

# Numerical Modelling

Jonathan GULA  
gula@univ-brest.fr

*the anatomy of an ocean model*

- **Lesson 1 : [D109]**
    - Introduction
    - Equations of motions
    - *Activity 1 [run an ocean model]*
  - **Lesson 2 : [B 012]**
    - Horizontal Discretization
    - *Activity 2 [Dynamics of an ocean gyre]*
  - **Lesson 3 : [D109]**
    - Presentation of the model CROCO
    - Dynamics of the ocean gyre
    - *Activity 2 [Dynamics of an ocean gyre]*
  - **Lesson 4 : [D109]**
    - Numerical schemes
    - *Activity 3 [Impacts of numerics]*
  - **Lesson 5 : [D109]**
    - Vertical coordinates
  - *Activity 4 [Impact of topography]*
  - **Lesson 6 : [D109]**
    - Boundary Forcings
    - Model parameterizations
    - *Activity 4 [Design a realistic simulation]*
  - **Lesson 7 : [D109]**
    - Diagnostics and validation
    - *Activity 5 [Analyze a realistic simulation]*
  - **Lesson 8 : [D109]**
    - *Work on your projet*
- Presentations and material will be available at :
- jgula.fr/ModNum/**

## #4 Solving the equations

---

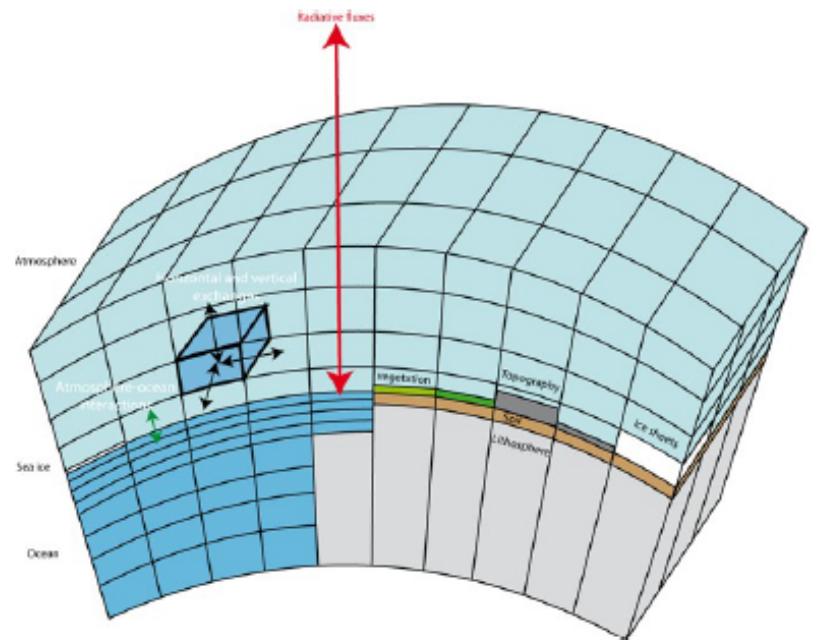
# Solving the equations ?

$$\frac{\partial u}{\partial t} + \vec{u} \cdot \vec{\nabla}_H u + w \frac{\partial u}{\partial z} - fv = -\frac{\partial_x P}{\rho_0} + \mathcal{F}_u + \mathcal{D}_u$$

$$\frac{\partial v}{\partial t} + \vec{u} \cdot \vec{\nabla}_H v + w \frac{\partial v}{\partial z} + fu = -\frac{\partial_y P}{\rho_0} + \mathcal{F}_v + \mathcal{D}_v$$

Equations

Discretization



# Solving the equations ?

$$\begin{aligned}\frac{\partial u}{\partial t} + \vec{u} \cdot \vec{\nabla}_H u + w \frac{\partial u}{\partial z} - fv &= -\frac{\partial_x P}{\rho_0} + \mathcal{F}_u + \mathcal{D}_u \\ \frac{\partial v}{\partial t} + \vec{u} \cdot \vec{\nabla}_H v + w \frac{\partial v}{\partial z} + fu &= -\frac{\partial_y P}{\rho_0} + \mathcal{F}_v + \mathcal{D}_v\end{aligned}$$

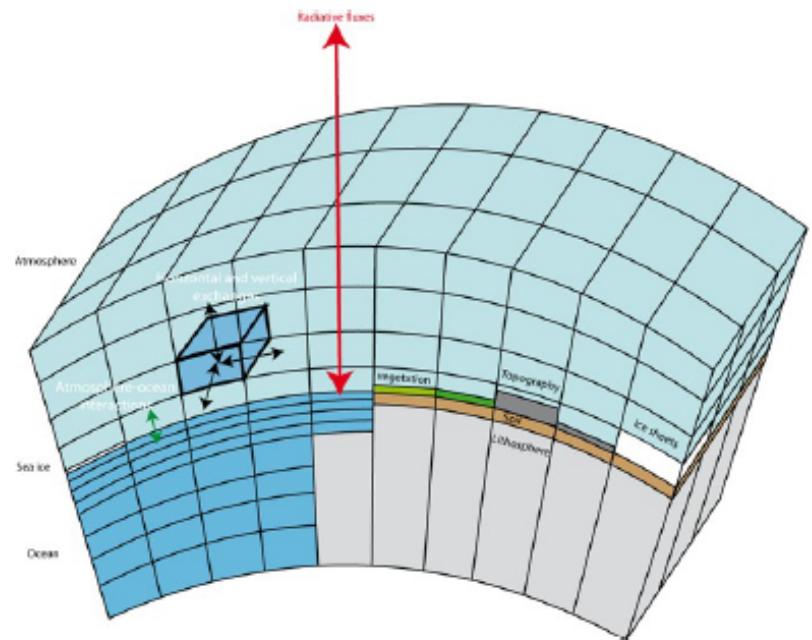
Equations

Time  
Stepping  
scheme

Horizontal  
Advection  
scheme

Vertical  
Advection  
scheme

Discretization



# Finite difference schemes

- A finite difference scheme is produced when the partial derivatives in the partial differential equations governing a physical phenomenon are replaced by a finite difference approximation.
- The result is a system of algebraic equations which, when solved, provide an approximation to the solution of the original partial differential equations at selected points of a solution grid.
-

# Taylor series expansion & truncation errors

- Definition of the derivative of a smooth function:

$$f'(x) = \lim_{\epsilon \rightarrow 0} \frac{f(x + \epsilon) - f(x)}{\epsilon}$$

- Taylor series expansion of  $f$  at point  $x$ :

$$\begin{aligned} f(x \pm \epsilon) &= f(x) + \sum_{j=1}^{\infty} \epsilon^j \frac{(\pm 1)^j}{j!} \frac{\partial^j}{\partial x^j} f \\ &= f(x) \pm \epsilon f'(x) + \frac{\epsilon^2}{2!} f''(x) \pm \frac{\epsilon^3}{3!} f'''(x) + \frac{\epsilon^4}{4!} f''''(x) + \dots \end{aligned}$$

# Taylor series expansion & truncation errors

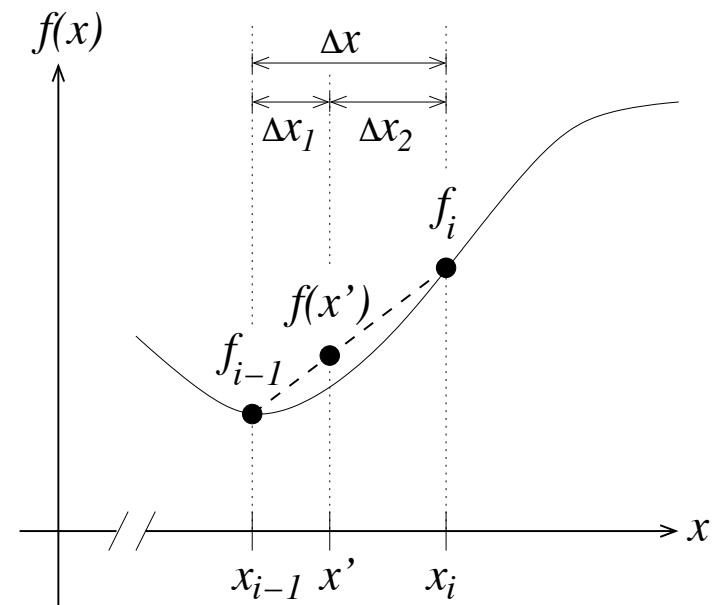
- Constructing a difference operator using Taylor series:

$$\begin{aligned}f_{i-1} &= f(x_i) - \Delta x f'(x_i) + \frac{\Delta x^2}{2!} f''(x_i) - \dots \\f_i &= f(x_i)\end{aligned}$$

where  $\Delta x = x_i - x_{i-1}$  is the grid spacing. Solving for  $f'(x_i)$  gives:

$$f'(x_i) \approx \frac{1}{\Delta x} (f_i - f_{i-1}) + \frac{\Delta x}{2} f''(x_i) + O(\Delta x)$$

Truncation  
error



# Spatial advection scheme:



First order

$$\frac{\partial u}{\partial x} = \frac{u_{i+1} - u_i}{\Delta x} + \mathcal{O}(\Delta x)$$

Downstream

$$\frac{\partial u}{\partial x} = \frac{u_i - u_{i-1}}{\Delta x} + \mathcal{O}(\Delta x)$$

Upstream

2<sup>nd</sup> order

$$\frac{\partial u}{\partial x} = \frac{u_{i+1} - u_{i-1}}{2\Delta x} + \mathcal{O}(\Delta x^2)$$

