

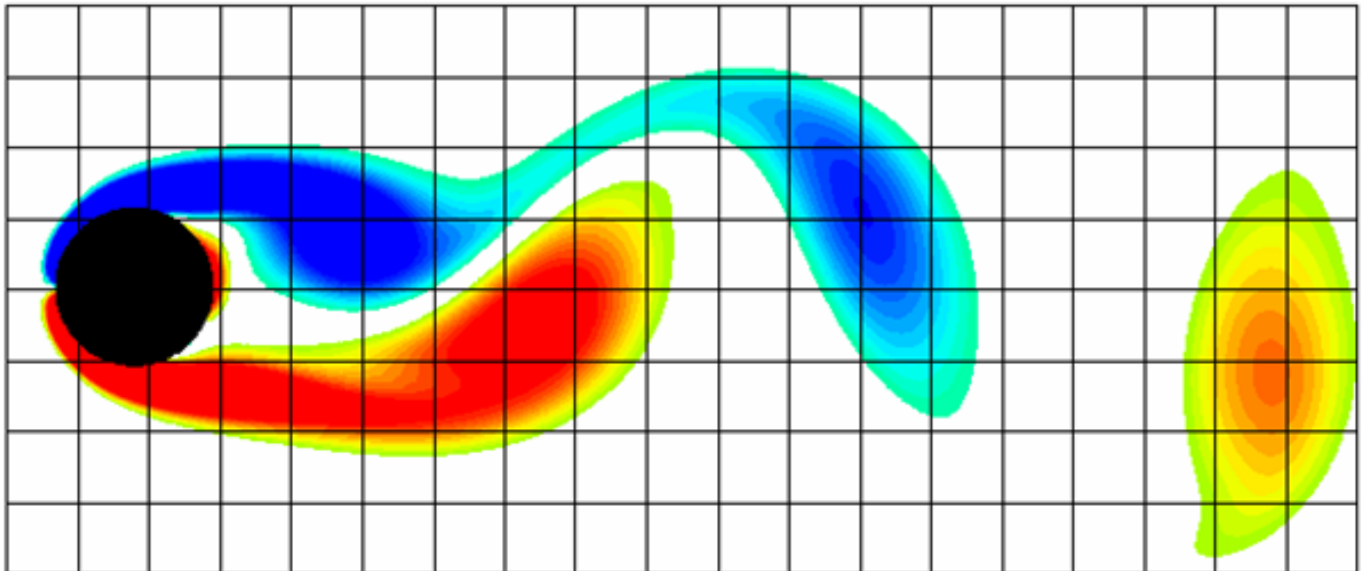


**UNIVERSITÉ
DE LORRAINE**



COMPUTATIONAL FLUID DYNAMICS

Lecture Notes - Master SRE



Olivier BOTELLA

LEMTA - Université de Lorraine - CNRS
2 avenue de la Forêt de Haye, TSA 60604
54518 Vandoeuvre (France)

Email : Olivier.Botella@univ-lorraine.fr

Lectures web page : <http://arche.univ-lorraine.fr/course/view.php?id=9396>

Lectures on Computational Fluid Dynamics (CFD)

Instructor : Olivier Botella

Master SRE, KazNU, March 2016

I. Basics of CFD and the finite-difference method (FDM) (4 days -16 h)

- 1) Day 1 : Introduction to CFD analysis - Partial Differential Equations (PDEs) for fluid flow problems - Finite Difference Method (FDM) (4h lecture)
- 2) Day 2: FDM for unsteady PDEs I: the heat equation. (2h lecture – 2h exercises)
- 3) Day 3: FDM for steady PDES: the convection-diffusion equation (1h lecture- 3h exercises)
- 4) Day 4: FDM for unsteady PDEs II: the transport equation. (2h lecture – 2h exercises)

II. Finite-volume method (FVM) for CFD (3 days- 12h)

- 1) Day 5: FVM for unsteady convection-diffusion equation. (2h lecture – 2h exercises)
- 2) Day 6: Meshes for CFD. FVM on unstructured meshes. (2h lecture – 2h exercises)
- 3) Day7: Solution of the Navier-Stokes equations. Introduction CFD analysis with ANSYS FLUENT software. (4h Lecture)

III. CFD lab exercises with ANSYS FLUENT. (3 days – 12h)

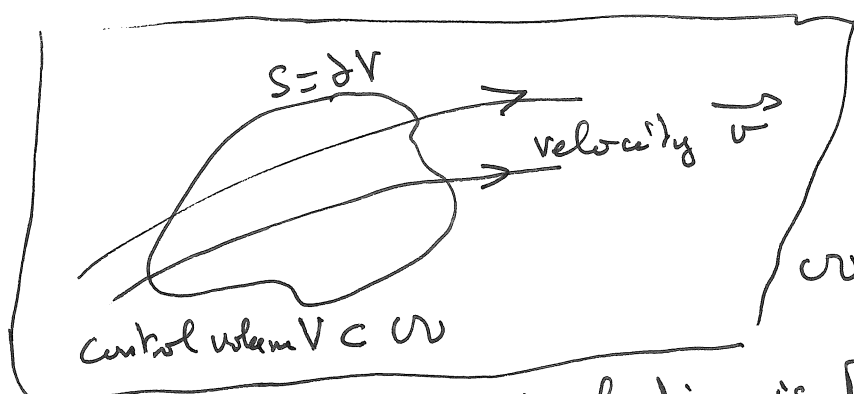
- 1) Day 8: Expansion flow in a pipe I: geometry & grid generation. (4h on PC)
- 2) Day 9: Expansion flow in a pipe II: flow computation in laminar and turbulent regimes (4h on PC).
- 3) Day 10: Flow pas circular cylinder (4h on PC).

IV. Day 11 : Written examination (2h).

(1)

Basics on Partial Differential Equations (PDEs)

I) Balance equations in Continuum Mechanics



U is the physical domain, its frontier is P or ∂U .

Let us the conservation law for a quantity $\Phi(x, t)$ inside a control volume V (e.g. $\Phi = T$, the momentum) is:

$$\text{rate of change of } \Phi \text{ inside } V = \text{Net fluxes at surface } S + \text{Sources or sinks inside } V$$

which has the mathematical form:

$$(1) \quad \frac{d}{dt} \int_V \rho \Phi dV = \underbrace{\int_S \vec{T} \cdot \vec{n} dS}_{I_1} + \underbrace{\int_V \rho q dV}_{I_2}$$

I_1 : Flux across S , due to surface stresses or pressure acting on V , represented by the stress vector $\vec{T}$

I_2 : production or dissipation inside V , due to external forces (e.g. gravity...) represented by q .

Theorem 1 (Reynolds Transport theorem)

(2)

$$(2) \forall V \subset \mathcal{V}, \quad \frac{d}{dt} \int_V \phi \, dV = \int_V \left(\frac{\partial \phi}{\partial t} + \nabla \cdot (\phi \vec{v}) \right) dV$$

Theorem 2 (Gauss's Divergence Theorem)

$$(3) \forall \text{ vector } \vec{u}, \quad \int_V \nabla \cdot \vec{u} \, dV = \int_S \vec{u} \cdot \vec{n} \, dS$$

By using Th. 1, Eq (4) ~~reads~~ leads to the integral form of the balance equation for Φ :

$$\int_V \frac{\partial \phi}{\partial t} \, dV + \int_S (\phi \vec{v} - \vec{t}) \cdot \vec{n} \, dS = \int_V q \, dV \quad (4).$$

After using ~~the~~ Th. 2 to transform surface integrals into volume integrals, we obtain the local form of the balance equation, ~~expressed~~

$$\forall \underline{x} \in \mathcal{V}, \quad \frac{\partial \phi}{\partial t} + \nabla \cdot (\phi \vec{v} - \vec{t}) = q \quad (5)$$

The balance eq's (4) or (5) represents the generic ^{if $\phi = T$ the eq. is for} equation of continuum mechanics.

Ex 1: (Fourier law for temperature). For heat transfer problems, $\vec{t} = \frac{k}{c_p} \nabla T$ where k is the thermal conductivity, c_p is the specific heat capacity. Eq (5) with $q=0$

yields the heat equation:

$$c_p \left(\frac{\partial T}{\partial t} + \nabla \cdot (\vec{v} T) \right) = k \nabla^2 T.$$

Example 2: (Answer of Mars) if we let $\phi = 1$, $\vec{F} = 0$ and (3)
 $\vec{q}$, Eq. (5) leads to the continuity eqn of fluid mechanics

$$\frac{\partial e}{\partial t} + \nabla \cdot (e \vec{v}) = 0.$$

II) A generic equation: the convec-diffusion equation:

As shown in the previous arch, all the balance laws of continuum mechanics have the same mathematical structure. It is then possible to devise build universal numerical methods to solve Eq. (4) or (5).

- The local form (Eq. (5)) is a partial differential eqn, and is mostly use in the finite-difference method (FDM)
- The Integral form (Eq. (4)) is used in the finite-volume method (FVM) as prototype eqn

We will focus on the FDM first, and consider the prototype equation convec diffusion equation:
 multidimensional

$$\frac{\partial e \phi}{\partial t} + \underbrace{\nabla \cdot (e \phi \vec{v})}_{\text{convection}} - \underbrace{\nabla \cdot (\gamma \nabla \phi)}_{\text{diffusion}} = q \quad (6)$$

which correspond to Eq (5) with Fick's law $\vec{F} = \gamma \nabla \phi$
 where $\gamma > 0$ is the viscosity. ~~Then I compare to the~~
~~transport of ϕ through convection~~ A 1D version of these
 eqn is obtained by letting $e = 1$, $\phi = u(x, t)$,
 $\vec{v} = a$ the convec velocity of arbitrary sign:

$$\frac{\partial u}{\partial t} + a \underbrace{\frac{\partial u}{\partial x}}_{\text{convec}} - \gamma \underbrace{\frac{\partial^2 u}{\partial x^2}}_{\text{diffus.}} = q. \quad (7)$$

Our presentation of the FOM will focus on this 1D eq. (4) which contains all the characteristics & basic properties of certain phenomena of astrium mechanics.

III) Fourier Analysis of the 1D wave equation

Let us find the solution of Eq. (7) with $q=0$ as a Fourier mode:

$$u(x, t) = \hat{u} e^{ikhx} \quad (i^2 = -1) \quad (8)$$

Eq. (7) becomes leads to the ordinary diff. eq (ODE):

$$\frac{d\hat{u}}{dt} + iah\hat{u} + \nu k^2 \hat{u} = 0$$

which is easily solved and leads to:

$$u(x, t) = \hat{u}_0 \underbrace{e^{-\nu k^2 t}}_I \underbrace{e^{i h(x-at)}}_{II} \quad (9)$$

where $\hat{u}_0$ depends on the initial condition.

• I: is a dissipation term, which is strictly decaying in time since $\nu > 0$.

• II: is a propagation term.

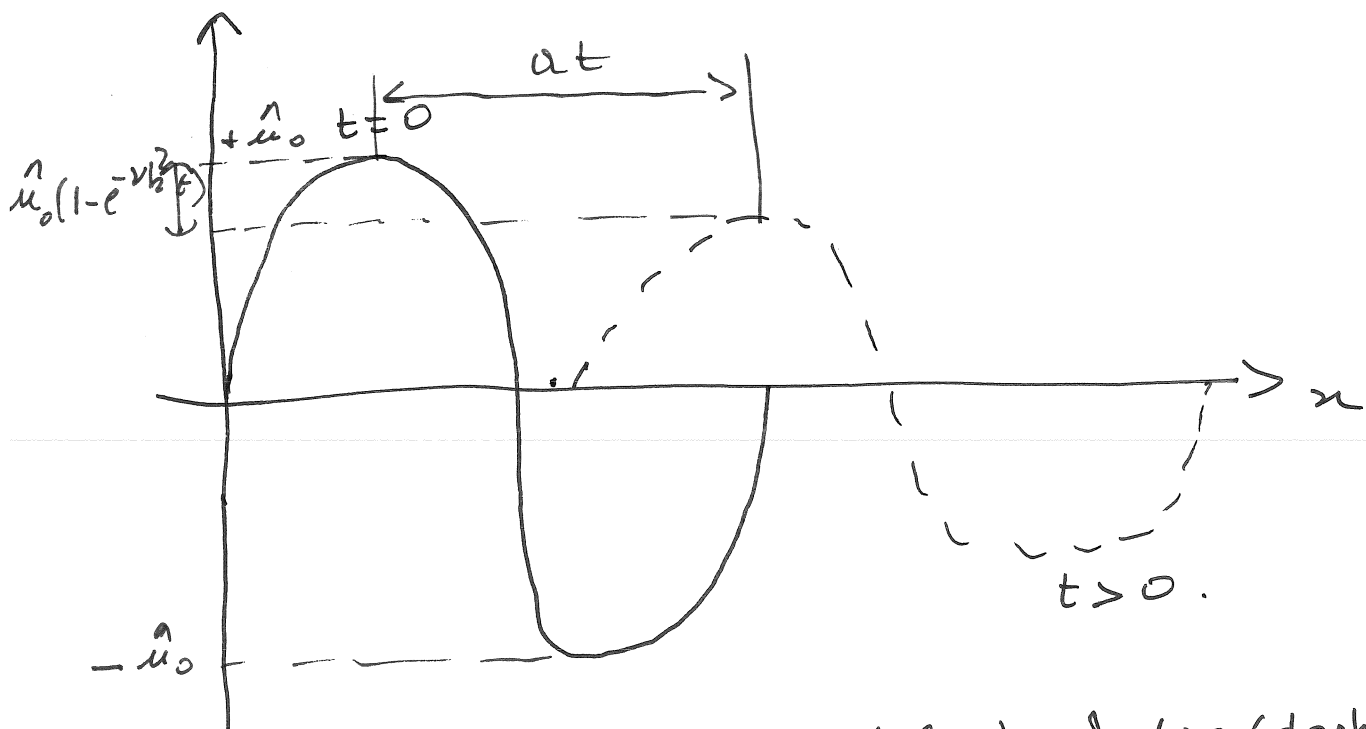


Fig. 1 Fourier solution at $t=0$ (solid line) and $t>0$ (dashed) for the case $\nu > 0$.

A sketch of the solution is given in Fig. 1. 2 particular cases are noteworthy:

(5)

Case 1: When there is no convection velocity ($\alpha = 0$), Eq. (9) reads:

$$\frac{\partial u}{\partial t} - \nu \frac{\partial^2 u}{\partial x^2} = 0 \quad (10)$$

whose solution (9) reads:

$$u(x, t) = u_0(t) e^{-\nu k^2 t}$$

The principal character of the solution is the exponential damping due to viscosity $\lim_{t \rightarrow \infty} u = 0$.

Eq (10) is known as the 1D heat equation, and is the prototype of parabolic equations.

Case 2: In the case of vanishing viscosity ($\nu = 0$) and $\alpha \neq 0$, Eq. (7) becomes:

$$\frac{\partial u}{\partial t} + \alpha \frac{\partial u}{\partial x} = 0 \quad (11)$$

whose solution reads:

$$u(x, t) = u_0 e^{-ik(x-\alpha t)} \quad (12),$$

whose solution is a wave which translates at constant speed α , without any damping.

Eq (11) is known as the 1D transport equation, and is the prototype of hyperbolic equations.

2.7 Question 7

Nous allons observer le déplacement du polluant en fonction de différentes valeurs de Δt . Dans un premier temps, nous allons choisir $\Delta t = 0.0025$.

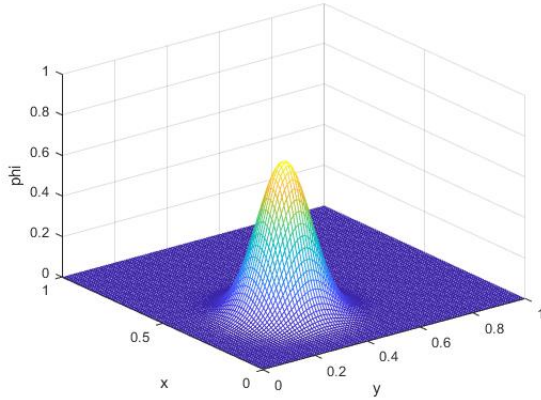


FIGURE 2.3 – Solution numérique pour $\Delta t = 0.0025$ au début du déplacement du polluant

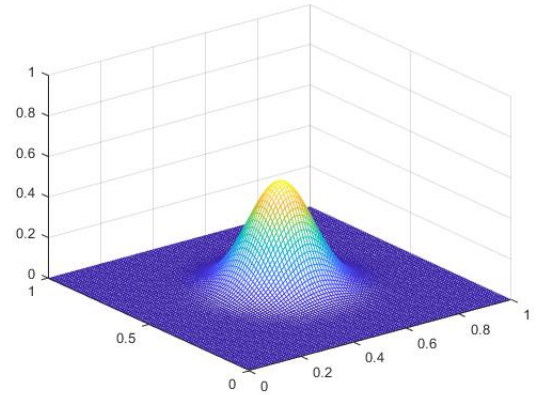


FIGURE 2.4 – Solution numérique pour $\Delta t = 0.0025$ pendant le déplacement du polluant

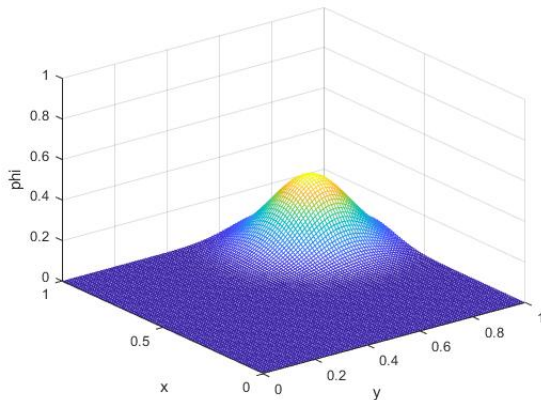


FIGURE 2.5 – Solution numérique pour $\Delta t = 0.0025$ à la fin du déplacement du polluant

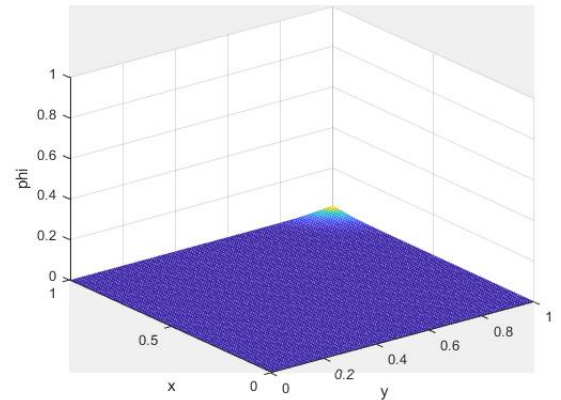


FIGURE 2.6 – Le polluant est sorti du domaine

Nous voyons que le polluant se déplace jusqu'à sortir du domaine. Maintenant, nous allons observer la solution numérique pour un pas de temps Δt plus faible que la condition de stabilité. Nous allons choisir $\Delta t = 0.0011$. Voici notre observation :

IV) ^{la and on} Classification of PDEs

⑥

In this course, a PDE will be often represent as:

$$\underline{A} u = f$$

where $\underline{A}$ is a given partial differential operator (e.g. $\underline{A} \cdot A = \frac{\partial}{\partial t} - \nabla^2$) and the relation $u(x, t)$ is $u(\underline{x}, t)$ where:

$t \in [0, T]$: the time domain

$\underline{x} \in \Omega \subset \mathbb{R}^2 \text{ or } \mathbb{R}^3$: the spatial domain.

IV-1) Classification of PDEs

Typical PDEs in continuous mechanics can be classified as in 3 groups: parabolic, ~~ell~~ hyperbolic and elliptic. The solution of each of these classes ~~posseds~~ group of PDEs possess common characterical properties, and so different numerical methods are needed for each class.

Physical problems which are time-dependent are represented by parabolic or hyperbolic equations, e.g.

• The heat equation

$$\frac{\partial u}{\partial t} - \nu \nabla^2 u = f$$

décroissance exponentielle de la chaleur due à la viscosité

is representative of parabolic ~~mat~~ problems.

• The wave and the transport equations:

$$\frac{\partial^2 u}{\partial t^2} - c^2 \frac{\partial^2 u}{\partial x^2}, \quad \frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} = 0$$

Solution est une onde se propageant à la vitesse c.

are representative of hyperbolic equations.

~~Physical problems~~ Physical problems which ~~sho~~ express a state of equilibrium are given by elliptic equations, such as the Poisson equation:

$$-\nabla^2 u = f.$$

IV-1) Boundary values (BCs)

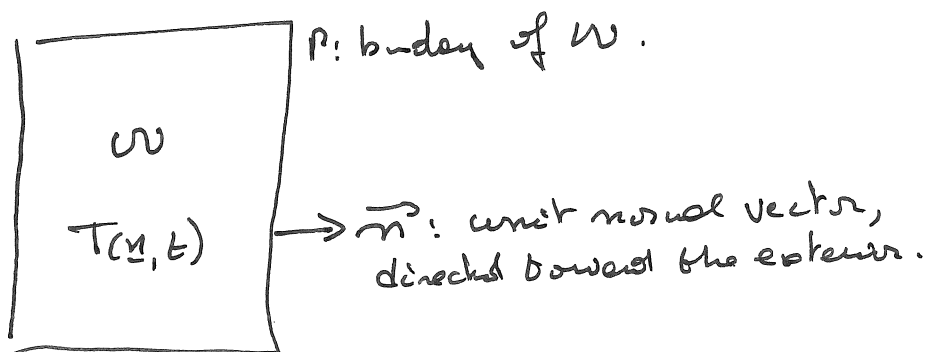
(4)

To be sure the soln is ^{at all points} unique, we must impose values of $u(x, t)$ at the boundaries. There are essentially 2 ^{mathematical} types of ~~boundary conditions~~ BCs

- Neumann BCs
- Dirichlet BCs.

Let's take as example, a vacuum flask (thermos)
the temperature distribution

heat flux:
 $q(x, t) = k \vec{\nabla} T$



1) Thermos is insulated (adiabatic condition)
We express the heat flux ~~towards~~ across P is zero:

$$\vec{q} \cdot \vec{n} \Big|_P = 0 \iff \frac{\partial T}{\partial \vec{n}} \Big|_P = 0.$$

The BC involves first order derivative of the soln at the boundary $\Rightarrow$ Neumann BC.

2) Temperature imposed at the boundary.

$$T \Big|_P = T_{out} \quad (\text{where } T_{out} \text{ is the outside temperature})$$

The BC involves the soln itself $\Rightarrow$ Dirichlet BC.

IV-3) Initial Conditions:

⑧

For time-dependent ~~solution~~ PDEs, the solution at the initial time must be provided (as for Cauchy problems for ODEs). For example:

$$u(t=0, x) = u_0(x), \quad \forall x \in \mathcal{D}.$$

Basics of the Finite-difference Method (FDM)

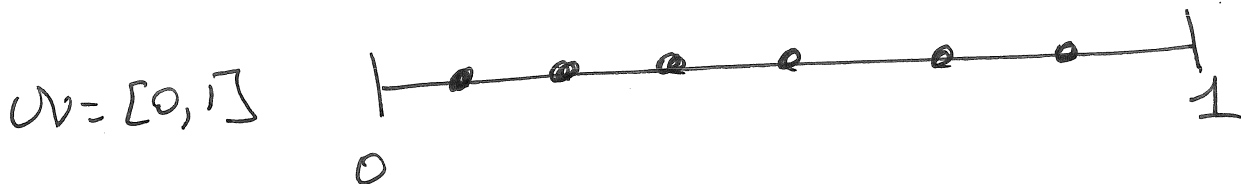
①

1) Main stages of the FD analysis of a continuous mechanics problem

Stage 1: ^{Preprocessing} Formulation of the problem
(governing eqts + BCs + initial cond.)

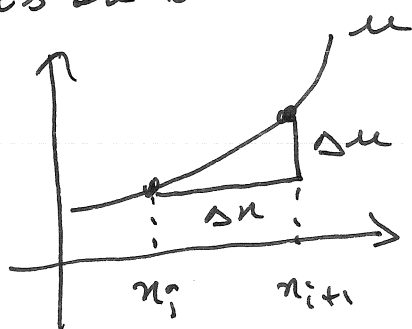
$$\underbrace{A} \text{ PDE operator } u = f \quad \text{in } \Omega.$$

— Construction of a computational grid (mesh)
 Ω_h for the spatial domain Ω .



Stage 2 | Solving

— Discretization of governing equations into a system of algebraic equations. Example: in FDM, derivatives of PDEs are substituted with finite-difference formulas



$$\frac{\partial u}{\partial x} \approx \frac{\Delta u}{\Delta x} = \frac{f_{i+1} - f_i}{x_{i+1} - x_i}$$

— Solving the algebraic eqts
(Gaussian elimination, iterative solutions, etc.)

Stage 3 | Post-Processing

②

- Analysis of results (in CFD: fluxes, rates, forces, etc...)

- Visualization of solution.

MATLAB

2) First notion of computational mesh

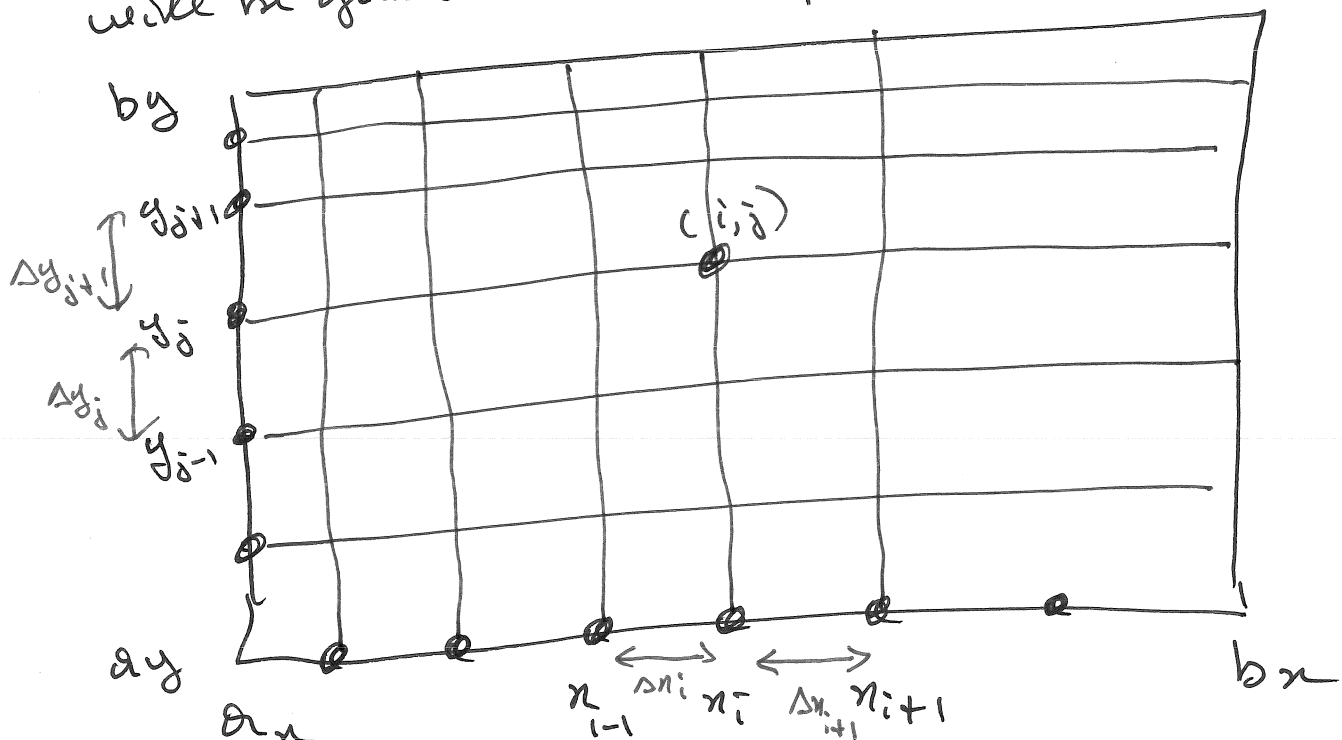
FDM_{1D} is suited for simple computational domains such as a square, rectangle, etc, which has the form:

$$D = [a_x, b_x] \times [a_y, b_y].$$

These domains are called Cartesian, and the most simple mesh for the geometries have the form:

$$D_h = \{ (x_i, y_j), i=1, \dots, N_x, j=1, \dots, N_y \}$$

A more detailed description of grids for complex domains will be given in a subsequent lecture.



