High-Fidelity Simulations for Turbulent Flows

Luca Sciacovelli

DynFluid Laboratory
Arts et Métiers Institute of Technology
http://savoir.ensam.eu/moodle

Master Recherche “Aérodynamique et Aéroacoustique”
2021 – 2022

Part VI

Implicit numerical schemes

Basics of Time-Marching Methods

Why implicit methods? Useful for:

- Steady-state solutions
- Unsteady problems with characteristic time scales large compared to Δt_{stab} constraints

A semi-discrete scheme in space yields the ODE

$$\boxed{\frac{\mathrm{d}w}{\mathrm{d}t} = R(w)}$$

- **Time** gives a **hyperbolic**/**parabolic** character
- This can be used to develop time-marching numerical methods:
  - Start with an initial guess and march the equations in time while applying BCs
  - The time-varying solution will evolve or the solution will asymptotically approach the steady-state solution (if any)

- If we seek a steady solution, time is not a variable, and the equation takes the form

$$R(w) = 0 \quad R \text{ called the Residual}$$

Options to solve the steady problem:

1. **Direct Methods**: assume the equation is mathematically **elliptic**, solve the system of eqs in a single process. Not applicable to nonlinear problems
2. **Iterative Methods**: Also assume the equation is elliptic; start with an initial guess and iterate
   - Newton or pseudo-Newton methods: solve the steady-state eqs using an iterative technique
     - Linearization of the problem and computation of Jacobian matrix needed
   - Time-marching or false transient methods: iterate the unsteady eqs until steady state is reached
     - It can be explicit or implicit (using known / unknown information respectively)

Source term discretization

- **Spatial** discretization: no particular problems
- **Time** discretization: **explicit** or **implicit**?

Linear model problem:

$$\frac{\mathrm{d}w}{\mathrm{d}t} = sw \qquad s = \mathrm{C}^{\mathrm{te}}$$

Case $s < 0$

$$w(t) = w(0)e^{-|s|t}$$

$$\frac{\Delta w}{w} = \frac{w(t+\Delta t) - w(t)}{w(t)} = e^{-|s|\Delta t} - 1$$

$$\implies -1 < \frac{\Delta w}{w} < 0$$

$$\frac{\Delta w^n}{\Delta t} = \frac{w^{n+1} - w^n}{\Delta t} = -|s|(w^n + \theta \Delta w^n)$$

$$\frac{\Delta w^n}{w} = \frac{-|s|\Delta t}{1 + \theta|s|\Delta t}$$

Scheme agrees with bounds of exact solution when:

- $\theta = 0 : \Delta t < \frac{1}{|s|} \qquad \theta = 1 : \forall \Delta t$
  - **Implicit helps**

Case $s > 0$

$$w(t) = w(0)e^{st}$$

$$\frac{\Delta w}{w} = \frac{w(t+\Delta t) - w(t)}{w(t)} = e^{s\Delta t} - 1$$

$$\implies \frac{\Delta w}{w} > -1$$

$$\frac{\Delta w^n}{\Delta t} = \frac{w^{n+1} - w^n}{\Delta t} = s(w^n + \theta \Delta w^n)$$

$$\frac{\Delta w^n}{w} = \frac{s\Delta t}{1 - \theta s\Delta t}$$

Scheme agrees with bounds of exact solution when:

- $\theta = 0 : \forall \Delta t \qquad \theta = 1 : \Delta t < \frac{1}{s}$
  - **Implicit useless!**

Deduction: make implicit only the negative part of the source terms, and use explicit for the positive part

Euler Methods

Simplest time integration methods: **Euler methods**, explicit (**forward** Euler) and implicit (**backward** Euler)

- Single stage and first-order accurate in time
- The l.h.s. of the equation is discretized as:

$$\frac{\Delta w}{\Delta t} = -[(1-\theta)R(w^n) + \theta R(w^{n+1})] \quad \text{with} \quad \Delta w = w^{n+1} - w^n$$

- Forward Euler: $\theta = 0 \implies w^{n+1} = w^n - \Delta t R(w^n)$
- Backward Euler: $\theta = 1 \implies w^{n+1}$ obtained by solving a system

FTCS Scheme

Remember the FTCS scheme for 1D advection-diffusion

$$\frac{w_j^{n+1} - w_j^n}{\Delta t} + a\frac{\delta\mu w_j^n}{\Delta x} = \nu\frac{\delta^2 w_j^n}{\Delta x^2}$$

- Stable under the condition $\dot{\nu} \leq \frac{1}{2}$ and $R_m \leq \frac{2}{\dot{a}}$
- Very restrictive for high-Re flows (clustered meshes close to solid walls)
- One way to fix the problems is to consider the following implicit scheme:

$$\frac{w_j^{n+1} - w_j^n}{\Delta t} + a\frac{\delta\mu w_j^{n+1}}{\Delta x} = \nu\frac{\delta^2 w_j^{n+1}}{\Delta x^2} \implies \left(\frac{\dot{a}}{2} - \dot{\nu}\right) w_{j+1}^{n+1} + (1 + 2\dot{\nu})w_j^{n+1} + \left(-\frac{\dot{a}}{2} - \dot{\nu}\right) w_{j-1}^{n+1} = w_j^n$$

  - The solution at time n is obtained by solving a linear system
  - **Amplification factor**:

$$G = \frac{1}{[1 + 2\dot{\nu}(1 - \cos\beta)] + i\dot{a}\sin\beta} = \frac{[1 + 2\dot{\nu}(1 - \cos\beta)] - i\dot{a}\sin\beta}{[1 + 2\dot{\nu}(1 - \cos\beta)]^2 + \dot{a}^2\sin^2\beta}$$

$$|G|^2 = \frac{[1 + 2\dot{\nu}(1 - \cos\beta)]^2 + \dot{a}^2\sin^2\beta}{\{[1 + 2\dot{\nu}(1 - \cos\beta)]^2 + \dot{a}^2\sin^2\beta\}^2} = \frac{1}{1 + 4\dot{\nu}(1 - \cos\beta) + 4\dot{\nu}^2(1 - \cos\beta)^2 + \dot{a}^2\sin^2\beta} \leq 1 \quad \forall\beta$$

  - Scheme **unconditionally stable**

Good and bad news for implicit schemes

In general, implicit schemes are **unconditionally stable** $\implies$ large Δt can be applied

- In practice, Δt **restricted** by
  - **Nonlinearities** of the flow equations (linearization needed)
  - **Accuracy** requirements for unsteady flow problems
  - It will nearly always be **much larger** than the explicit CFL-based Δt_{stab} limit
- Matrices have to be **inverted** at each time step
  - CPU cost per iteration and memory requirements significantly higher w.r.t. explicit methods
  - Interesting mainly for steady flows or slow unsteady flows

When to choose an implicit time integration?