Centered

4<sup>th</sup> order

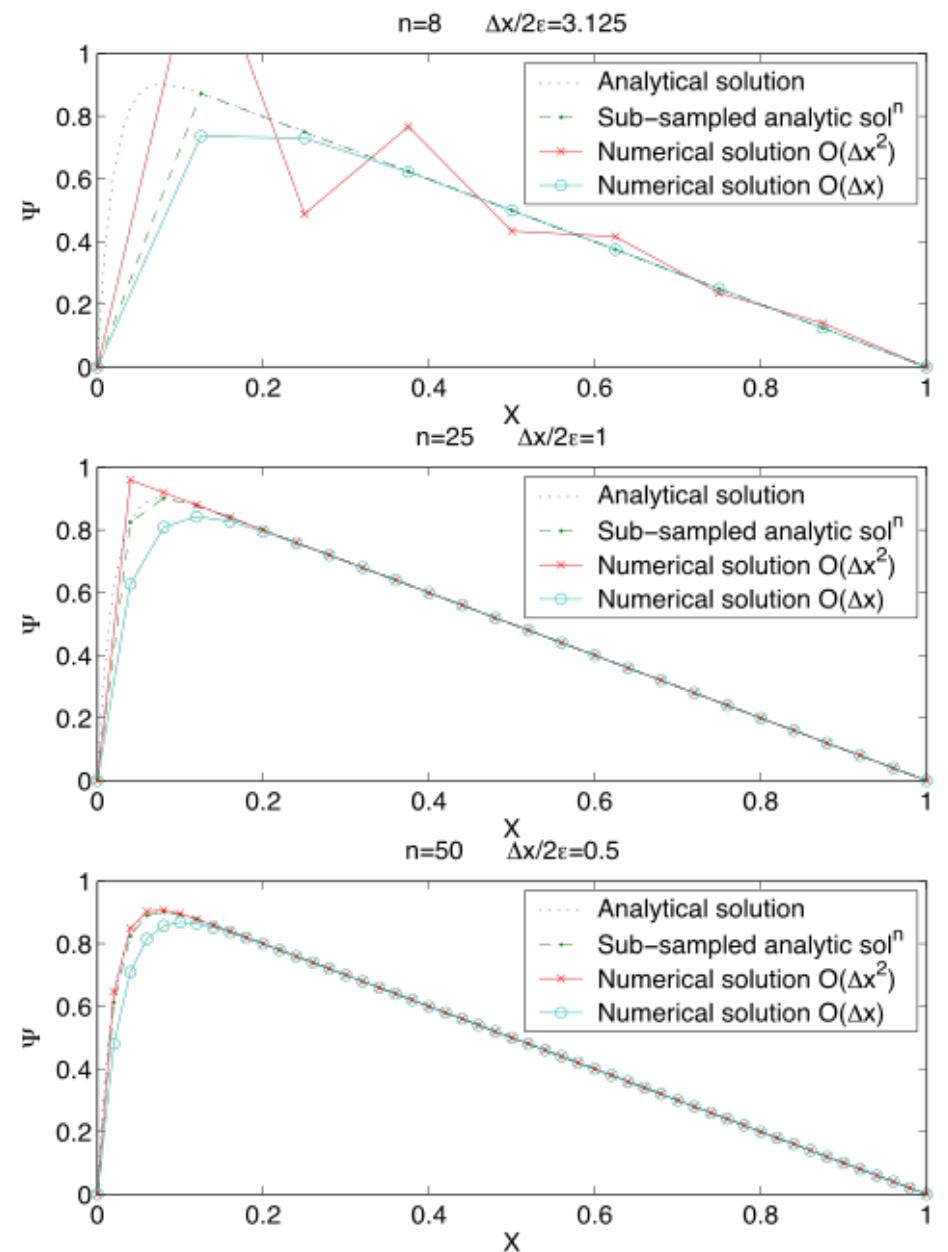
$$\frac{\partial u}{\partial x} = \frac{4}{3} \left( \frac{u_{i+1} - u_{i-1}}{2\Delta x} \right) - \frac{1}{3} \left( \frac{u_{i+2} - u_{i-2}}{2\Delta x} \right) + \mathcal{O}(\Delta x^4)$$

# Spatial advection scheme :

- Ex: Stommel equation in 1D

$$\epsilon \partial_{xx} \psi + \partial_x \psi = -1$$

$$\psi = C \left( e^{-\frac{x}{\epsilon}} - 1 \right) - x \quad \text{with} \quad C^{-1} = e^{-\frac{1}{\epsilon}} - 1.$$



# Solving the equations ?

$$\begin{aligned}\frac{\partial u}{\partial t} + \vec{u} \cdot \vec{\nabla}_H u + w \frac{\partial u}{\partial z} - fv &= -\frac{\partial_x P}{\rho_0} + \mathcal{F}_u + \mathcal{D}_u \\ \frac{\partial v}{\partial t} + \vec{u} \cdot \vec{\nabla}_H v + w \frac{\partial v}{\partial z} + fu &= -\frac{\partial_y P}{\rho_0} + \mathcal{F}_v + \mathcal{D}_v\end{aligned}$$

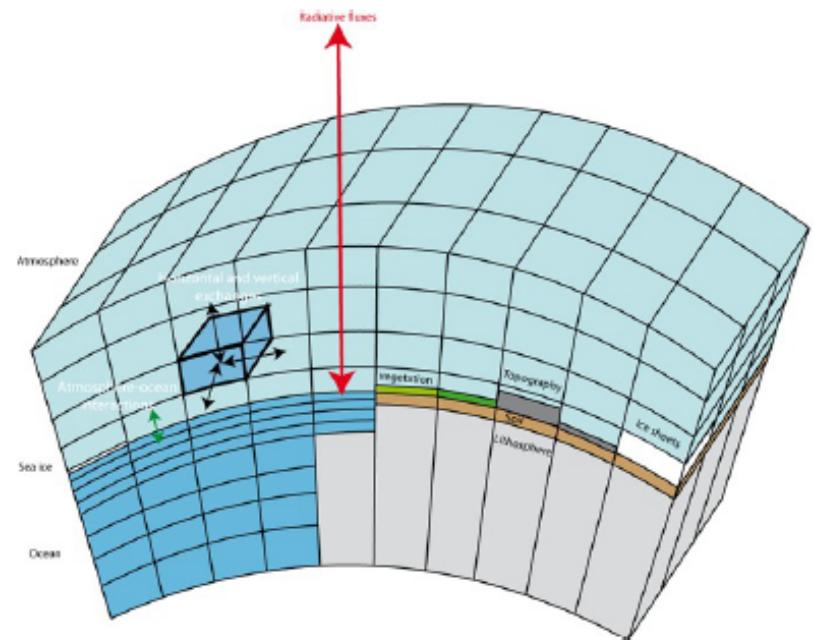
Equations

Time  
Stepping  
scheme

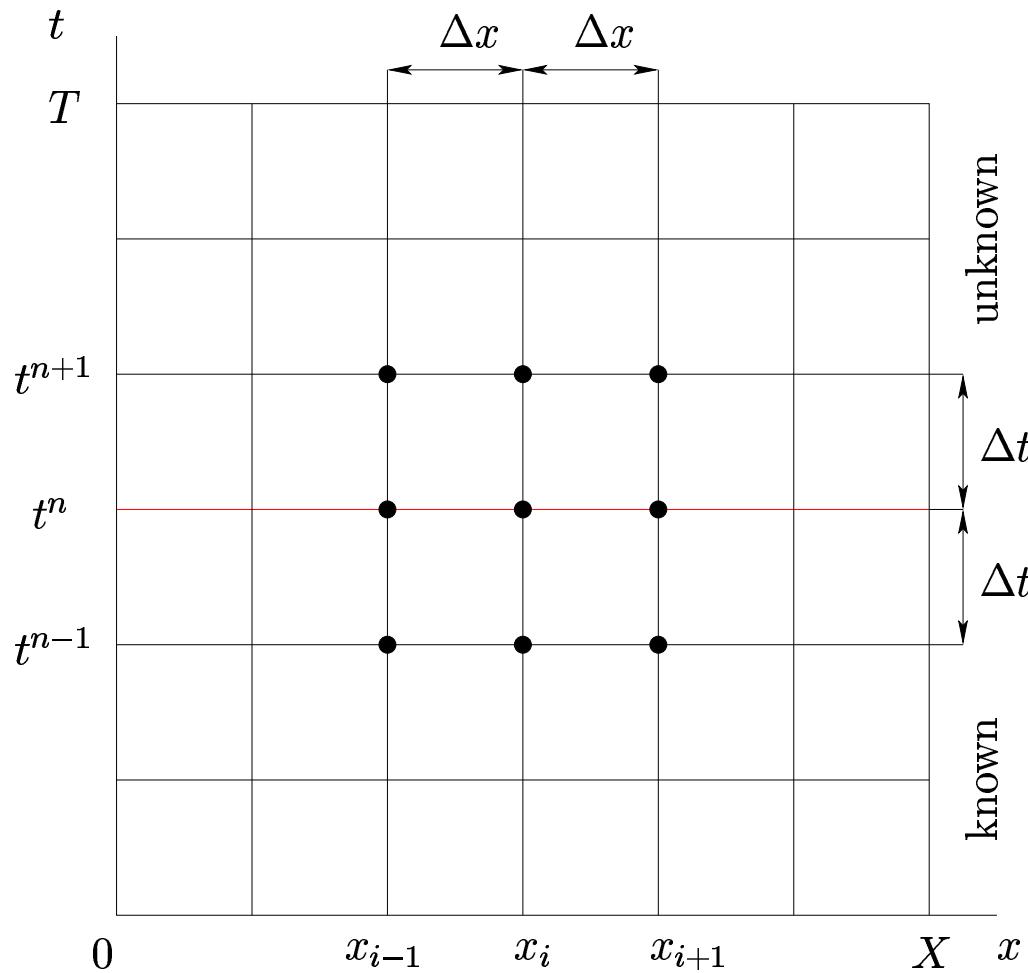
Horizontal  
Advection  
scheme

Vertical  
Advection  
scheme

Discretization



# Time-stepping schemes



# 2-levels Time-stepping scheme

First-order ODE system

$$\begin{cases} \frac{du}{dt} + R(u, t) = 0 & \text{for } t \in (t^n, t^{n+1}) \\ u(t^n) = u^n, & \forall n = 0, 1, \dots, M-1 \end{cases}$$

Standard  $\theta$ -scheme (finite difference discretization of the time derivative)

$$\boxed{\frac{u^{n+1} - u^n}{\Delta t} + [\theta R(u^{n+1}, t^{n+1}) + (1 - \theta)R(u^n, t^n)] = 0} \quad 0 \leq \theta \leq 1$$

where  $\Delta t = t^{n+1} - t^n$  is the time step and  $\theta$  is the implicitness parameter

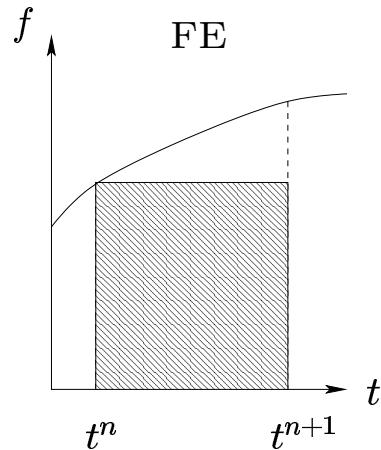
$\theta = 0$       *forward Euler scheme*      explicit,       $\mathcal{O}(\Delta t)$

