some fill-in, are often more efficient as well as more reliable.

In the ILU(p) method, a level of fill is attributed to each element that is processed by Gaussian elimination and dropping is based on the value of the level of fill.

The rationale is that the level of fill should be indicative of the size: the higher the level, the smaller the magnitude of the elements.

A size ϵ^k is attributed to any element whose level of fill is k, where $\epsilon < 1$.

An element $a_{i,j}$ is updated using $a_{i,j} = a_{i,j} - a_{i,k}a_{k,j}$ and, if $\texttt{lev}_{i,j}$ is the current level of fill of this element, then the size of the updated element should be,

$$\epsilon^{\texttt{lev}_{i,j}} - \epsilon^{\texttt{lev}_{i,k}}\epsilon^{\texttt{lev}_{k,j}} = \epsilon^{\texttt{lev}_{i,j}} - \epsilon^{\texttt{lev}_{i,k}+\texttt{lev}_{k,j}}.$$

The accuracy of the ILU(0) incomplete factorization may be insufficient to yield an adequate rate of convergence.

More accurate ILU factorizations that differ from ILU(0) by allowing some fill-in, are often more efficient as well as more reliable.

In the ILU(p) method, a level of fill is attributed to each element that is processed by Gaussian elimination and dropping is based on the value of the level of fill.

The rationale is that the level of fill should be indicative of the size: the higher the level, the smaller the magnitude of the elements.

A size ϵ^k is attributed to any element whose level of fill is k, where $\epsilon < 1$.

An element $a_{i,j}$ is updated using $a_{i,j} = a_{i,j} - a_{i,k}a_{k,j}$ and, if $\texttt{lev}_{i,j}$ is the current level of fill of this element, then the size of the updated element should be,

$$\epsilon^{\texttt{lev}_{i,j}} - \epsilon^{\texttt{lev}_{i,k}}\epsilon^{\texttt{lev}_{k,j}} = \epsilon^{\texttt{lev}_{i,j}} - \epsilon^{\texttt{lev}_{i,k}+\texttt{lev}_{k,j}}.$$

The accuracy of the ILU(0) incomplete factorization may be insufficient to yield an adequate rate of convergence.

More accurate ILU factorizations that differ from ILU(0) by allowing some fill-in, are often more efficient as well as more reliable.

In the ILU(p) method, a level of fill is attributed to each element that is processed by Gaussian elimination and dropping is based on the value of the level of fill.

The rationale is that the level of fill should be indicative of the size: the higher the level, the smaller the magnitude of the elements.

A size ϵ^k is attributed to any element whose level of fill is k, where $\epsilon < 1$.

An element $a_{i,j}$ is updated using $a_{i,j} = a_{i,j} - a_{i,k}a_{k,j}$ and, if $\texttt{lev}_{i,j}$ is the current level of fill of this element, then the size of the updated element should be,

$$\epsilon^{\texttt{lev}_{i,j}} - \epsilon^{\texttt{lev}_{i,k}}\epsilon^{\texttt{lev}_{k,j}} = \epsilon^{\texttt{lev}_{i,j}} - \epsilon^{\texttt{lev}_{i,k}+\texttt{lev}_{k,j}}.$$

Therefore, roughly speaking, the size of $a_{i,j}$ will be the maximum of the two sizes $\epsilon^{\mathtt{lev}_{i,j}}$ and $\epsilon^{\mathtt{lev}_{i,k}+\mathtt{lev}_{k,j}}$, so it is natiral to define the new level of fill as $\min(\mathtt{lev}_{i,j}\ ,\ \mathtt{lev}_{i,k}+\mathtt{lev}_{k,j})$.

In practice, the level of fill is actually shifted by -1 for convenience of notation and to conform with the definition used for ILU(0) (i.e. $\mathtt{lev}_{i,j}=0$ if $a_{i,j}\neq 0$ in the matrix A).

Definition 5.5

The initial level of fill of an element $a_{i,j}$ of a sparse matrix A is defined by,

$$\mathtt{lev}_{i,j}=\begin{cases}0 & \text{if } a_{i,j}\neq 0 \text{ or } i=j,\\ \infty & \text{otherwise.}\end{cases}$$

Each time this element is modified using,

$$a_{i,j}=a_{i,j}-a_{i,k}a_{k,j},$$

its level of fill must be updated by,

$$\mathtt{lev}_{i,j}:=\min(\mathtt{lev}_{i,j}\ ,\ \mathtt{lev}_{i,k}+\mathtt{lev}_{k,j}+1).$$

Therefore, roughly speaking, the size of $a_{i,j}$ will be the maximum of the two sizes $\epsilon^{\mathtt{lev}_{i,j}}$ and $\epsilon^{\mathtt{lev}_{i,k}+\mathtt{lev}_{k,j}}$, so it is natiral to define the new level of fill as $\min(\mathtt{lev}_{i,j}\ ,\ \mathtt{lev}_{i,k}+\mathtt{lev}_{k,j})$.

In practice, the level of fill is actually shifted by -1 for convenience of notation and to conform with the definition used for ILU(0) (i.e. $\mathtt{lev}_{i,j}=0$ if $a_{i,j}\neq 0$ in the matrix A).

Definition 5.5

The initial level of fill of an element $a_{i,j}$ of a sparse matrix A is defined by,

$$\mathtt{lev}_{i,j}=\begin{cases}0 & \text{if } a_{i,j}\neq 0 \text{ or } i=j,\\ \infty & \text{otherwise.}\end{cases}$$

Each time this element is modified using,

$$a_{i,j}=a_{i,j}-a_{i,k}a_{k,j},$$

its level of fill must be updated by,

$$\mathtt{lev}_{i,j}:=\min(\mathtt{lev}_{i,j}\ ,\ \mathtt{lev}_{i,k}+\mathtt{lev}_{k,j}+1).$$

Therefore, roughly speaking, the size of $a_{i,j}$ will be the maximum of the two sizes $\epsilon^{\mathtt{lev}_{i,j}}$ and $\epsilon^{\mathtt{lev}_{i,k}+\mathtt{lev}_{k,j}}$, so it is natiral to define the new level of fill as $\min(\mathtt{lev}_{i,j}\ ,\ \mathtt{lev}_{i,k}+\mathtt{lev}_{k,j})$.

In practice, the level of fill is actually shifted by -1 for convenience of notation and to conform with the definition used for ILU(0) (i.e. $\mathtt{lev}_{i,j}=0$ if $a_{i,j}\neq 0$ in the matrix A).

Definition 5.5

The initial level of fill of an element $a_{i,j}$ of a sparse matrix A is defined by,

$$\mathtt{lev}_{i,j}=\begin{cases}0 & \text{if } a_{i,j}\neq 0 \text{ or } i=j,\\ \infty & \text{otherwise.}\end{cases}$$

Each time this element is modified using,

$$a_{i,j}=a_{i,j}-a_{i,k}a_{k,j},$$

its level of fill must be updated by,

$$\mathtt{lev}_{i,j}:=\min(\mathtt{lev}_{i,j}\ ,\ \mathtt{lev}_{i,k}+\mathtt{lev}_{k,j}+1).$$

In the ILU(p) incomplete factorization method, all fill-in elements whose level of fill does not exceed p are kept.

The zero pattern for ILU(p) is the set,

$$\mathcal{P}_p = \{(i,j) | \texttt{lev}_{i,j} > p\},$$

where $\texttt{lev}_{i,j}$ is the level of fill value after all updates,

$$\texttt{lev}_{i,j} := \min(\texttt{lev}_{i,j} \ , \ \texttt{lev}_{i,k} + \texttt{lev}_{k,j} + 1) \quad (*).$$

Let $a_{i,\star}$ denotes the ith row of the matrix A.

In the ILU(p) incomplete factorization method, all fill-in elements whose level of fill does not exceed p are kept.

The zero pattern for ILU(p) is the set,

$$\mathcal{P}_p = \{(i,j) | \texttt{lev}_{i,j} > p\},$$

where $\texttt{lev}_{i,j}$ is the level of fill value after all updates,

$$\texttt{lev}_{i,j} := \min(\texttt{lev}_{i,j} \ , \ \texttt{lev}_{i,k} + \texttt{lev}_{k,j} + 1) \quad (*).$$

Let $a_{i,\star}$ denotes the ith row of the matrix A.

ILU(p) algorithm

For all nonzero elements $a_{i,j}$ define $\texttt{lev}_{i,j} = 0$

For $i = 2 \nearrow n$

For $k = 1 \nearrow i-1$ and for $\texttt{lev}_{i,j} \leq p$

$$a_{i,k} = \frac{a_{i,k}}{a_{k,k}}$$

$$a_{i,\star} = a_{i,\star} - a_{i,k} a_{k,\star}$$

Update the levels of fill of the nonzero $a_{i,j}$'s using $(*)$

End For

Replace anay element in row i with $\texttt{lev}_{i,j} > p$ with zero

End For

Incomplete factorization methods that rely on the levels of fill are blind to numerical values because elements that are dropped depend only on the structure of A.

A few alternative methods are available based on dropping elements in the Gaussian elimination process according to their magnitude rather than their locations.

A generic ILU algorithm with threshold (ILUT) can be derived by taking into account a set of rules for dropping small magnitude elements.

In the following, *applying a dropping rule to an element* will only mean *replacing the element with zero if it satisfies a set of crieria*.

A dropping rule can be applied to a whole row by applying the same rule to all the elements of the row.

In the following algorithm, w is a full-length working row used to accumulate linear combinations of sparse rows in the elimination and w_k is the kth entry of this row.

Incomplete factorization methods that rely on the levels of fill are blind to numerical values because elements that are dropped depend only on the structure of A.

A few alternative methods are available based on dropping elements in the Gaussian elimination process according to their magnitude rather than their locations.

