

# Numerical Linear Algebra for Computational Science and Information Engineering

## Lecture 08 Classical Iterative Methods for Linear Systems

Nicolas Venkovic  
nicolas.venkovic@tum.de

Group of Computational Mathematics  
School of Computation, Information and Technology  
Technical University of Munich

Winter 2025-26



# Outline I

1	Splitting methods	
	Section 7.1 in Darve & Wootters (2021)	2
2	Jacobi method	
	Section 7.2 in Darve & Wootters (2021)	5
3	Gauss-Seidel method	
	Section 7.3 in Darve & Wootters (2021)	7
4	Successive over-relaxation	
	Section 7.4 in Darve & Wootters (2021)	10
5	Homework problems	14
6	Practice session	15

## Towards iterative methods to solve linear systems

- ▶ So far, we saw how **direct methods** can be used to solve linear systems:
  - ① **Factorize** the matrix (e.g., LU or Cholesky factorization) with cost  $\mathcal{O}(n^3)$
  - ② **Solve** the system using the computed factors with cost  $\mathcal{O}(n^2)$
- ▶ **Direct methods** are **very stable** and **accurate**.

However, they can have a **very high computational cost**.

In general, direct methods are **not suitable for very large  $n$** .

- ▶ As a potentially more efficient way to solve linear systems, we will explore **iterative methods**.

Iterative methods are **inexact** in the sense that they rely on the generation of a sequence of approximate solutions which **converges towards the solution**, but the sequence is stopped at finite accuracy.

In general, iterative methods **do not require explicit access to the matrix** and they **rely on the matrix-vector kernel**  $x \mapsto Ax$ .

Iterative methods are particularly **recommended** for cases **where the matrix-vector product can be efficiently deployed**, e.g., as it is the case for sparse matrices.

# Splitting methods

Section 7.1 in Darve & Wootters (2021)

## General splitting methods

- ▶ Splitting methods are simple iterative methods to solve linear systems.
- ▶ Consider the  $A = M - N$  splitting of a matrix  $A$ , where  $M$  is non-singular.
- ▶ The linear system  $Ax = b$  can be recast as follows:

$$Mx - Nx = b$$

$$x - M^{-1}Nx = M^{-1}b$$

$$x = Gx + M^{-1}b$$

so that  $x$  is a **fixed point** of  $f : x \mapsto M^{-1}Nx + M^{-1}b$ , i.e.,  $x = f(x)$ , and where  $G := M^{-1}N$  is the **iteration matrix**.

- ▶ To solve a fixed point problem, one can start with any point  $x$ , and compute  $f(x)$ . Then compute  $f(f(x))$ , then  $f(f(f(x)))$  and so on, until the sequence converges. In particular, we consider the following

### Splitting method update rule

Given a matrix  $A = M - N$  where  $M$  is non-singular, the update rule for a general splitting method with a given  $x^{(0)}$  is

$$x^{(k+1)} := Gx^{(k)} + M^{-1}b \quad \text{where } G := M^{-1}N.$$

## General splitting methods, cont'd<sub>1</sub>

- ▶ The error  $e^{(k+1)} := x^{(k+1)} - x$  is such that

$$\begin{aligned}e^{(k+1)} &= Gx^{(k)} + M^{-1}b - Gx - M^{-1}b \\&= Gx^{(k)} - Gx \\&= Ge^{(k)}\end{aligned}$$

so that  $e^{(k)} = G^k e^{(0)}$ .

- ▶ The convergence theory depends on the iteration matrix  $G = M^{-1}N$ :

### Theorem (Convergence of splitting methods)

*Given  $b$  and  $A = M - N$  with non-singular  $A$  and  $M$ , the iteration*

$$x^{(k+1)} = Gx^{(k)} + M^{-1}b \quad \text{where } G := M^{-1}N$$

*converges for any starting  $x^{(0)}$  if and only if*

$$\rho(G) < 1$$

*where  $\rho(G)$  is the spectral radius, i.e., the largest modulus of eigenvalue of the iteration matrix  $G$ .*

## General splitting methods, cont'd<sub>2</sub>

- ▶ Even though analyzing the spectrum of the iteration matrix  $G$  is generally difficult, it is understood that, the smaller the modulus  $\rho(G)$ , the faster the convergence.

- ▶ How should we pick  $M$  and  $N$ ?

The selection of  $M$  and  $N$  may be guided by two desirable properties:

- Linear systems of the form  $Mz = d$  are easy to solve.

This suggest good choices for  $M$  are diagonal or triangular.

- The spectral radius  $\rho(G)$  is less than 1.