$\theta = 1/2$       *Crank-Nicolson scheme*      implicit,       $\mathcal{O}(\Delta t)^2$

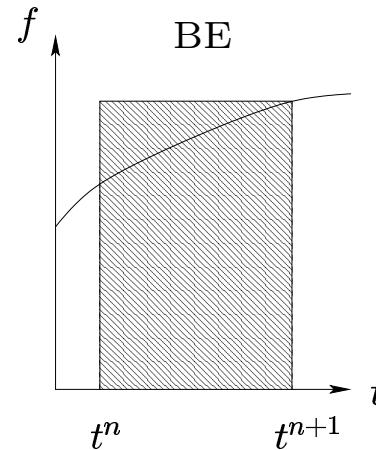
$\theta = 1$       *backward Euler scheme*      implicit,       $\mathcal{O}(\Delta t)$

# Example: standard time-stepping schemes

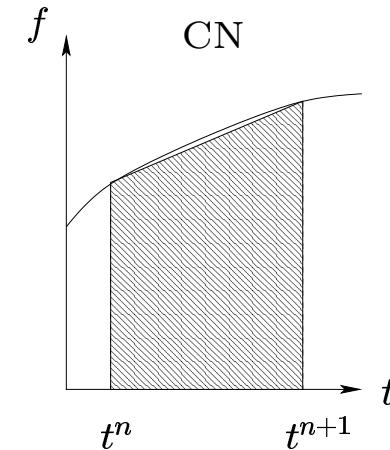
Numerical integration on the interval  $(t^n, t^{n+1})$



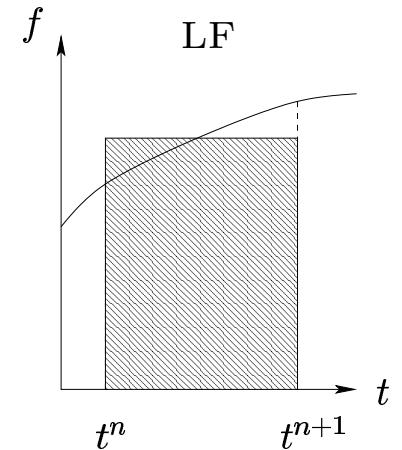
*left endpoint*



*right endpoint*



*trapezoidal rule*



*midpoint rule*

Forward Euler

$$u^{n+1} = u^n + f(t^n, u^n)\Delta t + \mathcal{O}(\Delta t)^2$$

Backward Euler

$$u^{n+1} = u^n + f(t^{n+1}, u^{n+1})\Delta t + \mathcal{O}(\Delta t)^2$$

Crank-Nicolson

$$u^{n+1} = u^n + \frac{1}{2}[f(t^n, u^n) + f(t^{n+1}, u^{n+1})]\Delta t + \mathcal{O}(\Delta t)^3$$

Leapfrog method

$$u^{n+1} = u^n + f(t^{n+1/2}, u^{n+1/2})\Delta t + \mathcal{O}(\Delta t)^3$$

# Explicit vs. implicit time discretization

Pros and cons of explicit schemes

- ⊕ easy to implement and parallelize, low cost per time step
- ⊕ a good starting point for the development of CFD software
- ⊖ small time steps are required for stability reasons, especially if the velocity and/or mesh size are varying strongly
- ⊖ extremely inefficient for solution of stationary problems unless *local time-stepping* i. e.  $\Delta t = \Delta t(\mathbf{x})$  is employed

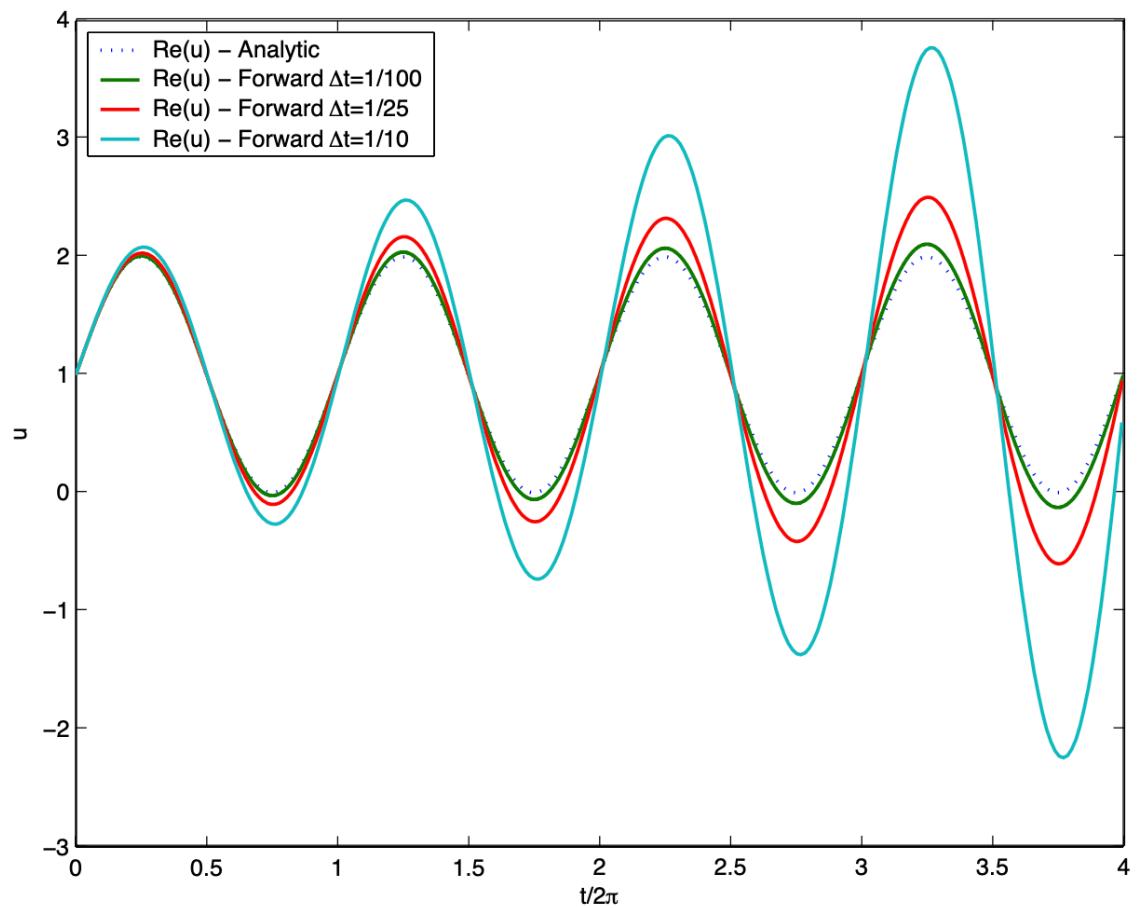
Pros and cons of implicit schemes

- ⊕ stable over a wide range of time steps, sometimes unconditionally
- ⊕ constitute excellent iterative solvers for steady-state problems
- ⊖ difficult to implement and parallelize, high cost per time step
- ⊖ insufficiently accurate for truly transient problems at large  $\Delta t$
- ⊖ convergence of linear solvers deteriorates/fails as  $\Delta t$  increases

# Explicit vs. implicit time discretization

$$\partial_t u + i f u = -\epsilon u + \tau$$

The forward method applied to a simple oscillation equation ( $f = 1$  and with no damping,  $\epsilon = 0$ ). The solutions all grow with time but as a function of the time-step.



# Predictor-corrector and multipoint methods

Objective: to combine the simplicity of explicit schemes and robustness of implicit ones in the framework of a fractional-step algorithm, e.g.,

1. Predictor  $\tilde{u}^{n+1} = u^n + f(t^n, u^n)\Delta t$  forward Euler
2. Corrector  $u^{n+1} = u^n + \frac{1}{2}[f(t^n, u^n) + f(t^{n+1}, \tilde{u}^{n+1})]\Delta t$  Crank-Nicolson  
or  $u^{n+1} = u^n + f(t^{n+1}, \tilde{u}^{n+1})\Delta t$  backward Euler

*Remark.* Stability still leaves a lot to be desired, additional correction steps usually do not pay off since iterations may diverge if  $\Delta t$  is too large

Order barrier: two-level methods are at most second-order accurate, so extra points are needed to construct higher-order integration schemes

Adams methods  $t^{n+1}, \dots, t^{n-m}, \quad m = 0, 1, \dots$

Runge-Kutta methods  $t^{n+\alpha} \in [t^n, t^{n+1}], \quad \alpha \in [0, 1]$

# Adams methods

Derivation: *polynomial fitting*

Truncation error:  $\epsilon_{\tau}^{\text{glob}} = \mathcal{O}(\Delta t)^p$

for polynomials of degree  $p - 1$  which  
interpolate function values at  $p$  points

Adams-Basforth methods (explicit)

$$p = 1 \quad u^{n+1} = u^n + \Delta t f(t^n, u^n) \quad \text{forward Euler}$$

$$p = 2 \quad u^{n+1} = u^n + \frac{\Delta t}{2} [3f(t^n, u^n) - f(t^{n-1}, u^{n-1})]$$

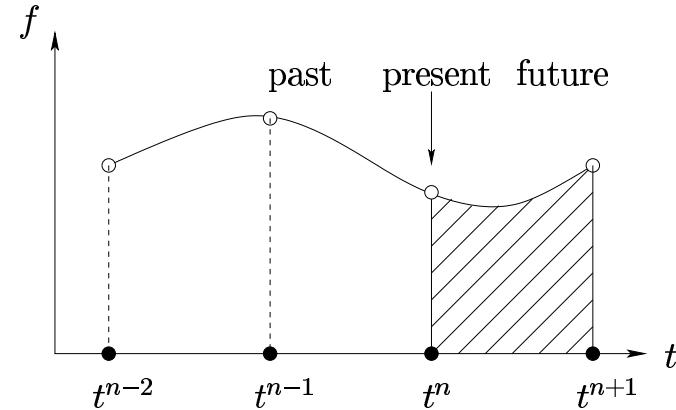
$$p = 3 \quad u^{n+1} = u^n + \frac{\Delta t}{12} [23f(t^n, u^n) - 16f(t^{n-1}, u^{n-1}) + 5f(t^{n-2}, u^{n-2})]$$

Adams-Moulton methods (implicit)

$$p = 1 \quad u^{n+1} = u^n + \Delta t f(t^{n+1}, u^{n+1}) \quad \text{backward Euler}$$

$$p = 2 \quad u^{n+1} = u^n + \frac{\Delta t}{2} [f(t^{n+1}, u^{n+1}) + f(t^n, u^n)] \quad \text{Crank-Nicolson}$$

$$p = 3 \quad u^{n+1} = u^n + \frac{\Delta t}{12} [5f(t^{n+1}, u^{n+1}) + 8f(t^n, u^n) - f(t^{n-1}, u^{n-1})]$$