A generic ILU algorithm with threshold (ILUT) can be derived by taking into account a set of rules for dropping small magnitude elements.

In the following, *applying a dropping rule to an element* will only mean *replacing the element with zero if it satisfies a set of crieria*.

A dropping rule can be applied to a whole row by applying the same rule to all the elements of the row.

In the following algorithm, w is a full-length working row used to accumulate linear combinations of sparse rows in the elimination and w_k is the kth entry of this row.

Incomplete factorization methods that rely on the levels of fill are blind to numerical values because elements that are dropped depend only on the structure of A.

A few alternative methods are available based on dropping elements in the Gaussian elimination process according to their magnitude rather than their locations.

A generic ILU algorithm with threshold (ILUT) can be derived by taking into account a set of rules for dropping small magnitude elements.

In the following, *applying a dropping rule to an element* will only mean *replacing the element with zero if it satisfies a set of crieria*.

A dropping rule can be applied to a whole row by applying the same rule to all the elements of the row.

In the following algorithm, w is a full-length working row used to accumulate linear combinations of sparse rows in the elimination and w_k is the kth entry of this row.

ILUT algorithm

For $i = 1 \nearrow n$
  $w = a_{i,\star}$
  For $k = 1 \nearrow i-1$ and when $w_k \neq 0$
    $w_k = \dfrac{w_k}{a_{k,k}}$
    Apply a dropping rule to w_k (1)

    If $w_k \neq 0$ **Then**
      $w = w - w_k * u_{k,\star}$
    Endif
  End For

  Apply a dropping rule to row w (2)
  For $j = 1, \cdots, i-1 : l_{i,j} = w_j$
  For $j = 1, \cdots, n : u_{i,j} = w_j$
  $w = 0$
End For

ILU(0) can be viewed as a particular case of the ILUT algorithm where the dropping rule is to drop elements that are in positions not belonging to the original structure of the matrix A.

In the ILUT(p,τ) factorization method, the following rules are used:

1. for (1), an element w_k is dropped (i.e. replaced with zero) if it is less than the relative tolerance τ_i obtained by multiplying τ by the original norm of the ith row.
2. for (2), a dropping rule of a different type is applied. First, drop again any element in the row with a magnitude that is below the relative tolerance τ_i. Then, keep only the p largest elements in the L part of the row and the p largest elements in the U part of the row in addition to the diagonal element, which is always kept.

The goal of the second dropping step is to control the number of elements per row.

Roughly speaking, p can be viewed as a parameter that helps control memory usage, while τ helps reduce the computational cost.

ILU(0) can be viewed as a particular case of the ILUT algorithm where the dropping rule is to drop elements that are in positions not belonging to the original structure of the matrix A.

In the ILUT(p,τ) factorization method, the following rules are used:

1. for (1), an element w_k is dropped (i.e. replaced with zero) if it is less than the relative tolerance τ_i obtained by multiplying τ by the original norm of the ith row.
2. for (2), a dropping rule of a different type is applied. First, drop again any element in the row with a magnitude that is below the relative tolerance τ_i. Then, keep only the p largest elements in the L part of the row and the p largest elements in the U part of the row in addition to the diagonal element, which is always kept.

The goal of the second dropping step is to control the number of elements per row.

Roughly speaking, p can be viewed as a parameter that helps control memory usage, while τ helps reduce the computational cost.

ILU(0) can be viewed as a particular case of the ILUT algorithm where the dropping rule is to drop elements that are in positions not belonging to the original structure of the matrix A.

In the ILUT(p,τ) factorization method, the following rules are used:

1. for (1), an element w_k is dropped (i.e. replaced with zero) if it is less than the relative tolerance τ_i obtained by multiplying τ by the original norm of the ith row.
2. for (2), a dropping rule of a different type is applied. First, drop again any element in the row with a magnitude that is below the relative tolerance τ_i. Then, keep only the p largest elements in the L part of the row and the p largest elements in the U part of the row in addition to the diagonal element, which is always kept.

The goal of the second dropping step is to control the number of elements per row.

Roughly speaking, p can be viewed as a parameter that helps control memory usage, while τ helps reduce the computational cost.

ILU(0) can be viewed as a particular case of the ILUT algorithm where the dropping rule is to drop elements that are in positions not belonging to the original structure of the matrix A.

In the ILUT(p,τ) factorization method, the following rules are used:

1. for (1), an element w_k is dropped (i.e. replaced with zero) if it is less than the relative tolerance τ_i obtained by multiplying τ by the original norm of the ith row.
2. for (2), a dropping rule of a different type is applied. First, drop again any element in the row with a magnitude that is below the relative tolerance τ_i. Then, keep only the p largest elements in the L part of the row and the p largest elements in the U part of the row in addition to the diagonal element, which is always kept.

The goal of the second dropping step is to control the number of elements per row.

Roughly speaking, p can be viewed as a parameter that helps control memory usage, while τ helps reduce the computational cost.

A variant for the second dropping step is to keep a number of elements equal to $\texttt{nl}(i) + p$ in the lower part and $\texttt{nu}(i) + p$ in the upper part of the row, where $\texttt{nl}(i)$ and $\texttt{nu}(i)$ are the number of the nonzero elements in the L part and the U part of the ith row of A.

The ILUT approach may fail for many of the matrices that arise from real applications for one of the following reasons:

- the ILUT procedure encounters a zero pivot,
- the ILUT procedure encounters an overflow or underflow condition because of exponential growth of the entries of the factors,
- the ILUT procedure terminates normally but the incomplete factorization preconditioner that is computed is *unstable*.

An unstable ILU factorization is one for which $M^{-1} = U^{-1}L^{-1}$ has a very large norm leading to poor convergence or divergence of the outer iteration.

A variant for the second dropping step is to keep a number of elements equal to $\texttt{nl}(i) + p$ in the lower part and $\texttt{nu}(i) + p$ in the upper part of the row, where $\texttt{nl}(i)$ and $\texttt{nu}(i)$ are the number of the nonzero elements in the L part and the U part of the ith row of A.

The ILUT approach may fail for many of the matrices that arise from real applications for one of the following reasons:

- the ILUT procedure encounters a zero pivot,
- the ILUT procedure encounters an overflow or underflow condition because of exponential growth of the entries of the factors,
- the ILUT procedure terminates normally but the incomplete factorization preconditioner that is computed is *unstable*.

An unstable ILU factorization is one for which $M^{-1} = U^{-1}L^{-1}$ has a very large norm leading to poor convergence or divergence of the outer iteration.

A variant for the second dropping step is to keep a number of elements equal to $\texttt{nl}(i) + p$ in the lower part and $\texttt{nu}(i) + p$ in the upper part of the row, where $\texttt{nl}(i)$ and $\texttt{nu}(i)$ are the number of the nonzero elements in the L part and the U part of the ith row of A.

The ILUT approach may fail for many of the matrices that arise from real applications for one of the following reasons:

- the ILUT procedure encounters a zero pivot,
- the ILUT procedure encounters an overflow or underflow condition because of exponential growth of the entries of the factors,
- the ILUT procedure terminates normally but the incomplete factorization preconditioner that is computed is *unstable*.

An unstable ILU factorization is one for which $M^{-1} = U^{-1}L^{-1}$ has a very large norm leading to poor convergence or divergence of the outer iteration.

ILUTP : ILUT with pivoting

ILUTP is based on column pivoting.

ILUTP uses a permutation array to hold the new orderings of the variables, along with the reverse permutation array.

At step i of the elimination process the largest entry in a row is selected and is defined to be the new ith variable. The two permutation arrays are then updated accordingly.

The matrix elements of L and U are kept in their original numbering, however when expanding the LU row that corresponds to the ith outer step of Gaussian elimination, the elements are loaded with respect to the new labeling using the permutation array.

At the end of the process, ther are two options:

- leave all elements labeled with respect to the original labeling but the variables (system unknowns) must then be permuted at each preconditioning step;
- apply the permutation to all elements of A as well as LU but this produces a permuted solution that must be permuted back at the end of the iteration phase.

ILUTP : ILUT with pivoting

ILUTP is based on column pivoting.

ILUTP uses a permutation array to hold the new orderings of the variables, along with the reverse permutation array.

At step i of the elimination process the largest entry in a row is selected and is defined to be the new ith variable. The two permutation arrays are then updated accordingly.

The matrix elements of L and U are kept in their original numbering, however when expanding the LU row that corresponds to the ith outer step of Gaussian elimination, the elements are loaded with respect to the new labeling using the permutation array.

At the end of the process, ther are two options:

- leave all elements labeled with respect to the original labeling but the variables (system unknowns) must then be permuted at each preconditioning step;
- apply the permutation to all elements of A as well as LU but this produces a permuted solution that must be permuted back at the end of the iteration phase.

ILUTP : ILUT with pivoting

ILUTP is based on column pivoting.

ILUTP uses a permutation array to hold the new orderings of the variables, along with the reverse permutation array.

At step i of the elimination process the largest entry in a row is selected and is defined to be the new ith variable. The two permutation arrays are then updated accordingly.

The matrix elements of L and U are kept in their original numbering, however when expanding the LU row that corresponds to the ith outer step of Gaussian elimination, the elements are loaded with respect to the new labeling using the permutation array.

At the end of the process, ther are two options:

- leave all elements labeled with respect to the original labeling but the variables (system unknowns) must then be permuted at each preconditioning step;
- apply the permutation to all elements of A as well as LU but this produces a permuted solution that must be permuted back at the end of the iteration phase.

Approximate inverse preconditioners

The ILU factorization techniques were developed originally for M-matrices arising from the discretization of PDEs of elliptic type, usually in one variable.

For the common situation where A is indefinite, standard ILU factorizations may face several difficulties, of which the best known is the fatal breakdown due to the encounter of a zero pivot.