(x_i, y_j) is called a node.

③

the grid size in the x and y direction is respectively:

$$\Delta x_i = x_i - x_{i-1}, \quad \Delta y_j = y_j - y_{j-1}$$

In the remainder of the section, we will mostly consider uniform grids where the grid size is constant:

$$\Delta x = \Delta x_i \quad \forall i \quad \Delta y = \Delta y_j \quad \forall j$$

We will use compact notation for the discrete solution given by the FDM (called discrete solution):

$$u_{i,j} \equiv \frac{\partial u}{\partial x} \bigg|_{i,j} = \frac{\partial u(x_i, y_j)}{\partial x}$$

3) Construction of Finite-difference formulas

Finite-difference approximation of derivatives stems directly from the definition of the derivative, i.e.:

$$\frac{\partial u}{\partial x} \bigg|_{i,j} = \frac{\partial u}{\partial x}(x_i) = \lim_{\Delta x \rightarrow 0} \frac{u(x_i + \Delta x) - u(x_i)}{\Delta x}$$

and are built by using Taylor series such as:

$$u(x + \Delta x) = \sum_{k \geq 0} u^{(k)}(x) \frac{\Delta x^k}{k!} \quad (1)$$

For example, on a uniform 1D grid, we can write:

$$(2) \quad u_{i+1} = u(x_i + \Delta x) = u_i + \Delta x \left(\frac{\partial u}{\partial x} \right)_i + \frac{\Delta x^2}{2} \left(\frac{\partial^2 u}{\partial x^2} \right)_i + \frac{\Delta x^3}{6} \left(\frac{\partial^3 u}{\partial x^3} \right)_i + O(\Delta x^4)$$

Definition (Big O notation)

Two functions f and g are such that $f = O(g)$

if there is $k > 0$ such that $\lim_{\Delta x \rightarrow 0} \left| \frac{f}{g} \right| = k$.

a constant

Analogously,

$$u_{i+1} = u_i + \Delta u \left(\frac{\partial u}{\partial x} \right)_i + \frac{\Delta u^2}{2} \left(\frac{\partial^2 u}{\partial x^2} \right)_i - \frac{\Delta u^3}{6} \left(\frac{\partial^3 u}{\partial x^3} \right)_i + O(\Delta u^4). \quad (3)$$

3-1) FD formula for the first derivative

From Eq. (2), we get:

$$\frac{u_{i+1} - u_i}{\Delta u} = \left(\frac{\partial u}{\partial x} \right)_i + \tau_i, \quad (4)$$

where τ_i is called the local truncation error (LTE), $\therefore$

$$\tau_i = \frac{\Delta u}{2} \left(\frac{\partial^2 u}{\partial x^2} \right)_i + O(\Delta u^2). \quad (5)$$

Definition (Local truncation error)

the LTE_i is the difference between the FD formula (called the numerical scheme) and the continuous formula (or equation). A mnemonic way to remember this is:

$$\text{FD scheme} = \text{continuous Eq}^t + \tau_i$$

Def (Consistency of a scheme)

A scheme is consistent if it has the property to represent "correctly" the continuous formula when the grid size Δu goes to zero. It means:

the scheme is consistent if $\lim_{\Delta u \rightarrow 0} \tau_i = 0$.

When $\tau_i = O(\Delta u^q)$, the method is consistent of order q . The higher is q , the smaller is the discretization error (see exercise A.1).

(5)

Application: The FD formula (4) has a LTE given by $\tau_i = O(\Delta u)$, it means it is consistent of order 1.

There are 2 other ways to define FD formulas for the first derivative by combining Taylor expansion (2) or (3).

Def (Usual FD of the first order derivative)

1) "Forward" FD formula; obtained by Eq. (2)

$$\delta_n^+ u_i = \frac{u_{i+1} - u_i}{\Delta u}, \quad \tau_i = \frac{\Delta u}{2} \left(\frac{\partial^2 u}{\partial u^2} \right)_i + O(\Delta u^2)$$

2) "Backward" FD formula; obtained by Eq. (3)

$$\delta_n^- u_i = \frac{u_i - u_{i-1}}{\Delta u}, \quad \tau_i = -\frac{\Delta u}{2} \left(\frac{\partial^2 u}{\partial u^2} \right)_i + O(\Delta u^2)$$

3) "Central" FD formula, obtained by combining Eq. (2) and (3)

$$\delta_n^0 u_i = \frac{u_{i+1} - u_{i-1}}{2\Delta u}, \quad \tau_i = \frac{\Delta u^2}{2} \left(\frac{\partial^3 u}{\partial u^3} \right)_i + O(\Delta u^3)$$

A sketch of these different FD derivatives is given in Fig. 2. The stencil, i.e. the nodes used by these formulas are:

- $\{u_{i+1}, u_i\}$ for δ_n^+
- $\{u_i, u_{i-1}\}$ for δ_n^-

• $\{u_{i-1}, u_{i+1}\}$ for δ_n^0 .

(6)

Formula δ_n^0 is called a centred formula because it uses a stencil centered around u_i .

Formulas δ_n^- and δ_n^+ which uses ~~more~~ ^{more} points at one node or the other of u_i are called biased formulas.

→ It is the only one whose LTE is second-order, and hence is more accurate. A systematic construction of FD formula of higher order will be presented in the exercise by using more points in the stencil.

3-2) FD formula for the second derivative

It is customary to use centred FD formula for $\frac{\partial^2 u}{\partial x^2}$.
In Exercise A.1, you will show by Taylor expansion that:

$$\frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} = \left(\frac{\partial^2 u}{\partial x^2} \right)_i + \frac{\Delta x^2}{12} \left(\frac{\partial^4 u}{\partial x^4} \right)_i + O(\Delta x^3)$$

this leads to the following definition:

Def: (Usual FD formula for 2nd derivative)

the centred formula

$$\delta_n^0 u_i = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2}$$

has a LTE of order 2.

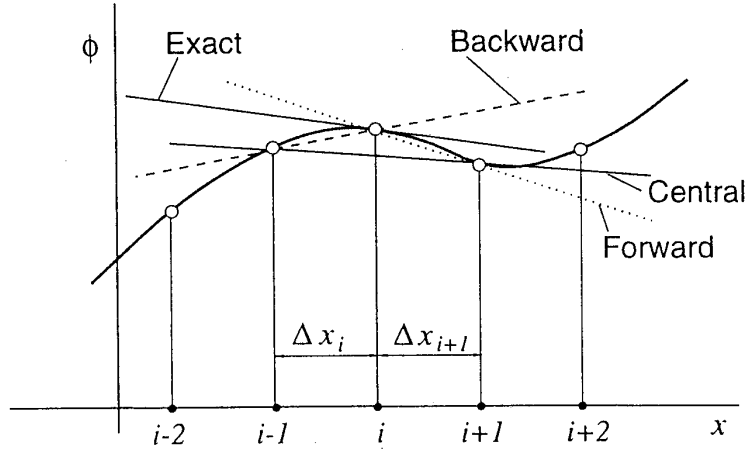


Fig. 5.2. Approximations usuelles de la dérivée première au nœud x_i .

- Schéma décentré amont (“upwind”, ou “backward”), en $\mathcal{O}(\Delta x)$:

$$\delta_x^- u_i \equiv \frac{u_i - u_{i-1}}{\Delta x}, \quad (5.13)$$

- Schéma décentré aval (“forward”), en $\mathcal{O}(\Delta x)$:

$$\delta_x^+ u_i \equiv \frac{u_{i+1} - u_i}{\Delta x}, \quad (5.14)$$

- Schéma centré (“central”), en $\mathcal{O}(\Delta x^2)$:

$$\delta_x^0 u_i \equiv \frac{u_{i+1} - u_{i-1}}{2\Delta x}. \quad (5.15)$$

Une interprétation géométrique de ces formules est présentée sur la figure 5.2. Il est évident d’après cette figure que certaines formules sont meilleures que d’autres. Les propriétés de chacune de ces approximations seront mises en évidence dans le chapitre 9.

5.3.2 Approximation de la dérivée seconde

L’approximation usuelle de la dérivée seconde est la formule centrée à trois points, déjà présentée dans le paragraphe 3.1 :

$$\frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} = \left(\frac{\partial^2 u}{\partial x^2} \right)_i + \frac{\Delta x^2}{12} \left(\frac{\partial^4 u}{\partial x^4} \right)_i + \mathcal{O}(\Delta x^3). \quad (5.16)$$

Définition 5.2 (Approximation de la dérivée seconde). L’approximation usuelle est la formule centrée à trois points :

$$\delta_{xx}^0 u_i \equiv \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2}. \quad (5.17)$$

dont l’erreur de troncature est en $\mathcal{O}(\Delta x^2)$.

Remarque 5.3. Il est possible de construire des schémas pour les dérivées secondes (ou d’ordre plus élevé) à partir de formules pour les dérivées d’ordre inférieur. Par exemple,

$$\delta_{xx}^0 u_i = \delta_x^+ (\delta_x^- u_i) = \delta_x^- (\delta_x^+ u_i) = \frac{1}{\Delta x} (\delta_x^+ - \delta_x^-) u_i,$$

par contre

$$\delta_x^0 \delta_x^0 u_i = \frac{u_{i+2} - 2u_i + u_{i-2}}{(2\Delta x)^2} \neq \delta_{xx}^0 u_i.$$

FDM for time-dependent Equations I: Parabolic Eqs.

①

We consider in this section the prototype of parabolic PDEs, the heat equation:

$$\frac{\partial u}{\partial t} - \nu \frac{\partial^2 u}{\partial x^2} = f(t, x) \quad (1)$$

with the initial condition $u(t=0, x) = u_0(x)$.

We will review in this section:

• Construction of FD schemes by the method of lines (MOL)

• Use of time-stepping schemes for ODEs (see appendix):

ODE: ordinary differential equations

– Forward Euler method (FE1)

– Backward Euler method (BE1)

• Truncation error ^(LTE) of the FD schemes,

as seen in previous section:

$$\underbrace{\frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2}}_{\text{SO}_{xx} u_i} = \left(\frac{\partial^2 u}{\partial x^2} \right)_i + \underbrace{\frac{\Delta x^2}{12} \left(\frac{\partial^4 u}{\partial x^4} \right)_i + O(\Delta x^4)}_{\tau_i = O(\Delta x^2): \text{truncation error.}}$$

• Stability of FD schemes for long-time integration: Von-Neumann stability analysis.

1) Construction of FD schemes by the M.O.L (2)

The method of lines (M.O.L) is decomposed in 2 steps:

Step 1] Approximate the spatial derivatives in (1) by FD formulas, i.e:

$$\forall u_i \text{ on the mesh, } \frac{du_i(t)}{dt} - \nu \delta_{xx}^0 u_i(t) = f_i(t) \quad (2)$$

In this way, the PDE is transformed in a set of N ODEs, which reads in matrix form:

$$\frac{d}{dt} \underline{U}(t) - \nu \underline{K} \underline{U}(t) = \underline{F}(t) \quad (3)$$

with $\underline{U}(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_N(t) \end{bmatrix}$, $\underline{F}(t) = \begin{bmatrix} f_1(t) \\ f_2(t) \\ \vdots \\ f_N(t) \end{bmatrix}$

$$\text{and } \underline{K} = \frac{-1}{\Delta x^2} \begin{bmatrix} 2 & -1 & & (0) \\ -1 & 2 & -1 & \\ & \ddots & \ddots & \ddots \\ (0) & & -1 & 2 \end{bmatrix} = \frac{1}{\Delta x^2} \text{tridiag} \begin{pmatrix} +1, -2, +1 \\ N \end{pmatrix}$$

is the matrix representation of δ_{xx}^0

Step 2] The system of ODEs (3) can now be solved by a time-stepping scheme for ODEs, and we consider here the FE1 method from the Appendix, which gives for Eq. (3):

the FE1 scheme for the 1D heat Equation: (3)

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} - \nu \delta_{xx}^0 u_i^n = f_i^n \quad (4),$$

where we remind that $u_i^n \equiv u(x_i, t_n)$ with $x_i = i\Delta x$ and $t_n = n\Delta t$. this scheme leads in matrix form:

$$\frac{1}{\Delta t} (\underline{U}^{n+1} - \underline{U}^n) - \nu \underline{K} \underline{U}^n = \underline{F}^n \quad (5).$$

By using Taylor expansion around (t_n, x_i) , the LTE of the FE1 scheme is:

$$\tau_i^n = \frac{\Delta t}{2} \left(\frac{\partial^2 u}{\partial t^2} \right)_i^n - \frac{\nu \Delta x^2}{12} \left(\frac{\partial^4 u}{\partial x^4} \right)_i^n + O(\Delta t^2, \Delta x^4) \quad (6)$$

$$= O(\Delta t, \Delta x^2)$$

In order to implement the scheme in a computer program, we write (4) in its iterative form:

$$\underbrace{u_i^{n+1}}_{\text{unknown at } t_n} = \underbrace{B_{i,i-1} u_{i-1}^n + B_{i,i} u_i^n + B_{i,i+1} u_{i+1}^n}_{\text{known at } t_n} + \Delta t f_i^n \quad (7a)$$

$$= \sum_{j=i-1}^{i+1} B_{i,j} u_j^n + \Delta t f_i^n \quad (7b)$$

where the non-zero coefficients of the iteration matrix $\underline{B}$:

$$B_{i,i-1} = \tau \quad B_{i,i} = 1 - 2\tau, \quad B_{i,i+1} = \tau \quad (8a)$$

where the diffusion coefficient τ is:

$$\tau = \nu \frac{\Delta t}{\Delta x^2} \quad (8b)$$

The iterative form (7) can also be written in matrix notation: (4)

$$\underline{U}^{n+1} = \underline{B} \underline{U}^n + \Delta t \underline{F}^n \quad (9)$$

where $\underline{B} = \text{tridiag}(1, 1-2\alpha, 1)$ is the iteration matrix.

As a summary, the important properties of the FE1 scheme (4) are:

- Truncation error $\tau_i = O(\Delta t, \Delta x^2)$: the scheme is first-order in time, second-order in space.

- Explicit: According to the iterative form (7) or (9), there is no need to solve an algebraic equation to obtain u_i^{n+1} from previous time levels.

- It is a 2-level scheme: only levels u_i^{n+1} and u_i^n are present in (4).

2) The Backward Euler scheme (BE1)

The BE1 scheme is obtained by applying the backward Euler scheme to the system of ODEs (2):

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} - \nu \delta_{xx}^0 u_i^{n+1} = f_i^{n+1} \quad (10)$$

In the exercise on the θ -scheme, we will show that the truncation error is:

$$\tau_i^n = \frac{\Delta t}{2} \frac{d}{dt} \left\{ \frac{\partial u}{\partial t} - 2\nu \frac{\partial^2 u}{\partial x^2} + f \right\}_i^n - \nu \frac{\Delta x^2}{12} \left(\frac{\partial^4 u}{\partial x^4} \right)_i^n \quad (11)$$

$$= O(\Delta t, \Delta x^2)$$

that shows that the scheme is 1st order in time and 5th order in space, like the FE1 scheme.

The iterative form of the BE1 scheme is:

$$u_i^{n+1} - \nu \Delta t \delta_{xx}^0 u_i^{n+1} = u_i^n + \nu \Delta t f_i^{n+1} \quad (12)$$

which is expanded as:

$$\underbrace{-2}_{B_{i,i+1}^0} u_{i+1}^{n+1} + \underbrace{(1+2\nu)}_{B_{i,i}^0} u_i^{n+1} - \underbrace{2}_{B_{i,i-1}^0} u_{i-1}^{n+1} = \underbrace{u_i^n}_{B_{i,i}^1} + \nu \Delta t f_i^{n+1} \quad (13)$$

and can be written in matrix form:

$$\underline{B}^0 \underline{U}^{n+1} = \underline{B}^1 \underline{U}^n + \Delta t \underline{F} \quad (13b)$$

with $\underline{B}^0 = \text{tridiag}(-2, 1+2\nu, -2)$, $\underline{B}^1 = \underline{I}_d$.

In summary, the important properties of BE1 scheme (10) are:

- Truncation error: $\tau_i = O(\Delta t, \Delta x^2)$

- 2-level scheme: u_i^{n+1} and u_i^n are present.

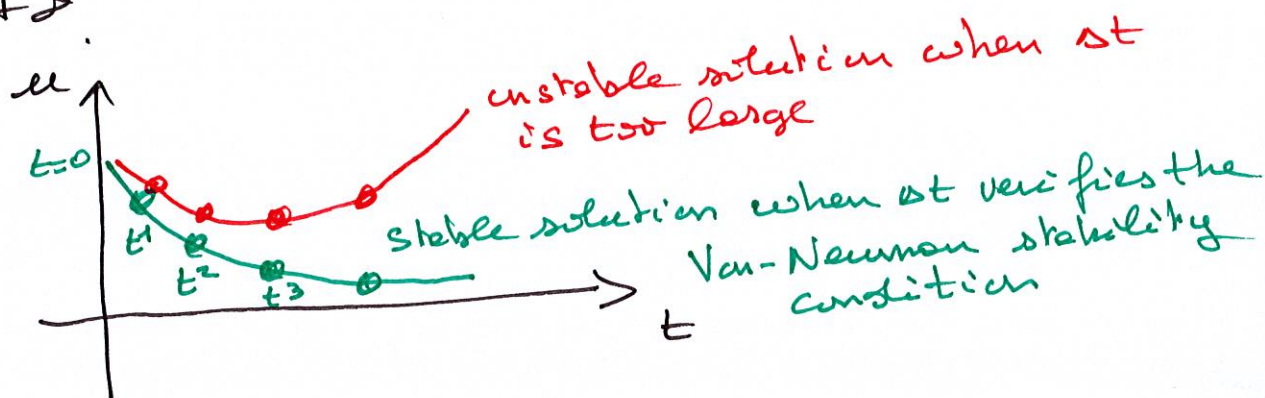
- Implicit scheme: According to (13), there needs to solve an algebraic set of linear equations (ie, a linear system) to get u_i^{n+1} from u_i^n .

Hence, the BE1 scheme needs more computational effort than the (explicit) FE1 scheme. However, we will see next that BE1 is more stable than FE1.

~~the EBT scheme (9) since there is a linear system (5) to solve at each time step to get U^{n+1} . But, as we shall see next, the BE1 is much more stable when we use it to get u at large t .~~

3) (Long-time) stability of FD schemes for unsteady PDEs

The (long-time, or absolute) stability is the property of a FD scheme to provide a numerical solution that does not blow up to $\pm \infty$ when $t \rightarrow \infty$.



Stability analysis is important for determining the values of the time-step Δt that give stable computation, on a given mesh of size N_x .

Practical criteria for the stability of FD schemes is achieved by performing the Von-Neuman (VN) analysis (which amounts to perform a Fourier Analysis). Let us consider the 2-level unforced ($f=0$) scheme:

$$\underline{\underline{B}}^0 \underline{\underline{U}}^{n+1} = \underline{\underline{B}}^1 \underline{\underline{U}}^n \quad (14)$$

We assume the solution to (14) is a Fourier (6) mode (see Lecture notes on Basics of PDEs)

$$u_i^n = \hat{u}^n e^{i k x_i^0}, \quad i^2 = -1, \quad x_i = i \Delta x \quad (15)$$

Eq. (14) becomes:

$$g_0(k \Delta x) \hat{u}^{n+1} = g_1(k \Delta x) \hat{u}^n \quad (16)$$

that we rewrite as:

$$\hat{u}^{n+1} = g(k \Delta x) \hat{u}^n, \quad (17)$$

where $g(k \Delta x) = g_1(k \Delta x) / g_0(k \Delta x)$ is the amplification factor of the FD scheme, which can be a complex number. As in the case of ODEs (see appendix), the geometric progression (17) is bounded when $n \rightarrow +\infty$ if the VN stability criteria is met:

Definition (Von-Neumann stability)

A FD scheme is stable for values of Δt that verifies:

$$(18) \quad |g(k \Delta x)| \leq 1, \quad \forall k \Delta x \in [-\pi, \pi]$$

Remark: In an exercise, it is useful to consider the auxiliary variable $\xi \in [0, 1]$ such as:

$$\xi = \frac{1 - \cos(k \Delta x)}{2} \quad (19a)$$

Another useful identity is:

$$\sin^2(\frac{k \Delta x}{2}) = 4\xi(1-\xi) \quad (19b)$$

Using the variable ζ , the VN condition (18)

(7)

becomes:

$$|g(\zeta)| \leq 1, \quad \forall \zeta \in [0, 1] \quad (20)$$

As a first example, let us apply VN stability analysis to the FE ~~1D~~ scheme (7), whose iterative form ~~reads~~ reads with $f \equiv 0$:

$$u_i^{n+1} = r u_{i+1}^n + (1-2r) u_i^n + r u_{i-1}^n$$

After using (15), i.e: $u_i^n = \hat{u}^n e^{ik i \Delta x}$, we get:

$$\hat{u}^{n+1} e^{i k i \Delta x} = \hat{u}^n e^{i k i \Delta x} [r e^{i k \Delta x} + (1-2r) + r e^{-i k \Delta x}]$$

we get the amplification factor of FE:

$$\Rightarrow \frac{\hat{u}^{n+1}}{\hat{u}^n} = g(k \Delta x) = 1 - 2r + 2r \cos(k \Delta x) \\ = 1 - 4r \zeta \text{ using (15a).}$$

The VN condition (20) is:

$$-1 \leq 1 - 4r \zeta \leq 1 \quad \forall \zeta \in [0, 1]$$

$$0 \leq r \zeta \leq \frac{1}{2} \quad \forall \zeta \in [0, 1]$$

This inequality is the strictest for $\zeta = 1$:

$$0 \leq r \leq \frac{1}{2}$$

Since $r = \frac{\nu \Delta t}{\Delta x^2}$, we finally get the stability condition of the FE scheme for the 1D heat equation:

$$\Delta t \leq \frac{\Delta x^2}{2\nu} \quad (21)$$

The FE scheme is said to be conditionally stable, because for a given mesh of size Δx , we should take a value of Δt that verifies (21)

to obtain a stable computation.

(8)

The stability condition (21) is very strict, because when the mesh size is halved (ie, $\Delta x' = \Delta x/2$), the time-step $\Delta t'$ should be divided by 4:

$$\Delta t' \leq \frac{\Delta x'^2}{2\nu} = \frac{1}{4} \frac{\Delta x^2}{2\nu}.$$

It means that on very fine meshes, the value of Δt becomes very small, leading to very large computational time. For this reason, it is preferable to use implicit time integration for viscous problems such as the heat equation: As we will see next, implicit methods such as the BE scheme are stable for any value of Δt .

To verify this, let us consider the BE scheme (12):

$$-r u_{i+1}^{n+1} + (1+2r) u_i^{n+1} - r u_{i-1}^{n+1} = u_i^n,$$

Using $u_i^n = \hat{u}^n e^{ikh_i \Delta x}$, we get:

$$\hat{u}^{n+1} e^{ikh_i \Delta x} [-r e^{ikh_{i+1} \Delta x} + (1+2r) - r e^{-ikh_{i-1} \Delta x}] = \hat{u}^n e^{ikh_i \Delta x},$$

after simplification we get the amplification factor of BE scheme:

$$g(s) = \frac{1}{1+4rs}.$$

Since $\forall s \in [0, 1]$, $1+4rs \geq 1$, we get:

$$0 \leq g(s) \leq 1 \quad \forall s \in [0, 1],$$

that shows that BE is unconditionally stable: for any value of Δt , the BE gives stable computations.

Appendix A: Methods for Ordinary Differential Equations (ODEs)

We consider the initial value problem:

$$(1a) \quad \dot{y} = f(t, y(t)), \quad t \in [0, T] = \mathcal{U}$$

$$(1b) \quad y(t=0) = y_0$$

often called a Cauchy problem.

I) Building of basic numerical schemes

the basic methods used in CFD (Euler scheme, Crank-Nicolson method) will be built here by means of numerical quadrature.

First, the time domain $\mathcal{U} = [0, T]$ is subdivided as

$$\text{as } \{t_n = n \Delta t, \quad n = 0, \dots, N\}$$

where $\Delta t = \frac{T}{N}$ is the time-step.

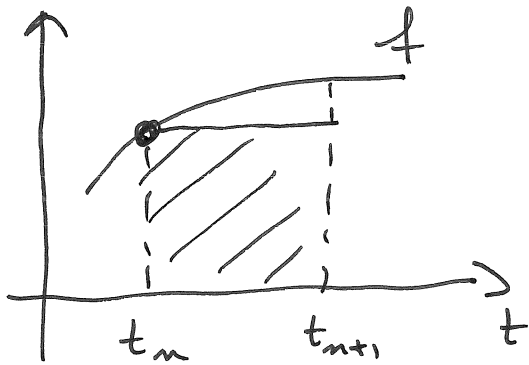
The EDO (1a) is first integrated in $[t_n, t_{n+1}]$ to yield:

$$y(t_{n+1}) - y(t_n) = \int_{t_n}^{t_{n+1}} f(t, y(t)) dt \quad (2)$$

~~Now~~ No discrete approximation has yet been performed. Now, the RHS of (2) will be approximated with numerical quadrature based on Lagrange polynomials.