- Let be T the smallest physical time scale to be captured:
  - If $T \approx \mathcal{O}((\Delta t)_{\text{stab}})$ or less $\implies$ **explicit** method
  - If $T \gg \mathcal{O}((\Delta t)_{\text{stab}}) \implies$ **implicit** method
  - For steady-state problems, $T \to \infty$: implicit methods are definitely more efficient

Linear Advection problem

$$\boxed{\frac{\partial w}{\partial t} + a\frac{\partial w}{\partial x} = 0}$$

- Implicit version of **FOU** scheme:

$$\frac{w_j^{n+1} - w_j^n}{\Delta t} + a\frac{\delta\mu w_j^{n+1}}{\Delta x} = \frac{1}{2}|a|\frac{\delta^2 w_j^{n+1}}{\Delta x^2}$$

$$\left(\frac{\dot{a} - |\dot{a}|}{2}\right) w_{j+1}^{n+1} + (1+|\dot{a}|)w_j^{n+1} - \left(\frac{\dot{a} + |\dot{a}|}{2}\right) w_{j-1}^{n+1} = w_j^n$$

- Implicit version of the **BTCS** scheme:

$$\frac{w_j^{n+1} - w_j^n}{\Delta t} + a\frac{\delta\mu w_j^{n+1}}{\Delta x} = 0 \implies \frac{\dot{a}}{2} w_{j+1}^{n+1} + w_j^{n+1} - \frac{\dot{a}}{2} w_{j-1}^{n+1} = w_j^n$$

- More generally, a linear 3-point implicit scheme can be written as $Aw^{n+1} = w^n$, with $w = (w_1, ..., w_N)$, and A a tridiagonal matrix, *i.e.*:

$$A = \begin{bmatrix} b & c & 0 & \dots & \dots & 0 \\ a & b & c & 0 & \dots & 0 \\ 0 & a & b & c & 0 & 0 \\ & & & \dots & & \\ 0 & \dots & a & b & c & 0 \\ 0 & \dots & 0 & a & b & c \\ 0 & \dots & \dots & 0 & a & b \end{bmatrix}$$

- Expression in **Delta Form**:

Define the solution increment: $\Delta w_j^n = w_j^{n+1} - w_j^n$
The preceding schemes can be re-written as:

FOU

$$\frac{\Delta w_j^n}{\Delta t} + a\frac{\delta\mu(\Delta w_j^n + w_j^n)}{\Delta x} = \frac{1}{2}|a|\frac{\delta^2(\Delta w_j^n + w_j^n)}{\Delta x^2}$$

$$\left(1 + \dot{a}\delta\mu - \frac{1}{2}|\dot{a}|\delta^2\right)\Delta w_j^n = -\dot{a}\delta\mu w_j^n + \frac{1}{2}|\dot{a}|\delta^2 w_j^n$$

$$= \Delta w_{j,exp}^n$$

$$\implies \quad A_{\mathrm{FOU}}\Delta w_j^n = \Delta w_{exp,j}^{FOU,n}$$

BTCS

$$\frac{\Delta w_j^n}{\Delta t} + a\frac{\delta\mu\Delta w_j^n}{\Delta x} = -a\frac{\delta\mu w_j^n}{\Delta x}$$

$$(1 + \dot{a}\delta\mu)\Delta w_j^n = -\dot{a}\delta\mu w_j^n = \Delta w_{j,exp}^n$$

$$\implies \quad A_{\mathrm{BTCS}}\Delta w_j^n = \Delta w_{exp,j}^{CS,n}$$

- A is called an iteration matrix or **mass matrix**

Diagonal Dominance

- For 3-point schemes, A is a tridiagonal matrix $\Longrightarrow$ efficiently solved via **Thomas algorithm**
- For Larger stencils, A is a band-matrix with more than 3 non-zero diagonals
- If an iterative algorithm is adopted (*e.g.* Jacobi, Gauss-Seidel,...), a solution is obtained if the iteration matrix is **diagonally dominant**, i.e. $A = [a_{ij}]$ such that $|a_{ii}| \geq \sum_{j\neq i} |a_{ij}|$

Implicit FOU

- **Diagonal dominance**:

$$|a_{ii}| = (1+|\dot{a}|) > \left|\frac{\dot{a}-|\dot{a}|}{2}\right| + \left|\frac{\dot{a}+|\dot{a}|}{2}\right| = |\dot{a}| \quad \forall \dot{a}$$

Strictly diagonally dominant

- **Amplification factor**:

$$G = \frac{1+|\dot{a}|(1-\cos\beta) - i\dot{a}\sin\beta}{[1+|\dot{a}|(1-\cos\beta)]^2 + \dot{a}^2\sin^2\beta}$$

$$|G|^2 = \frac{1}{\left(1+2|\dot{a}|\sin^2\frac{\beta}{2}\right)^2 + 4|\dot{a}|^2\sin^2\frac{\beta}{2}\left(1-\sin^2\frac{\beta}{2}\right)}$$

$$= \frac{1}{1+8|\dot{a}|\sin^2\frac{\beta}{2}} \leq 1 \quad \forall \dot{a}$$

  - Unconditionally stable
  - $G \to 0$ for $|\dot{a}| \to \infty$: the larger the CFL, the more quickly errors are damped

BTCS

- **Diagonal Dominance**:

$$|a_{ii}| = 1 > \left|\frac{\dot{a}}{2}\right| + \left|\frac{\dot{a}}{2}\right| = |\dot{a}| \iff \text{CFL} = |\dot{a}| < 1$$

Again a CFL condition, nonsense!

- **Amplification Factor**:

$$G = \frac{1}{1+i\dot{a}\sin\beta}$$

$$|G|^2 = \frac{1}{1+4|\dot{a}|^2\sin^2\frac{\beta}{2}\cos^2\frac{\beta}{2}} \leq 1 \quad \forall \dot{a}$$

  - Unconditionally stable
  - $G \to 0$ for $|\dot{a}| \to \infty$, but $G = 1$ for $\beta = \pm\pi$: **Undamped** cell-to-cell oscillations!