However, there are other problems that are just as serious.

Consider an incomplete factorization of the form,

$$A = LU + E,$$

where E is the error.

The preconditioned matrices associated with the different forms of ILU preconditioning are similar to,

$$L^{-1}AU^{-1} = \mathrm{Id} + L^{-1}EU^{-1}.$$

Approximate inverse preconditioners

The ILU factorization techniques were developed originally for M-matrices arising from the discretization of PDEs of elliptic type, usually in one variable.

For the common situation where A is indefinite, standard ILU factorizations may face several difficulties, of which the best known is the fatal breakdown due to the encounter of a zero pivot.

However, there are other problems that are just as serious.

Consider an incomplete factorization of the form,

$$A = LU + E,$$

where E is the error.

The preconditioned matrices associated with the different forms of ILU preconditioning are similar to,

$$L^{-1}AU^{-1} = \mathrm{Id} + L^{-1}EU^{-1}.$$

Approximate inverse preconditioners

What is sometimes missed is the fact that the error matrix E is not as important as the *preconditioned* error matrix $L^{-1}EU^{-1}$. Indeed, the original matrix A may lead to L^{-1} or U^{-1} inverse factors with very large norm, causing $L^{-1}EU^{-1}$ to very large.

It can be observed experimentally that ILU preconditioners can be very poor in these situations, which often arise when the matrices are indefinite or have large nonsymmetric parts.

One possible remedy is to try to find a preconditioner that does not require solving a linear system.

For example, the original system can be preconditioned by a marix M that is a direct approximation of the inverse of A.

Approximate inverse preconditioners

What is sometimes missed is the fact that the error matrix E is not as important as the *preconditioned* error matrix $L^{-1}EU^{-1}$. Indeed, the original matrix A may lead to L^{-1} or U^{-1} inverse factors with very large norm, causing $L^{-1}EU^{-1}$ to very large.

It can be observed experimentally that ILU preconditioners can be very poor in these situations, which often arise when the matrices are indefinite or have large nonsymmetric parts.

One possible remedy is to try to find a preconditioner that does not require solving a linear system.

For example, the original system can be preconditioned by a marix M that is a direct approximation of the inverse of A.

Approximate inverse preconditioners

A simple technique for finding approximate inverses of arbitrary sparse matrices is to attempt to find a sparse matrix M that minimizes the Frobenius norm of the residual matrix $\mathrm{Id} - AM$,

$$F(M) = \| \mathrm{Id} - AM \|_F^2 .$$

A matrix M whose value $F(M)$ is small would be a right approximate inverse of A (similarly a left approximate inverse can be defined by considering $\mathrm{Id} - MA$).

The objective function decouples into the sum of the squares of the 2-norms of the individual columns of the residual matrix,

$$F(M) = \sum_{j=1}^{n} \| e_j - Am_j \|_2^2,$$

in which e_j and m_j are the jth columns of the identity matrix and the matrix M respectively.

Approximate inverse preconditioners

A simple technique for finding approximate inverses of arbitrary sparse matrices is to attempt to find a sparse matrix M that minimizes the Frobenius norm of the residual matrix $\mathrm{Id} - AM$,

$$F(M) = \| \mathrm{Id} - AM \|_F^2 .$$

A matrix M whose value $F(M)$ is small would be a right approximate inverse of A (similarly a left approximate inverse can be defined by considering $\mathrm{Id} - MA$).

The objective function decouples into the sum of the squares of the 2-norms of the individual columns of the residual matrix,

$$F(M) = \sum_{j=1}^{n} \| e_j - Am_j \|_2^2,$$

in which e_j and m_j are the jth columns of the identity matrix and the matrix M respectively.

Approximate inverse preconditioners

There are two different ways to proceed in order to minimize $F(M)$: the function $F(M)$ can be minimized globally as a function of the sparse matrix M, or the individual functions,

$$f_j(m) = \| e_j - Am_j \|_2^2,$$

can be minimized.

The second approach is appealing for parallel computers, although there is also parallelism to be exploited in the first approach.

In the second approach, each minimization can be performed by taking a sparse initial guess and solving approximately the n parallel linear subproblems,

$$Am_j = e_j, \quad j = 1, \cdots, n,$$

with a few steps of a nonsymmetric descent-type method (such as MR or GMRES).

Approximate inverse preconditioners

There are two different ways to proceed in order to minimize $F(M)$: the function $F(M)$ can be minimized globally as a function of the sparse matrix M, or the individual functions,

$$f_j(m) = \| e_j - Am_j \|_2^2,$$

can be minimized.

The second approach is appealing for parallel computers, although there is also parallelism to be exploited in the first approach.

In the second approach, each minimization can be performed by taking a sparse initial guess and solving approximately the n parallel linear subproblems,

$$Am_j = e_j, \quad j = 1, \cdots, n,$$

with a few steps of a nonsymmetric descent-type method (such as MR or GMRES).

Approximate inverse (AINV) algorithm via MR iteration

Set $M = M_0$
For each column $j = 1 \nearrow n$
  Define $m_j = Me_j$
  For $i = 1 \nearrow n_i$
    $r_j = e_j - Am_j$

$$\alpha_j = \frac{< r_j, Ar_j >}{< Ar_j, Ar_j >}$$

    $m_j = m_j + \alpha_j r_j$
    Apply numerical dropping to m_j
  End For
End For

- n_i is the number of iterations to solve $Am_j = e_j$ approximately for each column
- M_0 is a initial guess

Approximate inverse (AINV) algorithm via MR iteration

Set $M = M_0$
For each column $j = 1 \nearrow n$
  Define $m_j = Me_j$
  For $i = 1 \nearrow n_i$
    $r_j = e_j - Am_j$

$$\alpha_j = \frac{< r_j, Ar_j >}{< Ar_j, Ar_j >}$$

    $m_j = m_j + \alpha_j r_j$
    Apply numerical dropping to m_j
  End For
End For

- n_i is the number of iterations to solve $Am_j = e_j$ approximately for each column
- M_0 is a initial guess

Proposition 5.6

Assume that A is nonsingular and that the residual of the AINV M satisfies the relation,

$$\| \mathrm{Id} - AM \| < 1,$$

where $\| . \|$ is any consistent matrix norm[a]. Then M is nonsingular.

[a] $\| AB \| \leq \| A \| \| B \|$.

The result follows from the equality,

$$AM = \mathrm{Id} - (\mathrm{Id} - AM) = \mathrm{Id} - N.$$

Since $\| N \| < 1$, then $\mathrm{Id} - N$ is nonsingular.

Proposition 5.6

Assume that A is nonsingular and that the residual of the AINV M satisfies the relation,

$$\| \mathrm{Id} - AM \| < 1,$$

where $\| . \|$ is any consistent matrix norm[a]. Then M is nonsingular.

[a] $\| AB \| \leq \| A \| \| B \|$.

The result follows from the equality,

$$AM = \mathrm{Id} - (\mathrm{Id} - AM) = \mathrm{Id} - N.$$

Since $\| N \| < 1$, then $\mathrm{Id} - N$ is nonsingular.

Outline

Domain decomposition methods

Generalities

Domain decomposition (DD) is a very natural framework in which to develop solution methods for parallel computers.

Although the idea is quite old and it has been used for many years, mainly in structural mechanics (substructuring methods), the interest in DD was renewed from the end of the 80's, especially with the advent of distributed memory parallel computers.

DD principles can be used for designing,

- iterative solvers for linear systems of equations,
- algebraic preconditioners for Krylov iterative methods,
- coupling strategies for multi-model problems.

DD methods are generally used for the solution of linear systems arising from the discretization of a PDE and are closely related to a partitioning of the domain on which the PDE is to be solved.

DD methods can be studied at the continuous or discrete levels.

Domain decomposition methods

Generalities

Domain decomposition (DD) is a very natural framework in which to develop solution methods for parallel computers.

Although the idea is quite old and it has been used for many years, mainly in structural mechanics (substructuring methods), the interest in DD was renewed from the end of the 80's, especially with the advent of distributed memory parallel computers.

DD principles can be used for designing,

- iterative solvers for linear systems of equations,
- algebraic preconditioners for Krylov iterative methods,
- coupling strategies for multi-model problems.

DD methods are generally used for the solution of linear systems arising from the discretization of a PDE and are closely related to a partitioning of the domain on which the PDE is to be solved.

DD methods can be studied at the continuous or discrete levels.

Generalities

Domain decomposition (DD) is a very natural framework in which to develop solution methods for parallel computers.

Although the idea is quite old and it has been used for many years, mainly in structural mechanics (substructuring methods), the interest in DD was renewed from the end of the 80's, especially with the advent of distributed memory parallel computers.

DD principles can be used for designing,

- iterative solvers for linear systems of equations,
- algebraic preconditioners for Krylov iterative methods,
- coupling strategies for multi-model problems.

DD methods are generally used for the solution of linear systems arising from the discretization of a PDE and are closely related to a partitioning of the domain on which the PDE is to be solved.

DD methods can be studied at the continuous or discrete levels.

Generalities

For constructing algorithms for parallel computers, a good principle is to divide the problem into smaller pieces, solve the subproblems in parallel and then paste the local results together (divide and conquer strategy).

DD methods proceed in a similar way,

1. the domain Ω (or preferably the problem) is split into subdomains (or subproblems),
2. a problem is defined and solved on each subdomain in parallel,
3. the partial solutions are glued together to get the global solution.

DD methods are generally separated into two main categories: overlapping and non-overlapping methods.

Generalities

For constructing algorithms for parallel computers, a good principle is to divide the problem into smaller pieces, solve the subproblems in parallel and then paste the local results together (divide and conquer strategy).

DD methods proceed in a similar way,

1. the domain Ω (or preferably the problem) is split into subdomains (or subproblems),
2. a problem is defined and solved on each subdomain in parallel,
3. the partial solutions are glued together to get the global solution.