# Adams methods

Predictor-corrector algorithm

1. Compute  $\tilde{u}^{n+1}$  using an Adams-Bashforth method of order  $p - 1$
2. Compute  $u^{n+1}$  using an Adams-Moulton method of order  $p$  with predicted value  $f(t^{n+1}, \tilde{u}^{n+1})$  instead of  $f(t^{n+1}, u^{n+1})$

Pros and cons of Adams methods

- ⊕ methods of any order are easy to derive and implement
- ⊕ only one function evaluation per time step is performed
- ⊕ error estimators for ODEs can be used to adapt the order
- ⊖ other methods are needed to start/restart the calculation
- ⊖ time step is difficult to change (coefficients are different)
- ⊖ tend to be unstable and produce nonphysical oscillations

# Runge-Kutta methods

Multipredictor-multicorrector algorithms of order  $p$

$$p = 2 \quad \tilde{u}^{n+1/2} = u^n + \frac{\Delta t}{2} f(t^n, u^n) \quad \text{forward Euler / predictor}$$

$$u^{n+1} = u^n + \Delta t f(t^{n+1/2}, \tilde{u}^{n+1/2}) \quad \text{midpoint rule / corrector}$$

$$p = 4 \quad \tilde{u}^{n+1/2} = u^n + \frac{\Delta t}{2} f(t^n, u^n) \quad \text{forward Euler / predictor}$$

$$\hat{u}^{n+1/2} = u^n + \frac{\Delta t}{2} f(t^{n+1/2}, \tilde{u}^{n+1/2}) \quad \text{backward Euler / corrector}$$

$$\tilde{u}^{n+1} = u^n + \Delta t f(t^{n+1/2}, \hat{u}^{n+1/2}) \quad \text{midpoint rule / predictor}$$

$$u^{n+1} = u^n + \frac{\Delta t}{6} [f(t^n, u^n) + 2f(t^{n+1/2}, \tilde{u}^{n+1/2}) \quad \text{Simpson rule}$$

$$+ 2f(t^{n+1/2}, \hat{u}^{n+1/2}) + f(t^{n+1}, \tilde{u}^{n+1})] \quad \text{corrector}$$

*Remark.* There exist ‘embedded’ Runge-Kutta methods which perform extra steps in order to estimate the error and adjust  $\Delta t$  in an adaptive fashion

# Runge-Kutta methods

Pros and cons of Runge-Kutta methods

- ⊕ self-starting, easy to operate with variable time steps
- ⊕ more stable and accurate than Adams methods of the same order
- ⊖ high order approximations are rather difficult to derive;  $p$  function evaluations per time step are required for a  $p$ -th order method
- ⊖ more expensive than Adams methods of comparable order

Adaptive time-stepping strategy       $\Delta t$      $\Delta t$

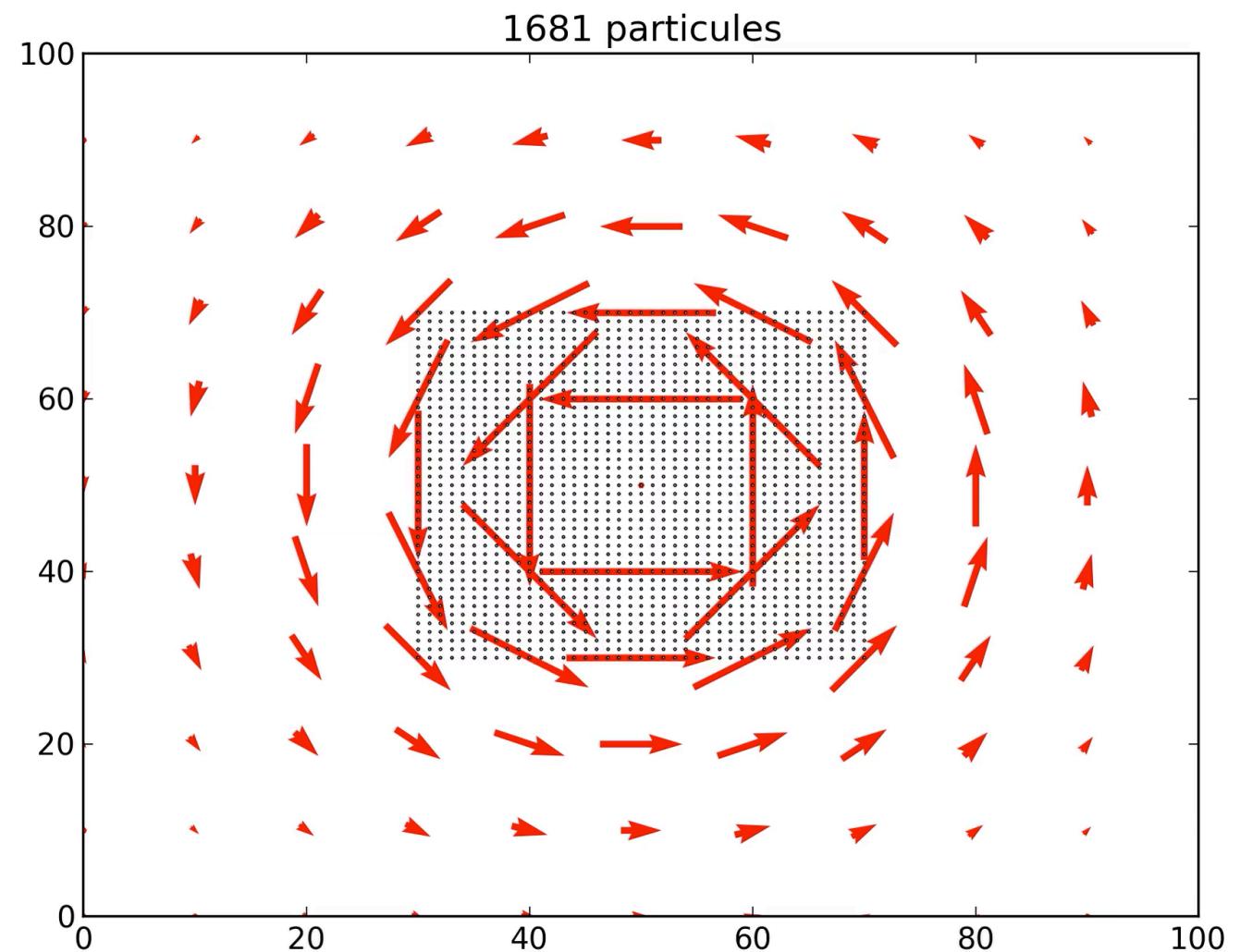
makes it possible to achieve the desired accuracy at a relatively low cost

Explicit methods: *use the largest time step satisfying the stability condition*

Implicit methods: *estimate the error and adjust the time step if necessary*

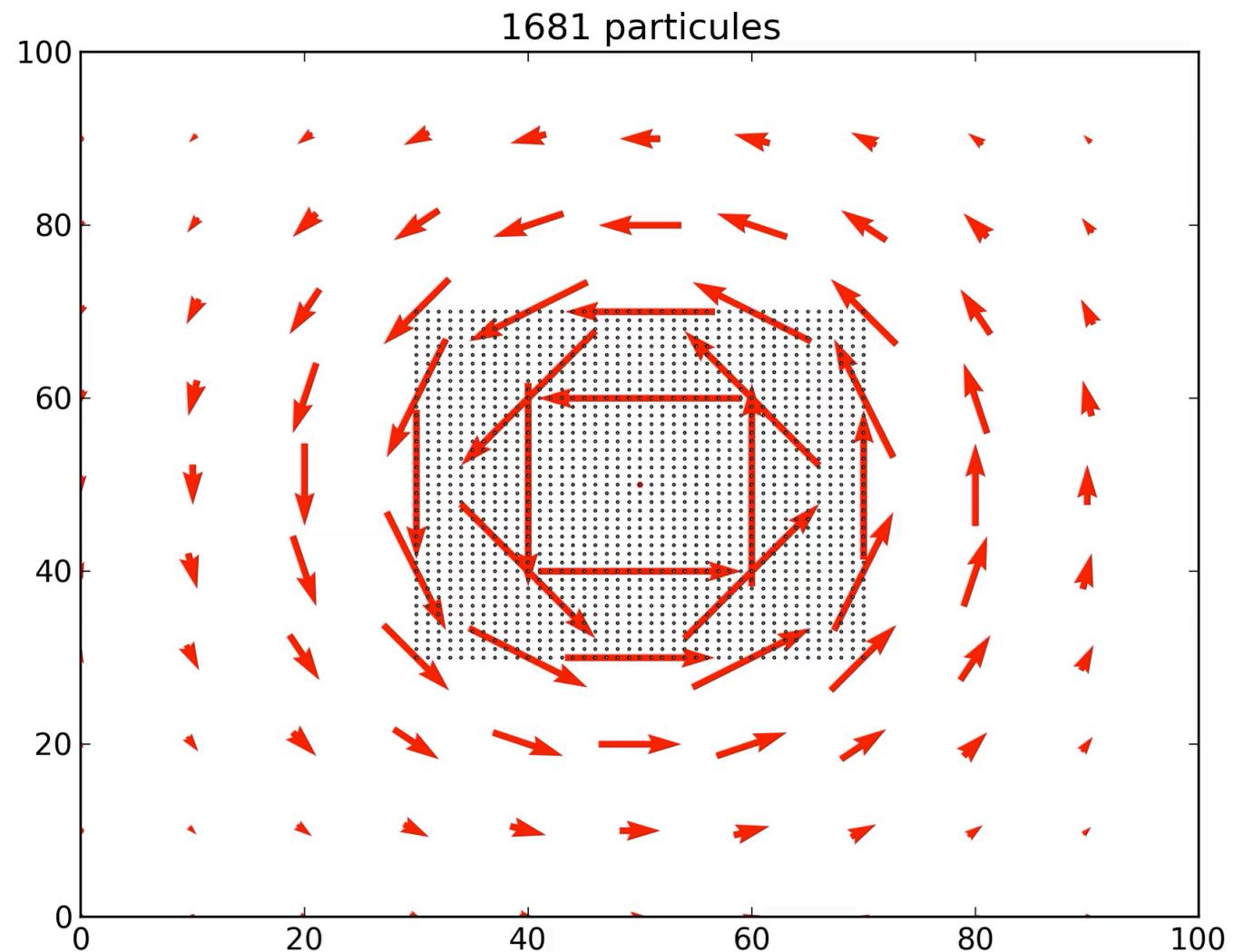
# Time-stepping scheme

- Example using a FE time-stepping scheme for advection of Lagrangian particles in a time-invariant velocity field:

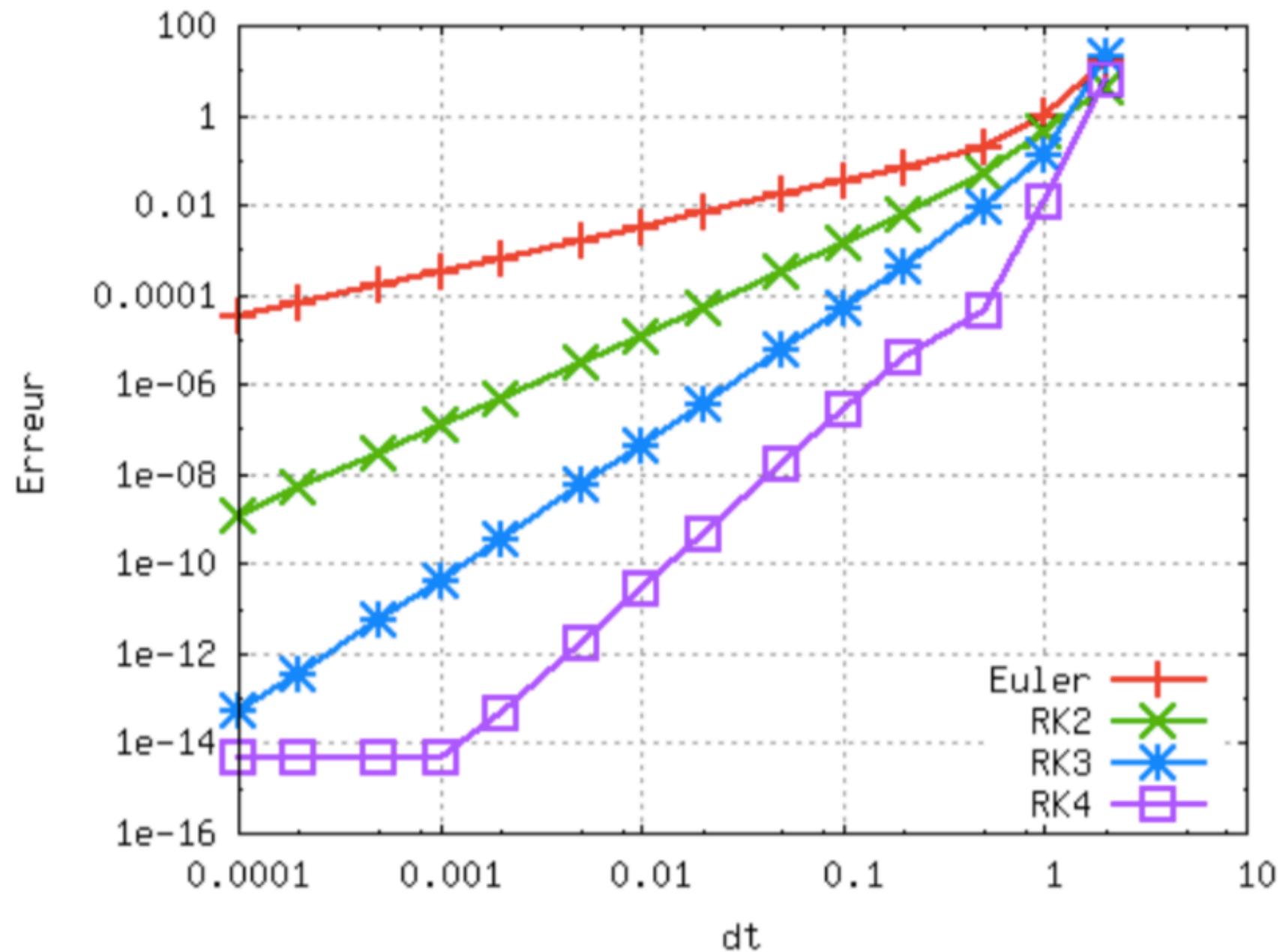


# Time-stepping scheme

- Example using a RK4 time-stepping scheme for advection of Lagrangian particles in a time-invariant velocity field:



# Time-stepping scheme



# Finite difference schemes

- A finite difference scheme is produced when the partial derivatives in the partial differential equations governing a physical phenomenon are replaced by a finite difference approximation.
- The result is a system of algebraic equations which, when solved, provide an approximation to the solution of the original partial differential equations at selected points of a solution grid.
- The Most important characteristic of a scheme are:  
**Consistence, Convergence and Stability**

# Finite difference schemes

- **Consistence** = A finite difference scheme is consistent if the operator reduces to the original differential equation as the increments in the independent variables vanish ( $\delta x_i, \delta t \rightarrow 0$ )
- **Convergence** = The finite-difference solution approaches the true solution to the partial differential equation as the increments go to zero ( $\delta x_i, \delta t \rightarrow 0$ )
- **Stability** = The error caused by a small perturbation in the numerical solution remains bound.
- *for a well-posed initial-value problem, the Lax-Richtmyer theorem states that, for a consistent numerical method, stability and convergence are equivalent.*

# Stability:

- The basic idea for the convergence and stability analysis for a linear partial differential equation consists in writing the solution to the equation as a complex Fourier series and analyzing a generic component of the solution.
- Example: ***The advection equation***

$$\frac{\partial T}{\partial t} + u \frac{\partial T}{\partial x} = 0$$

# Stability:

Example: ***The advection equation ( $u > 0$ )***  $\frac{\partial T}{\partial t} + u \frac{\partial T}{\partial x} = 0$

We use a **forward Euler in time and a downstream scheme in space**:

$$\frac{\partial T}{\partial t} = \frac{T(t + \delta t) - T(t)}{\delta t} = \frac{T_{i,n+1} - T_{i,n}}{\delta t}$$

$$\frac{\partial T}{\partial x} = \frac{T(x + \delta x) - T(x)}{\delta x} = \frac{T_{i+1,n} - T_{i,n}}{\delta x}$$

Such that:

$$T_{i,n+1} = T_{i,n} - u \frac{\delta t}{\delta x} (T_{i+1,n} - T_{i,n})$$

# Stability:

Example: ***The advection equation***

$$\frac{\partial T}{\partial t} + u \frac{\partial T}{\partial x} = 0$$

$$T_{i,n+1} = T_{i,n} - u \frac{\delta t}{\delta x} (T_{i+1,n} - T_{i,n})$$

We introduce:  $T_n = \hat{T}_n e^{ikx}$

We get  $\hat{T}_{n+1} e^{ikx} = \hat{T}_n e^{ikx} - u \frac{\delta t}{\delta x} (\hat{T}_n e^{ik(x+\delta x)} - \hat{T}_n e^{ikx})$

# Stability:

Example: ***The advection equation***

$$\frac{\partial T}{\partial t} + u \frac{\partial T}{\partial x} = 0$$

$$T_{i,n+1} = T_{i,n} - u \frac{\delta t}{\delta x} (T_{i+1,n} - T_{i,n})$$

We introduce:  $T_n = \hat{T}_n e^{ikx}$

We get  $\hat{T}_{n+1} e^{ikx} = \hat{T}_n e^{ikx} - u \frac{\delta t}{\delta x} (\hat{T}_n e^{ik(x+\delta x)} - \hat{T}_n e^{ikx})$

The amplification is  $A = \frac{\hat{T}_{n+1}}{\hat{T}_n} = 1 - u \frac{\delta t}{\delta x} (e^{ik\delta x} - 1)$

# Stability:

The amplification is  $A = \frac{\hat{T}_{n+1}}{\hat{T}_n} = 1 - u \frac{\delta t}{\delta x} (e^{ik\delta x} - 1)$

In general a numerical scheme is

unstable if	$ A  > 1$
stable if	$ A  < 1$

For the Euler/downstream:

$$|A|^2 = |1 - C(e^{ik\delta x} - 1)|^2$$

With

$$C = u \frac{\delta t}{\delta x}$$

so  $|A|^2 = (1 - C \cos(k\delta x) + C)^2 + C^2 \sin(k\delta x)^2$

# Stability:

$$|A|^2 = (1 - C\cos(k\delta x) + C)^2 + C^2\sin(k\delta x)^2$$

$$|A|^2 = (1 + C)^2 - 2(1 + C)C\cos(k\delta x) + C^2$$

$$\begin{aligned} |A|^2 &= 1 + 2(1 + C)C(1 - \cos(k\delta x)) \\ &> 0 \end{aligned}$$

For positive u:

$$|A| > 1$$

**The Euler/downstream is unconditionally unstable**

# Stability:

- Example: ***The advection equation***

$$\frac{\partial T}{\partial t} + u \frac{\partial T}{\partial x} = 0$$

- Activity:

- Write a discrete equation using a **forward Euler in time and an upstream scheme in space**:
- Check its stability

# Stability:

Example: ***The advection equation ( $u > 0$ )***  $\frac{\partial T}{\partial t} + u \frac{\partial T}{\partial x} = 0$

We use a **forward Euler in time and an upstream scheme in space**:

$$\frac{\partial T}{\partial t} = \frac{T(t + \delta t) - T(t)}{\delta t} = \frac{T_{i,n+1} - T_{i,n}}{\delta t}$$

$$\frac{\partial T}{\partial x} = \frac{T(x) - T(x - \delta x)}{\delta x} = \frac{T_{i,n} - T_{i-1,n}}{\delta x}$$

Such that:

$$T_{i,n+1} = T_{i,n} - u \frac{\delta t}{\delta x} (T_{i,n} - T_{i-1,n})$$

# Stability:

Example: ***The advection equation***

$$\frac{\partial T}{\partial t} + u \frac{\partial T}{\partial x} = 0$$

$$T_{i,n+1} = T_{i,n} - u \frac{\delta t}{\delta x} (T_{i,n} - T_{i-1,n})$$

We introduce:  $T_n = \hat{T}_n e^{ikx}$

We get

$$A = \frac{\hat{T}_{n+1}}{\hat{T}_n} = 1 - u \frac{\delta t}{\delta x} (1 - e^{-ik\delta x})$$

# Stability:

$$|A|^2 = (1 + C)^2 + 2(1 + C)C \cos(k\delta x) + C^2$$

$$|A|^2 = 1 + 2C(1 - C)(\cos(k\delta x) - 1)$$

For positive u:  $|A| < 1$  if  $C < 1$

- The Euler/upstream is conditionally stable if

$$C = u \frac{\delta t}{\delta x} < 1$$

# Stability

Courant-Friedrichs-Levy (CFL) stability criterion (1d) :

$$C = u \frac{\delta t}{\delta x} \leq C_{max}$$

... the time step must be kept small enough so that information has enough time to propagate through the space discretization

- For an explicit solver, typically:  $C_{max} = 1$
- The CFL condition is a necessary condition, but may not be sufficient for stability.