Defect-correction approach

- Consider the following scheme: $\left(1 + \dot{a}\delta\mu - \frac{1}{2}|\dot{a}|\delta^2\right)\Delta w_j^n = -\dot{a}\delta\mu w_j^n \quad \Longrightarrow \quad A_{\text{FOU}}\Delta w_j^n = \Delta w_{exp,j}^{CS,n}$
- A_{FOU} also known as **Roe-Harten implicit phase**
  - Same iteration matrix as FOU $\Longrightarrow$ **diagonal dominance** ensured
- **Amplification factor**:

$$G = \frac{1 + |\dot{a}|(1 - \cos\beta)}{1 + |\dot{a}|(1 - \cos\beta) + i\dot{a}\sin\beta}$$

$$|G|^2 = \frac{\left(1 + 2|\dot{a}|\sin^2\frac{\beta}{2}\right)^2}{\left(1 + 2|\dot{a}|\sin^2\frac{\beta}{2}\right)^2 + 4|\dot{a}|^2\sin^2\frac{\beta}{2}\cos^2\frac{\beta}{2}} = \frac{1}{1 + \frac{4|\dot{a}|^2\sin^2\frac{\beta}{2}\cos^2\frac{\beta}{2}}{\left(1+2|\dot{a}|\sin^2\frac{\beta}{2}\right)^2}} = \frac{1 + 4|\dot{a}|\sin^2\frac{\beta}{2} + 4|\dot{a}|^2\sin^4\frac{\beta}{2}}{1 + 4|\dot{a}|(1 + |\dot{a}|)\sin^2\frac{\beta}{2}} \leq 1 \quad \forall\dot{a}$$

  - Unconditionally stable; $G \to 0$ for $|\dot{a}| \to \infty$
- **Truncation error**:

$$\varepsilon = \underbrace{a\Delta t\frac{\partial^2 w}{\partial t\partial x}}_{1^{st}\text{-order error}} + \underbrace{a\frac{\Delta t^2}{2}\frac{\partial^3 w}{\partial t^2\partial x} - \frac{1}{2}|a|\Delta x\Delta t\frac{\partial^3 w}{\partial t\partial x^2}}_{2^{nd}\text{-order time errors}} + \underbrace{a\frac{\Delta x^2}{6}\frac{\partial^3 w}{\partial x^3}}_{2^{nd}\text{-order space error}} + \text{H.O.T.}$$

  - A_{FOU} introduces a first-order error that **vanishes** at steady state
  - Error is of **dissipative** nature $\Longrightarrow$ contributes to damp errors
- The Roe-Harten implicit phase can be combined with **different spatial discretizations**
  - Typically leads to **unconditionally stable** discretizations
  - Introduces numerical dissipation via the implicit phase, but dissipation at steady state depends on the **chosen spatial discretization**

Nonlinear equations

Consider the one-dimensional viscous Burgers' equation: $\boxed{\dfrac{\partial w}{\partial t} + w\dfrac{\partial w}{\partial x} = \nu \dfrac{\partial^2 w}{\partial x^2}}$

- **Conservative form**: $\dfrac{\partial w}{\partial t} + \dfrac{\partial F(w)}{\partial x} = \nu \dfrac{\partial^2 w}{\partial x^2}$ with $F(w) = \dfrac{w^2}{2}$
- Centred approximation for spatial derivatives + Backward Euler formulation:

$$\frac{\Delta w_j^n}{\Delta t} = -\frac{\delta \mu F_j^{n+1}}{\Delta x} + \frac{\nu}{2}\frac{\delta^2 w_j^{n+1}}{\Delta x^2}$$

  - This is now a **nonlinear** system of algebraic equations
  - No longer linear tridiagonal system, complication due to nonlinear flux F
  - Use **Taylor series expansions** of F at n^{th} time-level to convert to a linear system:

$$F_j^{n+1} = F_j^n + \Delta t \left.\frac{\partial F}{\partial t}\right|_j^n + \frac{\Delta t^2}{2}\left.\frac{\partial^2 F}{\partial t^2}\right|_j^n + \ldots = F_j^n + \Delta t A(w)\Delta w_j^n + \mathcal{O}(\Delta t^2), \quad \text{with} \quad A = \left.\frac{\partial F}{\partial w}\right|_j^n = w_j^n$$

$$\left(1 + \frac{\Delta t}{\Delta x}\delta\mu A - \frac{1}{2}\dot{\nu}\delta^2\right)\Delta w_j^n = -\Delta t\left(\frac{\delta\mu F_j^n}{\Delta x} + \frac{\nu}{2}\frac{\delta^2 w_j^n}{\Delta x^2}\right) \implies \text{Tridiagonal system}$$

  - Linearization introduces an **additional second-order error** in time (higher-order)
  - When looking for steady solutions, we do not care about time errors
  - Jacobian may be difficult to find for complex equations (e.g. Navier–Stokes)
  - For $\Delta t \to \infty$ we recover **Newton's method** for nonlinear systems of equations

Systems of equations

Consider the 1D Euler eqs: $\dfrac{\partial w}{\partial t} + \dfrac{\partial F}{\partial x} = 0$ with $w = \begin{bmatrix} \rho \\ \rho u \\ \rho E \end{bmatrix}$ $F = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u H \end{bmatrix}$ $A(w) = \dfrac{\partial F}{\partial w}$

- Discretize this by any **spatial approximation** scheme: $\dfrac{\partial w}{\partial t} + R(w) = 0$
  - R spatial discretization operator (nonlinear because of F)
- Apply an implicit **time discretization** (1^{st}-order B.E. sufficient for steady solutions): $\dfrac{\Delta w^n}{\Delta t} + R(w^{n+1}) = 0$
  - Nonlinear system of algebraic equations at each time step
- **Linearize w.r.t.** w: $\left[\mathcal{I} + \Delta t \left.\dfrac{\partial R}{\partial w}\right|^n\right] \Delta w^n = -\Delta t R(w^n) \implies$ Linear system with a band-blocked matrix
- For a 3-point implicit phase, one has a 3x3 block tridiagonal system ($\implies$ block Thomas algorithm)
  - Computation of the Jacobian may be **difficult and expensive**
    - Often replaced by an **approximate Jacobian**! **Example**: Roe-Harten implicit phase for systems

$$\left[\mathcal{I} + \frac{\Delta t}{\Delta x}\delta A_R^n \mu - \frac{1}{2}\frac{\Delta t}{\Delta x}\delta |A_R^n| \delta\right] \Delta w_j^n = -\Delta t R(w^n) \quad \text{with} \quad A_{R,j+\frac{1}{2}} = \text{Roe average at the interface}$$

    - May be reduced to a scalar system by replacing matrices by their spectral radius

$$\left[\mathcal{I} + \frac{\Delta t}{\Delta x}\delta \rho(A_R^n) \mu - \frac{1}{2}\frac{\Delta t}{\Delta x}\delta \rho(A_R) \delta\right] \Delta w_j^n = -\Delta t R(w^n) \quad \text{with} \quad \rho(A^n_{R,j+\frac{1}{2}}) = \text{spectral radius of } A_{R,j+\frac{1}{2}}$$