DD methods are generally separated into two main categories: overlapping and non-overlapping methods.

Domain decomposition methods

Generalities

For constructing algorithms for parallel computers, a good principle is to divide the problem into smaller pieces, solve the subproblems in parallel and then paste the local results together (divide and conquer strategy).

DD methods proceed in a similar way,

1. the domain Ω (or preferably the problem) is split into subdomains (or subproblems),
2. a problem is defined and solved on each subdomain in parallel,
3. the partial solutions are glued together to get the global solution.

DD methods are generally separated into two main categories: overlapping and non-overlapping methods.

Generalities

When designig a DD method for solving a linear system, a second source of distinction comes with the strategy used for solving the subproblems: on can use either a direct method (Gaussian elimination) or a (preconditioned) iterative method.

The ultimate goal is to develop a solution algorithm whose complexity is proportional to the number of unknowns (numerical efficiency) and whose performance weakly depends on the number of subdomains (scalability).

In the following, we will consider linear systems resulting from the discretization of second order elliptic PDEs.

Generalities

When designig a DD method for solving a linear system, a second source of distinction comes with the strategy used for solving the subproblems: on can use either a direct method (Gaussian elimination) or a (preconditioned) iterative method.

The ultimate goal is to develop a solution algorithm whose complexity is proportional to the number of unknowns (numerical efficiency) and whose performance weakly depends on the number of subdomains (scalability).

In the following, we will consider linear systems resulting from the discretization of second order elliptic PDEs.

Generalities

When designig a DD method for solving a linear system, a second source of distinction comes with the strategy used for solving the subproblems: on can use either a direct method (Gaussian elimination) or a (preconditioned) iterative method.

The ultimate goal is to develop a solution algorithm whose complexity is proportional to the number of unknowns (numerical efficiency) and whose performance weakly depends on the number of subdomains (scalability).

In the following, we will consider linear systems resulting from the discretization of second order elliptic PDEs.

The classical Schwarz alternating method

The domain Ω is split into two overlapping subdomains Ω_1 and Ω_2.

Let Γ_i for $i = 1, 2$ be the part of the boundary Ω_i enclosed in Ω.

Roughly speaking, the classical Schwarz alternating algorithm amounts to:

1. guess a value for the unknowns on the inner boundary Γ_1,
2. solve exactly the problem in Ω_1,
3. use the computed values on the inner boundary Γ_2 to solve exactly the problem in Ω_2,
4. repeat the process until convergence.

The classical Schwarz alternating method

The domain Ω is split into two overlapping subdomains Ω_1 and Ω_2.

Let Γ_i for $i = 1, 2$ be the part of the boundary Ω_i enclosed in Ω.

Roughly speaking, the classical Schwarz alternating algorithm amounts to:

1. guess a value for the unknowns on the inner boundary Γ_1,
2. solve exactly the problem in Ω_1,
3. use the computed values on the inner boundary Γ_2 to solve exactly the problem in Ω_2,
4. repeat the process until convergence.

The classical Schwarz alternating method

We consider the Poisson model problem,

$$-\Delta u = f \quad \text{in} \quad \Omega, \quad u|_{\partial\Omega} = 0.$$

The Schwarz alternating algorithm can be formulated at the PDE level, assuming $u^{(1)}$ is given,

1. solve $-\Delta u^{(2m)} = f$ in Ω_1, with $u^{(2m)}|_{\Gamma_1} = u^{(2m-1)}|_{\Gamma_1}$ and $u^{(2m)}|_{\partial\Omega_1 \cap \partial\Omega} = 0$.
2. solve $-\Delta u^{(2m+1)} = f$ in Ω_2, with $u^{(2m+1)}|_{\Gamma_2} = u^{(2m)}|_{\Gamma_2}$ and $u^{(2m+1)}|_{\partial\Omega_2 \cap \partial\Omega} = 0$.

The variational form of the problem is,

$$a(u, v) = f(v), \quad \forall v \in H_0^1(\Omega),$$

where the bilinear form a is,

$$a(u, v) = \int_\Omega \nabla u . \nabla v dx$$

Let $V_1 = H_0^1(\Omega_1)$ and $V_2 = H_0^1(\Omega_2)$, and let the projectors P_1 and P_2 defined by,

$$a(P_i u, w) = a(u, w), \quad \forall w \in V_i, \quad i = 1, 2.$$

The classical Schwarz alternating method

We consider the Poisson model problem,

$$-\Delta u = f \quad \text{in} \quad \Omega, \quad u|_{\partial\Omega} = 0.$$

The Schwarz alternating algorithm can be formulated at the PDE level, assuming $u^{(1)}$ is given,

1. solve $-\Delta u^{(2m)} = f$ in Ω_1, with $u^{(2m)}|_{\Gamma_1} = u^{(2m-1)}|_{\Gamma_1}$ and $u^{(2m)}|_{\partial\Omega_1 \cap \partial\Omega} = 0$.
2. solve $-\Delta u^{(2m+1)} = f$ in Ω_2, with $u^{(2m+1)}|_{\Gamma_2} = u^{(2m)}|_{\Gamma_2}$ and $u^{(2m+1)}|_{\partial\Omega_2 \cap \partial\Omega} = 0$.

The variational form of the problem is,

$$a(u, v) = f(v), \quad \forall v \in H_0^1(\Omega),$$

where the bilinear form a is,

$$a(u, v) = \int_\Omega \nabla u . \nabla v dx$$

Let $V_1 = H_0^1(\Omega_1)$ and $V_2 = H_0^1(\Omega_2)$, and let the projectors P_1 and P_2 defined by,

$$a(P_i u, w) = a(u, w), \quad \forall w \in V_i, \quad i = 1, 2.$$

The classical Schwarz alternating method

We consider the Poisson model problem,

$$-\Delta u = f \quad \text{in} \quad \Omega, \quad u|_{\partial\Omega} = 0.$$

The Schwarz alternating algorithm can be formulated at the PDE level, assuming $u^{(1)}$ is given,

1. solve $-\Delta u^{(2m)} = f$ in Ω_1, with $u^{(2m)}|_{\Gamma_1} = u^{(2m-1)}|_{\Gamma_1}$ and $u^{(2m)}|_{\partial\Omega_1 \cap \partial\Omega} = 0$.
2. solve $-\Delta u^{(2m+1)} = f$ in Ω_2, with $u^{(2m+1)}|_{\Gamma_2} = u^{(2m)}|_{\Gamma_2}$ and $u^{(2m+1)}|_{\partial\Omega_2 \cap \partial\Omega} = 0$.

The variational form of the problem is,

$$a(u, v) = f(v), \quad \forall v \in H_0^1(\Omega),$$

where the bilinear form a is,

$$a(u, v) = \int_\Omega \nabla u . \nabla v dx$$

Let $V_1 = H_0^1(\Omega_1)$ and $V_2 = H_0^1(\Omega_2)$, and let the projectors P_1 and P_2 defined by,

$$a(P_i u, w) = a(u, w), \quad \forall w \in V_i, \quad i = 1, 2.$$

The classical Schwarz alternating method

We consider the Poisson model problem,

$$-\Delta u = f \quad \text{in} \quad \Omega, \quad u|_{\partial\Omega} = 0.$$

The Schwarz alternating algorithm can be formulated at the PDE level, assuming $u^{(1)}$ is given,

1. solve $-\Delta u^{(2m)} = f$ in Ω_1, with $u^{(2m)}|_{\Gamma_1} = u^{(2m-1)}|_{\Gamma_1}$ and $u^{(2m)}|_{\partial\Omega_1 \cap \partial\Omega} = 0$.
2. solve $-\Delta u^{(2m+1)} = f$ in Ω_2, with $u^{(2m+1)}|_{\Gamma_2} = u^{(2m)}|_{\Gamma_2}$ and $u^{(2m+1)}|_{\partial\Omega_2 \cap \partial\Omega} = 0$.

The variational form of the problem is,

$$a(u, v) = f(v), \quad \forall v \in H_0^1(\Omega),$$

where the bilinear form a is,

$$a(u, v) = \int_\Omega \nabla u . \nabla v dx$$

Let $V_1 = H_0^1(\Omega_1)$ and $V_2 = H_0^1(\Omega_2)$, and let the projectors P_1 and P_2 defined by,

$$a(P_i u, w) = a(u, w), \quad \forall w \in V_i, \quad i = 1, 2.$$

The classical Schwarz alternating method

We consider the Poisson model problem,

$$-\Delta u = f \quad \text{in} \quad \Omega, \quad u|_{\partial\Omega} = 0.$$

The Schwarz alternating algorithm can be formulated at the PDE level, assuming $u^{(1)}$ is given,

1. solve $-\Delta u^{(2m)} = f$ in Ω_1, with $u^{(2m)}|_{\Gamma_1} = u^{(2m-1)}|_{\Gamma_1}$ and $u^{(2m)}|_{\partial\Omega_1 \cap \partial\Omega} = 0$.
2. solve $-\Delta u^{(2m+1)} = f$ in Ω_2, with $u^{(2m+1)}|_{\Gamma_2} = u^{(2m)}|_{\Gamma_2}$ and $u^{(2m+1)}|_{\partial\Omega_2 \cap \partial\Omega} = 0$.

The variational form of the problem is,

$$a(u, v) = f(v), \quad \forall v \in H_0^1(\Omega),$$

where the bilinear form a is,

$$a(u, v) = \int_\Omega \nabla u . \nabla v dx$$

Let $V_1 = H_0^1(\Omega_1)$ and $V_2 = H_0^1(\Omega_2)$, and let the projectors P_1 and P_2 defined by,

$$a(P_i u, w) = a(u, w), \quad \forall w \in V_i, \quad i = 1, 2.$$

The classical Schwarz alternating method

The functions defined only on subdomains are extended by 0 to $H_0^1(\Omega)$.