I-1) Forward Euler method (FE1) (2)

The simplest method stems from the approximation of (2) by using values at time t_n :



$$\int_{t_n}^{t_{n+1}} f dt \approx \Delta t f(t_n, y^n)$$

where we used the compact notation:

$$\underbrace{y^n}_{\text{approx. solution.}} \approx \underbrace{y(t_n)}_{\text{exact solution}}$$

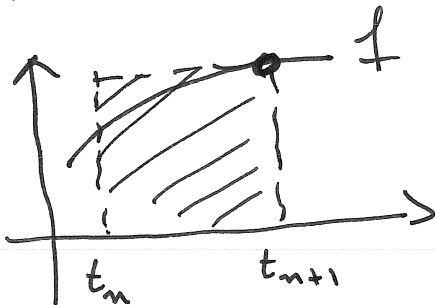
the forward Euler scheme reads then:

$$\frac{y^{n+1} - y^n}{\Delta t} = f(t_n, y^n) \quad (3)$$

Its acronym is FE1, because we shall see later that the method is first order in time.

I-2) Backward Euler Method (BE1)

If the RHS of (2) is approximated with end point t_{n+1} :



$$\int_{t_n}^{t_{n+1}} f dt \approx \Delta t f(t_{n+1}, y^{n+1})$$

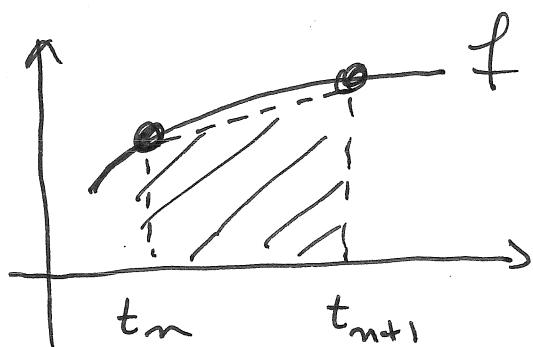
we obtain the backward Euler method (BE1):

$$\frac{y^{n+1} - y^n}{\Delta t} = f(t_{n+1}, y^{n+1}) \quad (4)$$

I-3] Trapezoidal method

(3)

The RHS of (2) can be approximated by the Trapezoidal method (use of Lagrange polynomial of order 2)



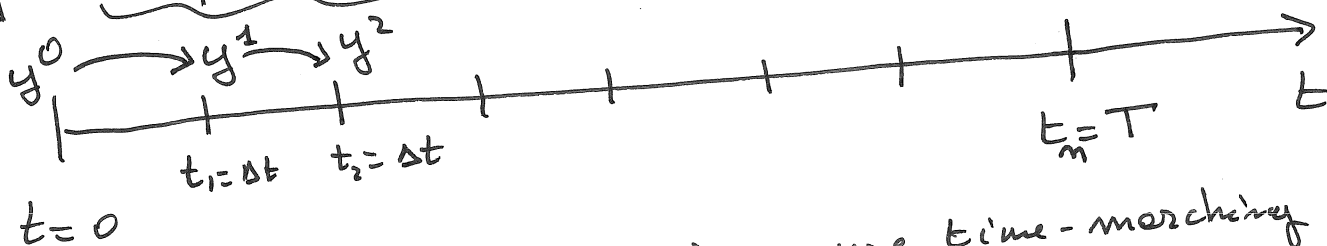
$$\int_{t_n}^{t_{n+1}} f dt \approx \frac{\Delta t}{2} [f(t_{n+1}, y^{n+1}) + f(t_n, y^n)]$$

The resulting trapezoidal method:

$$\frac{y^{n+1} - y^n}{\Delta t} = \frac{1}{2} [f(t_{n+1}, y^{n+1}) + f(t_n, y^n)] \quad (5)$$

is often called the Crank-Nicolson method (CN) when used for time integration of PDEs.

II] Use of time integration scheme. Implicit or explicit method



For ~~integrating~~ solving ODE (1), we use time-marching by starting with the initial condition at $t_0=0$:

$$y^0 = y(0),$$

Then, y^1 at $t=t_1$ is calculated with FE1, for example, written in its iterative form:

$$\underbrace{y^1}_{\text{Unknown at } t=t_1} = \underbrace{y^0 + \Delta t f(t_0, y^0)}_{\text{known at } t=t_1} \quad (6)$$

(4)

This expression is ready to be programmed in any language, such as FORTRAN or MATLAB. Since there is no equation to solve to obtain y^1 from values at previous time-level, the FE1 scheme is called explicit. Furthermore, Eq. (6) involves the solution at only time levels to and t_1 , thus FE1 is called a two-level scheme.

By contrast, the BE1 scheme leads to the iterative equation:

$$y^1 - \Delta t f(t_1, y^1) = y^0, \quad (7)$$

the solution of an equation is needed to obtain y^1 , resulting in more computational work. The BE1 is called an implicit scheme, and is also a two-level scheme.

The CN (trapezoidal) scheme is an implicit two-level scheme.

III) Truncation error of time-integrated scheme.

The local truncation error of a scheme (LTE) is the difference between the numerical scheme and the ODE at time t_n . It can be summarized as:

$$\text{Scheme at } t_n = \text{ODE at } t_n + \tau^n$$

where τ^n is the LTE at time t_n . It is usually performed by using Taylor expansion of the solution, i.e.

$$y^{n+1} = y^n + \Delta t \dot{y}^n + \frac{\Delta t^2}{2} \ddot{y}^n + O(\Delta t^3) \quad (8)$$

Let us calculate the LTE of backward-Euler scheme (5) by introducing this Taylor expansion in (3):

$$\underbrace{\frac{y^{n+1} - y^n}{\Delta t} - f(t_n, y_n)}_{\text{FE1 scheme at } t_n} = \frac{y^n + \Delta t \dot{y}^n + \frac{\Delta t^2}{2} \ddot{y}^n + O(\Delta t^3) - y^n}{\Delta t} - f(t_n, y_n)$$

$$= \underbrace{\dot{y}^n - f(t_n, y_n)}_{\text{ODE at } t_n} + \tau^n$$

where the LTE is $\tau^n = \frac{\Delta t}{2} \ddot{y}^n + O(\Delta t^2)$.

Hence, the FE1 scheme is first-order accurate.

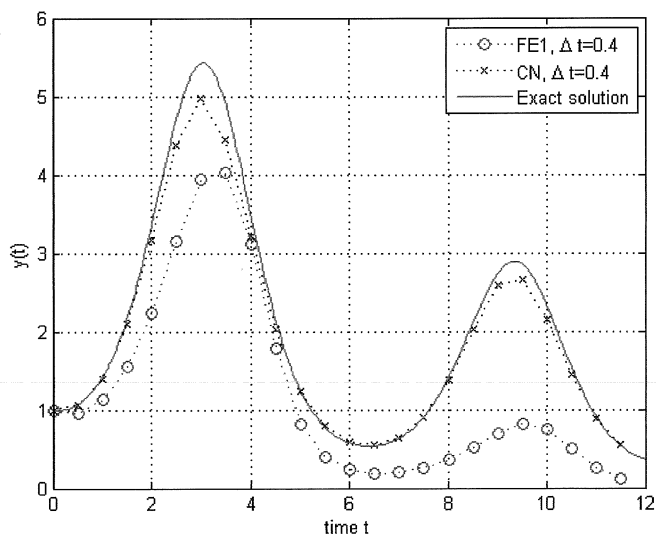
In a similar fashion, we can show the the LTE of

BE1 is:

$$\tau^n = -\frac{\Delta t}{2} \ddot{y}^n + O(\Delta t^2)$$

and the LTE of CN scheme is:

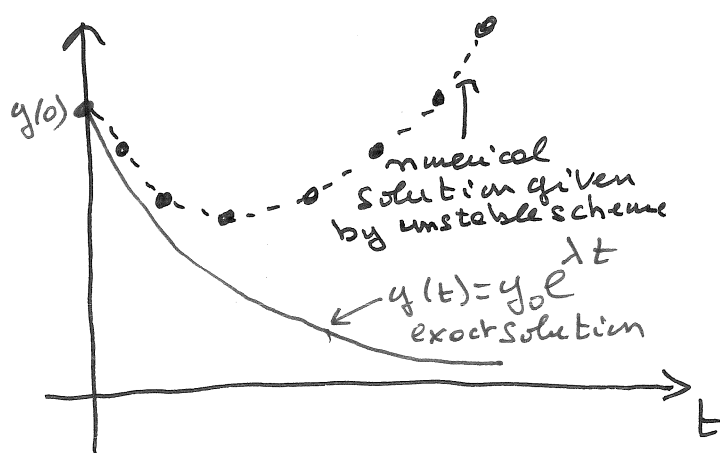
$$\tau^n = \frac{\Delta t^2}{12} \ddot{y}^n + O(\Delta t^3)$$



IV

Stability of a numerical scheme

(6)



The stability (also called absolute stability) is the property of the scheme to yield a numerical solution who does not diverge to ∞ when t is large.

Let us consider the model problem for $\lambda < 0$:

$$\begin{cases} \dot{y} = \lambda y, & t \in [0, +\infty[\\ y(0) = y_0 \end{cases} \quad (9)$$

Its exact solution is:

$$y(t) = y_0 e^{\lambda t},$$

and since $\lambda < 0$ the solution decays with time ($\lim_{t \rightarrow \infty} y(t) = 0$).

When BE1 is applied to model problem (9), the iterative equation reads:

$$y^{n+1} = y^n + \Delta t \lambda y^n \Rightarrow y^n = \sigma y^{n-1} = \dots = (\sigma)^n y^0$$

where $\sigma = 1 + \lambda \Delta t$ is the amplification factor of the scheme. In order to have a solution that does not blow up in time, we must have:

$$|\sigma| < 1 \quad (10)$$

$$\text{which leads to } 0 < \Delta t < -\frac{2}{\lambda}.$$

Hence, the BE1 scheme is stable if we use a time-step that verifies the stability condition:

$$\Delta t < \frac{2}{-\lambda} \quad (11)$$

We say that BE1 is conditionally stable.

- If we use BE1 to solve model problem (9),
the amplification factor is $\sigma = \frac{1}{1 - \lambda \Delta t}$,

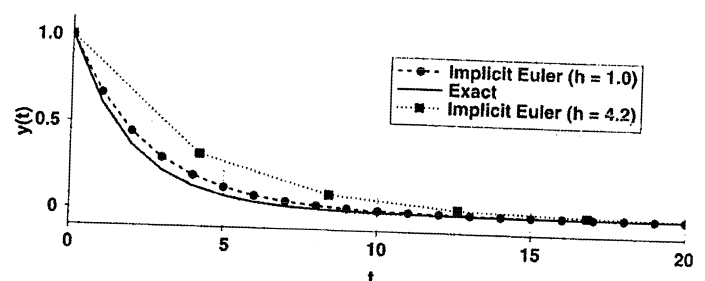
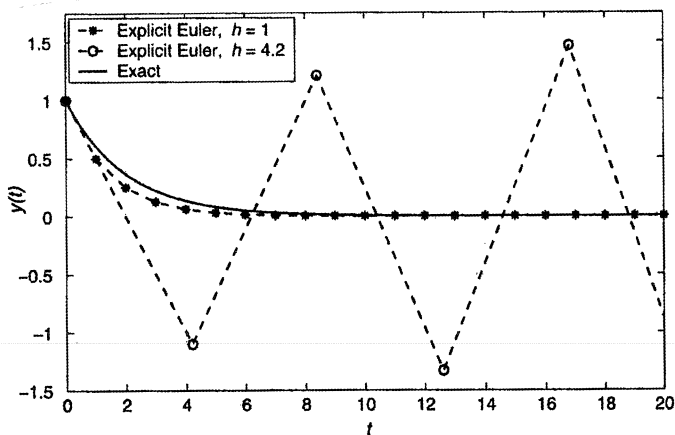
and stability condition $|\sigma| < 1$ leads to $|1 - \lambda \Delta t| > 1$,
which is always verified for $\Delta t > 0$. Hence, the BE1 is
stable with no conditions.

- The amplification factor for CN scheme is

$$\sigma = \frac{1 + \lambda \Delta t/2}{1 - \lambda \Delta t/2}$$

and stability condition (10) is verified for all $\Delta t > 0$. As
the BE1 scheme, the CN scheme is always stable.

Application: We consider the model problem $\dot{y} + \frac{1}{2}y = 0$.
the stability condition of BE1 is $\Delta t < 4$, while BE1 and CN
are stable for all $\Delta t > 0$. Fig. 1 shows computations
with FE with various values of Δt , the one with $\Delta t = 4.2$
is unstable. Fig. 2 shows that BE1 is stable for all values of Δt .



(1)

FDM for time-dependent Eq^s II : Hyperbolic PDEs

1) Introduction

As seen in the 1st lecture, the generic eq^t for fluid flows and heat transfer problem is the 1D convection diffusion eq^t:

$$\frac{\partial u}{\partial t} + a \underbrace{\frac{\partial u}{\partial x}}_{\text{convection}} - \underbrace{\nu \frac{\partial^2 u}{\partial x^2}}_{\text{diffusion}} = 0 \quad (1)$$

In the case of vanishing viscosity ($\nu=0$), this eq^t reduces to the transport eq^t:

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0. \quad (2)$$

The Fourier solution of this eq^t reads:

$$u(x, t) = \hat{u}_0 e^{\pm k(x-at)},$$

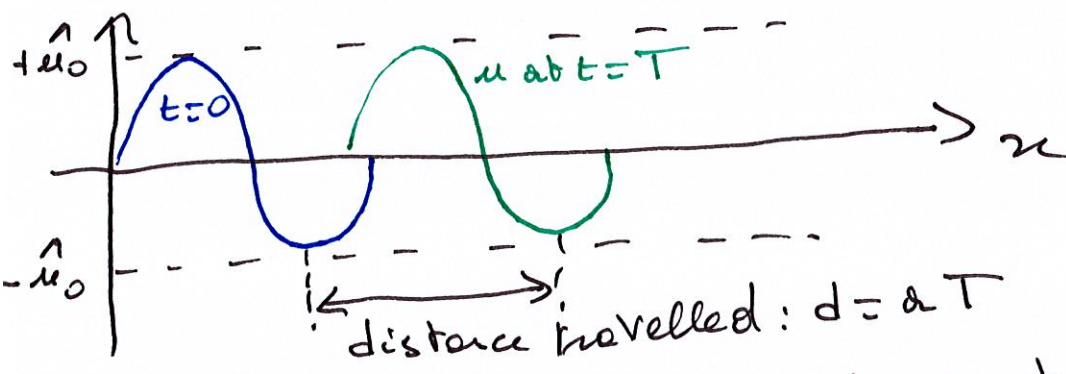


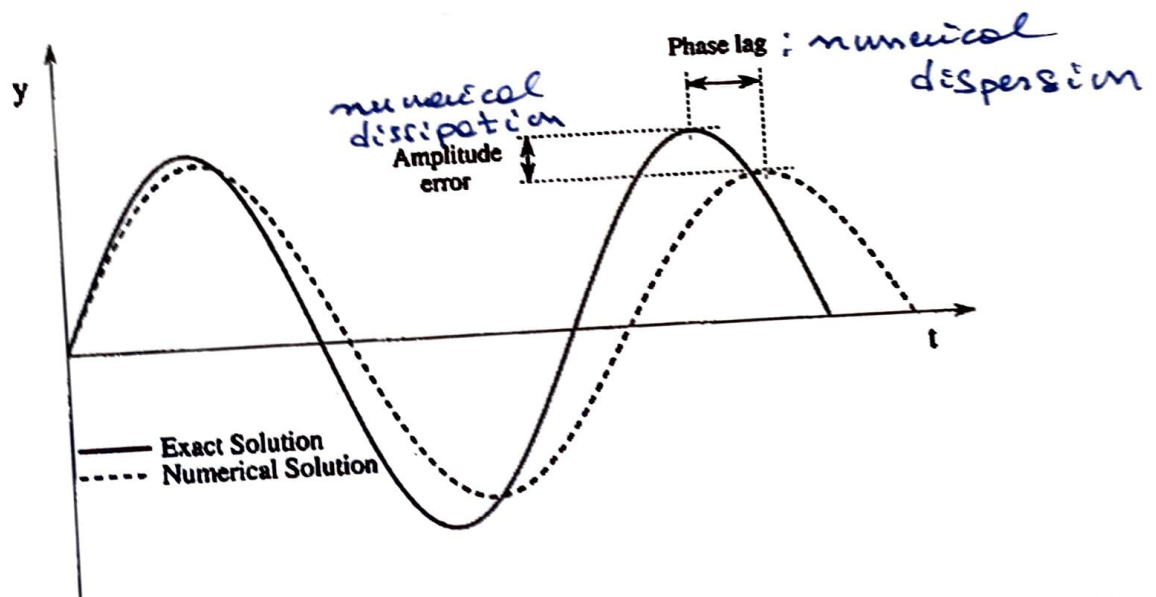
Fig 1: Fourier solution of the transport eq^t at $t=0$ (blue) and $t=T$ (green) for convection velocity $a > 0$. which shows that the solution corresponds to the propagation of the initial condition $u_0(x)$ at speed a with no distortion or dissipation.

(2)

The transport eqt. (1) is the prototype of hyperbolic equations, which model the dynamics of physical waves. An example is the conservation of mass for compressible fluids:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{v}) = 0 \quad (3)$$

The numerical solution of hyperbolic problems is particularly difficult, because the FD scheme should alter as ~~much~~ little as possible the amplitude of the solution wave (phenomenon of numerical dissipation) and its speed (phenomenon of numerical dispersion).



In the following, we will focus on the analysis of FD schemes for the transport eqt (2) with $\alpha > 0$ (solution travelling to the right). These FD schemes, which use the important concepts of upwinding and artificial dissipation, are widely used for the convection term of conservation eqt. in commercial codes such as ANSYS FLUENT.

2) The failure of central schemes

(3)

As in the lecture on parabolic eq's, the simplest scheme we consider for eq. (2) is the FE scheme with central differencing of the convection term:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + a \delta_n^0 u_i^n = 0 \quad (4)$$

The truncation error is:

$$\begin{aligned} \tau_i &= \frac{\Delta t}{2} \frac{\partial u}{\partial t} \Big|_i^n + a \frac{\Delta x^2}{2} \frac{\partial^3 u}{\partial x^3} \Big|_i^n + O(\Delta t^2, \Delta x^4) \\ &= O(\Delta t, \Delta x^2), \end{aligned}$$

so the scheme is 1st order in time and 2nd order in space.

The iterative form of (4) is:

$$u_i^{n+1} = u_i^n - \frac{C}{2} (u_{i+1}^n - u_{i-1}^n) \quad (5)$$

where C is the Courant number: (important definition)

$$C = \frac{a \Delta t}{\Delta x} \quad (6)$$

The scheme is explicit, with 2 levels: n and $n+1$.

Let us investigate the temporal stability of the scheme.

By using $u_i^n = \hat{u}^n e^{ik_i \Delta x}$ in (5), we obtain

$$g(k \Delta x) = 1 - \frac{i}{2} C \sin(k \Delta x), \quad (7)$$

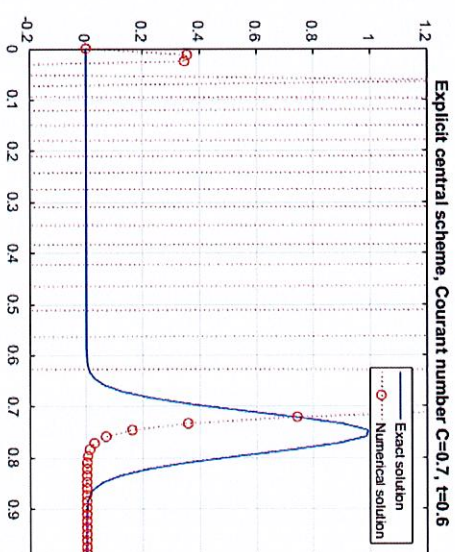
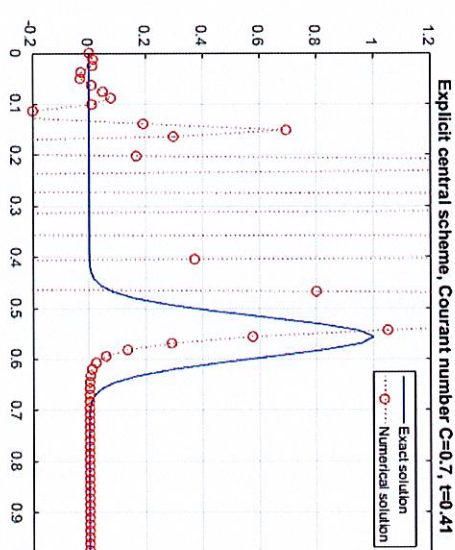
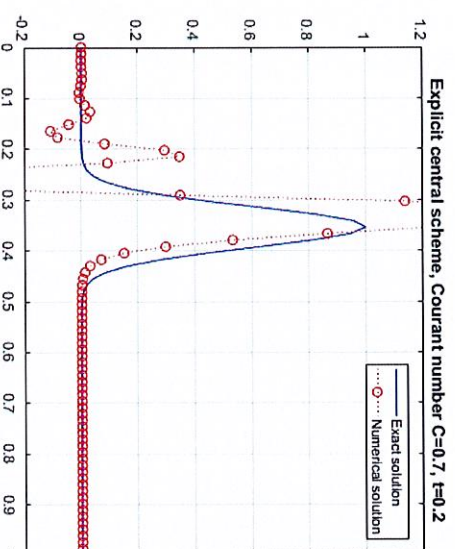
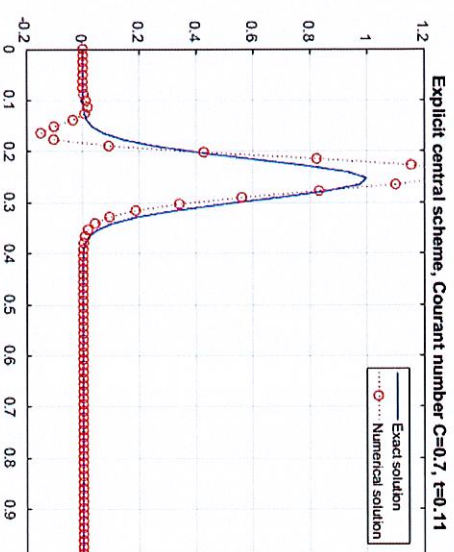
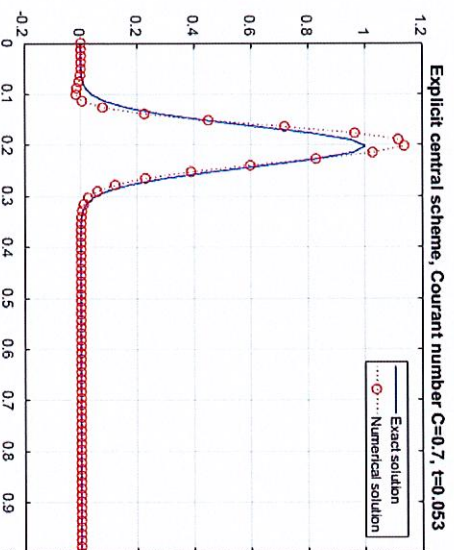
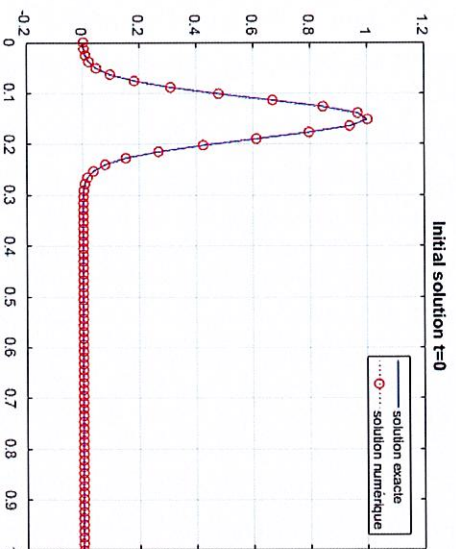
whose modulus verifies:

$$\forall k \Delta x \in [-\pi, \pi], |g(k \Delta x)|^2 = 1 + C^2 \sin^2(k \Delta x) \geq 1 \quad (8)$$

Hence, the VN stability condition $|g| \leq 1$ is never verified: for any value of Δt , the scheme is never stable.

Transport Eq. : FE central scheme

Fig 2: Solution of the transport Eq. $\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0$ with FE central scheme.



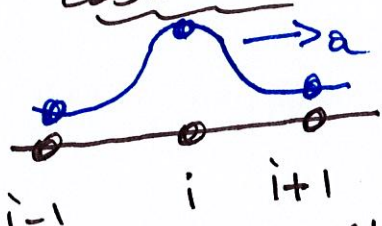
- Scheme is unstable for any value of $C=a \Delta t / \Delta x$

As shown in Fig. 2, the numerical solution exhibits spurious ~~numeric~~ oscillations during the time integration, which become larger as $t \rightarrow +\infty$.
 $\Rightarrow$ the FE central scheme has no practical use. (4)

3) The use of upwinding and the CFL condition

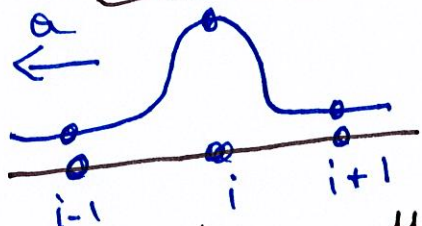
The failure of the FE central scheme (4) is caused by the central approximation of the convection term. As seen in the exercise on the steady convection-diffusion PDE, a simple remedy is to use upwinding according to the convection velocity:

case $a > 0$



use $\delta_x u_i = \frac{u_i - u_{i-1}}{\Delta x}$

case $a \leq 0$



use $\delta_x u_i = \frac{u_{i+1} - u_i}{\Delta x}$

3-1] The upwind scheme

For the transport eq. (2) with $a > 0$, the FE scheme with upwind discretization is:

$$\frac{u_{i+1}^{n+1} - u_i^n}{\Delta t} + a \delta_x^- u_i^n = 0 \quad (9)$$

This 2-level explicit scheme is simply called the upwind scheme. Its truncation error is:

$$\tau_i = \frac{\Delta t}{2} \frac{\partial u}{\partial t} \Big|_i^n - a \frac{\Delta x}{2} \frac{\partial^2 u}{\partial x^2} \Big|_i^n = O(\Delta t, \Delta x), \quad (10)$$

which has lower spatial accuracy than the central scheme (4).