Solution convergence acceleration

Several methods for accelerating the convergence of the solution to a steady-state solution:

- Use a uniform global CFL number which results in varying local time steps. Thus larger time steps are used in regions of larger cells
- Use an incrementing CFL number that starts with a small CFL number to get past initial transients then increases the CFL number to converge
- Use multi-grid techniques: use a set of nested spatial meshes (fine-medium-coarse) and transfer information from fine grids to coarse, and back again

Solve linear systems: direct methods

Gaussian Elimination

- **Pivoting**: rearrange equations to put the largest coefficient on the main diagonal
- **Eliminate** the column below main diagonal
- **Repeat** until the last equation is reached
- **Back-substitution**:

$$\begin{array}{llll} a_{11}w_1 & + a_{12}w_2 & + \ldots = & b_1 \\ a_{21}w_1 & + a_{22}w_2 & + \ldots = & b_2 \\ \vdots & \ddots & & \vdots \\ a_{N1}w_1 & + a_{N2}w_2 & + \ldots = & b_N \end{array} \rightarrow \begin{array}{llll} a_{11}w_1 & + a_{12}w_2 & + \ldots = & b_1 \\ & + a'_{22}w_2 & + \ldots = & b_2 \\ \vdots & \ddots & & \vdots \\ & & + a_{NM}w_N = & b_N \end{array}$$

- Not vectorize/parallelize well, rarely used in CFD
- For a general $N \times N$ dense matrix: $\mathcal{O}(N^3)$ operations and $\mathcal{O}(N^2)$ memory locations
  - **Example**: Poisson problem on 100^3 grid. $N = 10^6 \implies N^2 = 10^{12}$ (8 TB needed) $N^3 = 10^{18}$ FLOP (several months on a PC)
  - More sophisticated algorithms can solve the problem in seconds

Special case: TDMA (Thomas algorithm)

Tridiagonal system of the form

$$a_j^n w_{j-1}^{n+1} + b_j^n w_j^{n+1} + c_j^n w_{j+1}^{n+1} = d_j^n$$

1. Look for a solution of the form

$$w_j^{n+1} = R_j w_{j-1}^{n+1} + T_j$$

   with R_j and T_j coefficients to be determined
2. Plug this in the tridiagonal system:

$$a_j^n w_{j-1}^{n+1} + b_j^n w_j^{n+1} + c_j^n (R_{j+1} w_j^{n+1} + T_{j+1}) = d_j^n$$

$$\implies a_j^n w_{j-1}^{n+1} + (b_j^n + c_j^n R_{j+1}) w_j^{n+1} = d_j^n - c_j^n T_{j+1}$$

   And by identification:

$$R_j = -\frac{a_j^n}{b_j^n + c_j^n R_{j+1}} \qquad T_j = \frac{d_j^n - c_j^n T_{j+1}}{b_j^n + c_j^n R_{j+1}}$$

   Backward Loop!
3. The solution is then found via forward substitution

$$w_j^{n+1} = R_j w_{j-1}^{n+1} + T_j$$

- Many other algorithms exist for sparse matrices

Fast Direct Methods

- Direct methods exist for some special cases
  - Simple domains (rectangles)
  - Simple equations (separable vars)
  - Simple BCs (periodic or zeroed values)
- Based on the FFT: $w_j = \sum_{l=1}^{N} \widehat{w}_j e^{i\frac{2\pi}{N}lj}$ and $\widehat{w}_j = \sum_{l=1}^{N} w_j e^{-i\frac{2\pi}{N}lj}$
  - Can be evaluated in $2N\log_2 N$ FLOPS (Cooley-Tukey algorithm)
  - Take the double FFT

$$\Delta w_{j,k} = (w_{j+1,k} + w_{j-1,k} + w_{j,k+1} + w_{j,k-1} - 4w_{j,k}) = b_{j,k}$$

$$= \sum_{l=1}^{N}\sum_{m=1}^{N} \widehat{w}_{l,m} e^{i\frac{2\pi}{N}(kl+jm)} \left[e^{i\frac{2\pi}{N}l} + e^{-i\frac{2\pi}{N}l} + e^{i\frac{2\pi}{N}m} + e^{-i\frac{2\pi}{N}m} - 4\right]$$

$$= \sum_{l=1}^{N}\sum_{m=1}^{N} \widehat{w}_{l,m} e^{i\frac{2\pi}{N}(kl+jm)} \left[2\cos\frac{2\pi}{N}l + 2\cos\frac{2\pi}{N}m - 4\right] = \sum_{l=1}^{N}\sum_{m=1}^{N} \widehat{b}_{l,m} e^{i\frac{2\pi}{N}(kl+jm)}$$

- Solve $\widehat{w}_{l,m} = \dfrac{\widehat{b}_{l,m}}{2\left[\cos\frac{2\pi}{N}l + \cos\frac{2\pi}{N}m - 2\right]}$ $(\star)$

Algorithm:

1. Find $\widehat{b}_{l,m}$ by FFT
2. Find $\widehat{w}_{l,m}$ as in $(\star)$
3. Find $w_{i,j}$ by inverse FFT

Iterative Methods

- In CFD, the cost of direct methods may be too high (very large matrices)
- Purpose of iteration methods: drive the residual and the iteration error to be zero
- Rapid convergence of an iterative method is key to its effectiveness

$$Ax = b \qquad Ax^m = b - r^m \qquad \varepsilon^m = x - x^m \qquad A\varepsilon^m = r^m$$

- x^m: approximate solution after m iterations
- r^m: residual
- ε^m: iteration error

Typical iterative methods

Stationary methods

1. Jacobi
2. Gauss-Seidel
3. Successive Over-Relation (SOR)

Non-stationary methods

1. Alternate Direction Implicit (ADI) method
2. Steepest descent
3. CG - Conjugate Gradient
4. PCG - Preconditioned Conjugate Gradient Method
5. BICG - BiConjugate Gradient Method
6. BICGSTAG - BiConjugate Gradient Stabilized Method
7. CGS - Conjugate Gradients Squared Method
8. GMRES - Generalized Minimum Residual Method

Iterative Methods

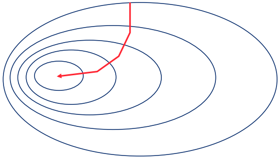
Most of these methods formulated as minimization techniques, where the following function is minimized:

$$\phi = \frac{1}{2}\vec{x}^T A\vec{x} - \vec{x}^T\vec{b} \quad \text{for which} \quad \nabla\phi(x) = Ax - b$$