# Stability

Courant-Friedrichs-Levy (CFL) stability criterion :

$$C = \delta t \sum_{i=1}^n \frac{u_i}{\delta x_i} \leq C_{max}$$

... the time step must be kept small enough so that information has enough time to propagate through the space discretization

- For an explicit solver, typically:  $C_{max} = 1$
- The CFL condition is a necessary condition, but may not be sufficient for stability.

# Stability:

- Example: ***The advection equation***

$$\frac{\partial T}{\partial t} + u \frac{\partial T}{\partial x} = 0$$

- Activity:

- Write discrete equation using a **forward Euler in time and a centered scheme** in space:
- Check its stability

# Stability:

- Example: ***The advection equation***

$$\frac{\partial T}{\partial t} + u \frac{\partial T}{\partial x} = 0$$

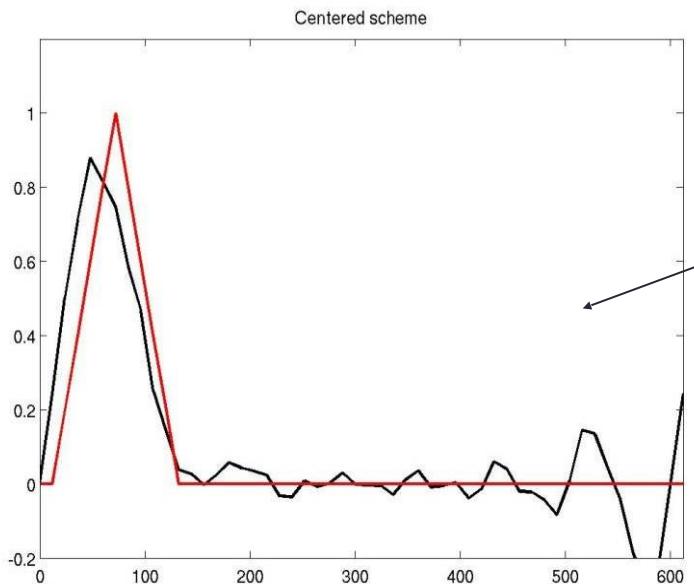
- forward Euler in time and **a centered scheme** in space:

$$T_{i,n+1} = T_{i,n} - u \frac{\delta t}{\delta x} \frac{T_{i+1,n} - T_{i-1,n}}{2}$$

$$|A|^2 = 1 + C^2 \sin(k\delta x)^2$$

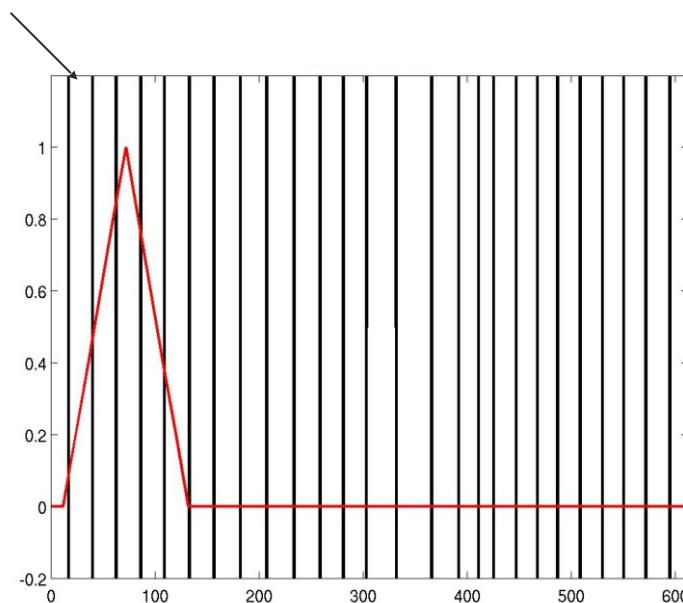
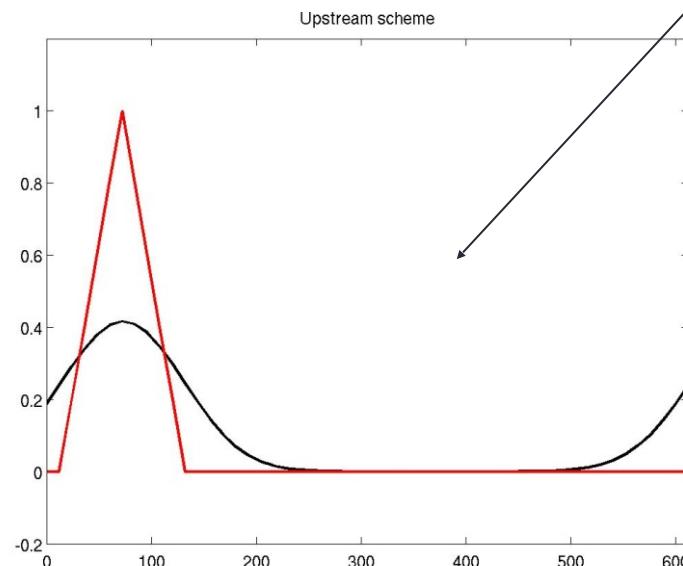
- **The Euler/centered (FCTS) is unconditionally unstable**

# Numerical properties



A numerical scheme can be:

- **Dispersive**: ripples, overshoot and extrema (Leapfrog/Centered)
- **Diffusive** (Euler /Upstream)
- **Unstable** (Euler/Centered)



# Stability:

- Example: ***The advection equation***

$$\frac{\partial T}{\partial t} + u \frac{\partial T}{\partial x} = 0$$

- One way to stabilise the Euler/Centered is to use a **Lax Friedrichs**:

$$T_{i,n}$$



$$T_{i,n+1} = \frac{T_{i+1,n} + T_{i-1,n}}{2} - u \frac{\delta t}{\delta x} \frac{T_{i+1,n} - T_{i-1,n}}{2}$$

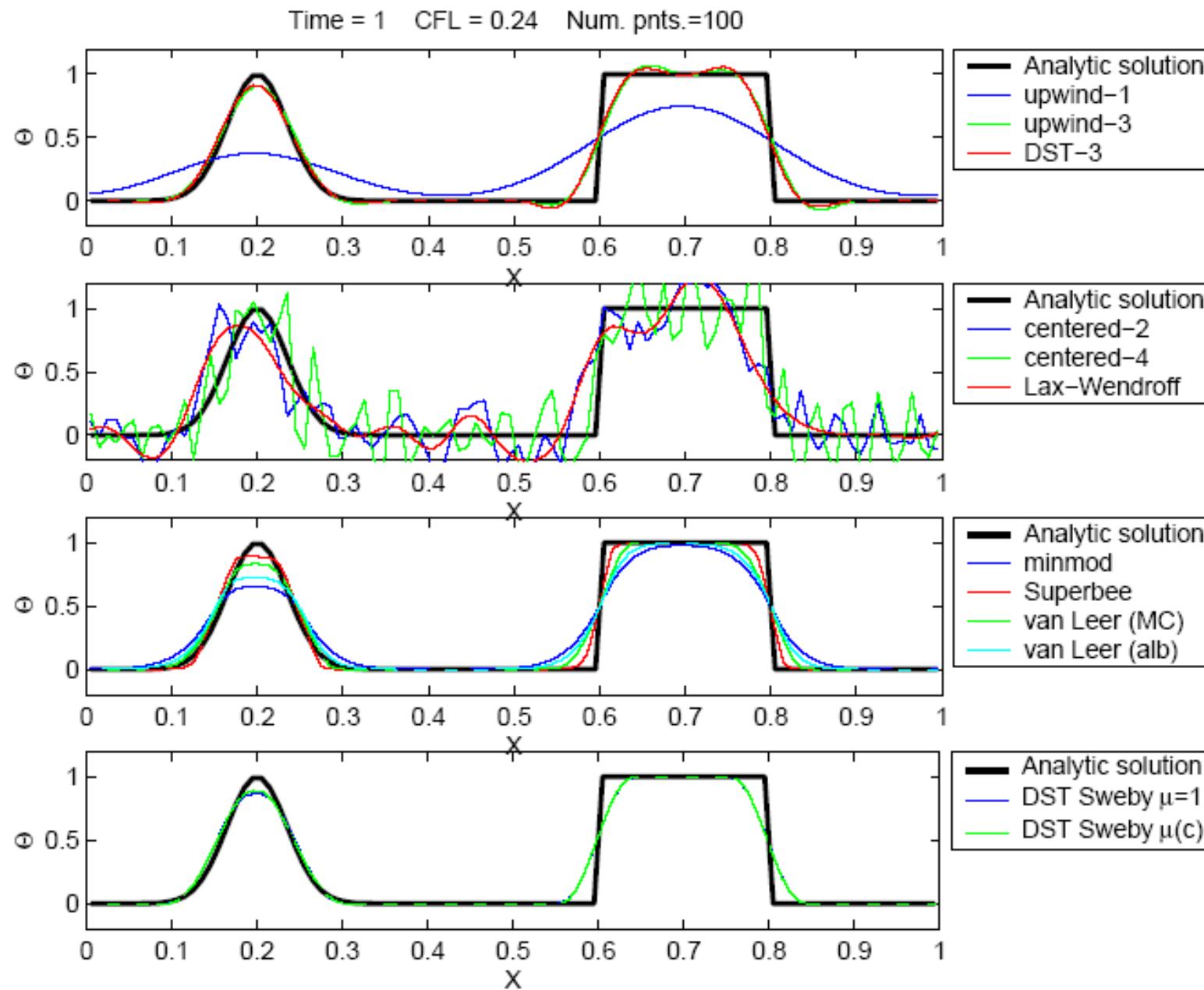
Stable (if  $C < 1$ ) but diffusive

# Stability:

- Example: ***The advection equation***  $\frac{\partial T}{\partial t} + u \frac{\partial T}{\partial x} = 0$
- We can have a 2<sup>nd</sup> order in time by using:
$$T_{n+1} = T_n + \delta t T_t + \frac{1}{2} \delta t^2 T_{tt} + \dots$$
$$T_{tt} = -u T_{tx} = u^2 T_{xx}$$
- Such that
$$T_{i,n+1} = T_{i,n} - u \frac{\delta t}{\delta x} \frac{T_{i+1,n} - T_{i-1,n}}{2} + \left( u \frac{\delta t}{\delta x} \right)^2 \frac{T_{i+1,n} + T_{i-1,n} - 2T_{i,n}}{2}$$