Let's turn to the stability analysis of the upwind scheme. The iterative form of (9) is:

$$u_i^{n+1} = (1-c) u_i^n + c u_{i-1}^n, \quad (11)$$

and its amplification factor is:

$$\begin{aligned} g(k\Delta x) &= 1 - c(1 - \cos k\Delta x) - i c \sin k\Delta x \\ &= 1 - 2c\zeta - i[4c\zeta(1-\zeta)]^{\frac{1}{2}} \end{aligned} \quad (12)$$

by using the auxiliary variable $\zeta = \frac{1 - \cos k\Delta x}{2}$ of previous lecture. It follows that:

$$|g(\zeta)|^2 = 1 + 4c\zeta(c-1). \quad (13)$$

Let us determine for which values of Δt the VN condition $|g|^2 \leq 1$ is fulfilled. For $a > 0$ we have $c = a\Delta t/\Delta x > 0$, so the upwind scheme is stable if:

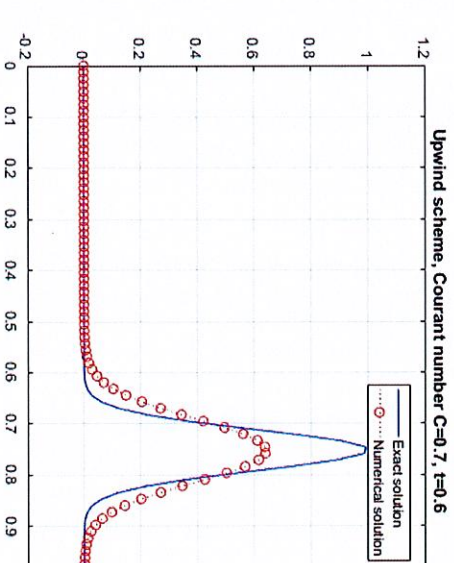
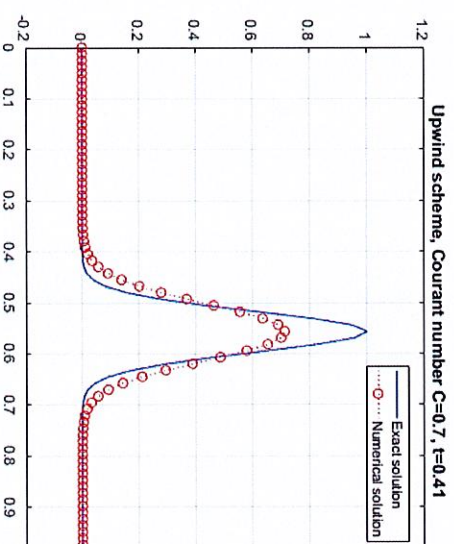
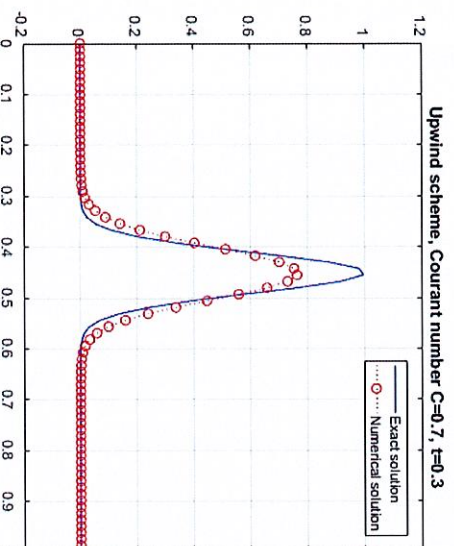
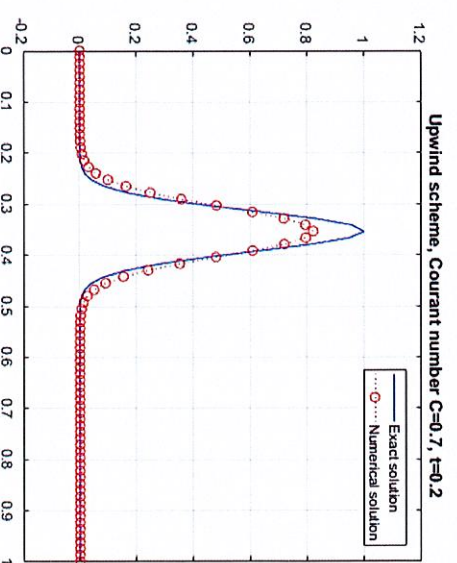
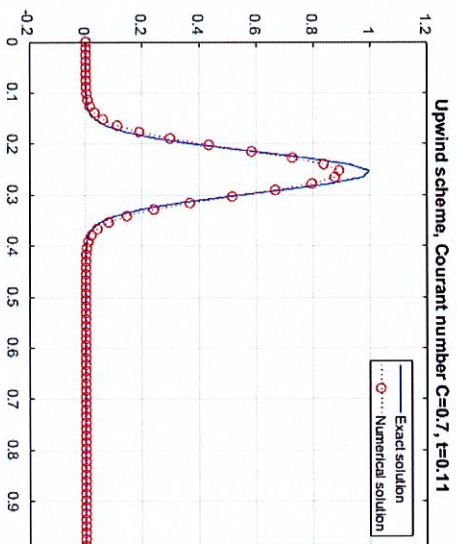
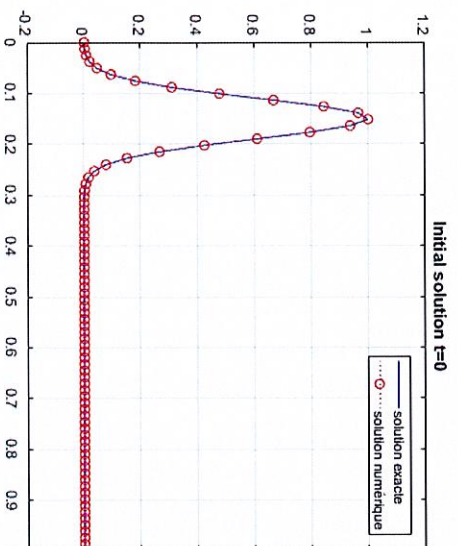
$$c \leq 1 \iff \Delta t \leq \frac{\Delta x}{a} \quad (14)$$

This stability condition is called the CFL condition, after the mathematicians who found it: Courant, Friedrich and Levy.

3-2) Upwind methods and the concept of numerical dissipation.

As shown in Fig. 3, the use of upwind discretisation $\delta_x^- u$ instead of central discretisation $\delta_x^0 u$ in the FE scheme has the property to stabilize the computations. However, the amplitude

Transport Eq. : Upwind scheme



Upwind scheme for the transport Eq. with $C=0.7$ (CFL condition is verified)

- Scheme is stable if the CFL condition is verified: $C=a \Delta t/\Delta x < 1$
- Numerical diffusion is high ($O(\Delta x)$)
- Numerical dispersion is low ($O(\Delta x)$)
- FD solution is monotone (no oscillations)

Fig. 3

of the solution wave is damped as time grows. (6)

To understand this phenomenon, let us rewrite the upwind operator as:

$$\begin{aligned} S_u^- u_i &= \frac{u_i - u_{i-1}}{\Delta u} = \frac{u_{i+1} - u_{i-1}}{2 \Delta u} - \frac{u_{i+1} - 2u_i + u_{i-1}}{2 \Delta x} \\ &= S_u^0 u_i - \frac{\Delta u}{2} S_{xx}^0 u_i \end{aligned} \quad (15)$$

Hence, the upwind scheme (9) can be written as:

$$\underbrace{\frac{u_i^{n+1} - u_i^n}{\Delta t}}_{\text{FE central scheme (4)}} + \alpha \underbrace{S_{xx}^0 u_i^n}_{\text{artificial dissipation term}} - \varepsilon \underbrace{S_{xx}^0 u_i^n}_{\text{artificial dissipation term}} = 0, \quad (16)$$

with $\varepsilon = \alpha \frac{\Delta u}{2}$ the artificial viscosity.

Eq. (16) shows that the upwind scheme can be interpreted as the central scheme (4) to which is added a dissipation term $\varepsilon S_{xx}^0 u_i^n$ that stabilizes the computation.

As shown in Fig 3, the artificial dissipation term has the consequence to damp the numerical solution, whereas the exact solution keeps the same amplitude for all time $t > 0$.

~~Nevertheless~~ Nevertheless, the numerical dissipation of the upwind scheme is too strong, since the truncation error is only first-order accurate. Actual CFD codes use dissipative schemes of higher order, such as the Lax-Wendroff scheme.

4) The Lax-Wendroff (LW) scheme.

(7)

As the ~~upwind~~ upwind scheme (9), the LW scheme proposes a stabilization of the central scheme (4) by the concept of artificial dissipation. We will see that the LW is more accurate than the upwind scheme, and that its truncation error is second-order.

The LW scheme is not built by the M.O.L., but instead by considering the following Taylor expansion of the solution:

$$u(x_i, t_n + \Delta t) = u(x_i, t_n) + \Delta t \frac{\partial u}{\partial t}(x_i, t_n) + \frac{\Delta t^2}{2} \frac{\partial^2 u}{\partial t^2}(x_i, t_n) + O(\Delta t^3) \quad (16)$$

Then, the time-derivatives are transformed into space-derivatives using the transport eq. (2):

$$\frac{\partial u}{\partial t} = -a \frac{\partial u}{\partial x}, \quad (17a)$$

$$\begin{aligned} \frac{\partial^2 u}{\partial t^2} &= \frac{\partial}{\partial t} \left(\frac{\partial u}{\partial t} \right) = \frac{\partial}{\partial t} \left(-a \frac{\partial u}{\partial x} \right) = -a \frac{\partial}{\partial x} \left(\frac{\partial u}{\partial t} \right) \\ &= +a^2 \frac{\partial^2 u}{\partial x^2}, \end{aligned} \quad (17b)$$

by using Schwartz' theorem. After inserting (17) into (16), we get:

$$u(x_i, t_{n+1}) = u(x_i, t_n) - a \Delta t \frac{\partial u}{\partial x}(x_i, t_n) + \frac{a^2 \Delta t^2}{2} \frac{\partial^2 u}{\partial x^2}(x_i, t_n) + O(\Delta t^3) \quad (18)$$

After replacing spatial derivatives with central FD formulas, (18) reads:

$$u_i^{n+1} = u_i^n - a \Delta t \sum_n^0 u_i^n + \frac{a^2 \Delta t^2}{2} \sum_{nn}^0 u_i^n, \quad (19) \quad (8)$$

that finally yields the Lax-Wendroff scheme:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + a \sum_{nn}^0 u_i^n - \underbrace{\frac{a^2 \Delta t}{2} \sum_{nn}^0 u_i^n}_{\varepsilon > 0: \text{artificial viscosity}} = 0. \quad (20)$$

The main properties of the LW scheme are:

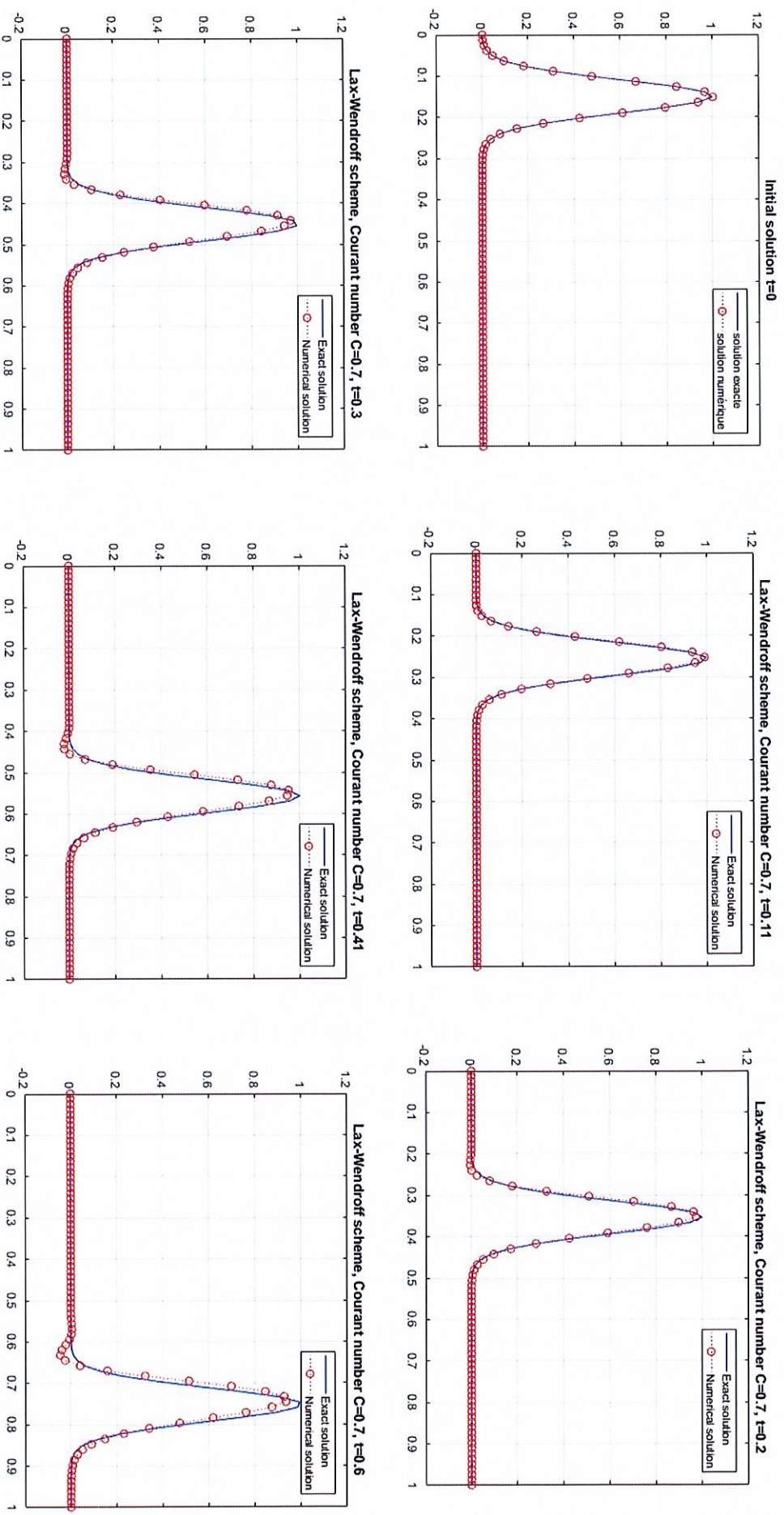
- It is an explicit 2-level scheme.
- By construction, the truncation error is $\tau_i = O(\Delta t^2, \Delta x^2)$.
- For convection velocity $a > 0$ or $a < 0$, the artificial viscosity $\varepsilon = \frac{a^2 \Delta t}{2} > 0$: LW scheme is valid ~~whatever~~ whatever the sign of the waves.
- The scheme is stable under the CFL condition: (see exercise)

$$\frac{|a| \Delta t}{\Delta x} < 1.$$

According to the computations shown in Fig. ⁴~~3~~,
 L-W scheme gives much less numerical damping
 than the upwind scheme, thanks to its higher ac-
 -curacy.

Transport Eq. : Lax-Wendroff scheme

Fig. 4: LW scheme for the transport Eq. with $C=0.7$ CFL condition verified)



- Scheme is stable if the CFL condition is verified: $C=a \Delta t/\Delta x < 1$
- Numerical diffusion is low ($O(\Delta x^2)$)
- Numerical dispersion is higher than for Upwind scheme (FD solution travels slower)
- FD solution is not monotone (small oscillations are visible)

5) Implementation of boundary conditions (BCs) (9)

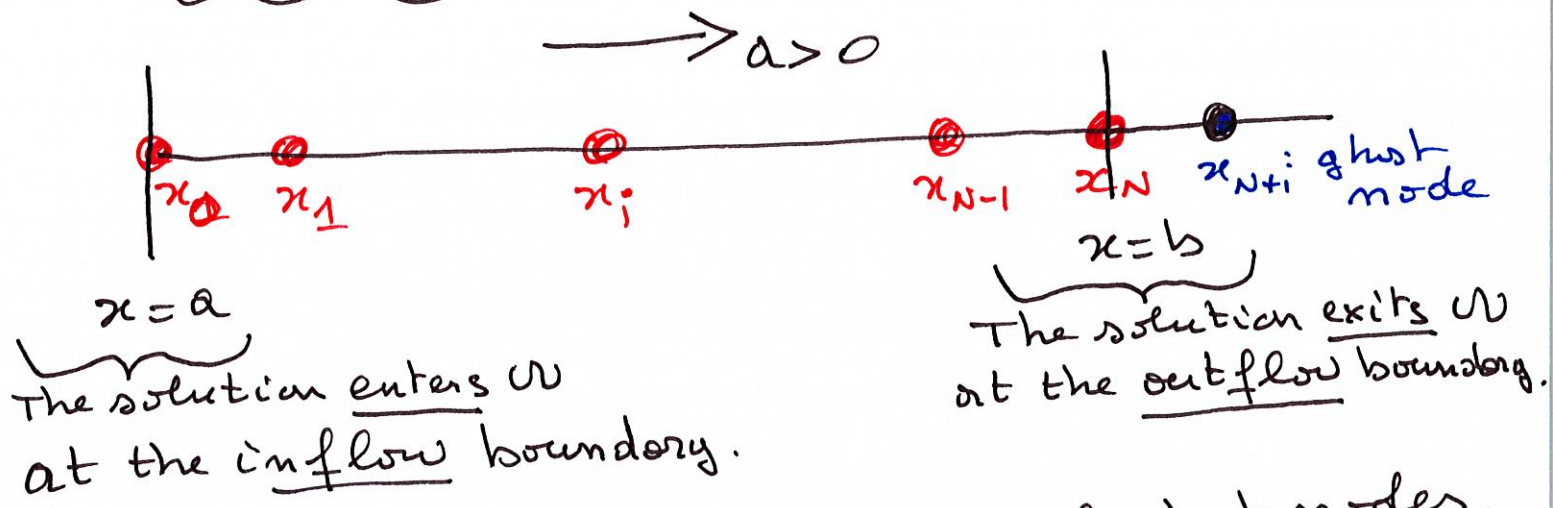


Fig. 4: FD mesh and the concept of ghost nodes.

Fig. 4 sketches the mesh $\{x_i = i\Delta x; i=0, \dots, N\}$ of the domain $U = [a, b]$. To obtain a unique solution of the transport eq. (2), we need to provide a BC at the inflow boundary $x=a$:

$$u(x=a, t) = u_{in}(t), \quad \forall t > 0, \quad (21)$$

but at the outflow boundary $x=b$, the mathematical theory states there is no need to provide a BC.

Hence, the numerical solution with the upwind scheme (5) is:

$$u_0^{n+1} = u_{in}(t_{n+1}), \quad (22)$$

and at the other nodes $i=1, \dots, N$:

$$u_i^{n+1} = u_i^n + a \Delta t \left(\frac{u_i^n - u_{i-1}^n}{\Delta x} \right). \quad (23)$$

When we consider the LW scheme (10), we observe we can only apply the scheme at the inner nodes $i = 1, \dots, N-1$:

$$u_i^{n+1} = u_i^n - \alpha \Delta t \left[\frac{u_{i+1}^n - u_{i-1}^n}{2 \Delta x} \right] + \frac{\alpha^2 \Delta t^2}{2} \left[\frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2} \right] \quad (24)$$

but we cannot use it at the outflow node x_N , because the central FD formulas use the value u_{N+1}^n which is outside the domain Ω (see Fig. 4). To solve this problem, CFD codes use the concept of "ghost nodes".

- Add the ghost node x_{N+1} as in Fig. 3

- Define the value of u_{N+1}^n by linear extrapolation:

$$u_N = \frac{u_{N-1} + u_{N+1}}{2} \Rightarrow u_{N+1} = 2u_N - u_{N-1}$$

- After inserting this value of u_{N+1} in the LW scheme (24), we obtain:

$$u_N^{n+1} = u_N^n - \alpha \Delta t \left[\frac{u_N^n - u_{N-1}^n}{\Delta x} \right] + \frac{\alpha^2 \Delta t^2}{2} \left[\frac{0}{\Delta x^2} \right]$$

and we recover thus the upwind scheme (23) at the outflow boundary x_N .

6) Summary of the Lecture

(11)

Central FD for the convection term give rises to spurious oscillations of the numerical solution, which may lead to instabilities.

For this reason, CFD codes prefer to use upwind methods that stabilizes the computations thru artificial dissipation.

It is preferable to use upwind methods of better accuracy than 1st-order. For example in ANSYS Fluent:

- Upwind Second-order

- Q.U.I.C.K

The CFL condition for stability:

$$C = a \frac{\Delta t}{\Delta x} < 1$$

is common to most of upwind methods. The CFL condition is less restrictive than the condition $\Delta t < \Delta x^2 / 2\nu$ for explicit methods for the heat equation. It is ok to use explicit time-stepping for convection, but not for diffusion.

①

Lecture notes:

Finite-volume Method (FVM)

1) Introduction

For the convection-diffusion balance equation:

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\vec{v} \phi) - \nabla \cdot (\nu \nabla \phi) = q \quad (1)$$

with non-constant coefficients:

- Convection velocity $\vec{v} = \begin{pmatrix} u(x, y) \\ v(x, y) \end{pmatrix}$

- viscosity $\nu = \nu(x, y)$

The previous lectures focused on the basics of the FVM in 1D. We have seen:

- Truncation error of a scheme

- Discretization formulas for:

- diffusion: centred method $\delta_{xx}^0 \phi$

- Convection: centred method $\delta_x^0 \phi$

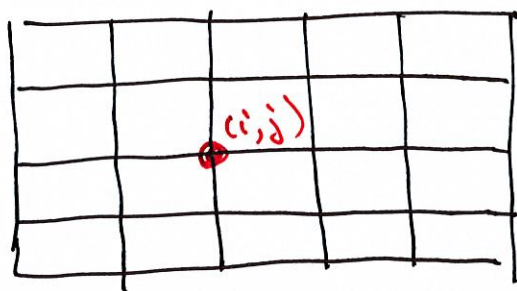
- upwind method δ_x^-, δ_x^+

- Time-integration based on the M.O.L. using EP1, ER1, CN, etc ...

- Stability of the numerical solution (e.g.: CFL criterion)

②

However, the FDM method applies only to Cartesian mesh:



and its application is restricted to simple geometries.

What remains to be described in this lecture:

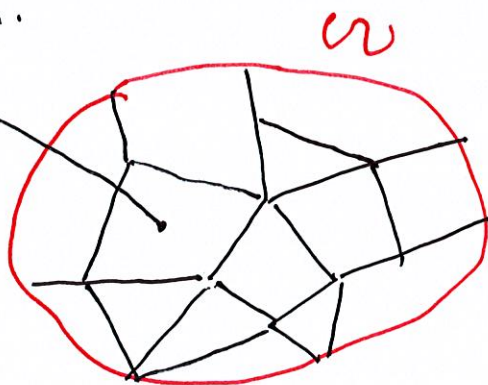
- Application to "real-life" problems (fluid flows, thermal problems) in 2D or 3D

- Numerical methods applicable to complex geometries: the FVM method.



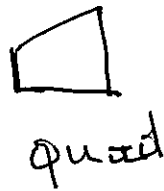
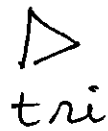
FVM has now a widespread use for engineering applications. Its main characteristic is to use the integral formulation of balance equations, in contrast to the FDM that uses its differential formulation.

CV
(Control Volume)



③ • In the FVM, the physical domain Ω is divided in CVs, and the balance equation is discretized in each CV $\Rightarrow$ 1 algebraic equation per CV.

• FVM can be applied to CVs of arbitrary shape :



and thus can be applied to complex geometries, in analogy to FDM.

• Nevertheless, FDM and FVM have common grounds. For simple applications using:

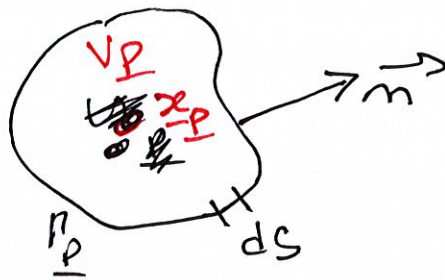
- Cartesian grids
- Constant coefficients (e.g. $\vec{u}$, ν in Eq. (1))
- uniform grid spacing Δx , Δy

the FVM and FDM are identical. Thus, the basic notions of FDM (truncation error, stability, upwinding, ...) apply to FVM. For more general application, it is preferable to use the FVM.

④ 2) Principles of the FVM

• Volume of CV_P is V_P

• Center is x_P



The basics of FDM are explained for the convection-diffusion Eq. (1), ~~in the~~ which is written in integral form in the CV CV_P by using the flux-divergence theorem:

$$\forall \text{ vector } \vec{V}, \quad \int_{CV_P} \nabla \cdot \vec{V} dV = \int_{P_P} \vec{V} \cdot \vec{n} dS \quad (2)$$

The integral form of (1) is thus:

$$\underbrace{\frac{d}{dt} \int_{CV_P} \phi dV}_{\text{Volume integrals}} + \underbrace{\int_{P_P} (\vec{F}^c - \vec{F}^d) \cdot \vec{n} dS}_{\text{Surface integrals}} = \underbrace{\int_{CV_P} q dV}_{\text{Volume int.}} \quad (3)$$

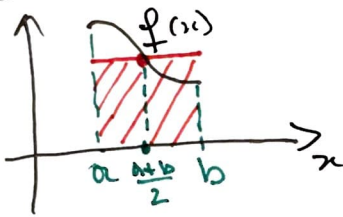
with the "numerical" fluxes $\vec{F} = \begin{pmatrix} F_u \\ F_v \end{pmatrix}$

• Convective fluxes $\vec{F}^c = \phi \vec{v} = \begin{pmatrix} \phi u \\ \phi v \end{pmatrix} \quad (4a)$

• Diffusive fluxes $\vec{F}^d = \nu \vec{\nabla} \phi = \begin{pmatrix} \nu \partial \phi / \partial x \\ \nu \partial \phi / \partial y \end{pmatrix} \quad (4b)$

⑤ The integrals in Eq. (3) will be approximated using the mid-point quadrature, which is a second-order discretization.

Definition (Mid-point quadrature in 1D)



$$\int_a^b f(x) dx \approx (b-a) f\left(\frac{a+b}{2}\right) \quad (5)$$

With this quadrature, the volume integrals are easily calculated, for example:

$$\int_{V_P} \Phi dV \approx V_P \Phi_P \Rightarrow \Phi_P \approx \frac{1}{V_P} \int_{V_P} \Phi dV \quad (6)$$

which means that Φ_P (value at center x_P) can be considered as the volume average of Φ in V_P .