The goal is finding x^* s.t.

$$\nabla\phi(x^*) = 0 \iff Ax^* - b = 0$$

- Direction and step are selected in a "best way"
- Many methods only work for symmetric systems
- Usually the system is **preconditioned** to make it better behaved



Preconditioning

If A^{-1} was known, the solution could be written in a straightforward way:

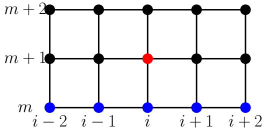
$$A\vec{x} = \vec{b} \qquad \text{with} \quad A^{-1}A = I$$
$$A^{-1}A\vec{x} = A^{-1}\vec{b}$$
$$\mathcal{I}\vec{x} = \vec{x} = A^{-1}\vec{b}$$

- Iterative methods generally converge faster if A is "close" to the identity matrix $\mathcal{I}$
- If we build a matrix M somewhat close to A^{-1}, then the systems

$$MA\vec{x} = M\vec{b} \quad \text{and} \quad A\vec{x} = \vec{b}$$

  - Have the same solution
  - The first should be easier to solve
  - M has a lower **condition number**

Jacobi Method (method of simultaneous displacements)



Jacobi method:

$$x_i^{m+1} = \frac{1}{a_{ii}}\left(b_i - \sum_{j\neq i}^{n} a_{ij}x_j^m\right), \qquad (i = 1, 2, .., n)$$

- Examples for the BTCS scheme applied to the steady 2D advection-diffusion equation:

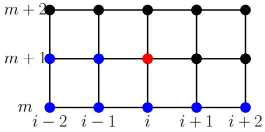
$$a\frac{\partial w}{\partial x} + b\frac{\partial w}{\partial y} - \nu\left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2}\right) = 0$$

$$a\frac{w_{j+1,k} - w_{j-1,k}}{2\Delta x} + b\frac{w_{j,k+1} - w_{j,k-1}}{2\Delta y} - \nu\frac{w_{j+1,k} - 2w_{j,k} + w_{j-1,k}}{\Delta x^2} - \nu\frac{w_{j,k+1} - 2w_{j,k} + w_{j,k-1}}{\Delta y^2} = 0$$

- Start with a guess solution $w_{j,k}^0$, then compute new approximation as:

$$a\frac{w_{j+1,k}^m - w_{j-1,k}^m}{2\Delta x} + b\frac{w_{j,k+1}^m - w_{j,k-1}^m}{2\Delta y} - \nu\frac{w_{j+1,k}^m - 2w_{j,k}^{m+1} + w_{j-1,k}^m}{\Delta x^2} - \nu\frac{w_{j,k+1}^m - 2w_{j,k}^{m+1} + w_{j,k-1}^m}{\Delta y^2} = 0$$

Gauss-Seidel method (method of successive displacements) and SOR



Gauss-Seidel method: similar to Jacobi, but most recently computed values of the $m+1$ it. are used as soon as available:

$$x_i^{m+1} = \frac{1}{a_{ii}} \left(b_i - \sum_{j<i} a_{ij} x_j^{m+1} - \sum_{j>i} a_{ij} x_j^m \right) \qquad (i = 1, 2, .., n)$$

- Examples for the BTCS scheme applied to the steady 2D advection-diffusion equation:
  - Gauss Seidel:

$$a\frac{w_{j+1,k}^m - w_{j-1,k}^m}{2\Delta x} + b\frac{w_{j,k+1}^m - w_{j,k-1}^{m+1}}{2\Delta y} - \nu\frac{w_{j+1,k}^m - 2w_{j,k}^{m+1} + w_{j-1,k}^m}{\Delta x^2} - \nu\frac{w_{j,k+1}^m - 2w_{j,k}^{m+1} + w_{j,k-1}^{m+1}}{\Delta y^2} = 0$$

  - Line Gauss-Seidel (solution of a tridiagonal system per each row or column):

$$a\frac{w_{j+1,k}^{m+1} - w_{j-1,k}^{m+1}}{2\Delta x} + b\frac{w_{j,k+1}^m - w_{j,k-1}^{m+1}}{2\Delta y} - \nu\frac{w_{j+1,k}^m - 2w_{j,k}^{m+1} + w_{j-1,k}^m}{\Delta x^2} - \nu\frac{w_{j,k+1}^m - 2w_{j,k}^{m+1} + w_{j,k-1}^{m+1}}{\Delta y^2} = 0$$

- Increasing the number of updated point speeds-up convergence

Successive Over-Relaxation: the updated solution is weighted by a relaxation factor

$$x_i^{m+1} = (1-\omega)x_i^m + \frac{\omega}{a_{ii}} \left(b_i - \sum_{j<i} a_{ij} x_j^{m+1} - \sum_{j>i} a_{ij} x_j^m \right)$$

$$(i = 1, 2, .., n)$$

- Speed-up convergence for stable problems
- Stabilize to some extent unstable problems
- It should be $0 < \omega < 2$
  (1.5 is a good starting value)

Stationary Methods

Consider the 2D Poisson equation (with $\Delta x = \Delta y = \Delta h$) :

$$\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} = s \quad \Longrightarrow \quad \frac{w_{i+1,j} - 2w_{i,j} + w_{i+1,j}}{\Delta x^2} + \frac{w_{i,j+1} - 2w_{i,j} + w_{i,j-1}}{\Delta y^2} = s_{i,j}$$

Jacobi $\quad w_{i,j}^{n+1} = \frac{1}{4}\left[w_{i+1,j}^{n} + w_{i-1,j}^{n} + w_{i,j+1}^{n} + w_{i,j-1}^{n} - \Delta h^2 s_{i,j}\right]$

Gauss-Seidel $\quad w_{i,j}^{n+1} = \frac{1}{4}\left[w_{i+1,j}^{n} + w_{i-1,j}^{n+1} + w_{i,j+1}^{n} + w_{i,j-1}^{n+1} - \Delta h^2 s_{i,j}\right]$

SOR $\quad w_{i,j}^{n+1} = \frac{\omega}{4}\left[w_{i+1,j}^{n} + w_{i-1,j}^{n+1} + w_{i,j+1}^{n} + w_{i,j-1}^{n+1} - \Delta h^2 s_{i,j}\right] + (1-\omega)w_{i,j}^{n}$