- ▶ We will see several examples of splitting methods, namely

- Jacobi method
- Gauss-Seidel method
- Over-relaxation method

# Jacobi method

Section 7.2 in Darve & Wootters (2021)

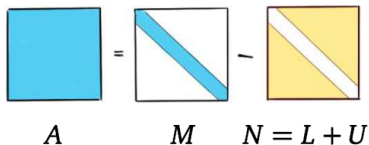


## Jacobi method

► Let  $A = D - L - U$  where

- $D$  is diagonal
- $L$  is strictly lower-triangular, i.e., with zeros on the diagonal
- $U$  is strictly upper-triangular

Then, the splitting is clearly unique given  $A$  as we choose  $M = D$  and  $N = L + U$ :



Darve, E., & Wootters, M. (2021). Numerical linear algebra with Julia. Society for Industrial and Applied Mathematics.

► The Jacobi splitting leads to the following iteration

### Jacobi iterations

Suppose  $A = D - U - L$  as above. The update formula for Jacobi iteration is given by

$$Dx^{(k+1)} = (L + U)x^{(k)} + b.$$

## Jacobi method, cont'd

- The convergence of Jacobi iterations is as follows:

### Theorem (Convergence of Jacobi iterations)

*If  $A$  is strictly diagonally dominant, i.e.,*

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}|$$

*for  $i = 1, 2, \dots, n$ , then Jacobi iterations converge for any initial guess  $x^{(0)}$ .*

Note that this condition is not necessary to ensure convergence.

The necessary condition to ensure convergence remains that the iteration matrix  $G_{\text{Jacobi}} = D^{-1}(L + U)$  has a spectral radius smaller than one.

- The Jacobi method is especially simple to implement.

It is also well-suited for parallel implementation as we have

$$\begin{aligned} x_i^{(k+1)} &= \left( b_i + (L + U)[i, :]x^{(k)} \right) / d_{ii} \\ x_i^{(k+1)} &= \left( b_i - (A - D)[i, :]x^{(k)} \right) / d_{ii}. \end{aligned}$$

# Gauss-Seidel method

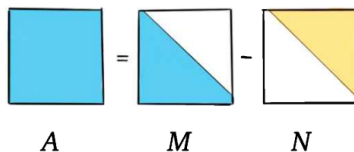
Section 7.3 in Darve & Wootters (2021)

## Gauss-Seidel method

► Let  $A = D - L - U$  where

- $D$  is diagonal
- $L$  is strictly lower-triangular, i.e., with zeros on the diagonal
- $U$  is strictly upper-triangular

Then, the splitting is clearly unique given  $A$  as we choose  $M = D - L$  and  $N = U$ :



Darve, E., & Wootters, M. (2021). Numerical linear algebra with Julia. Society for Industrial and Applied Mathematics.

► The Gauss-Seidel splitting leads to the following iteration

### Gauss-Seidel iterations

Suppose  $A = D - U - L$  as above. The update formula for Gauss-Seidel iteration is given by

$$(D - L)x^{(k+1)} = Ux^{(k)} + b.$$

## Gauss-Seidel method, cont'd<sub>1</sub>

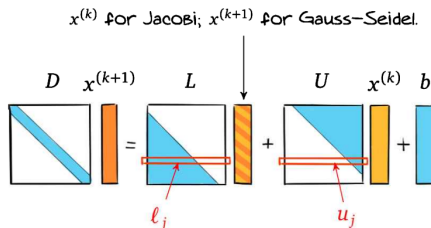
- Intuitively, "putting more information in  $M$ " should help with convergence of the method, and this is indeed the case, i.e.,  $\rho(G_{\text{GS}}) = \rho(G_{\text{Jacobi}})^2$ .
- On the other hand, solving triangular systems with  $M = D - L$  is more involved than solving diagonal systems with  $M = D$ .
- We can compare Gauss-Seidel to Jacobi iterations as follows:

$$(D - L)x^{(k+1)} = Ux^{(k)} + b$$

$$Dx^{(k+1)} = Lx^{(k+1)} + Ux^{(k)} + b \quad (\text{Gauss-Seidel})$$

$$Dx^{(k+1)} = Lx^{(k)} + Ux^{(k)} + b \quad (\text{Jacobi})$$

and see they are very similar except for one term:



## Gauss-Seidel method, cont'd<sub>2</sub>

- The convergence of Gauss-Seidel iterations is as follows:

### Theorem (Convergence of Gauss-Seidel iterations)

*If  $A$*

- *is strictly diagonally dominant, or*
- *is symmetric positive definite (SPD)*

*then Gauss-Seidel iterations converge irrespective of the initial guess  $x^{(0)}$ .*

Note that these conditions are not necessary to ensure convergence.

The necessary condition to ensure convergence remains that the iteration matrix  $G_{\text{GS}} = (D - L)^{-1}U$  has a spectral radius smaller than one.

# Successive over-relaxation

Section 7.4 in Darve & Wootters (2021)

## Successive over-relaxation

- ▶ Successive over-relaxation (SOR) consists of introducing a parameter to a splitting method in order to get a handle of the speed of convergence.
- ▶ In particular, we use a parameter  $\omega$  to boost convergence.