After discretization of the volume integrals, the convection-diffusion Eq. (3) reads in V_P :

$$\frac{d}{dt} \Phi_P + \frac{1}{V_P} \left[\vec{F}_c(\Phi_P) - \vec{F}_d(\Phi_P) \right] \cdot \vec{n} dS = q_P, \quad (7)$$

and the finite-volume discretization will be achieved by the approximation of the numerical fluxes (4) (convective and diffusive) on the FV mesh.

$$\frac{d}{dt} \Phi_P + \frac{1}{V_P} \int_{P_P} \vec{F} \cdot \vec{n} dS = q_P, \text{ with } \vec{F} = \vec{F}_c - \vec{F}_d \quad (7)$$

⑥

3) Flux evaluation on a Cartesian mesh.

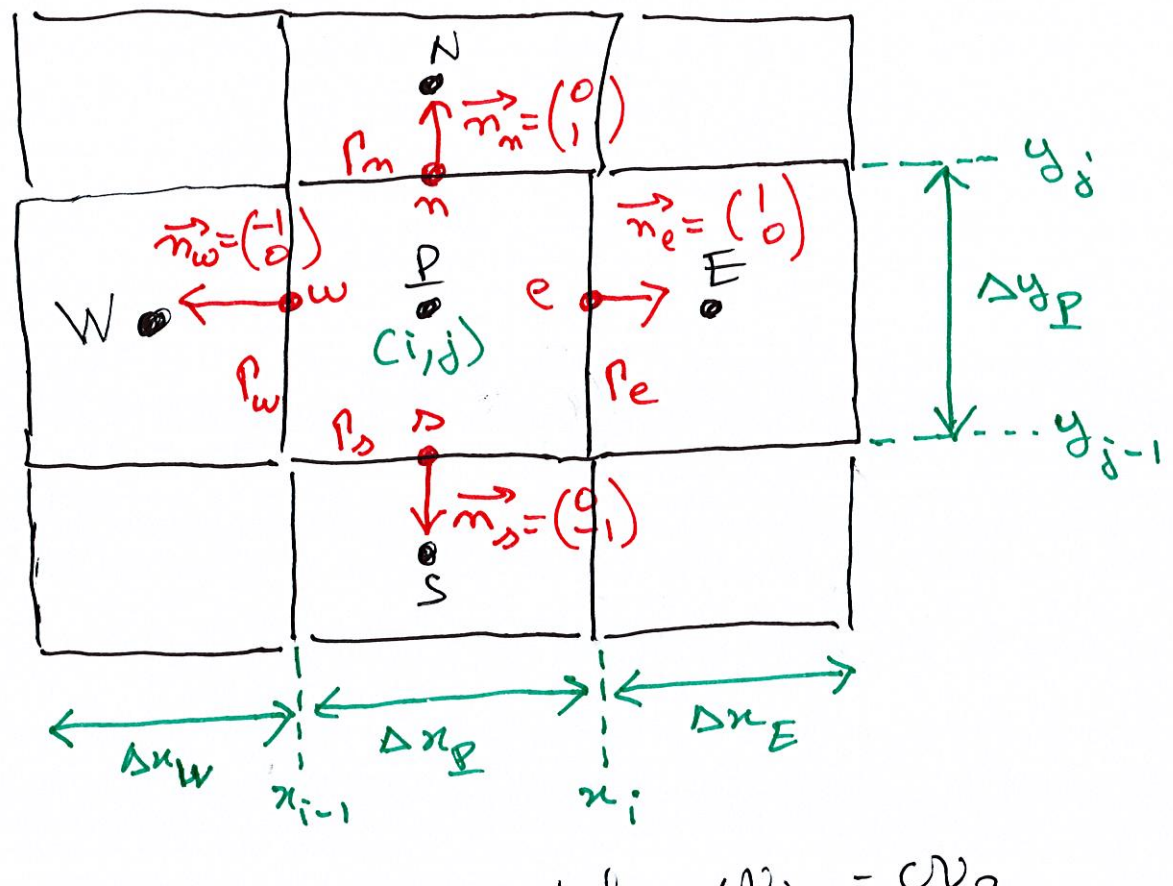


Fig 1: Compass notation used for $U_{i,j} = CV_P$

The flux evaluation is best described by using the compass notation of Fig. 1. In particular:

- The volume of cell CV_P is $V_P = \Delta x_P \Delta y_P$. Its center is x_P
- Cell P shares frontiers ("faces") with 4 neighboring cells: E, W, N and S . (Capital letters used for CVs)
- Thus, the surface Γ_P of cell CV_P is decomposed in 4 elementary faces Γ_f :

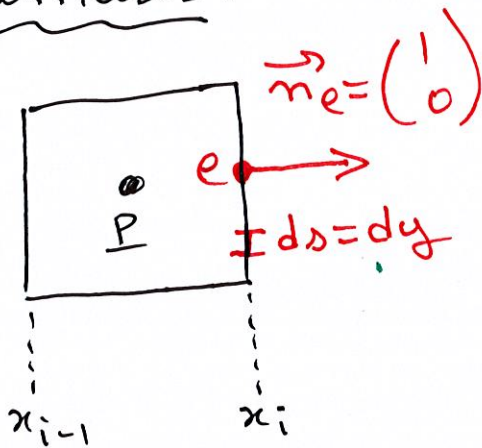
$$(8) \Gamma_P = \Gamma_S \cup \Gamma_W \cup \Gamma_N \cup \Gamma_E$$
 (small letters for faces)
- Each face Γ_f has a center (x_f) and a normal vector $\vec{n}_f$

⑦ With these definitions, the numerical flux (4) is decomposed as:

$$\frac{1}{V_P} \int_{P_P} \vec{F} \cdot \vec{n} dS = \frac{1}{V_P} \sum_{\text{face } f} \int_{P_f} \vec{F} \cdot \vec{n}_f dS \quad (9)$$

$$= \frac{1}{\Delta x_P \Delta y_P} \left[\int_{P_e} F_x dy - \int_{P_w} F_x dy + \int_{P_n} F_y dx - \int_{P_s} F_y dx \right]$$

Remark 2: The calculation on face P_e is done as:



$$\int_{P_e} \vec{F} \cdot \vec{n}_e dS = \int_{P_e} \begin{pmatrix} F_x \\ F_y \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} dS$$

$$= \int_{P_e} F_x(x_i, y) dy$$

$$\approx \Delta y_P F_x(x_e)$$

by using mid-point rule.

After a similar calculation on all 4 faces, the numerical flux (9) reads finally:

$$\frac{1}{V_P} \int_{P_P} \vec{F} \cdot \vec{n} dS = \frac{(F_x)_e - (F_x)_w}{\Delta x_P} + \frac{(F_y)_n - (F_y)_s}{\Delta y_P} \quad (10)$$

notation for $F_x(x_e)$

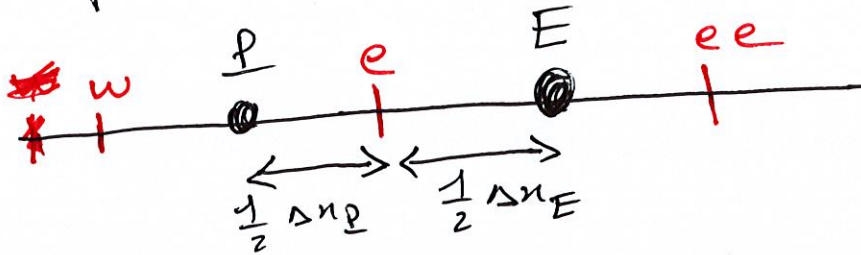
This equation involves the flux vectors $\vec{F}$ calculated at the face centers of CV_P . To achieve the FVM discretization, we need to interpolate them at the cell centers P, W, E, N and S . We need to distinguish the case of diffusive $\vec{F}^D$ and convective $\vec{F}^C$ fluxes.

⑧ 4) Interpolation of diffusive fluxes $\vec{F}^d = \nu \nabla \Phi$

When $\vec{F}$ is given by (4b), Eq. (10) reads:

$$\frac{1}{V_P} \int_{\vec{r}_P} \vec{F} \cdot \vec{n} dS = \frac{\nu_e \left(\frac{\partial \phi}{\partial x} \right)_e - \nu_w \left(\frac{\partial \phi}{\partial x} \right)_w}{\Delta x_P} + \frac{\nu_n \left(\frac{\partial \phi}{\partial y} \right)_n - \nu_s \left(\frac{\partial \phi}{\partial y} \right)_s}{\Delta y_P} \quad (11)$$

in the case of non-constant viscosity, it is evaluated at ~~cell~~ face centers.



For calculating $\left(\frac{\partial \phi}{\partial x} \right)_e$ at face center x_e , we consider the centred FD formula:

$$\left(\frac{\partial \phi}{\partial x} \right)_e = \frac{\phi_E - \phi_P}{x_E - x_P} = \frac{\phi_E - \phi_P}{\frac{1}{2} \Delta x_P + \frac{1}{2} \Delta x_E} \quad (12)$$

and use similar FD formulas for the other 3 faces, i.e. at the north face:

$$\left(\frac{\partial \phi}{\partial y} \right)_n = \frac{\phi_N - \phi_P}{\frac{1}{2} \Delta y_N + \frac{1}{2} \Delta y_P} \quad (13)$$

The resulting FV discretization of (11) is equivalent to the centred FD formulas $\delta_{xx}^0 \Phi_P$ and $\delta_{yy}^0 \Phi_P$. In effect, in the case of uniform mesh ($\Delta x, \Delta y = \text{const}$) and constant viscosity, the first term in the RHS of (11) simplifies as:

9

$$\frac{v_e \left(\frac{\partial \phi}{\partial n} \right)_e - v_w \left(\frac{\partial \phi}{\partial n} \right)_w}{\Delta x_P} = v \left[\frac{\frac{\phi_E - \phi_P}{\Delta x} - \frac{\phi_P - \phi_W}{\Delta x}}{\Delta x} \right]$$

$$= v \left(\frac{\phi_E - 2\phi_P + \phi_W}{\Delta x^2} \right) \quad (14)$$

$$= v \delta_{xx}^0 \Phi_P.$$

The same simplification applies in the y direction.
 As a conclusion, in the case $\Delta x, \Delta y$ and $v = \text{constant}$, FVM is strictly equivalent to FDM, and the lectures on FDM tells that the discretization is second-order accurate.

5) Interpolation of convective fluxes $\vec{F}^c = \phi \vec{u}$

When $\vec{F}$ is given by (4a), Eq. (10) reads:

$$\frac{1}{V_P} \int_{P_P} \vec{F} \cdot \vec{n} dS = \frac{u_e \Phi_e - u_w \Phi_w}{\Delta x_P} + \frac{v_n \Phi_n - v_s \Phi_s}{\Delta y_P}.$$

We assume the convection velocity is known at face centers; it remains to interpolate the face values Φ_e, Φ_w , etc. ...

There are several possibility to perform these interpolation which leads, as in the FDM, to central and upwind formulas.

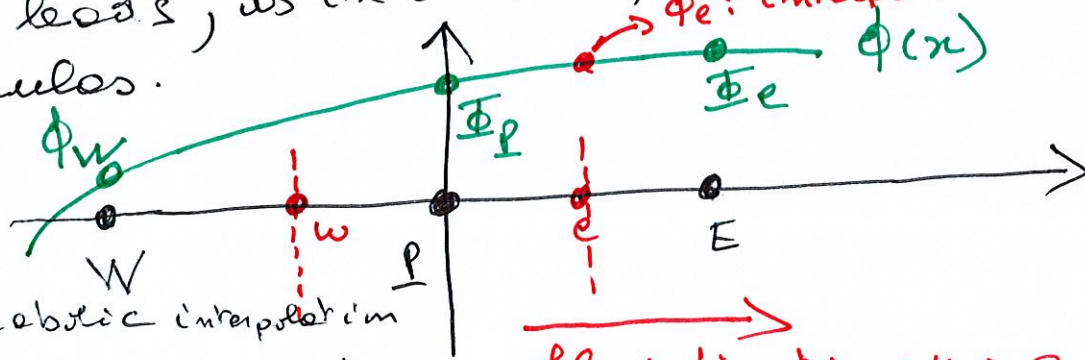


Fig 2: Parabolic interpolation of ϕ_e by the QUICK scheme

flow direction: $u_e > 0$

We will discuss the interpolation ~~of~~ ϕ_e as shown in Fig. 2.

5-1) Central Differencing scheme: (CDS).

(10)

The ~~simplest~~ ^{most common} ~~method~~ ^{method} is the central interpolation:

$$\phi_e \approx \frac{1}{2} \phi_E + \frac{1}{2} \phi_P \quad (16)$$

such as the flux on face e needs:

$$\mu_e \phi_e \approx \mu_e \left(\frac{1}{2} \phi_E + \frac{1}{2} \phi_P \right). \quad (17)$$

This FVM scheme is usually called CDS and, in the case of a constant coefficient ($\vec{v} = c\vec{e}$; $\Delta x, \Delta y$ constant) is equivalent to the central FD $\nabla^2 \phi_P$ since, in this case, the first term in Eq. (15) simplifies as:

$$\begin{aligned} \frac{\mu_e \phi_e - \mu_w \phi_w}{\Delta x_P} &= \frac{\mu \left(\frac{1}{2} \phi_E + \frac{1}{2} \phi_P \right) - \mu \left(\frac{1}{2} \phi_W + \frac{1}{2} \phi_P \right)}{\Delta x} \\ &= \mu \frac{\phi_E - \phi_W}{2\Delta x} = \mu \nabla^2 \phi_P. \end{aligned} \quad (18)$$

From the FDM lecture, we can infer that CDS is $O(\Delta x^2)$ but, when convection dominates ($Pe = \frac{u \Delta x}{\nu} \gg 1$), ~~leads~~ to large oscillations in the solution. In this case, it is advised to use upwind-biased interpolation.

5-2) Upwind-biased schemes.

These schemes are based on the flow direction (see Fig. 2) and distinguish the case where $\mu_e < 0$ and $\mu_e > 0$.

(9) S-2-1) Upwind Differencing Scheme (UDS)

This scheme consists in a constant interpolation with respect to the local flow direction u_e :

$$\phi_e = \begin{cases} \phi_P & \text{if } u_e \geq 0, \\ \phi_E & \text{if } u_e < 0. \end{cases} \quad (19)$$

In the case of a constant ~~local~~ ^{constant} velocity, the UDS is equivalent to the upwind FD scheme S_n^- or S_n^+ .

For example if $u_w = u_e = u > 0$, the first term in

Eq. (15) simplify as:

$$\frac{u_e \phi_e - u_w \phi_w}{\Delta x} = \frac{u \phi_P - u \phi_w}{\Delta x} = u S_n^- \phi_P. \quad (20)$$

For the FD lectures, we can state that UDS gives no oscillation, in contrary to CDS, but ~~is very diffusive~~ is only $O(\Delta x)$ and leads to large diffusive errors.

In order to enhance ~~the~~ the accuracy of UDS, we need to consider upwind-biased interpolation with polynomials of higher-order ^{than} ~~than~~ the constant polynomial used in (19).



S-2-2 | QUICK Scheme

Q.U.I.C.K = Quadratic Upwind Interpol for Convection Kinematics.

For $u > 0$

this scheme uses a ~~quadratic~~ quadratic polynomial (ie: $p(x) = ax^2 + b + c$) ~~that interpolates ϕ at~~ points E, P and W, as shown in Fig. (2) to obtain the interpolation formula:

$$\phi_e = \frac{3}{8} \phi_E + \frac{3}{4} \phi_P - \frac{1}{8} \phi_W, \quad (21)$$

as seen ~~in~~ in Exercise B.2. This interpolation can be written as:

$$\phi_e = \frac{\phi_{i+1} + \phi_i}{2} - \frac{\Delta x^2}{8} \delta_{xx}^0 \phi_i \quad (22)$$

which shows that the scheme scheme stabilizes the CDS scheme by adding a second-order diffusion term.

~~Due to its good~~ ~~properties of accuracy and stability~~, the QUICK has seen a widespread use in commercial CFD like such as ANSYS FLUENT.

6) Implementation of boundary conditions (BCs)

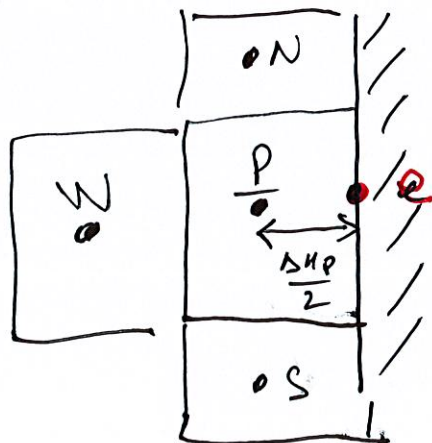


Fig. 3 sketch of the mesh near the East boundary.

Let us consider the case depicted in Fig. 3, (13)
 where the CV CV_P is next to the East boundary.
 The interpolations of fluxes at the w, s, n are
 the same as in Section 4 and 5, but the
 flux $(F_x)_e$ at the east face needs to be
 modified to take into account Dirichlet BCs:

$$\phi_e = f_e, \quad f_e \text{ is a given data} \quad (23)$$

or Neuman BCs:

$$\left(\frac{\partial \phi}{\partial n} \right)_e = g_e, \quad g_e \text{ is a given data.} \quad (24)$$

6-1) BCs in convective fluxes

For the convective flux $F_x = u \phi$, the Dirichlet
 BC (23) is easily implemented as:

$$(F_x)_e = u_e \phi_e = u_e f_e. \quad (25)$$

The implementation of the Neuman BC (24) is more
 involved since it necessitates the Taylor expansion:

$$\phi_P \approx \phi_e - \frac{\Delta x_P}{2} \left(\frac{\partial \phi}{\partial x} \right)_e. \quad (26)$$

From this Taylor expansion, we obtain:

$$(F_x)_e = u_e \phi_e = u_e \left(\phi_P + \frac{\Delta x_P}{2} g_e \right) \quad (27)$$



6-2) BCs in diffusive fluxes

For the diffusive flux $F_n = \nu \frac{\partial \phi}{\partial n}$, the Neuman BC (24) is easily implemented as:

$$(F_n)_e = \nu_e \left(\frac{\partial \phi}{\partial n} \right)_e = \nu_e g_e. \quad (28)$$

The Dirichlet boundary condition ~~requires~~ requires the Taylor expansion (26), ie:

$$\left(\frac{\partial \phi}{\partial n} \right)_e \approx \frac{\phi_e - \phi_P}{\Delta x_P/2} \quad (29)$$

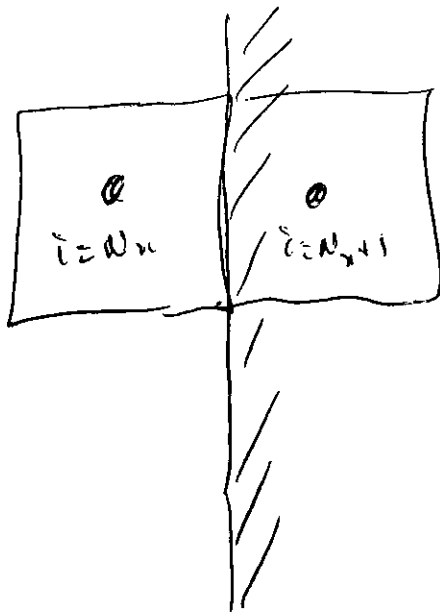
which yields finally:

$$(F_n)_e = \nu_e \frac{\phi_e - \phi_P}{\Delta x_P/2}. \quad (30)$$

Remark: In a computer code, these BCs are efficiently implemented by using the concept of ghost-cells described in the next pages.

Utilisation des Ghost-cell pour
imposer les conditions aux limites

(1)



Direchlet $\frac{\phi_{N_{n+1}} + \phi_{N_n}}{2} = f_e$

$\Rightarrow \phi_{N_{n+1}} = 2f_e - \phi_{N_n}$

Neumann

$\frac{\phi_{N_{n+1}} - \phi_{N_n}}{\Delta u} = g_e$

$\Rightarrow \phi_{N_{n+1}} = \phi_{N_n} + \Delta u g_e$

• Utilisation des Formules centrées usuelles:

$\rightarrow$ Cas Neumann:

$F_e^c = (u_e \phi_e) = u_e \frac{\phi_{N_{n+1}} + \phi_{N_n}}{2} = u_e \phi_{N_n} + \frac{\Delta x}{2} g_e$

$F_d^e = \left(\frac{\partial \phi}{\partial n} \right)_e = \frac{\phi_{N_{n+1}} - \phi_{N_n}}{\Delta u} = g_e$

on retrouve les conditions aux limites du cas.

Sect. 15.3, 4.

→ Cos Dirichlet

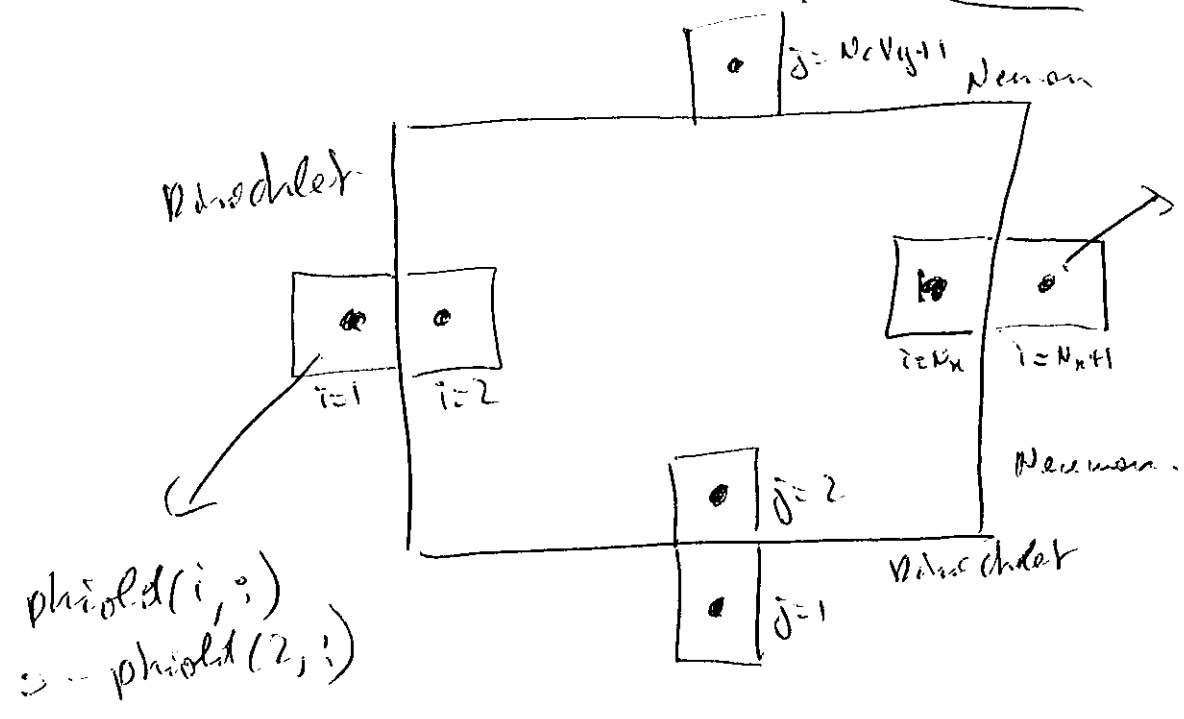
$$F_c^e = u \frac{\phi_{N_{x+1}} + \phi_{N_x}}{2} = u f_e.$$

$$F_d^e = \left(\frac{\partial \phi}{\partial u} \right)_e = \frac{\phi_{N_{x+1}} - \phi_{N_x}}{\Delta x} = \frac{f_e - \phi_{N_x}}{\Delta x / 2}$$

idem cons.

Application on TP

$$\phi_{old}(:, N_{cvg}+1) = \phi_{old}(:, N_{cvg})$$

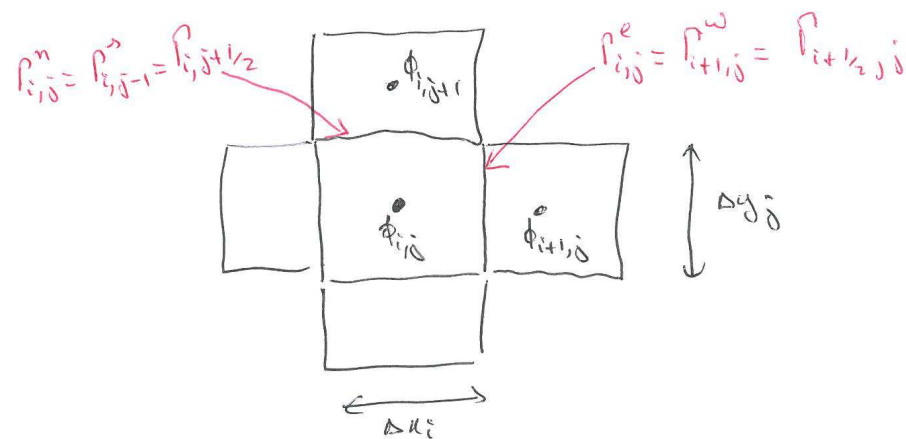


19/04/2014

Section 15.5 Global (and local) Conservation properties of FVM

FV Discretization of convection-diffusion eqn reads:

for cell $\omega_{i,j}$ (1)
$$\frac{d}{dt} \phi_{i,j} + \frac{F_{i,j}^e - F_{i,j}^w}{\Delta x_i} + \frac{F_{i,j}^n - F_{i,j}^s}{\Delta y_j} = q_{i,j}$$



An important property that FVM should verify is the local conservation of fluxes

$$F_{i,j}^e = F_{i+1,j}^w, \quad F_{i,j}^n = F_{i,j+1}^s \quad (2)$$

This is usually verified by even discretization, e.g. CDS discretization of convective flux:

$$F_{i,j}^e = u \left(\frac{\phi_{i+1,j} + \phi_{i,j}}{2} \right) = F_{i+1,j}^w \equiv F_{i+1,j}$$

We can thus drop the supers notation and rewrite (2) simply (1) as:

$$\frac{d}{dt} \phi_{i,j} + \frac{(F_{i+1/2,j} - F_{i-1/2,j})}{\Delta x_i} + \frac{(F_{i,j+1/2} - F_{i,j-1/2})}{\Delta y_j} = q_{i,j} \quad (3)$$

The local conservation of numerical fluxes (2) is a property of FVM as long as the PDE is written in conservative form:

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (F(\phi)) = q \quad (4)$$

$$\Rightarrow \frac{d}{dt} \int_{\omega} \phi dV + \int_{\partial} F(\phi) dS = \int_{\omega} q dV \quad (5)$$

Example 1: the convection-diffusion eqn in conservative form is:

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\vec{\sigma} \phi - \nu \nabla \phi) = q \quad (6)$$

The non conservative (or convective) form is:

$$\frac{\partial \phi}{\partial t} + \vec{\sigma} \cdot \nabla \phi - \nu \nabla^2 \phi = q \quad (7)$$

The convective and conservative are analytically identical when $\nabla \cdot \vec{\sigma} = 0$, but the discretization of (7) may lead to the violation of local conservation property (2) $\Rightarrow$ creation of artificial source or sink terms in the FV equations.