Then,

$$\begin{cases} a(u^{(2m)} - u, v_1) = 0, & \forall v_1 \in V_1, \quad u^{(2m)} - u^{(2m-1)} \in V_1, \\ a(u^{(2m+1)} - u, v_2) = 0, & \forall v_2 \in V_2, \quad u^{(2m+1)} - u^{(2m)} \in V_2. \end{cases}$$

It is easy to see that,

$$\begin{cases} u - u^{(2m)} &= (I - P_1)(u - u^{(2m-1)}), \\ u - u^{(2m+1)} &= (I - P_2)(u - u^{(2m)}). \end{cases}$$

Therefore,

$$u - u^{(2m+1)} = (I - P_2)(I - P_1)(u - u^{(2m-1)}).$$

showing the multiplicative nature of the alternating Schwarz algorithm.

The classical Schwarz alternating method

The functions defined only on subdomains are extended by 0 to $H_0^1(\Omega)$.

Then,

$$\begin{cases} a(u^{(2m)} - u, v_1) = 0, & \forall v_1 \in V_1, \quad u^{(2m)} - u^{(2m-1)} \in V_1, \\ a(u^{(2m+1)} - u, v_2) = 0, & \forall v_2 \in V_2, \quad u^{(2m+1)} - u^{(2m)} \in V_2. \end{cases}$$

It is easy to see that,

$$\begin{cases} u - u^{(2m)} &= (I - P_1)(u - u^{(2m-1)}), \\ u - u^{(2m+1)} &= (I - P_2)(u - u^{(2m)}). \end{cases}$$

Therefore,

$$u - u^{(2m+1)} = (I - P_2)(I - P_1)(u - u^{(2m-1)}).$$

showing the multiplicative nature of the alternating Schwarz algorithm.

The classical Schwarz alternating method

The mathematical formulation of the problem for studying convergence is in terms of the iterated projections of the error,

$$e^{(2m)} = (I - P_1)e^{(2m-1)}, \quad e^{(2m+1)} = (I - P_2)e^{(2m)}.$$

Theorem 6.1

(P.L. Lions, 1988)

If $V = \overline{V_1 + V_2}$, where the overbar denotes the closure of the set, then $e^{(k)} \to 0$.

The classical Schwarz alternating method

The mathematical formulation of the problem for studying convergence is in terms of the iterated projections of the error,

$$e^{(2m)} = (I - P_1)e^{(2m-1)}, \quad e^{(2m+1)} = (I - P_2)e^{(2m)}.$$

Theorem 6.1

(P.L. Lions, 1988)

If $V = \overline{V_1 + V_2}$, where the overbar denotes the closure of the set, then $e^{(k)} \to 0$.

Algebraic form of the Schwarz alternating algorithm

Let $Ax = b$ be the linear system associated to the discretization of a 2D second order elliptic equation in a rectangle using a five point finite difference scheme.

For a row-wise ordering of the unknowns, the matrix A has a block structure,

$$A = \begin{pmatrix} D_1 & -B_2^t & & & \\ -B_2 & D_2 & -B_3^t & & \\ & \ddots & \ddots & \ddots & \\ & & -B_{n-1} & D_{n-1} & -B_n^t \\ & & & -B_n & D_n \end{pmatrix}.$$

Algebraic form of the Schwarz alternating algorithm

Let us assume a two-subdomain decomposition of the domain Ω. Then the matrix A^1 associated to Ω_1 is,

$$A^1 = \begin{pmatrix} D_1 & -B_2^t & & & \\ -B_2 & D_2 & -B_3^t & & \\ & \ddots & \ddots & \ddots & \\ & & -B_{p-2} & D_{p-2} & -B_{p-1}^t \\ & & & -B_{p-1} & D_{p-1} \end{pmatrix},$$

and the matrix A^2 corresponding to Ω_2 is,

$$A^2 = \begin{pmatrix} D_{l+1} & -B_{l+2}^t & & & \\ -B_{l+2} & D_{l+2} & -B_{l+3}^t & & \\ & \ddots & \ddots & \ddots & \\ & & -B_{n-1} & D_{n-1} & -B_n^t \\ & & & -B_n & D_n \end{pmatrix},$$

with $p - 1 > l + 1$ (overlapping decomposition).

Algebraic form of the Schwarz alternating algorithm

Let us denote the matrix A in block form as,

$$A = \begin{pmatrix} A^1 & A^{1,2} \\ X & X \end{pmatrix},$$

or,

$$A = \begin{pmatrix} Y & Y \\ A^{2,1} & A^2 \end{pmatrix}.$$

Note that $A^{1,2}$ has only one nonzero block in the left lower corner and $A^{2,1}$ is zero except for the upper right block.

Algebraic form of the Schwarz alternating algorithm

Let x_1 and x_2 be the partitioning of the unknown vectors such that,

$$x_1 = (x_{1,1}, \cdots, x_{1,p-1})^t \quad , \quad x_2 = (x_{2,l+1}, \cdots, x_{2,n})^t \quad ,$$

and let b_1 and b_2 the corresponding partitioned right-hand side vectors.

We extend the vectors x_1 and x_2 to Ω by completing with the components of the previous iterate of the Schwarz algorithm i.e. we define $x^{(2m)}$ by $x_1^{(2m)}$ for the first $p-1$ components, and by $x_2^{(2m-1)}$ for the components p to n (which we denote by $x_{1,2}^{(2m-1)}$).

Similarly, $x^{(2m+1)}$ is defined by by $x_1^{(2m)}$ for the first l components (which we denote by $x_{2,1}^{(2m)}$), and by $x_2^{(2m+1)}$ for the components $l+1$ to n

Algebraic form of the Schwarz alternating algorithm

Let x_1 and x_2 be the partitioning of the unknown vectors such that,

$$x_1 = (x_{1,1}, \cdots, x_{1,p-1})^t \quad , \quad x_2 = (x_{2,l+1}, \cdots, x_{2,n})^t \quad ,$$

and let b_1 and b_2 the corresponding partitioned right-hand side vectors.

We extend the vectors x_1 and x_2 to Ω by completing with the components of the previous iterate of the Schwarz algorithm i.e. we define $x^{(2m)}$ by $x_1^{(2m)}$ for the first $p-1$ components, and by $x_2^{(2m-1)}$ for the components p to n (which we denote by $x_{1,2}^{(2m-1)}$).

Similarly, $x^{(2m+1)}$ is defined by by $x_1^{(2m)}$ for the first l components (which we denote by $x_{2,1}^{(2m)}$), and by $x_2^{(2m+1)}$ for the components $l+1$ to n

Algebraic form of the Schwarz alternating algorithm

The Schwarz alternating algorithm can be written as,

$$\begin{cases} x_1^{(2m)} & = & x_1^{(2m-1)} & + & (A^1)^{-1}\left[b_1 - A^1 x_1^{(2m-1)} - A^{1,2} x_{1,2}^{(2m-1)}\right], \\ x_2^{(2m+1)} & = & x_2^{(2m)} & + & (A^2)^{-1}\left[b_2 - A^2 x_2^{(2m)} - A^{2,1} x_{2,1}^{(2m)}\right]. \end{cases}$$

Note that the expressions within [.] are restrictions of the residual to Ω_1 and Ω_2 respectively.

Then, we can write globally,

$$\begin{cases} x^{(2m)} & = x^{(2m-1)} & + & \begin{pmatrix} (A^1)^{-1} & 0 \\ 0 & 0 \end{pmatrix}\left[b - Ax^{(2m-1)}\right], \\ x^{(2m+1)} & = x^{(2m)} & + & \begin{pmatrix} 0 & 0 \\ 0 & (A^2)^{-1} \end{pmatrix}\left[b - Ax^{(2m)}\right]. \end{cases}$$

Algebraic form of the Schwarz alternating algorithm

The Schwarz alternating algorithm can be written as,

$$\begin{cases} x_1^{(2m)} & = & x_1^{(2m-1)} & + & (A^1)^{-1}\left[b_1 - A^1 x_1^{(2m-1)} - A^{1,2} x_{1,2}^{(2m-1)}\right], \\ x_2^{(2m+1)} & = & x_2^{(2m)} & + & (A^2)^{-1}\left[b_2 - A^2 x_2^{(2m)} - A^{2,1} x_{2,1}^{(2m)}\right]. \end{cases}$$

Note that the expressions within [.] are restrictions of the residual to Ω_1 and Ω_2 respectively.

Then, we can write globally,

$$\begin{cases} x^{(2m)} & = x^{(2m-1)} & + & \begin{pmatrix} (A^1)^{-1} & 0 \\ 0 & 0 \end{pmatrix}\left[b - Ax^{(2m-1)}\right], \\ x^{(2m+1)} & = x^{(2m)} & + & \begin{pmatrix} 0 & 0 \\ 0 & (A^2)^{-1} \end{pmatrix}\left[b - Ax^{(2m)}\right]. \end{cases}$$

Algebraic form of the Schwarz alternating algorithm

By eliminating $x^{(2m)}$ we obtain,

$$x^{(2m+1)} = x^{(2m-1)} + \left[\begin{pmatrix} (A^1)^{-1} & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & (A^2)^{-1} \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 0 & (A^2)^{-1} \end{pmatrix} A \begin{pmatrix} (A^1)^{-1} & 0 \\ 0 & 0 \end{pmatrix} \right] [b - Ax^{(2m-1)}].$$

This shows that the Schwarz alternating method is nothing else than a preconditioned Richardson iteration.

We introduce the restriction operators R_1 and R_2 such that,

$$x_1 = R_1 x \quad \text{and} \quad x_2 = R_2 x.$$

R_1 is simply the matrix $(I_{p-1} \quad 0)$ (of size $p-1 \times n$) and R_2 the matrix $(0 \quad I_{n-l+1})$ (of size $n-l+1 \times n$).