The idea is to start with the Gauss-Seidel update step as follows:

$$Dx_{\text{GS}}^{(k+1)} = Lx_{\text{GS}}^{(k+1)} + Ux^{(k)} + b$$

$$x_{\text{GS}}^{(k+1)} = D^{-1} \left( Lx_{\text{GS}}^{(k+1)} + Ux^{(k)} + b \right)$$

$$x_{\text{GS}}^{(k+1)} = x^{(k)} + \left[ D^{-1} \left( Lx_{\text{GS}}^{(k+1)} + Ux^{(k)} + b \right) - x^{(k)} \right]$$

so that  $x_{\text{GS}}^{(k+1)} = x^{(k)} + \Delta x_{\text{GS}}^{(k)}$  where

$$\Delta x_{\text{GS}}^{(k)} = D^{-1} \left( Lx_{\text{GS}}^{(k+1)} + Ux^{(k)} + b \right) - x^{(k)}$$

is the update to  $x^{(k)}$  in a Gauss-Seidel iteration.

In SOR, the idea is to scale this correction by a parameter  $0 < \omega < 2$ :

$$x_{\text{SOR}}^{(k+1)} = x^{(k)} + \omega \Delta x_{\text{GS}}^{(k)}.$$



## Successive over-relaxation, cont'd

- ▶ When  $\omega \approx 0$ , we are very cautious and only make small corrections to  $x^{(k)}$ .  
When  $\omega = 1$ , we recover a Gauss-Seidel iteration.  
When  $\omega \approx 2$ , we are very confident in the Gauss-Seidel correction and apply it twice instead of once.
- ▶ The update formula of the SOR sequence is given as follows:

$$\begin{aligned}x_{\text{SOR}}^{(k+1)} &= x_{\text{SOR}}^{(k)} + \omega \left[ D^{-1} \left( Lx_{\text{SOR}}^{(k+1)} + Ux_{\text{SOR}}^{(k)} + b \right) - x_{\text{SOR}}^{(k)} \right] \\&= (1 - \omega)x_{\text{SOR}}^{(k)} + \omega \left[ D^{-1} \left( Lx_{\text{SOR}}^{(k+1)} + Ux_{\text{SOR}}^{(k)} + b \right) \right]\end{aligned}$$

which yields the following iterations:

### SOR iterations

Let  $\omega \in (0, 2)$ , and suppose  $A = D - L - U$  as above. The update formula for SOR iterations is

$$(D - \omega L)x_{\text{SOR}}^{(k+1)} = ((1 - \omega)D + \omega U)x_{\text{SOR}}^{(k)} + \omega b.$$

## Successive over-relaxation, cont'd

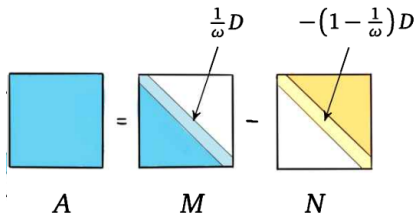
- Since the update formula can be written as

$$\left(\frac{1}{\omega}D - L\right) x_{\text{SOR}}^{(k+1)} = \left(\left(\frac{1}{\omega} - 1\right)D + U\right) x_{\text{SOR}}^{(k)} + b$$

and that we have

$$\left(\frac{1}{\omega}D - L\right) - \left(\left(\frac{1}{\omega} - 1\right)D + U\right) = D - L - U = A$$

we can say that SOR is a splitting method with  $M = \frac{1}{\omega}D - L$  and  $N = \left(\frac{1}{\omega} - 1\right)D + U$  such that  $A = M - N$ :



## Successive over-relaxation, cont'd

- ▶ Although SOR is a heuristic, it can lead to significant improvements in the convergence rate when  $\omega$  is chosen appropriately.

### Theorem (Convergence of SOR iterations)

*If  $A$  is symmetric positive definite (SPD), then SOR iterations converge irrespective of the initial guess  $x^{(0)}$ , for any  $\omega \in (0, 2)$ .*

Note that this condition is not necessary to ensure convergence.

The necessary condition to ensure convergence remains that the iteration matrix

$$\begin{aligned} G &= \left( \frac{1}{\omega} D - L \right)^{-1} \left( \left( \frac{1}{\omega} - 1 \right) D + U \right) \\ &= (D - \omega L)^{-1} ((1 - \omega) D + \omega U) \end{aligned}$$

has a spectral radius smaller than one.

# Homework problems

## Homework problem

Turn in **your own** solution to the following problem:

**Pb. 18** Let  $A = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$ . Analyze the spectrum of the iteration matrix

and show whether

- (a) A Jacobi iteration would converge.
- (b) A Gauss-Seidel iteration would converge.
- (c) A SOR iteration would converge with  $\omega = 1/2$ .

# Practice session

## Practice session

- 1 Implement Jacobi, Gauss-Seidel and SOR iterations.
- 2 Benchmark your implementations.