See !

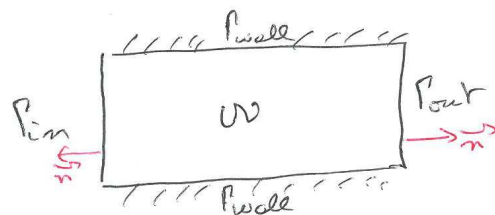
• Exercise

• lecture on NS eq^t in allocated arrangement

(3)

Example 2:

Conservation of mass in incompressible flows



The continuity eq^t reads:

$$\nabla \cdot \vec{v} = 0 \quad \forall \underline{x} \in U \quad (8a)$$

If we integrate this equation on whole domain U :

$$(8b) \quad \underbrace{\int_P \vec{v} \cdot \vec{n} ds}_{\dot{m}: \text{flow rate across } P} = 0 \quad \text{for } P = P_{in} \cup P_{out} \cup P_{wall}$$

which leads to:

$$\dot{m}_{in} - \dot{m}_{out} = 0 \quad (9)$$

since $\dot{m}_{wall} = 0$ due to B.C.s.

Hence, the global conservation of mass in U leads to a compatibility condition for B.C.s at P_{in} and P_{out} :

$$\int_{P_{in}} \vec{v}_{in} \cdot \vec{n} ds = \int_{P_{out}} \vec{v}_{out} \cdot \vec{n} ds \quad (10)$$

Rq1: Conservation of mass is one of the global compatibility properties of NS eq^ts (along with conservation of momentum and kinetic energy). (4)

Compatibility condition (10) must be verified analytically to lead to a well-posed problem (solution exists and is unique). If the FV discretized version of this condition is violated, the discrete ~~NS~~ FV eq^ts may not be solvable $\Rightarrow$ blow up of the solution.

What is the equivalent of (8)-(10) in FVM?

In a cell $U_{i,j}$, Eq. (8a) leads to:

$$\dot{m}_{i,j} = \dot{m}_e^{i,j} - \dot{m}_w^{i,j} + \dot{m}_n^{i,j} - \dot{m}_s^{i,j} = 0 \quad (11)$$

(the relation between flow rate and $\vec{v}_{i,j}$ will be detailed in NS lectures). If local conservation (2) holds in all cells $U_{i,j}$, i.e.:

$$\dot{m}_e^{i,j} = \dot{m}_w^{i+1,j}, \quad \dot{m}_n^{i,j} = \dot{m}_s^{i,j+1}$$

then discrete continuity eq^t reads:

$$\dot{m}_{i+1/2,j} - \dot{m}_{i-1/2,j} + \dot{m}_{i,j+1/2} - \dot{m}_{i,j-1/2} = 0 \quad (12)$$

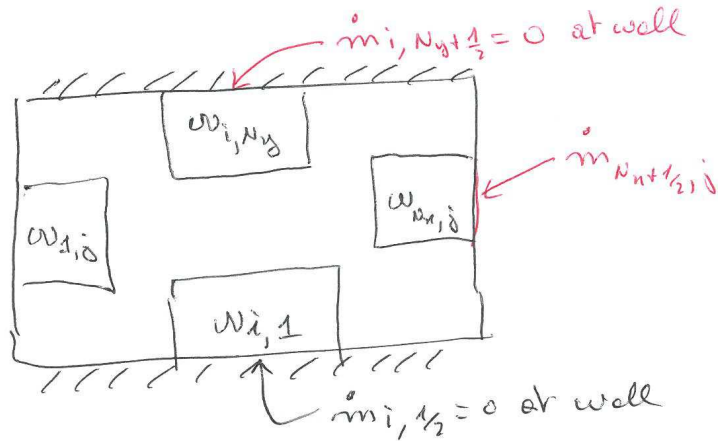
Now, discretization of continuity eq^t (8b) in whole fluid domain (8b) reads:

$$\sum_{i,j} \dot{m}_{i,j} = 0$$

$$\sum_{i=1}^{N_x} \sum_{j=1}^{N_y} (\dot{m}_{i+1/2,j} - \dot{m}_{i-1/2,j} + \dot{m}_{i,j+1/2} - \dot{m}_{i,j-1/2}) = 0 \quad (5)$$

this sum cancels out except at boundaries P_{in} and P_{out} :

$$\underbrace{\sum_{j=1}^{N_y} \dot{m}_{1/2,j}}_{\text{mass flow at } P_{in}} - \underbrace{\sum_{j=1}^{N_y} \dot{m}_{N_x+1/2,j}}_{\text{mass flow at } P_{out}} = 0 \quad (13)$$



Eqⁿ (13) is a discretization of ~~mass~~ global mass ~~conservation~~ (10).

Hence, FVM is globally conservative if:

- Local conservation of flux numerical fluxes (2) is verified. (easy)
- BCs are "correctly" imposed at boundary cells. (more difficult)

If this property is not verified, some terms in FVM eq^s act as source or sinks, which alter the discrete conservation of quantities such as mass, momentum or kinetic energy.

⇒ See lecture on FVM for NS eq^s.

DFV Discretization of the Navier-Stokes eqns: ①

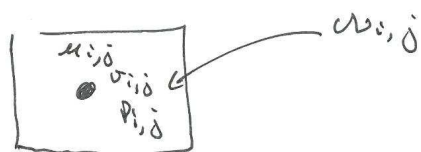
Navier-Stokes eqns:

18 Feb 2015
STG

1) Arrangement of the variables on a mesh:

2 popular arrangements:

1-1) Collocated grid arrangement.



flow variables
- All ~~variables~~ or stored ("collocated") at the center of the cell.

⇒ Easy to generalize this approach to non-structured grids and to apply techniques for ~~best~~ ^{straightforward} lectures.

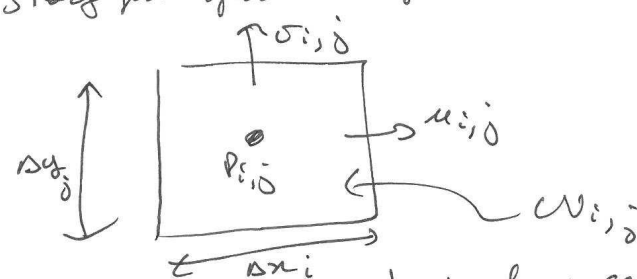
⇒ Most of commercial codes who use non-structured grids use collocated arrangement.

⇒ Problems related to ~~acoustic~~ ^{acoustic} properties.

② Staggered as we shall see next.

There is no problem related to pressure oscillations as we shall see next, which is related to the interpolation of pressure in collocated mesh. A common ^{straight} way to cure this pressure oscillations is to use a different interpolation of pressure (Moukhanov's interpolation of $\nabla \phi$ & χ see next lecture), which has the effect of ~~decreasing~~ ^{decreasing} the acoustic properties of the collocated approach.

2) Staggered grid arrangement ②



Velocities are stored at face centers.
- Pressure and other ~~scalar~~ ^{mass} scalar variables (Temperature, etc...) are stored at cell centers.

⇒ The continuity equation is discretized in a straightforward way without any interpolation, i.e.:

$$\dot{m}_e \approx \Delta y u_{i,j}$$

⇒ It is the first method for solving the NS eqn in velocity-pressure formulation. (MAC Meth of Harlow & Welch, 1964).

⇒ Generalization of mass, momentum, etc. are easily verified, ^{if it is a numerical method of NS eqns} there is no problem of pressure oscillations as in the collocated arrangement.

But:

• Since the variables ($u_{i,j}$, $v_{i,j}$, $w_{i,j}$) are stored at different locations in the cell, we must define $\neq$ control volumes for are needed for the momentum eqn in each direction, mass eqn.
⇒ Coupling the staggered method is more involved.

In the beginning of CFD 1970-1980, there were conceptual difficulties in generalizing the staggered method to non-structured mesh (3)
 $\Rightarrow$ Collocated methods were preferred, that is why current commercial CFD codes (e.g. FLUENT) still uses collocated method, despite of its flaws.

$\Rightarrow$ Staggered method or still be preferred choice when structured grid are used.

2) FVM for NS eqt on collocated grid (4)

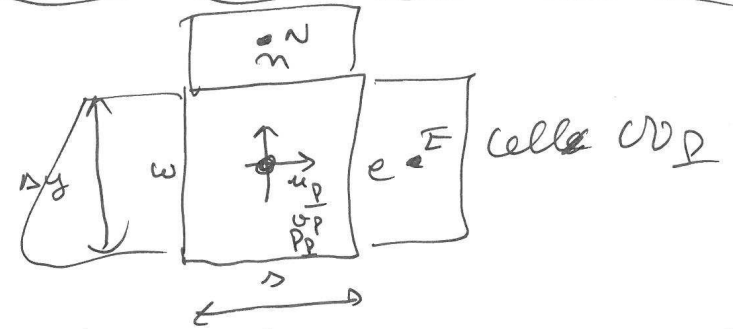


Fig 1: cell CV_P
 We will consider the case of a uniform mesh with cell $\Delta x \times \Delta y \Rightarrow V_P = \Delta x \Delta y$. The u - v variables are stored in the cell centers, i.e.:

$$\phi_P = \frac{1}{V} \int_{CV_P} \phi dV \quad \text{for } \phi = u, v, \text{ and } p.$$

The integral (or conservative) form of the NS eqts is:
 For $h = u, v$ ($\rho(u_h)_P = \rho_P$, $(u_h)_P = u_P$):
 Conservation of momentum:
 (1) $\frac{d}{dt} \left(\rho u \right)_P + \frac{1}{V_P} \int_{\Gamma_P} (F_h^c - F_h^d) \cdot \vec{n} dS + \frac{1}{V} \int_{\Gamma_P} p \vec{e}_h \cdot \vec{n} d\Omega = f_h$

for $h = u, v$.

where for putting the pressure term in integral form we have used Green's theorem:

$$\int_{CV} \nabla \phi dV = \int_{\Gamma} \phi \vec{n} dS \Leftrightarrow \int_{CV} \frac{\partial \phi}{\partial x_h} dV = \int_{\Gamma} \phi \vec{e}_h \cdot \vec{n} d\Omega$$

where $\vec{e}_h$ is the unit vector in the h th direction.

The convective and diffusive fluxes are respectively: (5)

$$(2) \quad F_h^c = \rho \vec{v} u_h, \quad F_h^d = V \frac{D}{Dt} = \frac{1}{2} (\underline{D} \vec{v} + \vec{v} \underline{D})$$

Conservation of mass:

$$\int_{\Omega} \vec{v} \cdot \vec{n} dS = 0. \quad (3)$$

To these NS equations, we must add boundary conditions on the boundary Γ (see lecture ---), and initial conditions for the velocity at $t=0$.

The NS eqt (1)-(3) needs to be discretized on the CV Ω_P of Reg. 1. Some remarks on the N-S eqts (1)-(3):

They are really models to be used with caution except that: there is no exact equation for pressure,

and no initial conditions.

There is no initial conditions for pressure.

The incompressibility constraint:

$$\int_{\Omega} \vec{v} \cdot \vec{n} dS = 0 \Leftrightarrow \nabla \cdot \vec{v} = 0.$$

must be fulfilled at each time. This is a real difference with the con-diff eqt, and we will see next that the pressure will be computed such as this incompressibility constraint is fulfilled at each time-step.

2-1) Discretization of the continuity eqt: (6)

By applying the midpoint rule to the 4 faces of control volume Ω_P , we get:

$$\int_{\Omega_P} \vec{v} \cdot \vec{n} dS \approx \Delta y (u_e - u_w) + \Delta x (v_n - v_s) \quad (4)$$

The velocity at the face centers need now to be interpreted to the center of the CV. This is done straightforwardly with central interpolation:

$$u_e \approx \frac{1}{2} (u_P + u_E), \quad v_n \approx \frac{1}{2} (v_P + v_N) \quad (5)$$

We finally get:

$$\begin{aligned} \int_{\Omega_P} \vec{v} \cdot \vec{n} dS &= \Delta y \frac{(u_E - u_W)}{2} + \Delta x \frac{(v_N - v_S)}{2} \\ &= V \left(\delta_x^0 u_{i,j} + \delta_y^0 v_{i,j} \right) \quad (6) \end{aligned}$$

by using stencil notation for central FD formulae.

→ discrete system of PDEs: $\approx \approx \approx$.

2-2) Secu-discretization of the momentum eqts:

For Eq. (1), the discretization of conv & diff flux follow the lines of lecture 1 on con-diff eqt, what remains is the pressure term; which reads in the x-dim:

$$\begin{aligned} \frac{1}{V} \int_{\Omega_P} p \vec{e}_x \cdot \vec{n} dS &= \frac{1}{V} \left[\int_{\Omega_e} p dV - \int_{\Omega_w} p dS \right] \\ &\approx \frac{1}{\Delta x} [p_e - p_w] \quad (7) \end{aligned}$$

We use standard central interpolation:

$$P_c \approx \frac{1}{2} (P_E + P_W) \quad , \quad P_w \approx \frac{1}{2} (P_E + P_W) \quad (8)$$

and we finally get:

$$\frac{1}{V} \int_{P_P} p \cdot \vec{e}_x \cdot \vec{n} ds \approx \frac{(P_E - P_W)}{2\Delta x} = \sum_n P_{i,j} \quad (9)$$

and for the y direction:

$$\frac{1}{V} \int_{P_P} p \cdot \vec{e}_y \cdot \vec{n} ds \approx \frac{(P_N - P_S)}{2\Delta y} = \sum_n P_{i,j} \quad (10)$$

In matrix formulation, the sea-surface NS eqs read:

$$\frac{d}{dt} U + \left(\frac{G_h(U) - K_h}{N_h} \right) U + G_h P = F \quad (11a)$$

$$D_h U = 0 \quad (11b)$$

where the unknowns are:

$$U = (u_1, u_2, \dots, u_N, v_1, \dots, v_N) = (u_l, v_l)_{l=1 \dots N_h}$$

$$P = (P_1, \dots, P_N)_{l=1 \dots N_h}$$

Since the collocated mesh has $l=1, \dots, N$ unknowns ($N=N_h N_y$).

Note that the velocity mesh have $2N$ unknowns in 2D.

The matrix G_h represents the discrete pressure gradient, such as:

$$G_h P = \begin{bmatrix} \delta_x^0 \\ \delta_y^0 \end{bmatrix} P = \begin{pmatrix} \delta_x^0 P_e \\ \delta_y^0 P_e \end{pmatrix}$$

The matrix D_h is the discrete divergence, such as:

$$D_h U = \begin{bmatrix} \delta_x^0 & \delta_y^0 \end{bmatrix} U = (\delta_x^0 u_l + \delta_y^0 v_l)_l$$

3) An "explicit" time-stepping scheme for the NS eqs - the Semi-Discrete EP1 (SDEP1).

Since Eq. (11a) is very similar to the cur-diff Eq, we let's apply the EP1 scheme:

$$\xrightarrow{2N \text{ eqs}} \frac{U^{n+1} - U^n}{\Delta t} + N_h U^n + G_h P^n = F^n \quad (12a)$$

Now, observe that the divergence eq has not ~~temporal~~ time derivatives. It must be resolved at each current time-step t^{n+1} , i.e:

$$N_h P^n \rightarrow D_h U^{n+1} = 0$$

this system runs $2N$ eqs

Note that this systema possess $3N$ unknowns: $U^{n+1} = (u^{n+1}, v^{n+1})$ and P^n .

this system runs $3N$ eqs, so we must consider $3N$ unknowns $U^{n+1} = (u^{n+1}, v^{n+1})$ and P^n which must be solved simultaneously. As we shall see next, this system is explicit only for the velocity, ~~and we~~ must derive a ~~pressure~~ eq a Poisson eq for the pressure in order to get P^n .

3-1] the EP1 as a fractional step scheme

For ease of computation, it is customary to rewrite EP1 scheme (12a) - (12b) as a 2-step schm, by introducing the auxiliary variable $\tilde{U}$, as:

$$\frac{\tilde{U} - U^n}{\Delta t} + N_h U^n = F^n \quad (13)$$

$$\text{Then: } \begin{cases} \frac{U^{n+1} - \tilde{U}}{\Delta t} + G_h P^n = 0 & (14a) \\ D_h U^{n+1} = 0 & (14b) \end{cases}$$

After cancel of (14a) and (14b) to get rid of U^{n+1} , we obtain the pressure eqⁿ for the pressure. (8)

$$D_h G_h P^n = \frac{1}{\Delta t} \sum D_h \vec{U}. \quad (15)$$

Algorithm (EPI schem for NS eq^t)

The soln of EPI schem (13)-(14) is actually done by the follow steps:

1) Initialization: at $t=0$: U^0 is given by initial cond.

2) Evolve from t^n to $t^{n+1} = t^n + \Delta t$. The soln (U^{n+1}, P^n) is obtained by the current steps:

a) Apply Eq. (13) to get ~~a predicted~~ the predicted velocity $\vec{U}$:

$$\vec{U} = U^n + \Delta t (F^n - A_h U^n) \quad (16a)$$

Note that $\vec{U}$ is not divergence-free, i.e. $\nabla \cdot \vec{U} \neq 0$.

b) Compute P^n by solving the pressure eqⁿ (15).

c) Now that P^n is known, use Eq (14a) to get U^{n+1} :

$$U^{n+1} = \vec{U} - \Delta t G_h P^n \quad (16b)$$

It can be verified a posteriori that U^{n+1} is divergence-free, i.e. $D_h U^{n+1} = 0$.

This algorithm is the prototype of the pressure-correction method, where P^n is computed and used to correct the velocity (16b) so that it is divergence-free at each time step. It means mass is conserved along the time integration. If it is not the case, the time-integration would blow up.

As a consequence, the EPI schem is explicit only

3-2) the Pressure Poisson Equation for the Pressure (PPE) (2)

Last paragraph showed that the EPI scheme (12) is explicit only for the velocity, and the pressure must be solved "implicitly" by the solution of the PPE:

(17) $A_h D_h G_h P^{n+1} = \sum D_h \vec{U}$, where $A_h = D_h G_h$ and $\sum$ depends on the time-stepping scheme. The explicit nature of the calculation of P is a common characteristic of all numerical methods for incompressible flows that needs at any time step to solve eqⁿ (17).

The pressure operator $A_h = D_h G_h$ is the product of the discrete divergence and gradient, so it should look like a discrete gradient. Let's have a formula:

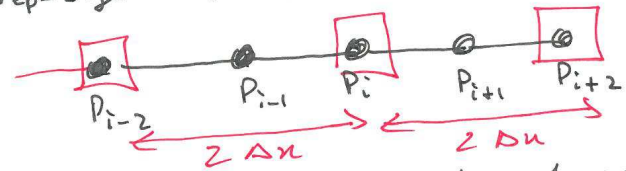
$$D_h G_h P^k = \begin{bmatrix} \delta_x^0 & \delta_y^0 \end{bmatrix} \begin{bmatrix} \delta_x^0 \\ \delta_y^0 \end{bmatrix} = \delta_x^0 \delta_x^0 P^k + \delta_y^0 \delta_y^0 P^k. \quad (17)$$

Note that $\delta_x^0 \delta_x^0 \neq \delta_{xx}^0$, but rather:

$$\delta_x^0 (\delta_x^0 P_{i,j}) = \frac{P_{i+3,j} - 2P_{i,j} + P_{i-2,j}}{(2\Delta x)^2} \quad (18a)$$

$$\delta_y^0 (\delta_y^0 P_{i,j}) = \frac{P_{i,j+2} - 2P_{i,j} + P_{i,j-2}}{(2\Delta y)^2} \quad (18b)$$

which shows that A_h corresponds to a discrete Laplacian of step-size $2\Delta x, 2\Delta y$, which skips a grid point:



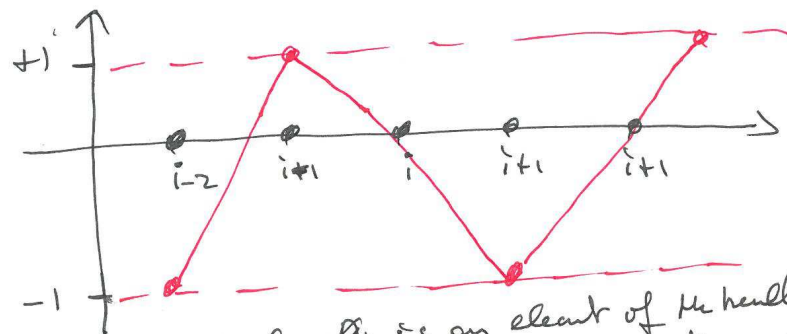
the skipping of every other grid point is dangerous, as we shall see next.

First, let us recall that the term: (10)

$$\pi_{i,j}^1 = cte \quad (15)$$

Cancels the product $A_h \pi_{i,j}^1$. It is the discrete version of the fact that pressure is defined up to a constant value. Math, it means that $\pi_{i,j}^1$ lies in the kernel of values A_h .

Let us also observe that term $\pi_{i,j}^2 = (-1)^i$, (20a)



also ~~cancels~~ A_h is an element of the kernel of A_h . Same for the term $\pi_{i,j}^3 = (-1)^j$ and $\pi_{i,j}^4 = (-1)^{i+j}$. (20b)

Hence, the solution of the PPE (17) has the form:

$$P_{i,j} + \sum_{m=1}^4 \alpha_m \pi_{i,j}^m$$

and is thus polluted by the spurious pressure terms π^2, π^3 and π^4 that make goods a note-to-note pressure oscillations $\Rightarrow$ this is the so-called Pressure checkerboard phenomenon.

Hence, the ~~use~~ collocation formulation yields a pressure solution which has oscillate, rendering this this method ~~is~~ unusable in practice. There are nonetheless 2 remedies:

- 1) Use the staggered formulation, where we see in the next lecture. (11)
- 2) Modify the ~~discretization~~ ~~in order~~ discretization of the continuity eq (4) in order to get a PPE which is free of pressure oscillations. This is the basis of the unimodular weighted unimodular of $\nabla \cdot \mathbf{u}$ (see lecture) which is used by all commercial codes that use collocation approach.

25 Feb 2015

(12)

Q1) FVM for NS eq^t on staggered grid

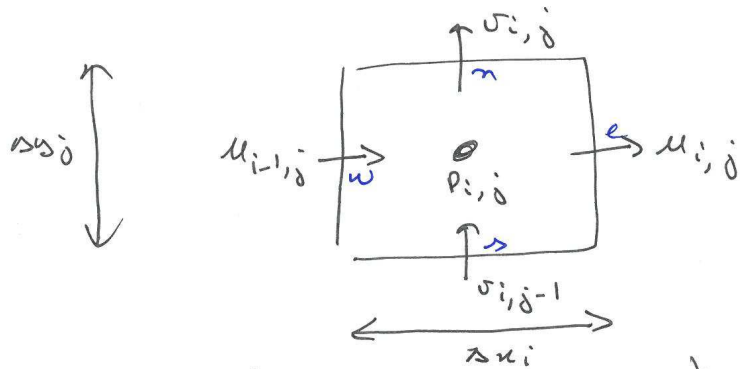


Fig 1: Control volume V_P for continuity eq^t:

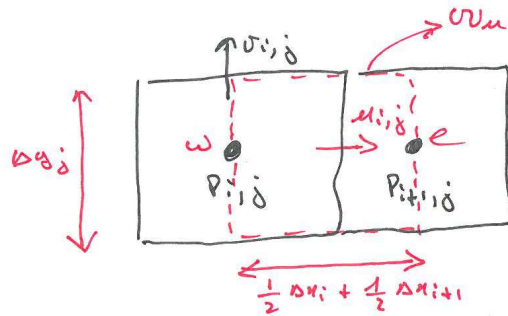


Fig 2: Staggered control volume V_u for the u -momentum eq^t. Its width $V_u = (\frac{1}{2}\Delta x_i + \frac{1}{2}\Delta x_{i+1}) = \Delta x_u$ on a uniform mesh.

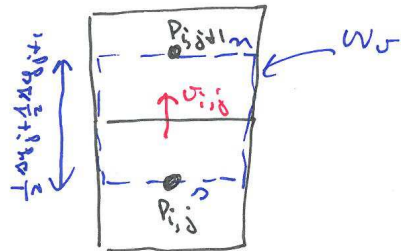


Fig 3: Staggered control volume V_v for the v -momentum eq^t. $V_v = \Delta x_v \Delta y_v$ on a uniform mesh.

(13)

Q1-1) Discretization of the continuity eq^t

As in Sect 2.1, the discretization of the continuity eq^t is:

$$\int_{P_P} \vec{v} \cdot \vec{n} dS \approx \Delta y (u_e - u_w) + \Delta x (v_n - v_s) \quad (21)$$

there is no need to do further simplification since $u_e = u_{i,j}$, $v_n = v_{i,j}$, etc... so we finally get:

$$\int_{P_P} \vec{v} \cdot \vec{n} dS \approx V_P (\bar{S}_x u_{i,j} + \bar{S}_y v_{i,j}) \quad (22)$$

by using the FD formulas:

$$\bar{S}_x \phi_{i,j} = \frac{\phi_{i,j} - \phi_{i-1,j}}{\Delta x}, \quad \bar{S}_y \phi_{i,j} = \frac{\phi_{i,j} - \phi_{i,j-1}}{\Delta y} \quad (23)$$

N.B: It does not mean that upwind biased formulae are used for calculating the diffusion divergence! everything is computed at face center with central formulae.

Q1-2) Semi-discretization of the momentum eq^t

The u -momentum eq^t needs to be discretized on CV V_u of Fig 2, ie:

$$\frac{d}{dt} u_{i,j} + \frac{1}{V_u} \int_{P_u} (F_x^c - F_x^d) \cdot \vec{n} dS + \frac{1}{V_u} \int_{P_u} \vec{p} \cdot \vec{e}_n \cdot \vec{n} dS = 0 \quad (24)$$

The discretization of F_x^c and F_x^d will be detailed in an exercise.

The pressure gradient needs to be approximated by:

$$\frac{1}{V_n} \int_{P_n} \vec{p} \cdot \vec{e}_n \cdot \vec{n} dS = \frac{1}{\Delta x \Delta y} \left[\frac{\int p dx}{\Delta y P_e} - \frac{\int p dy}{\Delta y P_w} \right] \quad (24)$$

$$= \frac{1}{\Delta x} (P_{i+1,j} - P_{i,j}) = \delta_n^+ P_{i,j} \quad (25)$$

As for the continuity eq^t, there is no need to make further interpolatⁿ $\Rightarrow$ big difference with the collocated arrangement (see Sect. 2.2).

Note that the pressure gradient for the y-moment eq^t (Fig. 5) reads:

$$\frac{1}{V_o} \int_{P_o} \vec{p} \cdot \vec{e}_y \cdot \vec{n} dS = \delta_y^+ P_{i,j} \quad (26)$$

3-3) The ED1 scheme on the PDE for staggered mesh.