Then,

$$A^1 = R_1 A R_1^t \quad \text{and} \quad A^2 = R_2 A R_2^t.$$

Algebraic form of the Schwarz alternating algorithm

By eliminating $x^{(2m)}$ we obtain,

$$x^{(2m+1)} = x^{(2m-1)} + \left[\begin{pmatrix} (A^1)^{-1} & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & (A^2)^{-1} \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 0 & (A^2)^{-1} \end{pmatrix} A \begin{pmatrix} (A^1)^{-1} & 0 \\ 0 & 0 \end{pmatrix} \right] [b - Ax^{(2m-1)}].$$

This shows that the Schwarz alternating method is nothing else than a preconditioned Richardson iteration.

We introduce the restriction operators R_1 and R_2 such that,

$$x_1 = R_1 x \quad \text{and} \quad x_2 = R_2 x.$$

R_1 is simply the matrix $(I_{p-1} \quad 0)$ (of size $p-1 \times n$) and R_2 the matrix $(0 \quad I_{n-l+1})$ (of size $n-l+1 \times n$).

Then,

$$A^1 = R_1 A R_1^t \quad \text{and} \quad A^2 = R_2 A R_2^t.$$

Algebraic form of the Schwarz alternating algorithm

By eliminating $x^{(2m)}$ we obtain,

$$x^{(2m+1)} = x^{(2m-1)} + \left[\begin{pmatrix} (A^1)^{-1} & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & (A^2)^{-1} \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 0 & (A^2)^{-1} \end{pmatrix} A \begin{pmatrix} (A^1)^{-1} & 0 \\ 0 & 0 \end{pmatrix} \right] [b - Ax^{(2m-1)}].$$

This shows that the Schwarz alternating method is nothing else than a preconditioned Richardson iteration.

We introduce the restriction operators R_1 and R_2 such that,

$$x_1 = R_1 x \quad \text{and} \quad x_2 = R_2 x.$$

R_1 is simply the matrix $(I_{p-1} \quad 0)$ (of size $p-1 \times n$) and R_2 the matrix $(0 \quad I_{n-l+1})$ (of size $n-l+1 \times n$).

Then,

$$A^1 = R_1 A R_1^t \quad \text{and} \quad A^2 = R_2 A R_2^t.$$

Algebraic form of the Schwarz alternating algorithm

In the first half step of the iteration, we have to restrict the residual by R_1, apply the inverse of $R_1 A R_1^t$ and extend the result by R_1^t,

$$x^{(2m)} = x^{(2m-1)} + R_1^t [R_1 A R_1^t]^{-1} R_1 [b - A x^{(2m-1)}].$$

Similarly, the second half step is,

$$x^{(2m+1)} = x^{(2m)} + R_2^t [R_2 A R_2^t]^{-1} R_2 [b - A x^{(2m)}].$$

Proposition 6.2

The matrix $P_i = R_i^t [R_i A R_i^t]^{-1} R_i A$, for $i = 1, 2$ is an orthogonal projection in the scalar product defined by A.

Algebraic form of the Schwarz alternating algorithm

In the first half step of the iteration, we have to restrict the residual by R_1, apply the inverse of $R_1 A R_1^t$ and extend the result by R_1^t,

$$x^{(2m)} = x^{(2m-1)} + R_1^t [R_1 A R_1^t]^{-1} R_1 [b - A x^{(2m-1)}].$$

Similarly, the second half step is,

$$x^{(2m+1)} = x^{(2m)} + R_2^t [R_2 A R_2^t]^{-1} R_2 [b - A x^{(2m)}].$$

Proposition 6.2

The matrix $P_i = R_i^t [R_i A R_i^t]^{-1} R_i A$, for $i = 1, 2$ is an orthogonal projection in the scalar product defined by A.

Algebraic form of the Schwarz alternating algorithm

We have,

$$P_i P_i = R_i^t [R_i A R_i^t]^{-1} R_i A R_i^t [R_i A R_i^t]^{-1} R_i A = P_i.$$

Moreover,

$$AP_i = AR_i^t [R_i A R_i^t]^{-1} R_i A = (AP_i)^t.$$

If $\varepsilon^{(m)}$ is the error, we have,

$$\varepsilon^{(2m)} = (I - P_1)\varepsilon^{(2m-1)} \quad \text{and} \quad \varepsilon^{(2m+1)} = (I - P_2)\varepsilon^{(2m)}.$$

Therefore, P_i is the discrete version of the projection operator introduced earlier.

We remark that if the restriction operators R_i are chosen properly, the Schwarz alternating method is a generalization of the block Gauss-Seidel algorithm where the restriction operators allow for overlapping of the blocks.

Algebraic form of the Schwarz alternating algorithm

We have,

$$P_i P_i = R_i^t [R_i A R_i^t]^{-1} R_i A R_i^t [R_i A R_i^t]^{-1} R_i A = P_i.$$

Moreover,

$$AP_i = AR_i^t [R_i A R_i^t]^{-1} R_i A = (AP_i)^t.$$

If $\varepsilon^{(m)}$ is the error, we have,

$$\varepsilon^{(2m)} = (I - P_1)\varepsilon^{(2m-1)} \quad \text{and} \quad \varepsilon^{(2m+1)} = (I - P_2)\varepsilon^{(2m)}.$$

Therefore, P_i is the discrete version of the projection operator introduced earlier.

We remark that if the restriction operators R_i are chosen properly, the Schwarz alternating method is a generalization of the block Gauss-Seidel algorithm where the restriction operators allow for overlapping of the blocks.

Algebraic form of the Schwarz alternating algorithm

We have,

$$P_i P_i = R_i^t [R_i A R_i^t]^{-1} R_i A R_i^t [R_i A R_i^t]^{-1} R_i A = P_i.$$

Moreover,

$$AP_i = AR_i^t [R_i A R_i^t]^{-1} R_i A = (AP_i)^t.$$

If $\varepsilon^{(m)}$ is the error, we have,

$$\varepsilon^{(2m)} = (I - P_1)\varepsilon^{(2m-1)} \quad \text{and} \quad \varepsilon^{(2m+1)} = (I - P_2)\varepsilon^{(2m)}.$$

Therefore, P_i is the discrete version of the projection operator introduced earlier.

We remark that if the restriction operators R_i are chosen properly, the Schwarz alternating method is a generalization of the block Gauss-Seidel algorithm where the restriction operators allow for overlapping of the blocks.

The additive Schwarz method

A way to get a parallel Schwarz method is to use instead a block Jacobi-like algorithm.

Roughly speaking, we solve independently for each subdomain using the interface conditions from the previous iteration, extend the results to the whole domain and sum them. Then, the algorithm proceeds by solving in parallel,

1. $-\Delta u^{(2m)} = f$ in Ω_1, with $u^{(2m)}|_{\Gamma_1} = u^{(2m-1)}|_{\Gamma_1}$ and $u^{(2m)}|_{\partial\Omega_1 \cap \partial\Omega} = 0$.
2. $-\Delta u^{(2m)} = f$ in Ω_2, with $u^{(2m)}|_{\Gamma_2} = u^{(2m-1)}|_{\Gamma_2}$ and $u^{(2m)}|_{\partial\Omega_2 \cap \partial\Omega} = 0$.

However, this method is intended to be a preconditioner for the CG method, which is defined by,

$$M^{-1} = \sum_{i=1,2} R_i^t [R_i A R_i^t]^{-1} R_i.$$

The additive Schwarz method

A way to get a parallel Schwarz method is to use instead a block Jacobi-like algorithm.

Roughly speaking, we solve independently for each subdomain using the interface conditions from the previous iteration, extend the results to the whole domain and sum them. Then, the algorithm proceeds by solving in parallel,

1. $-\Delta u^{(2m)} = f$ in Ω_1, with $u^{(2m)}|_{\Gamma_1} = u^{(2m-1)}|_{\Gamma_1}$ and $u^{(2m)}|_{\partial\Omega_1 \cap \partial\Omega} = 0$.
2. $-\Delta u^{(2m)} = f$ in Ω_2, with $u^{(2m)}|_{\Gamma_2} = u^{(2m-1)}|_{\Gamma_2}$ and $u^{(2m)}|_{\partial\Omega_2 \cap \partial\Omega} = 0$.

However, this method is intended to be a preconditioner for the CG method, which is defined by,

$$M^{-1} = \sum_{i=1,2} R_i^t [R_i A R_i^t]^{-1} R_i.$$

Algebraic DD methods without overlapping

Let us consider an elliptic second order PDE in a rectangle discretized by standard finite difference schemes.

Let Ω_1 and Ω_2 be the two subdomains and $\Gamma_{1,2}$ the interface which is a mesh line in the present case.

Let x_1 (respectively x_2) be the vector of unknowns in Ω_1 (respectively Ω_2), and $x_{1,2}$ be the vector of interface unknowns.

With this numbering of the unknowns, the linear system can be rewritten blockwise as,

$$\begin{pmatrix} A^1 & 0 & E^1 \\ 0 & A^2 & E^2 \\ (E^1)^t & (E^2)^t & A^{1,2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_{1,2} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_{1,2} \end{pmatrix}.$$

Algebraic DD methods without overlapping

Let us consider an elliptic second order PDE in a rectangle discretized by standard finite difference schemes.

Let Ω_1 and Ω_2 be the two subdomains and $\Gamma_{1,2}$ the interface which is a mesh line in the present case.

Let x_1 (respectively x_2) be the vector of unknowns in Ω_1 (respectively Ω_2), and $x_{1,2}$ be the vector of interface unknowns.