- Iterations carried out until the solution is sufficiently accurate
  - Define $R_{i,j} = \frac{w_{i+1,j} + w_{i-1,j} + w_{i,j+1} + w_{i,j-1} - 4w_{i,j}}{\Delta h^2} - s_{i,j} \quad \Longrightarrow \quad R_{i,j} = 0$ at steady-state
- What about **boundary conditions**?
  - **Dirichlet**: easily implemented
  - **Neumann**: The simplest approach is $\frac{\partial w}{\partial y} = 0 \Longrightarrow w_{i,0} - w_{i,1} = 0$ (1st order)
    - Update interior points $w_{i,1}, w_{i,2}, w_{i,3}$.. and then set $w_{i,0} = w_{i,1} \Longrightarrow$ this generally **does not converge**
    - Instead, try to **incorporate** BCs directly into the equations:

$$w_{i,1} = \frac{1}{4}\left[w_{i-1,1} + w_{i+1,1} + w_{i,2} + \overbrace{w_{i,0}}^{=w_{i,1}} - \Delta h^2 s_{i,j}\right] = \frac{1}{3}\left[w_{i-1,1} + w_{i+1,1} + w_{i,2} - \Delta h^2 s_{i,j}\right]$$

Iteration versus time integration

Iterative methods can sometimes be viewed as **integration in time**. **Example** for Jacobi:

The solution of $\boxed{\dfrac{\partial^2 w}{\partial x^2} + \dfrac{\partial^2 w}{\partial y^2} = 0}$ can be seen as the steady-state solution of $\boxed{\dfrac{\partial w}{\partial t} + \dfrac{\partial^2 w}{\partial x^2} + \dfrac{\partial^2 w}{\partial y^2} = 0}$

- Discretize with FTCS:

$$\frac{w_{i,j}^{n+1} - w_{i,j}^{n}}{\Delta t} = \frac{w_{i+1,j}^{n} + w_{i+1,j}^{n} + w_{i,j+1}^{n} + w_{i,j-1}^{n} - 4w_{i,j}^{n}}{\Delta h^2}$$

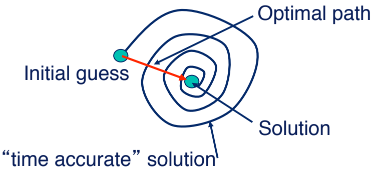
$$w_{i,j}^{n+1} = w_{i,j}^{n} + \frac{\Delta t}{\Delta h^2}\left(w_{i+1,j}^{n} + w_{i+1,j}^{n} + w_{i,j+1}^{n} + w_{i,j-1}^{n} - 4w_{i,j}^{n}\right)$$

$$w_{i,j}^{n+1} = \left[1 - \frac{4\Delta t}{\Delta h^2}\right] w_{i,j}^{n} + \frac{\Delta t}{\Delta h^2}\left(w_{i+1,j}^{n} + w_{i+1,j}^{n} + w_{i,j+1}^{n} + w_{i,j-1}^{n}\right)$$

- Select the maximum timestep: $\dfrac{\Delta t}{\Delta h^2} = \dfrac{1}{4}$:

$$w_{i,j}^{n+1} = \frac{1}{4}\left(w_{i+1,j}^{n} + w_{i+1,j}^{n} + w_{i,j+1}^{n} + w_{i,j-1}^{n}\right)$$

  - This is **exactly** the Jacobi iteration!



The fact that we are only interested in steady-state solution allows us to take “**short-cuts**” to get there as fast as possible

Multigrid methods (I)

- Commonly used for elliptic eqs
- Consider the 1D equation $\boxed{\dfrac{\partial^2 w}{\partial x^2} = s}$
- Search for the steady-state solution of $\dfrac{\partial w}{\partial t} = \nu \left(\dfrac{\partial^2 w}{\partial x^2} - s \right)$
  - Use **Fourier series** for analytic solution:

$$w = \sum_k a_k(t) e^{ikx} \qquad s = \sum_k b_k k^2 e^{ikx}$$

$$\sum_k \frac{da_k}{dt} e^{ikx} = -\nu \sum_k [a_k(t) - b_k] k^2 e^{ikx}$$

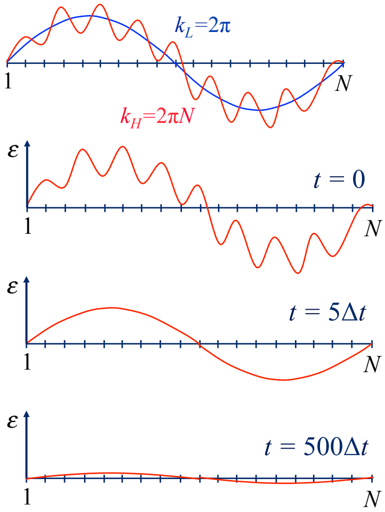
  - Solving for each k:

$$\frac{da_k}{dt} = -\nu k^2 [a_k(t) - b_k]$$

$$a_k(t) - b_k = [a_k(0) - b_k] e^{-\nu k^2 t} \quad (\varepsilon_k = \varepsilon_0 e^{-\nu k^2 t})$$

Therefore, $a_k(t \to \infty) = b_k$

Rate of convergence $\sim \nu k^2$



High wave number modes damps out faster!

Multigrid methods (II)

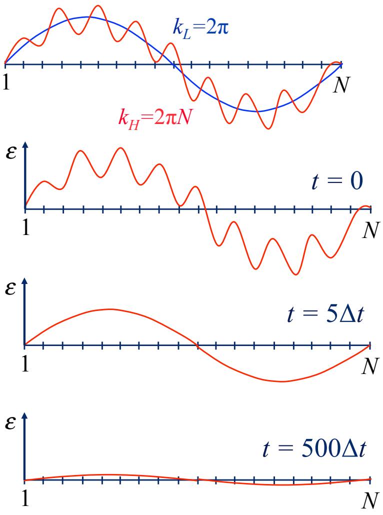
- For explicit time step, stability condition yields:

$$\frac{\nu \Delta t}{\Delta x^2} = \frac{1}{2} \quad \Longrightarrow \quad \Delta t = \frac{\Delta x^2}{2\nu}$$

- Error for a given k at time $t = n\Delta t = \dfrac{n\Delta x^2}{2\nu}$

$$\frac{\varepsilon}{\varepsilon_0} = e^{-\nu k^2 t} = e^{-\frac{k^2 n \Delta x^2}{2}} \Longrightarrow \begin{cases} \dfrac{\varepsilon_H}{\varepsilon_0} = \exp(-n\pi^2 N^2 \Delta x^2) & \text{for } k_H = 2\pi N \\ \dfrac{\varepsilon_L}{\varepsilon_0} = \exp(-n\pi^2 \Delta x^2) & \text{for } k_L = 2\pi \end{cases}$$

Short-wave errors decay much faster



High wave number modes damps out faster!