As for the collocated mesh (Sect 3.), we will use the ED1 scheme (12), or equivalently the fractional-step sch^m (13)-(14), where now the pressure and divergence unknown are:

$$D_h = \begin{bmatrix} \delta_n^- & \delta_n^- \end{bmatrix}, \quad G_h = \begin{bmatrix} \delta_n^+ \\ \delta_y^+ \end{bmatrix} \quad (27)$$

and the PDE (15) reads: pressure the following pressure eqⁿ:

$$D_h G_h p_{\ell} = (\delta_n^- \delta_n^+ + \delta_y^- \delta_y^+) p_{\ell} \quad (28)$$

and it is well known that (see FD exercise 1)

$$\delta_n^- \delta_n^+ = \delta_{nn}^0, \quad \delta_y^- \delta_y^+ = \delta_{yy}^0 \quad (29)$$

So that the PDE now looks like a standard Poisson equation with step-size Δx and Δy :

$$\frac{P_{i+1,j} - 2P_{i,j} + P_{i-1,j}}{\Delta x^2} + \frac{P_{i,j+1} - 2P_{i,j} + P_{i,j-1}}{\Delta y^2} = \frac{1}{\Delta t} (\delta_n^- \bar{u}_{i,j} + \delta_y^- \bar{v}_{i,j}) \quad (30)$$

We can now readily observed that the spurious pressure terms (20) of the collocated method are no more part of the kernel, only $\bar{u}_{i,j} = c_k$ cancels Eq. (30).
the physical pressure mode.

$\Rightarrow$ There is no spurious pressure oscillations in the staggered method!

Basics of FVM on unstructured 2D meshes

1) Introduction

We will review the FV discretization of the unsteady convection-diffusion equation on 2D unstructured meshes:

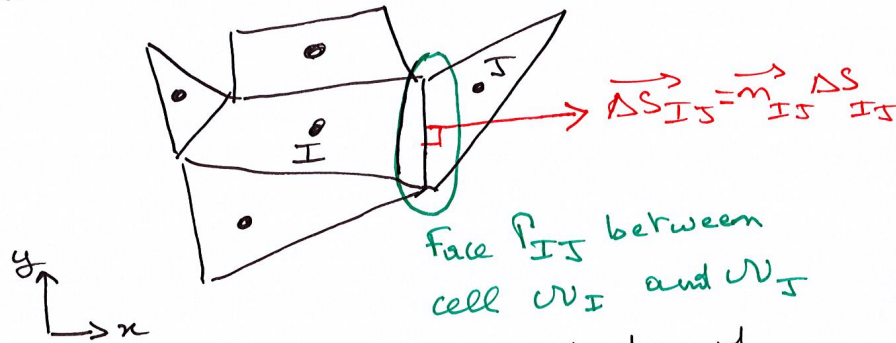


Fig 1: Example of a 2D unstructured grid. This grid is hybrid (with Tri and Quad cells).

Unstructured mesh in 2D have:

- Quadrangle cells ("Quad") with 4 faces, and 4 neighboring cells.
- Triangular cells ("Tri") with 3 faces

We consider only the cell-centred FV method, where the unknown Φ_I is located at the cell centers (centroids, barycenters).

2) Discretization of the convection-diffusion Eq^t

The integral form of the convection-diffusion in cell V_I is:

$$\frac{d}{dt} \int_{V_I} \phi dV + \int_{P_I} \vec{F} \cdot \vec{\Delta S} = \int_{V_I} q dV \quad (1)$$

where:

- $\vec{F} = \vec{F}^c - \vec{F}^d$ are the convection and diffusive flows:
 - $\vec{F}^c = \phi \vec{u}$
 - $\vec{F}^d = \nu \vec{\nabla} \phi$

• $\vec{\Delta S}$ is the area-vector of the faces of V_I , such that:

$$\vec{\Delta S}_{IJ} = \vec{n}_{IJ} \Delta S_{IJ} \quad (2)$$

- with:
- $\vec{n}_{IJ}$ is the normal vector of the face between I and J ($\|\vec{n}_{IJ}\| = 1$)
 - ΔS_{IJ} is the area of the face.

By using mid-point rule, the FV discretization of Eq. (1) is:

$$V_I \frac{d}{dt} \phi_I + \sum_{\text{faces of } V_I} \vec{F}_f \cdot \vec{\Delta S}_f = V_I q_I \quad (3)$$

where, for example,

$$\vec{F}_f \cdot \vec{\Delta S}_f = F_x(x_f) \Delta S_f^x + F_y(x_f) \Delta S_f^y$$

with $\vec{\Delta S}_f = \begin{pmatrix} \Delta S_f^x \\ \Delta S_f^y \end{pmatrix}$ and x_f is the center of the face.

Example: On the rectangular cell of Fig. 2

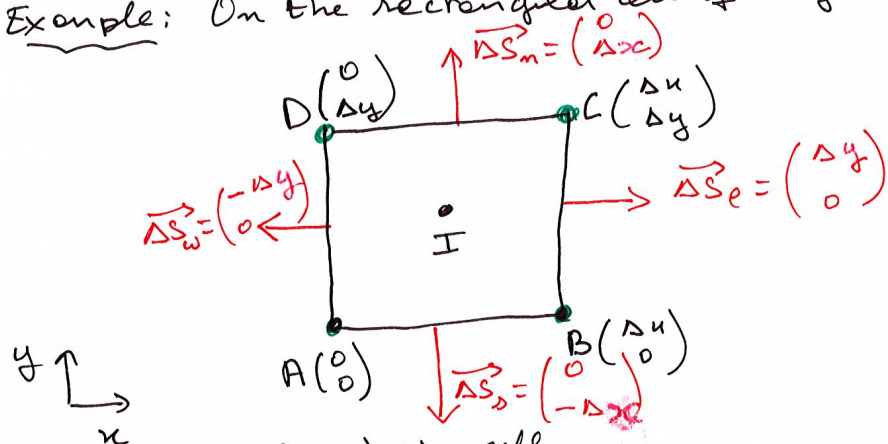


Fig. 2 A Cartesian cell
we recover the usual FV discretization on structured grid:

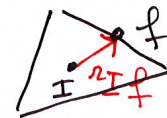
$$(4) \quad \frac{d}{dt} \phi_I + \frac{(F_x)_e - (F_x)_w}{\Delta x} + \frac{(F_y)_n - (F_y)_s}{\Delta y} = q_I$$

3) Calculation of the geometry parameters of a cell

In a structured mesh, we only need to store in the PC memory the grid size ($\Delta x, \Delta y$) to perform a FV discretization such as (4).

To perform the discretization (2) on an unstructured mesh, we need the following geometry parameters of the cells:

- Volume V_I
- N_f , the number of faces of V_I
- For all faces, the face-area vectors $\vec{\Delta S}_f$
- $\vec{r}_{If}$, the vector joining the center of cell V_I to the center of the face.



Since these geometry parameters are not as easily calculated as in a Cartesian mesh, they need to be stored in the PC memory.

example: For a 2D quad mesh, we store for each cell:

- the volume V_I
- the face-area vectors $\begin{pmatrix} \Delta S_f^x \\ \Delta S_f^y \end{pmatrix}$ for the 4 faces
- The distance $\begin{pmatrix} r_{If}^x \\ r_{If}^y \end{pmatrix}$ to the 4 faces.

$\Rightarrow$ 17 reals for the N cells of the grid.

(5)

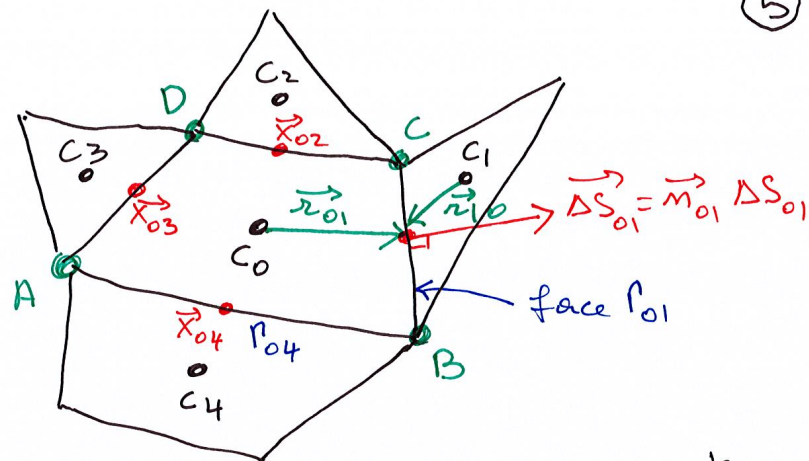


Fig. 3 Notations for calculating geometry parameters.

We consider the quad cell in Fig. 3, which has 4 neighboring cells C_1 to C_4 :

- The cell center is simply denoted C_0 .
- The volume of C_0 is V_0 .
- Cell C_0 has 4 vertices (or nodes) A, B, C and D with coordinates in the basis (x, y) : $\vec{X}_A = \begin{pmatrix} x_A \\ y_A \end{pmatrix} = x_A \vec{x} + y_A \vec{y}$.
- The face between C_0 and C_1 is denoted P_{01}
 - $\vec{DS}_{01}$ is the face-area vector (oriented from centre 0 to face P_{01})
 - $\vec{x}_{01}$ is the center of face P_{01}
 - $\vec{r}_{01}$ is the distance between center C_0 and the center of face P_{01} .

3-1) Calculation of cell volumes with the Gauss formula (6)

The volume of any type of cells in 2D and 3D is computed with the Gauss formula:

$$V_0 = \frac{1}{N_{\text{dim}}} \sum_{i=1}^4 \vec{X}_{0i} \cdot \vec{DS}_{0i} \quad (4)$$

with $N_{\text{dim}} = 2$ in 2D ($N_{\text{dim}} = 3$ in 3D).

Proof: For any position vector $\vec{X} = \begin{pmatrix} x \\ y \end{pmatrix}$ we have the identity $\nabla \cdot \vec{X} = 2$ in 2D. After integration in cell C_0 , we get $\int_{C_0} \nabla \cdot \vec{X} dV = 2 V_0$.

The flux-divergence theorem gives:

$$V_0 = \frac{1}{2} \int_{C_0} \nabla \cdot \vec{X} dV = \frac{1}{2} \int_{P_0} \vec{X} \cdot \vec{DS} = \frac{1}{2} \sum_{\text{faces}} \vec{X}_f \cdot \vec{DS}_f$$

Remark 1: The point $\vec{X}_f$ can be any point on face f ,

since:

$$\vec{X}_f \cdot \vec{DS}_f = [\vec{X}_{f'} + (\vec{X}_f - \vec{X}_{f'})] \cdot \vec{DS}_f = \vec{X}_{f'} \cdot \vec{DS}_f + \underbrace{(\vec{X}_f - \vec{X}_{f'}) \cdot \vec{DS}_f}_{\perp \vec{DS}_f}$$

$\Rightarrow$ choose $\vec{X}_f$ that gives the easiest calculations!

Remark 2: The volume of the Cartesian cell of (4)

Fig. 2 is calculated as:

$$\begin{aligned} 2V_I &= \vec{B} \cdot \vec{\Delta S}_e + \vec{C} \cdot \vec{\Delta S}_n + \vec{D} \cdot \vec{\Delta S}_w + \vec{A} \cdot \vec{\Delta S}_s \\ &= \Delta x \Delta y + \Delta x \Delta y + 0 + 0 \\ \Rightarrow V_I &= \Delta x \Delta y. \end{aligned}$$

3-2) Calculation of face-area vectors

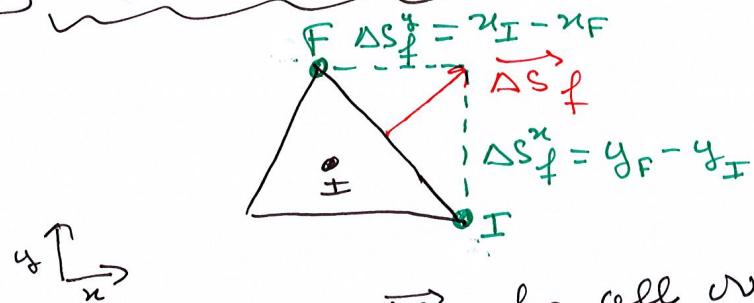


Fig. 4: Calculation of $\vec{\Delta S}_f$ for cell V_I .
The vector $\vec{\Delta S}_f$ is easily calculated from the coordinates of its initial vertex I and final vertex F

$$\vec{\Delta S}_f = \begin{pmatrix} \Delta S_f^x \\ \Delta S_f^y \end{pmatrix} = \begin{pmatrix} y_F - y_I \\ x_I - x_F \end{pmatrix} \quad (5)$$

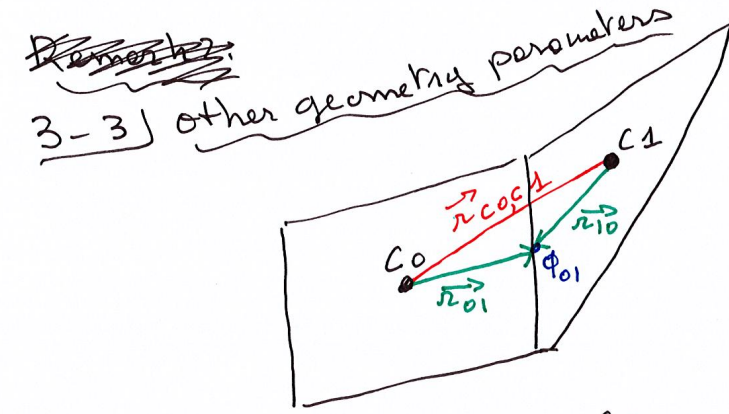
Remark 1: For the Cartesian cell of Fig. 2, Eq. (5) recovers the face-area vectors of the 4 faces, for example:

• face e: $I=B, F=C$; $\vec{\Delta S}_e = \begin{pmatrix} y_C - y_B \\ x_B - x_C \end{pmatrix} = \begin{pmatrix} \Delta y - 0 \\ \Delta x - \Delta x \end{pmatrix}$

• face n: $I=C, F=D \Rightarrow \vec{\Delta S}_n = \begin{pmatrix} \Delta y - \Delta y \\ \Delta x - 0 \end{pmatrix}$ (8)

• face w: $I=D, F=A \Rightarrow \vec{\Delta S}_w = \begin{pmatrix} 0 - \Delta y \\ 0 - 0 \end{pmatrix}$

• face s: $I=A, F=B \Rightarrow \vec{\Delta S}_s = \begin{pmatrix} 0 - 0 \\ 0 - \Delta x \end{pmatrix}$



For calculating the viscous fluxes (section 6), it will be useful to define the distance between Centroids C_0 and C_1 as:

$$\vec{r}_{C_0 C_1} = \vec{r}_{01} - \vec{r}_{10}$$

It will also be useful to interpolate the solution at face center of P_{01} . The most accurate way is to use the length-weighted average of cell values ϕ_0 and ϕ_1 :

$$\phi_{01} = d_{10} \phi_0 + (1 - d_{10}) \phi_1 \quad (6a)$$

$$\text{with } d_{10} = |\vec{r}_{10}| / |\vec{r}_{C_0 C_1}| \quad (6b)$$

A less accurate option is to use arithmetic average: (9)

$$\phi_{o1} \cong \frac{1}{2} \bar{\phi}_0 + \frac{1}{2} \bar{\phi}_1 \quad (7)$$

4) Green-Gauss calculation of gradients

An important tool for the calculation of fluxes in unstructured mesh is the calculation of the solution gradient $\nabla \phi$ at the center of the cells. This calculation is based on the Green-Gauss (GG) formula, which states that for any closed domain Ω (of boundary Γ):

$$\int_{\Omega} \nabla \phi \, dV = \int_{\Gamma} \bar{\phi} \, \vec{n} \, dS \quad (8)$$

In cell C_0 , discretization with the midpoint rule gives:

$$\nabla \bar{\phi} \Big|_{C_0} \cong \frac{1}{V_0} \sum_{\text{faces of } C_0} \bar{\phi}_f \vec{\Delta S}_f \quad (9)$$

The gradient components are thus:

$$\frac{\partial \phi}{\partial x} \Big|_{C_0} \cong \frac{1}{V_0} \sum_f \bar{\phi}_f \Delta S_f^x ; \quad \frac{\partial \phi}{\partial y} \Big|_{C_0} \cong \frac{1}{V_0} \sum_f \bar{\phi}_f \Delta S_f^y$$

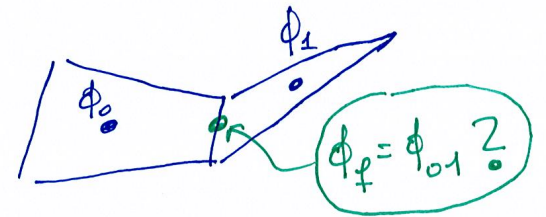
Remark: When the GG formula (9) is applied to $\bar{\phi} = 1$, one gets:

$$\sum_f \vec{\Delta S}_f = 0,$$

which states that the sum of the face area vectors sum up to zero. This property is useful for checking that in an exercise that the results obtained with Eq. (8) are correct.

The calculation of gradients with GG formula (9) will be useful for:

- Diffusive fluxes: calculation of its non-orthogonal contribution (see section 6)
- Convective fluxes: Implementation of high-order upwind discretization with the M.U.S.C.L. approach.



The values of $\bar{\phi}_f$ (or ϕ_{01} in the figure) needs now to be interpolated from values at the cell centers.

Commercial CFD codes like ANSYS FLUENT gives several different techniques to perform these

interpolation, with varying degrees of accuracy and applicability to the various type of cells encountered in actual 2D and 3D unstructured grids. Two type of interpolations are available: (11)

- Cell-based interpolation of ϕ_f
- node-based interpolation of ϕ_f

In addition, a closely related technique for computing $\nabla \phi|_{c_0}$ is available in the latest version of ANSYS FLUENT:

- Least-square calculation of gradients.

We will review these techniques now.

4-1) Cell-based interpolation of Φ_f

The easiest interpolation of ϕ_f from values at cell centers is to use the values at cell centers c_0 and c_1 by linear interpolation (6), or arithmetic interpolation:

$$\Phi_f \approx \frac{1}{2} \phi_0 + \frac{1}{2} \phi_1 \quad (10)$$

The gradient calculation Eq. (9), (10) is named G-G cell-based interpolation. It is the first gradient calculation method developed in CFD.

Its main properties are: (12)

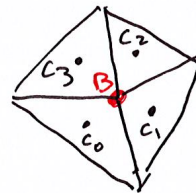
- fast calculation
- Valid for any type of 2D and 3D cells (Tri, quad, etc...)
- Low accuracy on ~~mixed~~ hybrid grids and highly distorted grids.

4-2) Node-based interpolation of Φ_f

An improvement of the cell-based interpolation is to define Φ_f from the nodal (vertex) values of cell c_0 , for example in the case of face P_{01} of Fig-3:

$$\Phi_{01} \approx \frac{1}{2} \Phi_B + \frac{1}{2} \Phi_C \quad (11)$$

The nodal values at B and C are constructed from weighted averages of cell-values surrounding the nodes, for examples:



$$\Phi_B = \frac{1}{4} (\phi_0 + \phi_1 + \phi_2 + \phi_3) \quad (12)$$

This method is called the G-G-node based calculation of gradients in ANSYS-FLUENT.

Its main properties are: (13)

- More accurate than the cell-based approach.
- More expensive to calculate.
- Recommended for tri (2D) and tet (3D) grids.
- Not available for polyhedral (pyramids, wedges) cells in 3D.

This method was for many years the best available method in ANSYS-FLUENT. In the latest version, the least-squares gradient calculation has replaced this method as the default gradient calculation available.

4-3) Least-squares (LS) gradient calculation.

This calculation is based on unidirectional Taylor expansion of the solution at cell centers (see notation of Fig. 3):

$$\Phi_1 \approx \Phi + \vec{\nabla} \Phi \cdot \vec{r}_{c0c1} + O(\|\vec{r}_{c0c1}\|^2) \quad (13)$$

If we neglect the second-order term, we get:

$$\vec{\nabla} \Phi \cdot \vec{r}_{c0c1} = \Phi_1 - \Phi_0. \quad (14)$$

Following this idea, the gradient of Φ is looked such that (14) is verified for all 4 neighboring cells of C_0 :

$$\forall i=1, \dots, 4: \vec{\nabla} \Phi \cdot \vec{r}_{c0ci} = \Phi_i - \Phi_0. \quad (15) \quad (14)$$

Eq. (15) is a linear system of 8 scalar equations for 2 scalar unknowns: $\frac{\partial \Phi}{\partial x}|_{C_0}$ and $\frac{\partial \Phi}{\partial y}|_{C_0}$.

This overdetermined system is solved by the least-squares method (see lectures on optimization).

This gradient calculation technique is called least-squares cell-based gradient evaluation in ANSYS-FLUENT. Its main properties are:

- Accuracy comparable to node-based GG (much better than cell-based GG).
- More expensive to calculate than GG methods.
- Recommended for hybrid grids.

5) Discretization of numerical fluxes

The geometry parameters of Section 3 and the gradients calculation of Section 4 are now used to calculate the convective $F_f^c = \phi \vec{u}$ and diffusive $F_f^d = \nu \vec{\nabla} \phi$ in the FV discretization given by Eq. (3):

$$\sum_{\substack{\text{faces} \\ \text{of } C_0}} \vec{F}_f \cdot \vec{\Delta S}_f = \sum_f F_f^u \Delta S_f^u + F_f^d \Delta S_f^d, \quad (16)$$

which involves, in particular, the interpolation of the fluxes at 'f' faces with respect to the unknowns at cell centers.

5-1) Calculation of diffusive fluxes $F_f^d = \nu \nabla \phi|_f$ (15)

Due to the non-orthogonality of unstructured grid, the calculation of F^d at cell faces is more involved than in Cartesian grids. On face $f = f_{01}$, the diffusive flux is calculated as:

$$F_f^d \cdot \vec{S}_f = \nu \vec{\nabla} \phi|_{o_1} \cdot \vec{S}_{o_1} = \Delta S_{o_1} \frac{\partial \phi}{\partial n}|_{o_1} \quad (17)$$

where $\frac{\partial \phi}{\partial n}|_{o_1}$ is the derivative in the direction $\vec{n}_{o_1}$.

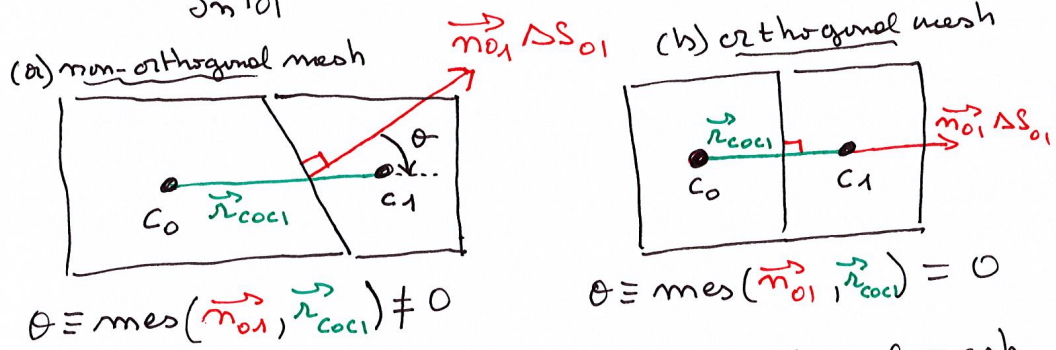


Fig. 5: Example of orthogonal and non-orthogonal mesh. For both type of mesh, $\vec{r}_{c_0c_1}$ is the line joining c_0 and c_1 , so the directional derivative in the $\vec{r}$ direction is given by the 2-point formula (2PT):

$$\frac{\partial \phi}{\partial r}|_{o_1} \approx \frac{\phi_1 - \phi_0}{\|\vec{r}_{c_0c_1}\|} \quad (18)$$

5-1-1) The 2PT approximation.

For the case of orthogonal mesh, since $\vec{n}_{o_1} \parallel \vec{r}_{c_0c_1}$ we can approximate $\frac{\partial \phi}{\partial r}|_{o_1}$ by the 2-PT formula (18)

which gives:

$$\frac{\partial \phi}{\partial n}|_{o_1} = \frac{\phi_1 - \phi_0}{\|\vec{r}_{c_0c_1}\|} \text{ in Eq. (17). } (19)$$

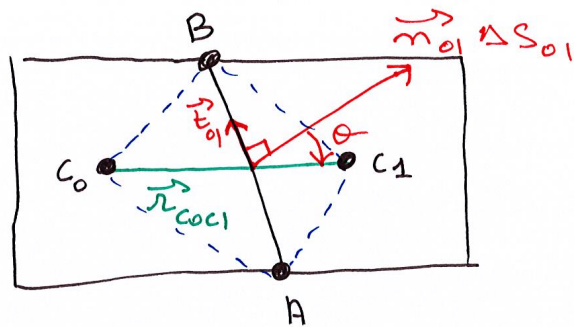
However, when the mesh is non-orthogonal (Fig. 5 left), the use of this 2-PT approximation gives large errors when the non orthogonality angle $\theta = \text{mes}(\vec{n}_{o_1}, \vec{r}_{c_0c_1})$ becomes large.

In this case, we need to add a non-orthogonal correction to the 2-PT approximation (19) by using the Diamond-scheme method (DCM)

5-1-2) The DCM approximation.

S-1-2] The DCM approximation

(17)



The name Diamond-cell method (DCM) comes from the fact that this gradient approximation was first constructed from the GG formula (8) in the diamond-shaped cell (C_0, A, C_1, B) . In these notes, the DCM formula is obtained by writing the gradient in the local basis $(\vec{n}_{o1}, \vec{t}_{o1})$ of face P_{o1} :

$$\nabla \phi|_{o1} = \left(\frac{\partial \phi}{\partial n} \right)_{o1} \vec{n}_{o1} + \left(\frac{\partial \phi}{\partial t} \right)_{o1} \vec{t}_{o1} \quad (20)$$

thus:

$$\nabla \phi|_{o1} \cdot \vec{r}_{coc1} = \left(\frac{\partial \phi}{\partial n} \right)_{o1} \underbrace{\vec{n}_{o1} \cdot \vec{r}_{coc1}}_{\cos \theta} + \left(\frac{\partial \phi}{\partial t} \right)_{o1} \underbrace{\vec{t}_{o1} \cdot \vec{r}_{coc1}}_{\sin \theta}$$

The directional derivative in the $\vec{r}$ direction is discretized with 2-pt formula (14):

$$\frac{\partial \phi}{\partial n}|_{o1} = \nabla \phi|_{o1} \cdot \vec{r}_{coc1} = \frac{\phi_1 - \phi_0}{\|\vec{r}_{coc1}\|}$$

which is substituted in Eq. (21) to give

the DCM approximation of the gradient:

$$\frac{\partial \phi}{\partial n}|_{o1} = \underbrace{\frac{\phi_1 - \phi_0}{\|\vec{r}_{coc1}\| \cos \theta}}_{(1)} - \underbrace{\left(\frac{\partial \phi}{\partial t} \right)_{o1} \tan \theta}_{(2)} \quad (18)$$

• Term (1): is the orthogonal contribution of the gradient. It is equal to 2-pt formula (19) when the grid is orthogonal ($\theta = 0$).