With this numbering of the unknowns, the linear system can be rewritten blockwise as,

$$\begin{pmatrix} A^1 & 0 & E^1 \\ 0 & A^2 & E^2 \\ (E^1)^t & (E^2)^t & A^{1,2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_{1,2} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_{1,2} \end{pmatrix}.$$

Algebraic DD methods without overlapping

A^1 (respectively A^2) is a matrix that represents the coupling of the unknowns local to Ω_1 (respectively Ω_2).

$A^{1,2}$ is matrix that represents the coupling of the unknowns on the interface.

E^1 (respectively E^2) represents the coupling of local unknowns of Ω_1 (respectively Ω_2) with interface unknowns.

Most algebraic non-overlapping DD methods are based on block Gaussian elimination (or approximate block Gaussian factorization) of the matrix A, and the goal is to solve the system by an iterative method.

There are basically two possibilities depending on the fact that we cannot (or do not want to) solve linear systems corresponding to subproblems like,

$$A^1 x_1 = c_1 \quad \text{and} \quad A^2 x_2 = c_2,$$

exactly with a direct method (or with a fast solver).

Algebraic DD methods without overlapping

A^1 (respectively A^2) is a matrix that represents the coupling of the unknowns local to Ω_1 (respectively Ω_2).

$A^{1,2}$ is matrix that represents the coupling of the unknowns on the interface.

E^1 (respectively E^2) represents the coupling of local unknowns of Ω_1 (respectively Ω_2) with interface unknowns.

Most algebraic non-overlapping DD methods are based on block Gaussian elimination (or approximate block Gaussian factorization) of the matrix A, and the goal is to solve the system by an iterative method.

There are basically two possibilities depending on the fact that we cannot (or do not want to) solve linear systems corresponding to subproblems like,

$$A^1 x_1 = c_1 \quad \text{and} \quad A^2 x_2 = c_2,$$

exactly with a direct method (or with a fast solver).

Algebraic DD methods without overlapping

A^1 (respectively A^2) is a matrix that represents the coupling of the unknowns local to Ω_1 (respectively Ω_2).

$A^{1,2}$ is matrix that represents the coupling of the unknowns on the interface.

E^1 (respectively E^2) represents the coupling of local unknowns of Ω_1 (respectively Ω_2) with interface unknowns.

Most algebraic non-overlapping DD methods are based on block Gaussian elimination (or approximate block Gaussian factorization) of the matrix A, and the goal is to solve the system by an iterative method.

There are basically two possibilities depending on the fact that we cannot (or do not want to) solve linear systems corresponding to subproblems like,

$$A^1 x_1 = c_1 \quad \text{and} \quad A^2 x_2 = c_2,$$

exactly with a direct method (or with a fast solver).

Algebraic non-overlapping DD methods with exact subdomain solvers

The local unknowns x_1 and x_2 are eliminated yielding a reduced system,

$$Sx_{1,2} = \tilde{b}_{1,2},$$

with,

$$S = A^{1,2} - (E^1)^t (A^1)^{-1} E^1 \; - \; (E^2)^t (A^2)^{-1} E^2,$$

and,

$$\tilde{b}_{1,2} = b_{1,2} - (E^1)^t (A^1)^{-1} b_1 - (E^2)^t (A^2)^{-1} b_2.$$

The matrix S is the Schur complement of $A^{1,2}$ in A.

Of course, $(A^1)^{-1}$ and $(A^2)^{-1}$ are dense matrices, therefore it is too costly to construct S.

A more economical approach is to solve the reduced system with matrix S using an iterative method.

Then, we need to know the properties of the Schur complement to be able to choose a suitable iterative method.

Moreover, if a Krylov method like CG or GMRES is adopted, knowing the properties of S will help in designing an appropriate preconditioner.

Algebraic non-overlapping DD methods with exact subdomain solvers

The local unknowns x_1 and x_2 are eliminated yielding a reduced system,

$$Sx_{1,2} = \tilde{b}_{1,2},$$

with,

$$S = A^{1,2} - (E^1)^t (A^1)^{-1} E^1 \; - \; (E^2)^t (A^2)^{-1} E^2,$$

and,

$$\tilde{b}_{1,2} = b_{1,2} - (E^1)^t (A^1)^{-1} b_1 - (E^2)^t (A^2)^{-1} b_2.$$

The matrix S is the Schur complement of $A^{1,2}$ in A.

Of course, $(A^1)^{-1}$ and $(A^2)^{-1}$ are dense matrices, therefore it is too costly to construct S.

A more economical approach is to solve the reduced system with matrix S using an iterative method.

Then, we need to know the properties of the Schur complement to be able to choose a suitable iterative method.

Moreover, if a Krylov method like CG or GMRES is adopted, knowing the properties of S will help in designing an appropriate preconditioner.

Algebraic non-overlapping DD methods with exact subdomain solvers

It can be proved that if A is a symmetric positive definite M-matrix, S is also a a symmetric positive definite M-matrix.

The following result can be proved by computing the eigenvalues of the matrix S (see G. Meurant, Studies in Mathematics and its Applications, Vol. 28, North Holland, 1999).

Theorem 6.3

For the Poisson model problem the condition number of the Schur complement is,

$$\mathrm{cond}(S) = \mathcal{O}\left(\frac{1}{h}\right).$$

Therefore, the reduced system has two interesting properties: the dimension of the system is smaller and the condition number is one order of magnitude better.

Algebraic non-overlapping DD methods with exact subdomain solvers

It can be proved that if A is a symmetric positive definite M-matrix, S is also a a symmetric positive definite M-matrix.

The following result can be proved by computing the eigenvalues of the matrix S (see G. Meurant, Studies in Mathematics and its Applications, Vol. 28, North Holland, 1999).

Theorem 6.3

For the Poisson model problem the condition number of the Schur complement is,

$$\texttt{cond}(S) = \mathcal{O}\left(\frac{1}{h}\right).$$

Therefore, the reduced system has two interesting properties: the dimension of the system is smaller and the condition number is one order of magnitude better.

Algebraic non-overlapping DD methods with exact subdomain solvers

One of the operations that must be performed when a Krylov method is used for solving the reduced system, is the product of the matrix S by a given vector p.

The product Sp can be computed easily as,

$$Sp = A^{1,2}p - (E^1)^t(A^1)^{-1}E^1p - (E^2)^t(A^2)^{-1}E^2p,$$

p being a vector defined on the interface $\Gamma_{1,2}$.

The vectors $w_i = (A^i)^{-1}E^ip$ for $i = 1, 2$ are computed by solving in parallel the local (subdomain) linear systems,

$$A^iw_i = E^ip.$$

and Sp is computed as,

$$Sp = A^{1,2}p - (E^1)^tw_1 - (E^2)^tw_2.$$

Overall, the resulting DD algorithm can be seen as a hybrid iterative-direct parallel solution method for the linear system $Ax = b$.

Algebraic non-overlapping DD methods with exact subdomain solvers

One of the operations that must be performed when a Krylov method is used for solving the reduced system, is the product of the matrix S by a given vector p.

The product Sp can be computed easily as,

$$Sp = A^{1,2}p - (E^1)^t(A^1)^{-1}E^1p - (E^2)^t(A^2)^{-1}E^2p,$$

p being a vector defined on the interface $\Gamma_{1,2}$.

The vectors $w_i = (A^i)^{-1}E^ip$ for $i = 1, 2$ are computed by solving in parallel the local (subdomain) linear systems,

$$A^iw_i = E^ip.$$

and Sp is computed as,

$$Sp = A^{1,2}p - (E^1)^tw_1 - (E^2)^tw_2.$$

Overall, the resulting DD algorithm can be seen as a hybrid iterative-direct parallel solution method for the linear system $Ax = b$.

Algebraic non-overlapping DD methods with exact subdomain solvers

One of the operations that must be performed when a Krylov method is used for solving the reduced system, is the product of the matrix S by a given vector p.

The product Sp can be computed easily as,

$$Sp = A^{1,2}p - (E^1)^t(A^1)^{-1}E^1p - (E^2)^t(A^2)^{-1}E^2p,$$

p being a vector defined on the interface $\Gamma_{1,2}$.

The vectors $w_i = (A^i)^{-1}E^ip$ for $i = 1, 2$ are computed by solving in parallel the local (subdomain) linear systems,

$$A^iw_i = E^ip.$$

and Sp is computed as,

$$Sp = A^{1,2}p - (E^1)^tw_1 - (E^2)^tw_2.$$

Overall, the resulting DD algorithm can be seen as a hybrid iterative-direct parallel solution method for the linear system $Ax = b$.

Algebraic non-overlapping DD methods with exact subdomain solvers

One of the operations that must be performed when a Krylov method is used for solving the reduced system, is the product of the matrix S by a given vector p.

The product Sp can be computed easily as,

$$Sp = A^{1,2}p - (E^1)^t(A^1)^{-1}E^1p - (E^2)^t(A^2)^{-1}E^2p,$$

p being a vector defined on the interface $\Gamma_{1,2}$.

The vectors $w_i = (A^i)^{-1}E^ip$ for $i = 1, 2$ are computed by solving in parallel the local (subdomain) linear systems,

$$A^iw_i = E^ip.$$

and Sp is computed as,

$$Sp = A^{1,2}p - (E^1)^tw_1 - (E^2)^tw_2.$$

Overall, the resulting DD algorithm can be seen as a hybrid iterative-direct parallel solution method for the linear system $Ax = b$.

Algebraic non-overlapping DD methods with approximate subdomain solvers

In the present case, the linear systems,

$$A^1 x_1 = c_1 \quad \text{and} \quad A^2 x_2 = c_2,$$

are solved using an iterative method, or we would like to replace A^1 and A^2 by approximations (preconditioners).

Let us consider in more details the second option. The problem is to find a global preconditioner M for the permuted form of the matrix A.