Idea behind Multigrid methods:

- A low wave number components on a fine grid becomes a high wave number component on a coarse grid, hence:
  - Use a corse grid to converge rapidly low-k components of solution
  - Map it onto the fine grid system to converge high-k components

2D Methods

2D heat equation: $$\frac{\partial w}{\partial t} = \nu\left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2}\right)$$

Apply FTCS: $$\frac{w_{i,j}^{n+1} - w_{i,j}^{n}}{\Delta t} = \nu\left[\frac{w_{i+1,j}^{n} - 2w_{i,j}^{n} + w_{i-1,j}^{n}}{\Delta x^2} + \frac{w_{i,j+1}^{n} - 2w_{i,j}^{n} + w_{i,j-1}^{n}}{\Delta y^2}\right]$$

if $\Delta x = \Delta y = \Delta h$: $$\frac{w_{i,j}^{n+1} - w_{i,j}^{n}}{\Delta t} = \frac{\nu}{\Delta h^2}\left[w_{i+1,j}^{n} + w_{i-1,j}^{n} + w_{i,j+1}^{n} + w_{i,j-1}^{n} - 4w_{i,j}^{n}\right]$$

$$\begin{bmatrix} w_{1,1}^{n+1} \\ w_{1,2}^{n+1} \\ \vdots \\ w_{1,J}^{n+1} \\ w_{2,1}^{n+1} \\ w_{2,2}^{n+1} \\ \vdots \\ \vdots \\ w_{I,J-1}^{n+1} \\ w_{I,J}^{n+1} \end{bmatrix} = \begin{bmatrix} w_{1,1}^{n} \\ w_{1,2}^{n} \\ \vdots \\ w_{1,J}^{n} \\ w_{2,1}^{n} \\ w_{2,2}^{n} \\ \vdots \\ \vdots \\ w_{I,J-1}^{n} \\ w_{I,J}^{n} \end{bmatrix} + \frac{\nu\Delta t}{\Delta h^2} \cdot \begin{bmatrix} -4 & 1 & 0 & 0 & \dots & 1 & 0 & \dots & \dots & \dots \\ 1 & -4 & 1 & 0 & \dots & 0 & 1 & \dots & \dots & \dots \\ 0 & 1 & -4 & 1 & \dots & \dots & 0 & 1 & \dots & \dots \\ \vdots & & & & & \vdots & & & & \vdots \\ 1 & 0 & & & & & & & & \\ 0 & 1 & & & & & & & & \\ 0 & 0 & 1 & & & & & & & \\ & & & & & & & 0 & 1 & -4 \end{bmatrix} \cdot \begin{bmatrix} w_{1,1}^{n} \\ w_{1,2}^{n} \\ \vdots \\ w_{1,J}^{n} \\ w_{2,1}^{n} \\ w_{2,2}^{n} \\ \vdots \\ \vdots \\ w_{I,J-1}^{n} \\ w_{I,J}^{n} \end{bmatrix}$$

- By considering BTCS (unconditionally stable), set of linear eqs to be solved at each iteration
  - If small Δt for accuracy, explicit method is competitive. Otherwise Jacobi, Gauss-Seidel, SOR..

Von Neumann Analysis

$$\widehat{w}_{i,j}^{n} = \widehat{w}^{n} e^{ikx} e^{imy} = \widehat{w}^{n} e^{ikh} e^{imh}$$

$$\widehat{w}^{n+1} = \widehat{w}^{n} + \frac{\nu \Delta t}{\Delta h^2} \left(e^{ik\Delta h} + e^{-ik\Delta h} + e^{im\Delta h} + e^{-im\Delta h} - 4 \right) \widehat{w}^{n}$$

$$\Longrightarrow \quad \frac{\widehat{w}^{n+1}}{\widehat{w}^{n}} = 1 + \frac{\nu \Delta t}{\Delta h^2} (2\cos k\Delta h + 2\cos m\Delta h - 4) = 1 + \frac{\nu \Delta t}{\Delta h^2} (2\cos\beta + 2\cos\xi - 4)$$

$$= 1 - \frac{4\nu \Delta t}{\Delta h^2} \left(\sin^2\frac{\beta}{2} + \sin^2\frac{\xi}{2} \right)$$

Worst case:

$$\frac{\widehat{w}^{n+1}}{\widehat{w}^{n}} = 1 - \frac{4\nu \Delta t}{\Delta h^2}(1+1)$$

$$\Longrightarrow \boxed{\frac{\nu \Delta t}{\Delta h^2} = \dot{\nu} \leq \frac{1}{4}}$$

- **Stability limits depend on dimensions!**

- **1D Case**:

$$\frac{\nu \Delta t}{\Delta x^2} \leq \frac{1}{2} \quad \text{and} \quad \frac{u^2 \Delta t}{\nu} \leq 2$$

- **2D Case**:

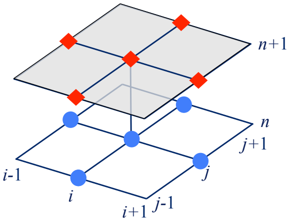
$$\frac{\nu \Delta t}{\Delta x^2} \leq \frac{1}{4} \quad \text{and} \quad \frac{(|u| + |v|)^2 \Delta t}{\nu} \leq 4$$

- **3D Case**:

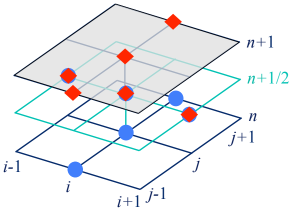
$$\frac{\nu \Delta t}{\Delta x^2} \leq \frac{1}{6} \quad \text{and} \quad \frac{(|u| + |v| + |w|)^2 \Delta t}{\nu} \leq 8$$

ADI Methods

2D Crank-Nicolson



ADI Method



- 2$^{\text{nd}}$-order in time may be achieved with **Crank-Nicolson**

$$w_{i,j}^{n+1} = w_{i,j}^{n} + \frac{\dot{\nu}}{2}\left(w_{i+1,j}^{n+1} + w_{i-1,j}^{n+1} + w_{i,j+1}^{n+1} + w_{i,j-1}^{n+1} - 4w_{i,j}^{n+1}\right) + \frac{\dot{\nu}}{2}\left(w_{i+1,j}^{n} + w_{i-1,j}^{n} + w_{i,j+1}^{n} + w_{i,j-1}^{n} - 4w_{i,j}^{n}\right)$$

  - Matrix equation **expensive** to solve
  - Can larger time-steps be achieved **without** having to solve it?