It is the **Lax-Wendroff Scheme** - **Stable but dispersive!**

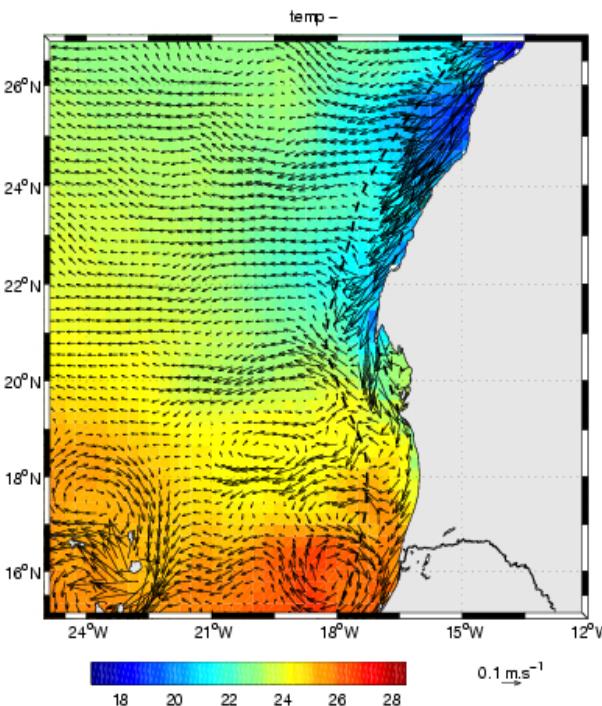
# Illustration of different numerical schemes



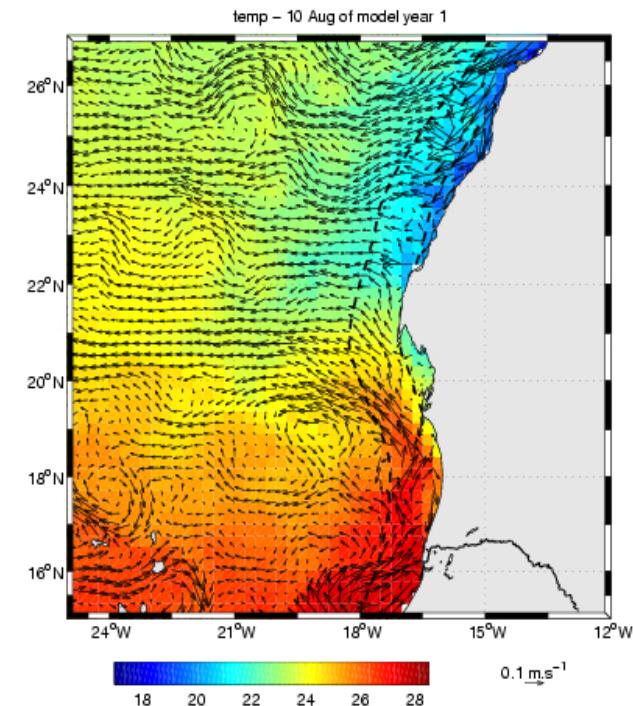
# Spatial advection scheme in ROMS/CROCO

- ❑ 3<sup>rd</sup> or 5<sup>th</sup> order, upstream-biased advection scheme : allows the generation of steep gradient, with a weak dispersion and weak diffusion.
- ❑ No need to impose explicit diffusion/ viscosity to avoid numerical noise (in case of 3D modeling)
- ❑ Effective resolution is improved :

NEMO -  
0.25 deg



ROMS -  
0.25 deg



# Upwind biased schemes :



- Upwind biased schemes:

Ex:  $\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0$

First order

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + a \frac{u_i^n - u_{i-1}^n}{\Delta x} = 0 \quad \text{for } a > 0$$

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + a \frac{u_{i+1}^n - u_i^n}{\Delta x} = 0 \quad \text{for } a < 0$$

2<sup>nd</sup> order

$$u_x^- = \frac{3u_i^n - 4u_{i-1}^n + u_{i-2}^n}{2\Delta x}$$

$$u_x^+ = \frac{-u_{i+2}^n + 4u_{i+1}^n - 3u_i^n}{2\Delta x}$$

3<sup>rd</sup> order

$$u_x^- = \frac{2u_{i+1} + 3u_i - 6u_{i-1} + u_{i-2}}{6\Delta x} \quad u_x^+ = \frac{-u_{i+2} + 6u_{i+1} - 3u_i - 2u_{i-1}}{6\Delta x}$$

# Available schemes in CROCO

- See formation\_adv\_diff.pdf

# Stability

- Courant-Friedrichs-Levy (CFL) stability criterion :

$$C = \delta t \sum_{i=1}^n \frac{u_i}{\delta x_i} \leq C_{max}$$

- Approximate limitations on time-step due to propagation processes in the ocean :

Process	Speed	Formula for maximum $\Delta t$	Maximum $\Delta t$ using $\Delta x = 20$ km	Maximum $\Delta t$ using $\Delta x = 200$ km
Sound waves, $c_s$	$1500 \text{ m s}^{-1}$	$\Delta x/c_s$	15 sec	2 min
External waves, $\sqrt{gH}$	$200 \text{ m s}^{-1}$	$\Delta x/\sqrt{gH}$	2 min	15 min
Internal waves, $NH$	$3 \text{ m s}^{-1}$	$\Delta x/NH$	2 hour	18 hour
Jets, $U$	$2 \text{ m s}^{-1}$	$\Delta x/U$	3 hour	1 day
Interior flow, $U$	$0.1 \text{ m s}^{-1}$	$\Delta x/U$	2 days	3 weeks

# Split-explicit method:

- Since the limitation on time-step due to external gravity waves is due only to the depth integrated equations we can try split out that part of the system and integrate it with a shorter time step than the rest of the model.
- We first obtain an approximation for the barotropic momentum equations:

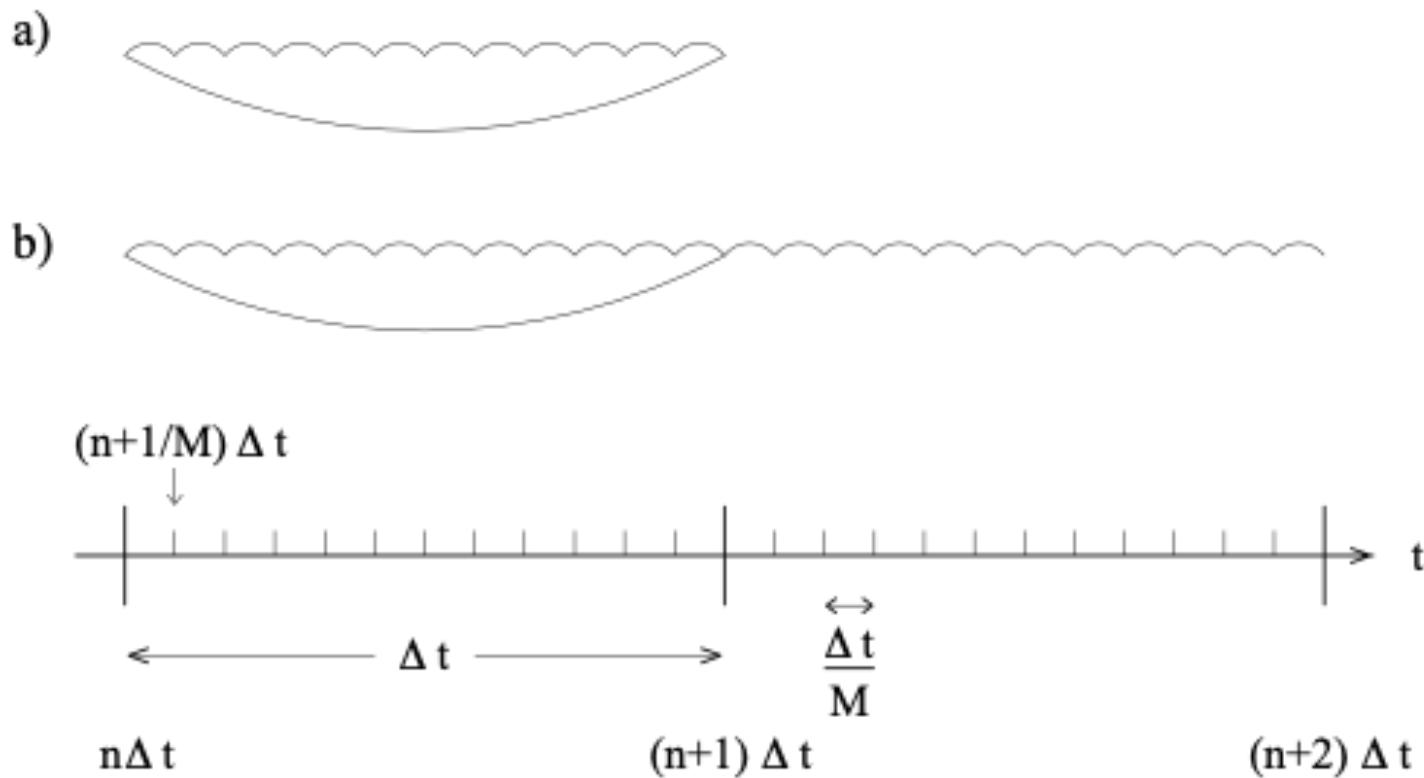
$$\partial_t \langle \vec{v}_h \rangle + g \nabla \eta = \langle \vec{G}_h \rangle$$

where  $\langle G_h \rangle$  is the depth average of all the terms in the full momentum equations.

- We then integrate this equation with the free-surface equation forward using a short time step,  $\Delta t/M$ , where  $\Delta$  is the regular time-step of the full model.