If we want to use a PCG method for solving the global linear system $Ax = b$, then M has to be symmetric positive definite.

We choose to define M in the form,

$$M = L \begin{pmatrix} (M^1)^{-1} & & \\ & (M^2)^{-1} & \\ & & (M^{1,2})^{-1} \end{pmatrix} L^t,$$

where M^1 (respectively M^2) is of the same order as A^1 (respectively A^2) and $M^{1,2}$ is of the same order as $A^{1,2}$.

Algebraic non-overlapping DD methods with approximate subdomain solvers

In the present case, the linear systems,

$$A^1 x_1 = c_1 \quad \text{and} \quad A^2 x_2 = c_2,$$

are solved using an iterative method, or we would like to replace A^1 and A^2 by approximations (preconditioners).

Let us consider in more details the second option. The problem is to find a global preconditioner M for the permuted form of the matrix A.

If we want to use a PCG method for solving the global linear system $Ax = b$, then M has to be symmetric positive definite.

We choose to define M in the form,

$$M = L \begin{pmatrix} (M^1)^{-1} & & \\ & (M^2)^{-1} & \\ & & (M^{1,2})^{-1} \end{pmatrix} L^t,$$

where M^1 (respectively M^2) is of the same order as A^1 (respectively A^2) and $M^{1,2}$ is of the same order as $A^{1,2}$.

Algebraic non-overlapping DD methods with approximate subdomain solvers

In the present case, the linear systems,

$$A^1 x_1 = c_1 \quad \text{and} \quad A^2 x_2 = c_2,$$

are solved using an iterative method, or we would like to replace A^1 and A^2 by approximations (preconditioners).

Let us consider in more details the second option. The problem is to find a global preconditioner M for the permuted form of the matrix A.

If we want to use a PCG method for solving the global linear system $Ax = b$, then M has to be symmetric positive definite.

We choose to define M in the form,

$$M = L \begin{pmatrix} (M^1)^{-1} & & \\ & (M^2)^{-1} & \\ & & (M^{1,2})^{-1} \end{pmatrix} L^t,$$

where M^1 (respectively M^2) is of the same order as A^1 (respectively A^2) and $M^{1,2}$ is of the same order as $A^{1,2}$.

Algebraic non-overlapping DD methods with approximate subdomain solvers

L is a block lower triangular matrix,

$$L = \begin{pmatrix} M^1 & & \\ & M^2 & \\ (E^1)^t & (E^2)^t & M^{1,2} \end{pmatrix}.$$

Then, we see how parallelism is introduced: at each PG iteration, we must solve a linear system like,

$$Mz = M \begin{pmatrix} z_1 \\ z_2 \\ z_{1,2} \end{pmatrix} = r = \begin{pmatrix} r_1 \\ r_2 \\ r_{1,2} \end{pmatrix}.$$

This is done by first solving $Ly = r$, where the first two steps are,

$$M^1 y_1 = r_1 \quad \text{and} \quad M^2 y_2 = r_2.$$

This can be done in parallel.

Algebraic non-overlapping DD methods with approximate subdomain solvers

L is a block lower triangular matrix,

$$L = \begin{pmatrix} M^1 & & \\ & M^2 & \\ (E^1)^t & (E^2)^t & M^{1,2} \end{pmatrix}.$$

Then, we see how parallelism is introduced: at each PG iteration, we must solve a linear system like,

$$Mz = M \begin{pmatrix} z_1 \\ z_2 \\ z_{1,2} \end{pmatrix} = r = \begin{pmatrix} r_1 \\ r_2 \\ r_{1,2} \end{pmatrix}.$$

This is done by first solving $Ly = r$, where the first two steps are,

$$M^1 y_1 = r_1 \quad \text{and} \quad M^2 y_2 = r_2.$$

This can be done in parallel.

Algebraic non-overlapping DD methods with approximate subdomain solvers

Then we solve for the interface problem,

$$M^{1,2} y_{1,2} = r_{1,2} - (E^1)^t y_1 - (E^2)^t y_2.$$

To obtain the global solution, we have to perform a backward solve step as,

$$\begin{pmatrix} I & 0 & (M^1)^{-1}E^1 \\ & I & (M^2)^{-1}E^2 \\ & & I \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_{1,2} \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_{1,2} \end{pmatrix}.$$

This implies that $z_{1,2} = y_{1,2}$ and,

$$\begin{cases} M^1 w_1 = E^1 z_{1,2}, & z_1 = y_1 - w_1, \\ M^2 w_2 = E^2 z_{1,2}, & z_2 = y_2 - w_2. \end{cases}$$

The last two stages can be done in parallel.

Algebraic non-overlapping DD methods with approximate subdomain solvers

Then we solve for the interface problem,

$$M^{1,2} y_{1,2} = r_{1,2} - (E^1)^t y_1 - (E^2)^t y_2.$$

To obtain the global solution, we have to perform a backward solve step as,

$$\begin{pmatrix} I & 0 & (M^1)^{-1} E^1 \\ & I & (M^2)^{-1} E^2 \\ & & I \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_{1,2} \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_{1,2} \end{pmatrix}.$$

This implies that $z_{1,2} = y_{1,2}$ and,

$$\begin{cases} M^1 w_1 = E^1 z_{1,2}, & z_1 = y_1 - w_1, \\ M^2 w_2 = E^2 z_{1,2}, & z_2 = y_2 - w_2. \end{cases}$$

The last two stages can be done in parallel.

Algebraic non-overlapping DD methods with approximate subdomain solvers

Then we solve for the interface problem,

$$M^{1,2} y_{1,2} = r_{1,2} - (E^1)^t y_1 - (E^2)^t y_2.$$

To obtain the global solution, we have to perform a backward solve step as,

$$\begin{pmatrix} I & 0 & (M^1)^{-1} E^1 \\ & I & (M^2)^{-1} E^2 \\ & & I \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_{1,2} \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_{1,2} \end{pmatrix}.$$

This implies that $z_{1,2} = y_{1,2}$ and,

$$\begin{cases} M^1 w_1 = E^1 z_{1,2}, & z_1 = y_1 - w_1, \\ M^2 w_2 = E^2 z_{1,2}, & z_2 = y_2 - w_2. \end{cases}$$

The last two stages can be done in parallel.

Algebraic non-overlapping DD methods with approximate subdomain solvers

The problem is now how to choose the approximations M^1, M^2 and $M^{1,2}$.

If we multiply together the three matrices whose product defines M, we obtain,

$$M = \begin{pmatrix} M^1 & 0 & E^1 \\ 0 & M^2 & E^2 \\ (E^1)^t & (E^2)^t & \tilde{M}^{1,2} \end{pmatrix},$$

where,

$$\tilde{M}^{1,2} = M^{1,2} + (E^1)^t (M^1)^{-1} E^1 \; + \; (E^2)^t (M^2)^{-1} E^2.$$

Therefore, as we would like M to be an approximation of A, it makes sense to choose,

$$M^1 \approx A^1 \quad \text{and} \quad M^2 \approx A^2,$$

and,

$$\tilde{M}^{1,2} \approx A^{1,2} \Longrightarrow M^{1,2} \approx A^{1,2} - (E^1)^t (M^1)^{-1} E^1 - (E^2)^t (M^2)^{-1} E^2.$$

Thus, if the inverse of M^1 (respectively M^2) is also a good approximation of the inverse of A^1 (respectively A^2), then $M^{1,2}$ must be an approximation of the Schur complement S.

Algebraic non-overlapping DD methods with approximate subdomain solvers

The problem is now how to choose the approximations M^1, M^2 and $M^{1,2}$.

If we multiply together the three matrices whose product defines M, we obtain,

$$M = \begin{pmatrix} M^1 & 0 & E^1 \\ 0 & M^2 & E^2 \\ (E^1)^t & (E^2)^t & \tilde{M}^{1,2} \end{pmatrix},$$

where,

$$\tilde{M}^{1,2} = M^{1,2} + (E^1)^t (M^1)^{-1} E^1 \; + \; (E^2)^t (M^2)^{-1} E^2.$$

Therefore, as we would like M to be an approximation of A, it makes sense to choose,

$$M^1 \approx A^1 \quad \text{and} \quad M^2 \approx A^2,$$

and,

$$\tilde{M}^{1,2} \approx A^{1,2} \Longrightarrow M^{1,2} \approx A^{1,2} - (E^1)^t (M^1)^{-1} E^1 - (E^2)^t (M^2)^{-1} E^2.$$

Thus, if the inverse of M^1 (respectively M^2) is also a good approximation of the inverse of A^1 (respectively A^2), then $M^{1,2}$ must be an approximation of the Schur complement S.

Algebraic non-overlapping DD methods with approximate subdomain solvers

The problem is now how to choose the approximations M^1, M^2 and $M^{1,2}$.

If we multiply together the three matrices whose product defines M, we obtain,

$$M = \begin{pmatrix} M^1 & 0 & E^1 \\ 0 & M^2 & E^2 \\ (E^1)^t & (E^2)^t & \tilde{M}^{1,2} \end{pmatrix},$$

where,

$$\tilde{M}^{1,2} = M^{1,2} + (E^1)^t (M^1)^{-1} E^1 \; + \; (E^2)^t (M^2)^{-1} E^2.$$

Therefore, as we would like M to be an approximation of A, it makes sense to choose,

$$M^1 \approx A^1 \quad \text{and} \quad M^2 \approx A^2,$$

and,

$$\tilde{M}^{1,2} \approx A^{1,2} \Longrightarrow M^{1,2} \approx A^{1,2} - (E^1)^t (M^1)^{-1} E^1 - (E^2)^t (M^2)^{-1} E^2.$$

Thus, if the inverse of M^1 (respectively M^2) is also a good approximation of the inverse of A^1 (respectively A^2), then $M^{1,2}$ must be an approximation of the Schur complement S.