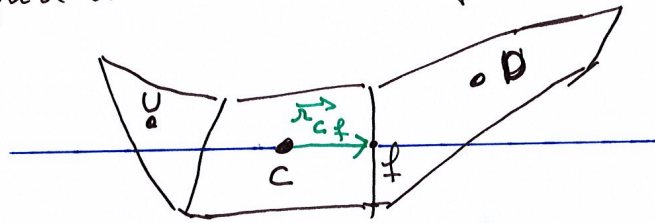
• Term (2): is the non-orthogonal correction. This term is zero when $\theta = 0$. In ANSYS FLUENT, this ~~term~~ term is calculated as:

$$\frac{\partial \phi}{\partial t}|_{o1} = \nabla \phi|_{o1} \cdot \vec{t}_{o1}$$

where $\nabla \phi|_{o1}$ is the face center gradient calculated with techniques discussed in Section 4.

5-2] Calculation of convective fluxes $F_f^c = (\vec{u} \cdot \vec{\Delta S})_f \phi_f$ (19)

As in the case of Cartesian grids, convective discretization involves centered (CDS) and upwind-type (UDS) methods. The implementation in unstructured grid will be discussed for the case of Fig. 5



Flow direction: $(\vec{u} \cdot \vec{\Delta S})_f > 0$

Fig. 5 Notation for convective discretization.

In the case the flow direction at face f of cell U_C is to the right, we have the following notation:

- C: is the central point
- D: Downstream point
- U: Upstream point.

As shown in the figure, these points are not aligned. Thus the implementation of high-order methods such as the k_2 -scheme (see exercise) will be more complicated than on regular grids.

The convective flux on face f is:

$$F_f^c \cdot \vec{\Delta S}_f = (\vec{u}_f \cdot \vec{\Delta S}_f) \phi_f, \quad (19)$$

and we will discuss the various interpolations of ϕ_f .

5-2-1] Upwind scheme (UDS) (20)

The first-order upwind interpolation is as easy as in Cartesian grid:

$$\phi_f = \begin{cases} \phi_c & \text{if } (\vec{u} \cdot \vec{\Delta S})_f > 0 \\ \phi_D & \text{otherwise.} \end{cases} \quad (20)$$

As mentioned in the previous lecture, the accuracy of UDS is very low, and more accurate methods are recommended.

5-2-2] Central scheme (CDS)

It is tempting to simply write the CDS as the arithmetic average:

$$\phi_f = \frac{1}{2} \phi_c + \frac{1}{2} \phi_D. \quad (21)$$

However, a more accurate implementation is due by considering the k_2 -scheme described in the next section.

5-2-3) Higher order interpolation for correction: the k -scheme ⁽²¹⁾

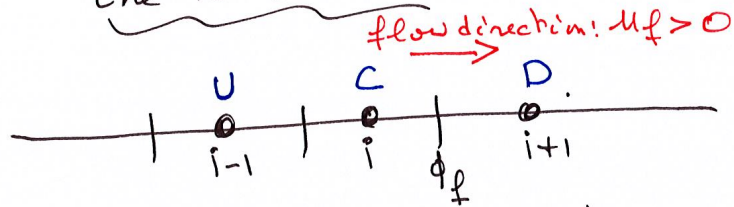


Fig. 6 Notation used for the k -scheme in a 1D grid

In exercise A.2, we have seen that the k -scheme use the following interpolation on a Cartesian grid:

$$\phi_f = \frac{\phi_{i+1} + \phi_i}{2} + \frac{(k-1)}{4} [\phi_{i+1} - 2\phi_i + \phi_{i-1}] \quad (22)$$

where the values of k gives the following methods:

- $k=1$: CDS scheme
- $k=-1$: upwind second-order
- $k=\frac{1}{2}$: Q.U.I.C.K scheme.

All these schemes are second-order accurate, and are available in ANSYS FLUENT.

For applying the k -scheme (22) to the unstructured mesh of Fig. 5, we need to rewrite as:

$$\phi_f = \underbrace{\phi_i}_{\phi_c} + \underbrace{\frac{(1-k)}{2} [\phi_{i+1} - \phi_{i-1}]}_{S_n^0 \phi_c \Delta x} + \underbrace{\frac{k}{2} [\phi_{i+1} - \phi_i]}_{S_n^0 \phi_f \Delta x} \quad (23)$$

As in Section 4.3, the k -scheme is extended to unstructured grids by remarking: (22)

$$S_n^0 \phi_c \Delta x \cong \vec{\nabla} \phi_c \cdot \vec{r}_{cf}, \quad (24a)$$

$$S_n^0 \phi_f \Delta x \cong \vec{\nabla} \phi_f \cdot \vec{r}_{cf}, \quad (24b)$$

which finally gives the generalization of the k -scheme:

$$\phi_f = \phi_c + (1-\frac{k}{2}) \vec{\nabla} \phi_c \cdot \vec{r}_{cf} + \frac{k}{2} \vec{\nabla} \phi_f \cdot \vec{r}_{cf}. \quad (24)$$

The gradients are calculated with the techniques described in Section 4.

14/03/2014

Analysis of the accuracy of FVM on arbitrary grids. Assessing the quality of the grid

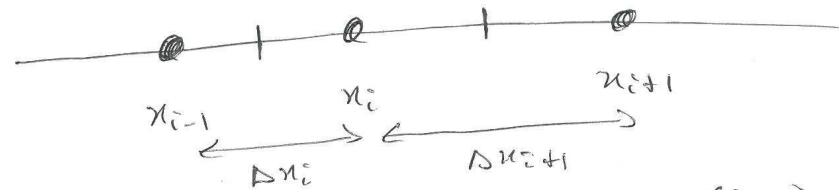
①

For meshing "real life" CFD problems, the grid is generally non-uniform (for clustering cells in high-gradient flow regions, like boundary layers), and meshing complex geometries give rise to mixed grids (tri and tet) which may be highly distorted, so that the accuracy and convergence of the computation may be severely diminished.

This lecture will review the accuracy properties of FVM on arbitrary mesh by checking the local truncation error (LTE) of the schemes. Then, we will derive some general guidelines for building high-quality grids for CFD in complex geometries.

②
1) Effects of grid non-uniformity: 1D example.

We consider a 1D mesh of cell $[x_{i-1/2}, x_{i+1/2}]$ of centroid x_i , such as the grid $\rightarrow$ step $\Delta x_i = x_i - x_{i-1}$ is non-uniform, i.e. $\Delta x_i \neq \Delta x_{i+1}$.



We will build a FD approximation of $(\frac{\partial u}{\partial x})_i$ by the usual method of Taylor expansions:

$$u_{i-1} = u_i - \Delta x_i \left(\frac{\partial u}{\partial x} \right)_i + \Delta x_i^2 \left(\frac{\partial^2 u}{\partial x^2} \right)_i - \frac{\Delta x_i^3}{3!} \left(\frac{\partial^3 u}{\partial x^3} \right)_i \quad (1)$$

$$u_{i+1} = u_i + \Delta x_{i+1} \left(\frac{\partial u}{\partial x} \right)_i + \Delta x_{i+1}^2 \left(\frac{\partial^2 u}{\partial x^2} \right)_i + \frac{\Delta x_{i+1}^3}{3!} \left(\frac{\partial^3 u}{\partial x^3} \right)_i \quad (2)$$

By combining Eqs (1) and (2), we get the FD quotient:

$$\frac{u_{i+1} - u_{i-1}}{\Delta x_i + \Delta x_{i+1}} = \left(\frac{\partial u}{\partial x} \right)_i + \tau_i \quad (3)$$

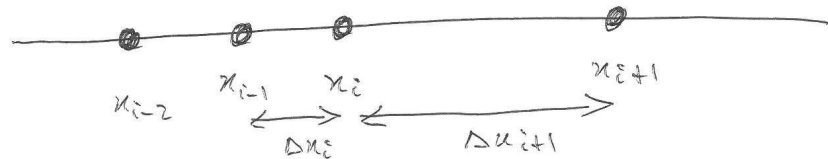
with the LTE:

$$\tau_i = (\Delta x_{i+1} - \Delta x_i) \left(\frac{\partial^2 u}{\partial x^2} \right)_i + \frac{1}{6} \frac{\Delta x_i^3 + \Delta x_{i+1}^3}{\Delta x_i + \Delta x_{i+1}} \left(\frac{\partial^3 u}{\partial x^3} \right)_i \quad (4)$$

On a uniform grid, $\Delta x_i = \Delta x \forall i$,
 we recover the usual CDS
 formula $\delta_x^0 u_i = \frac{u_{i+1} - u_{i-1}}{2 \Delta x}$ with LTE
 $\tau_i = O(\Delta x^2)$, but when $\Delta x_i \neq \Delta x_{i+1}$, the
 LTE is only first order!

Rule 1: In general, FD formulas lose 1 (or more!)
 order of accuracy on arbitrary grids.

This is especially true when there is an abrupt
 change in the grid, e.g. $\Delta x_{i+1} \gg \Delta x_i$



Local loss of accuracy near x_i is inevitable!

Rule 2: For "almost" uniform grids which are
 called smooth grids, such as:

$$\forall i, |\Delta x_{i+1} - \Delta x_i| = O(\Delta x_{\max}^2), \text{ where } \Delta x_{\max} = \max_i \Delta x_i$$

then Eq. (4) shows that $\tau_i = O(\Delta x_{\max}^2)$ and FD
 formula (3) recovers the second-order accuracy observed
 on uniform mesh.

Best practice 1: Avoid brutal change in
 the grid distribution or 2nd order accuracy is
 lost! (4)

Rule 3: Popular meshing strategies in grid
 generation software use a graded mesh such as:

$$\Delta x_{i+1} = r \Delta x_i, \text{ with clustering factor } r \geq 1.$$

For such grids, the LTE (4) can be written as:

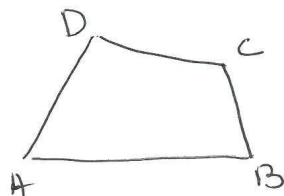
$$\tau_i = (r-1) \Delta x_i \left(\frac{\partial^2 u}{\partial x^2} \right)_i + O(\Delta x_i^2) \quad (5)$$

and if r is chosen close to 1 then LTE is
 near 2nd order and the local errors remain
 acceptable.

Best practice 2: Use smoothly varying grids
 (recommended values: $\frac{1}{4} \leq r \leq 1, 1-\frac{1}{4}$) in regions of
 strong flow variations to minimize numerical errors.

2) Error analysis for a 2D arbitrary quad (5)

We consider a quadrilateral cell $\mathcal{Q}_I$ (of volume V_I)



that is not highly distorted, ie such as faces lengths are:

$$x_D - x_A, x_C - x_D \approx \Delta x$$

$$V_I \approx \mathcal{O}(\Delta x \Delta y)$$

$$y_D - y_A, y_C - y_B \approx \Delta y$$

Note that a triangle is a degenerate case (eg. if $A=B$).

For a cell-vertex discretization of the steady compressible Euler eqs, ie:

$$\sum_{\text{faces}} \vec{F} \cdot \vec{\Delta S}_f = V_I q_I, \quad (6)$$

When we assume that $\Delta x \approx \Delta y = h$, Roe (1987) has shown that the LTE τ_I reads:

$$V_I \tau_I = \cancel{E_I} + \mathcal{O}(h^4) \quad (7)$$

with:

$$E_I = -E_2 \left(\frac{\partial^2 F_x^c}{\partial x^2} \right) + 2E_3 \left(\frac{\partial^2 F_x^c}{\partial x \partial y} \right) - E_4 \left(\frac{\partial^2 F_x^c}{\partial y^2} \right) - E_1 \left(\frac{\partial^2 F_y^c}{\partial x^2} \right) + 2E_2 \left(\frac{\partial^2 F_y^c}{\partial x \partial y} \right) - E_3 \left(\frac{\partial^2 F_y^c}{\partial y^2} \right) \quad (8)$$

$$E_1 = (x_A - x_C)(x_B - x_D)(x_A - x_B + x_C - x_D) \quad (9a)$$

$$= \mathcal{O}(\Delta x^3)$$

$$E_2 = 2V_I(x_A - x_B + x_C - x_D) + (x_A - x_C)(x_B - x_D) \times (y_A - y_B + y_C - y_D) \quad (9b)$$

$$= \mathcal{O}(\Delta x^2 \Delta y^2)$$

$$E_3 = 2V_I(y_A - y_B + y_C - y_D) + (y_A - y_C)(y_B - y_D)(x_A - x_B + x_D - x_C) \quad (9c)$$

$$= \mathcal{O}(\Delta x \Delta y^2)$$

$$E_4 = (y_A - y_C)(y_B - y_D)(y_A - y_B + y_C - y_D) \quad (9d)$$

$$= \mathcal{O}(\Delta y^3)$$

So that for a smooth quad such as $\Delta x \approx \Delta y \approx h$, the LTE

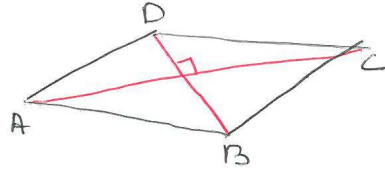
(7) reads:

$$\tau_I = \frac{\mathcal{O}(h^3)}{h^2} + \mathcal{O}(h^2). \quad (10)$$

Remark 1: For an arbitrary mesh $\tau_I = \mathcal{O}(h)$ the FVM is only 1st order accurate!

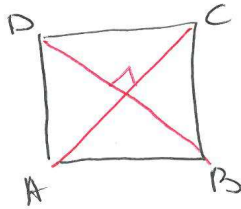
Remark 2: For particular cases of quadrilateral cells, because E_1 to E_4 in (9) cancel out such that $E_5 = 0$ and the method is second order accurate.

This is the case for parallelograms:



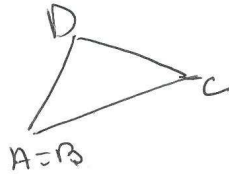
which are such that $\frac{\vec{x}_A + \vec{x}_C}{2} = \frac{\vec{x}_B + \vec{x}_D}{2}$.

The only interesting parallelogram cell is the rectangular cell:



⇒ On smooth Cartesian grids, the FVM is 2nd order accurate!

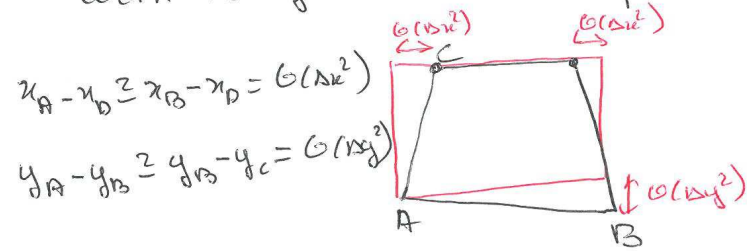
Remark 3: A triangle is a degenerate case of quadrilateral:



For triangles, cells, we always have $E_{\pm} \neq 0$
 ⇒ FVM on triangular grids is 1st-order accurate!

(4)

Remark 4: Let us consider a quad-cell with slight deviations from a ~~rect~~ rectangle:



$$x_A - x_B \approx x_B - x_D = O(hx^2)$$

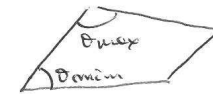
$$y_A - y_B \approx y_B - y_C = O(hy^2)$$

then $E_{\pm} = O(h^2)$ and the method would be second-order accurate.

This "slight deviation from rectangle" property is measured by the skewness factor Q_s .

For a quad, it is defined as:

$$Q_s = \max \left[\frac{\theta_{\max} - \pi/2}{\pi/2}, \frac{\pi/2 - \theta_{\min}}{\pi/2} \right] \in [0, 1].$$



$$Q_s \in [0, 1]$$



$$Q_s = 0.$$

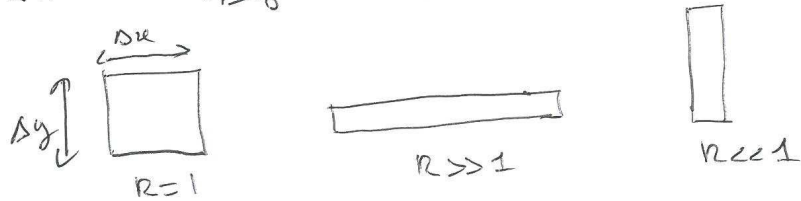
Quads are close to rectangles when $Q_s \approx 0$, hence numerical errors will be minimal for low skewness elements.

(8)

Remark 5) For arbitrary quad cells, the LTE (9)
 can be written as:

$$\begin{aligned} \gamma_{\pm} &= \underbrace{\frac{\mathcal{O}(\Delta x^3)}{\Delta x \Delta y}}_{E1} + \underbrace{\frac{\mathcal{O}(\Delta x \Delta y^2)}{\Delta x \Delta y}}_{E3} + \underbrace{\frac{\mathcal{O}(\Delta x^2 \Delta y)}{\Delta x \Delta y}}_{E2} + \underbrace{\frac{\mathcal{O}(\Delta y^3)}{\Delta x \Delta y}}_{E4} \\ &= \mathcal{O}(R \Delta x) + \mathcal{O}(\Delta y) + \mathcal{O}(\Delta x) + \mathcal{O}\left(\frac{\Delta y}{R}\right) \end{aligned}$$

with $R = \Delta x / \Delta y$ the cell aspect ratio:



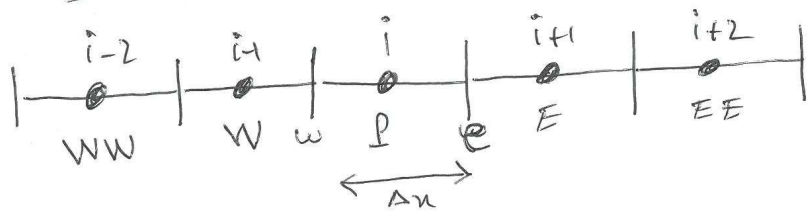
The LTE is minimal for cell with aspect ratio close to 1, but can become $\mathcal{O}(1)$ for cells with high aspect ratio!

3) Summary and best practice guidelines.

See slides.

Rhee - Chow momentum-weighted interpolation for collocated variable arrangement

①



1) Statement of the problem: pressure checkerboarding

In cell V_P the continuity eq. reads for face velocities:

$$\Delta y (u_e - u_w) + \Delta x (v_n - v_s) = 0 \quad (1)$$

In the collocated arrangement (pressure and velocities stored at cell center), the face velocities need to be interpolated. When we use central interpolation, as the one used for the discretization of momentum Eqs:

$$u_e = \frac{1}{2} (u_E + u_P), \quad (2)$$

the continuity eq. reads:

$$V_P \left[\underbrace{\frac{(u_E - u_w)}{2\Delta x} + \frac{(v_n - v_s)}{2\Delta y}}_{m_P} \right] = 0 \quad (3).$$

The PPE is constructed from the projection step (18.21a) of the EP1 scheme:

②

$$u^{n+1} = u^n - \Delta t \left(\frac{\partial P}{\partial x} \right) \Rightarrow u_E^{n+1} = u_E^n - \Delta t \left(\frac{\partial P}{\partial x} \right)_E \quad (4)$$

with $\left(\frac{\partial P}{\partial x} \right)_E = \frac{P_{EE} - P_P}{2\Delta x}$. When inserting Eq. (4)

into the continuity eq. (3), the PPE reads:

$$\left[\frac{P_{EE} - P_P}{2\Delta x} - \frac{P_P - P_{WW}}{2\Delta x} \right] + \left[\dots \right] = \Delta t \nabla \cdot u \quad (5)$$

The PPE amounts to a discrete Laplacian which uncouples odd nodal values (P_{WW}, P_P, P_{EE}) from even nodal values ($P_W, P_E, \dots$). In other words, this odd-even decoupling or checkerboarding effect leads to oscillations in the pressure field in the form $(-1)^i, (-1)^j$ and $(-1)^{i+j}$ in 2D, which renders the collocated method unsuitable for actual CFD computations.

2) Momentum-weighted (MWI) interpolation

Rhee & Chow (1983) observed that pressure checkerboarding is caused by the central ~~face~~ interpolation (2) that shuddles the ^{PPE} pressure stencil. They proposed a modification of (2) that uses pressure nodal values as weights in the fashion of the R-scheme for convection. This pressure

- weighted interpolation (PWI) is also known ③
as momentum-weighted interpolation (MWI)
because it is based on the discretized momentum
eqⁿ. that reads for cells U_P and U_E :

$$a_P u_P + \left(\frac{\partial P}{\partial x} \right)_P = - \left(\sum_{nb} a_{nb} u_{nb} \right)_P \quad (6a)$$

$$a_P u_E + \left(\frac{\partial P}{\partial x} \right)_E = - \left(\sum_{nb} a_{nb} u_{nb} \right)_E \quad (6b).$$

Note that the coefficients of the discretization ~~are~~
(a_P , a_{nb}) are identical in the case of uniform mesh.

their idea is to use momentum discretization (6)
to interpolate face velocity u_e as:

$$\begin{aligned} a_P u_e + \left(\frac{\partial P}{\partial x} \right)_e &= - \left(\sum_{nb} a_{nb} u_{nb} \right)_e \\ &= - \frac{1}{2} \left(\sum_{nb} a_{nb} u_{nb} \right)_E - \frac{1}{2} \left(\sum_{nb} a_{nb} u_{nb} \right)_P \\ &= \frac{1}{2} a_P u_P + \frac{1}{2} \left(\frac{\partial P}{\partial x} \right)_P + \frac{1}{2} a_P u_E + \frac{1}{2} \left(\frac{\partial P}{\partial x} \right)_E \end{aligned}$$

by using (6a) and (6b). Finally, the MWI for u_e reads:

$$u_e = \frac{1}{2} (u_E + u_P) + \frac{1}{a_P} \left[\frac{1}{2} \left(\frac{\partial P}{\partial x} \right)_P + \frac{1}{2} \left(\frac{\partial P}{\partial x} \right)_E - \left(\frac{\partial P}{\partial x} \right)_e \right] \quad (7)$$

with the pressure gradient at CV center evaluated as:

$$\left(\frac{\partial P}{\partial x} \right)_P = \frac{P_E - P_W}{2 \Delta x} \quad (8a)$$

and at face center as: ④

$$\left(\frac{\partial P}{\partial x} \right)_e = \frac{P_E - P_P}{\Delta x} \quad (8b)$$

The MWI interpolation (7) amounts to the central
interpolation (2) with additional pressure terms.

If we switch to index notations, Eq. (7) reads:

$$\begin{aligned} u_e &= \frac{1}{2} (u_i + u_{i+1}) + \frac{1}{a_P} \left[\frac{1}{2} \left(\frac{P_{i+1} - P_{i-1}}{2 \Delta x} \right) + \frac{1}{2} \left(\frac{P_{i+2} - P_i}{2 \Delta x} \right) - \left(\frac{P_{i+1} - P_i}{\Delta x} \right) \right] \\ &= \frac{1}{2} (u_i + u_{i+1}) + \frac{1}{4 a_P \Delta x} [P_{i+2} - 3 P_{i+1} + 3 P_i - P_{i-1}] \quad (9) \end{aligned}$$

In this interpolation, we recognize the non-centred
finite-difference quotient:

$$\left(\frac{d^3 P}{dx^3} \right)_i = \frac{P_{i+2} - 3 P_{i+1} + 3 P_i - P_{i-1}}{\Delta x^3} + O(\Delta x), \quad (10)$$

so that the PWI interpolation is equivalent to:

$$u_e = \frac{1}{2} (u_i + u_{i+1}) + \frac{\Delta x^2}{4 a_P} \left(\frac{\partial^3 P}{\partial x^3} \right)_i \quad (11)$$

the PWI
and amounts to add a small regularization term in
 $O(\Delta x^2)$ to the central interpolation (2), with the aim
of suppressing the checkerboard oscillations.
When the MWI (7) is used for discretizing
the continuity eqⁿ (1), we obtain:

$$\begin{aligned} \Delta y (u_e - u_w) &= \Delta y \left[\frac{1}{2} (u_i + u_{i+1}) + \frac{1}{4 a_P \Delta x} (P_{i+2} - 3 P_{i+1} + 3 P_i - P_{i-1}) \right. \\ &\quad \left. - \frac{1}{2} (u_{i-1} + u_i) + \frac{1}{4 a_P \Delta x} (P_{i+1} - 3 P_i + 3 P_{i-1} - P_{i-2}) \right] \end{aligned}$$

$$\Delta y (u_e - u_w) = V_p \left[\frac{u_E - u_W}{2 \Delta n} + \frac{1}{4 q_p \Delta x^2} (P_{i+2} - 4P_{i+1} + 6P_i - 4P_{i-1} + P_{i-2}) \right] \quad (5)$$

(12)

where the centred finite difference quotient appears:

$$\left(\frac{d^4 p}{dx^4} \right)_i = \frac{P_{i+2} - 4P_{i+1} + 6P_i - 4P_{i-1} + P_{i-2} + O(\Delta x^2)}{\Delta x^4}$$

with MWI interpolation.

The continuity eqⁿ is thus equivalent to:

$$\dot{m}_p + \frac{1}{4 q_p} \left[\underbrace{\Delta x^2 \left(\frac{\partial^4 p}{\partial x^4} \right)_i}_{\text{dissipative terms}} + \Delta y^2 \left(\frac{\partial^4 p}{\partial y^4} \right)_i \right] = 0 \quad (13)$$

which has to be compared with the continuity Eq. (3) obtained with central interpolation. The PWI has the effect to add a dissipative term to the continuity eqⁿ in order to damp the checkerboarding oscillations. In effect, checkerboarding is absent from the PPE obtained with PWI since spurious pressure solutions of the form $(-1)^i$ do not cancel in Eq. (12).

Concluding remarks: The PWI (or MWI) is an effective way to get rid of the checkerboarding problem of the collocated method, and is widely used in commercial codes such as FLUENT. The regularizing terms in Eq. (15) and (12) do not alter the truncation error of the collocated method (in $O(\Delta x^2)$) but they may have an effect on its stability. Ventures of the PWI (7) have been devised for the case of

non-uniform grids, non-rectangular (unstructured) cells, unsteady NS eqs, presence of body terms, etc.... (6)

We nonetheless observe that PWI affects the conservation properties of the method since Eq. (13) shows that total mass is not conserved during a computation.