- **Breakthrough**: **Alternating-Direction-Implicit (ADI)** method

$$\text{(I):}\quad w_{i,j}^{n+\frac{1}{2}} - w_{i,j}^{n} = \frac{\dot{\nu}}{2}\left[\left(w_{i+1,j}^{n+\frac{1}{2}} - 2w_{i,j}^{n+\frac{1}{2}} + w_{i-1,j}^{n+\frac{1}{2}}\right) + \left(w_{i,j+1}^{n} - 2w_{i,j}^{n} + w_{i,j-1}^{n}\right)\right]$$

$$\text{(II):}\quad w_{i,j}^{n+1} - w_{i,j}^{n+\frac{1}{2}} = \frac{\dot{\nu}}{2}\left[\left(w_{i+1,j}^{n+\frac{1}{2}} - 2w_{i,j}^{n+\frac{1}{2}} + w_{i-1,j}^{n+\frac{1}{2}}\right) + \left(w_{i,j+1}^{n+1} - 2w_{i,j}^{n+1} + w_{i,j-1}^{n+1}\right)\right]$$

  - Treat one row implicitly, then reverse roles and treat the other (with BTCS)
  - Instead of solving one set of linear equations for the 2D system, solve 1D equations for each grid line with an efficient tridiagonal algorithm
  - Unconditionally stable in 2D

Approximate Factorization Splitting

$$\delta_x^2(\bullet) = \frac{(\bullet)_{i-1,j} - 2(\bullet)_{i,j} + (\bullet)_{i+1,j}}{\Delta x^2} \qquad \delta_y^2(\bullet) = \frac{(\bullet)_{i,j-1} - 2(\bullet)_{i,j} + (\bullet)_{i,j+1}}{\Delta y^2}$$

By using the δ operators, rewrite the **2D Crank-Nicolson** scheme:

$$\frac{w^{n+1} - w^n}{\Delta t} = \frac{\nu}{2}\delta_x^2(w^{n+1} + w^n) + \frac{\nu}{2}\delta_y^2(w^{n+1} + w^n) + \mathcal{O}(\Delta t^2, \Delta x^2, \Delta y^2)$$

$$\left[1 - \frac{\nu\Delta t}{2}\delta_x^2 - \frac{\nu\Delta t}{2}\delta_y^2\right] w^{n+1} = \left[1 + \frac{\nu\Delta t}{2}\delta_x^2 + \frac{\nu\Delta t}{2}\delta_y^2\right] w^n + \Delta t\mathcal{O}(\Delta t^2, \Delta x^2, \Delta y^2)$$

$$\left[1 - \frac{\nu\Delta t}{2}\delta_x^2\right]\left[1 - \frac{\nu\Delta t}{2}\delta_y^2\right] w^{n+1} - \frac{\nu^2\Delta t^2}{4}\delta_x^2\delta_y^2 w^{n+1} = \left[1 + \frac{\nu\Delta t}{2}\delta_x^2\right]\left[1 + \frac{\nu\Delta t}{2}\delta_y^2\right] w^n - \frac{\nu^2\Delta t^2}{4}\delta_x^2\delta_y^2 w^n + \Delta t\mathcal{O}(\Delta t^2, \Delta x^2, \Delta y^2)$$

$$\left[1 - \frac{\nu\Delta t}{2}\delta_x^2\right]\left[1 - \frac{\nu\Delta t}{2}\delta_y^2\right] w^{n+1} = \left[1 + \frac{\nu\Delta t}{2}\delta_x^2\right]\left[1 + \frac{\nu\Delta t}{2}\delta_y^2\right] w^n + \frac{\nu^2\Delta t^2}{4}\delta_x^2\delta_y^2(w^{n+1} - w^n) + \Delta t\mathcal{O}(\Delta t^2, \Delta x^2, \Delta y^2)$$

ADI Method can be written as

$$\left[1 - \frac{\nu\Delta t}{2}\delta_x^2\right] w^{n+\frac{1}{2}} = \left[1 + \frac{\nu\Delta t}{2}\delta_y^2\right] w^n$$

$$\left[1 - \frac{\nu\Delta t}{2}\delta_y^2\right] w^{n+1} = \left[1 + \frac{\nu\Delta t}{2}\delta_x^2\right] w^{n+\frac{1}{2}}$$

- Eliminating $w^{n+\frac{1}{2}}$, it is shown that **ADI is an approximate factorization of Crank-Nicolson**, up to the red factor
- Several other splitting proposed in literature. **Why splitting?**
  - Stability limits of 1D cases apply
  - Different Δt can be used in different directions

ADI / Approximate Factoring

For multi-D systems, the Jacobian matrix is three-dimensional: $\dfrac{\partial R}{\partial w} = J = J_x + J_y + J_z$.

The linearized implicit method gives $[\mathcal{I} - \Delta t(J_x + J_y + J_z)]\,\Delta w = -R(w^n)$

- No longer tridiagonal matrix $\implies$ expensive to solve!
- $\implies$ put the system into a **factored** form to allow a series of 3 1D solutions:

$$[\mathcal{I} - \Delta t J_x][\mathcal{I} - \Delta t J_y][\mathcal{I} - \Delta t J_z]\Delta w = -R(w^n) \iff \begin{cases} [\mathcal{I} - \Delta t J_x]X = -R(w^n) & \to X \\ [\mathcal{I} - \Delta t J_y]Y = X & \to Y \\ [\mathcal{I} - \Delta t J_z]\Delta w = Y & \to \Delta w \end{cases}$$

A simple factorized implicit scheme for viscous problems is, e.g.:

$$\left(\mathcal{I} + \frac{\Delta t}{\Delta x}\delta A_R \mu - \frac{1}{2}\frac{\Delta t}{\Delta x}\delta |A_R| \delta - \frac{\Delta t}{\Delta x^2}\delta A^v \delta\right) \times$$
$$\left(\mathcal{I} + \frac{\Delta t}{\Delta y}\delta B_R \mu - \frac{1}{2}\frac{\Delta t}{\Delta y}\delta |B_R| \delta - \frac{\Delta t}{\Delta y^2}\delta B^v \delta\right) \times$$
$$\left(\mathcal{I} + \frac{\Delta t}{\Delta z}\delta C_R \mu - \frac{1}{2}\frac{\Delta t}{\Delta z}\delta |C_R| \delta - \frac{\Delta t}{\Delta z^2}\delta C^v \delta\right) \times \Delta w = -\Delta t[R(w) - R^v(w)]$$

with

- A_R, B_R, C_R: Roe averages
- A^v, B^v, C^v: viscous flux Jacobians
- **Matrix-free** methods obtained replacing the Jacobians by their spectral radii

- These techniques speed-up convergence toward the steady state, but destroy time-accuracy
- Moreover, implicit methods are sometimes too costly for unsteady problems
- Can they be adapted to **unsteady problems**? How to achieve **high-accuracy** in time?