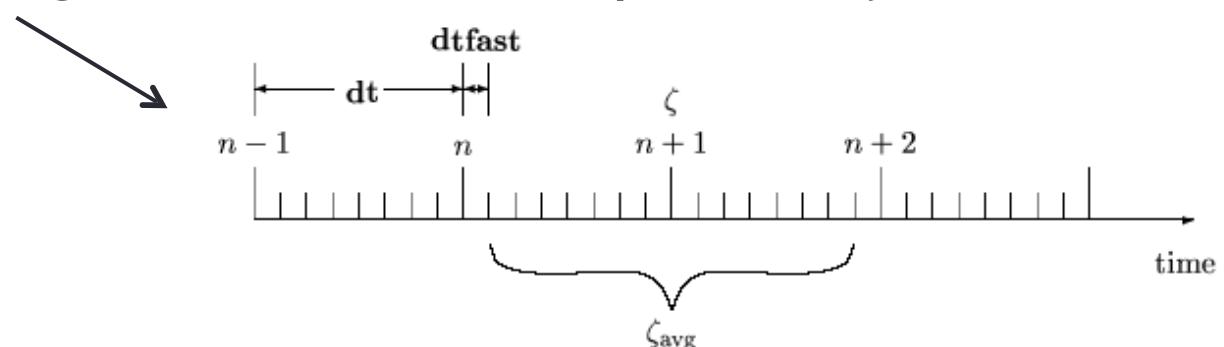
# Split-explicit method:

- We then integrate this equation with the free-surface equation forward using a short time step,  $\Delta t/M$ , where  $\Delta t$  is the regular time-step of the full model.

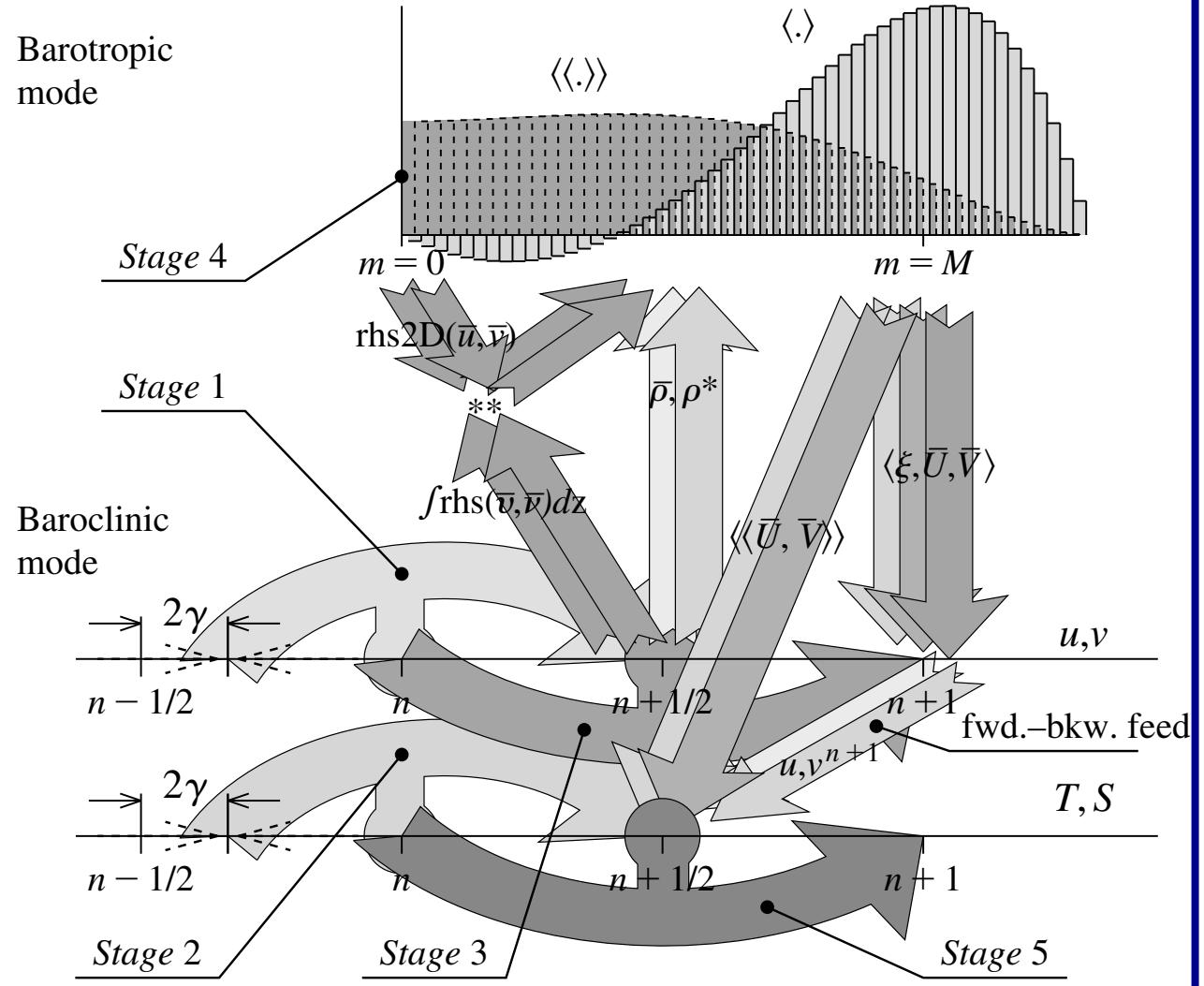


# Time Stepping in ROMS

- ROMS resolve free surface using the time-splitting method :
  - ❑ Direct integration of the barotropic equations
  - ❑ Getting the free surface is straight forward
  - ❑ Good parallelization performances
  - ❑ BUT difficulties to separate fast and slow modes : possible instabilities. To avoid that:
    - time averaging over the barotropic subcycle



# Time Stepping in ROMS



LeapFrog – third-order  
Adams-Moulton (LF–AM3)  
predictor-corrector step for  
the baroclinic (3D) mode with  
mode coupling during the  
corrector stage.

# Time Stepping in ROMS

- Shchepetkin, A., and J. McWilliams, 2008: Computational kernel algorithms for fine- scale, multiprocess, longtime oceanic simulations. *Handb. Nu- mer. Anal.*, 14, 121–183, doi:10.1016/S1570-8659(08)01202-0.

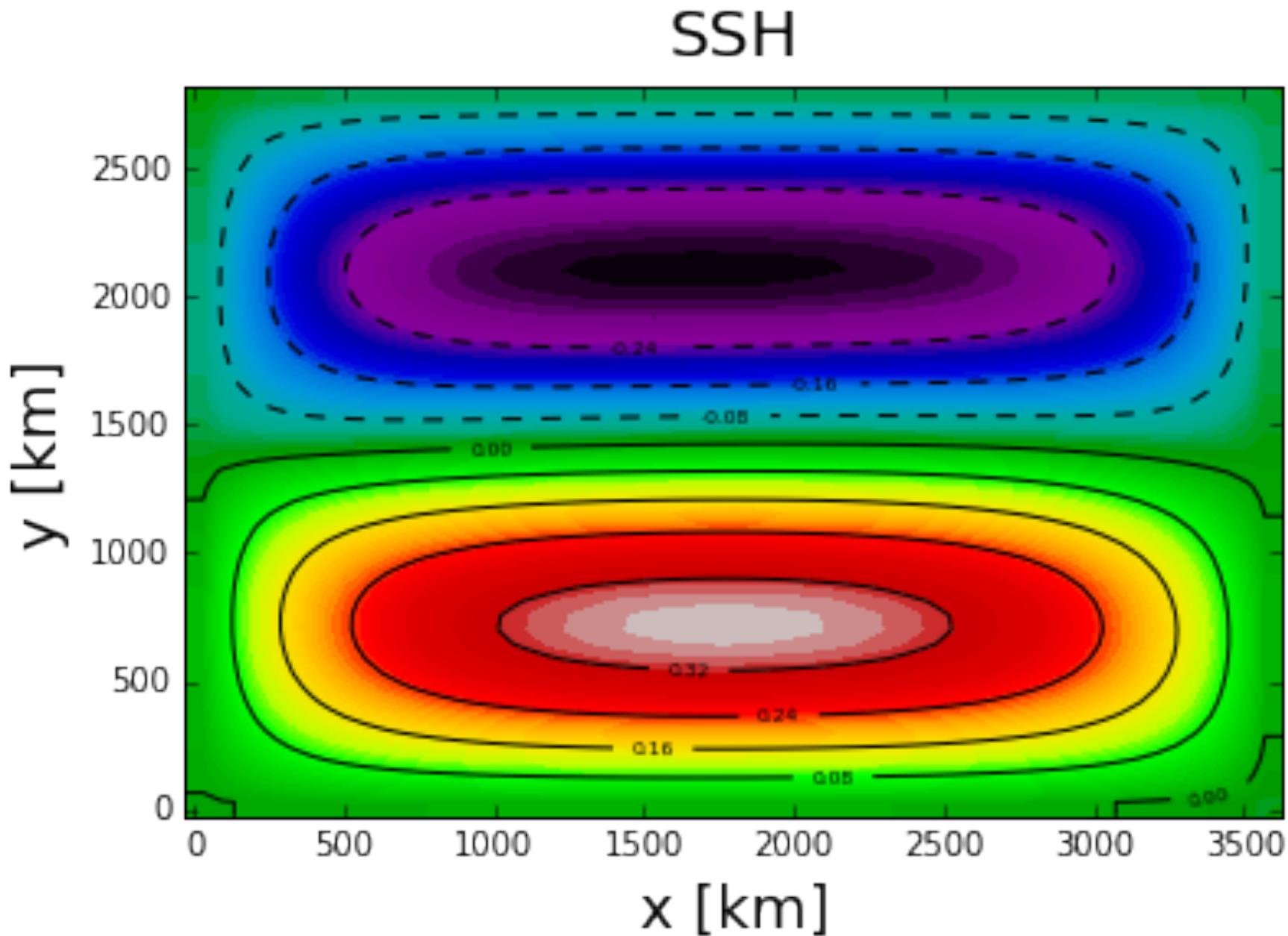
<http://jgula.fr/ModNum/ShchepetkinMcWilliams08.pdf>

- See also: ROMS time-stepping flow chart [from F. Lemarié]:

[http://jgula.fr/ModNum/croco\\_step\\_hydro\\_en.pdf](http://jgula.fr/ModNum/croco_step_hydro_en.pdf)

- See also: formation\_step.pdf

## Activity 3 - Run an idealized ocean basin III



# Example: 1D convection-diffusion equation

$$\begin{cases} \frac{\partial u}{\partial t} + v \frac{\partial u}{\partial x} = d \frac{\partial^2 u}{\partial x^2} & \text{in } (0, X) \times (0, T) \\ u(0) = u(1) = 0, \quad u|_{t=0} = u_0 & \\ f(t, u(t)) = -v \frac{\partial u}{\partial x} + d \frac{\partial^2 u}{\partial x^2} & \end{cases}$$

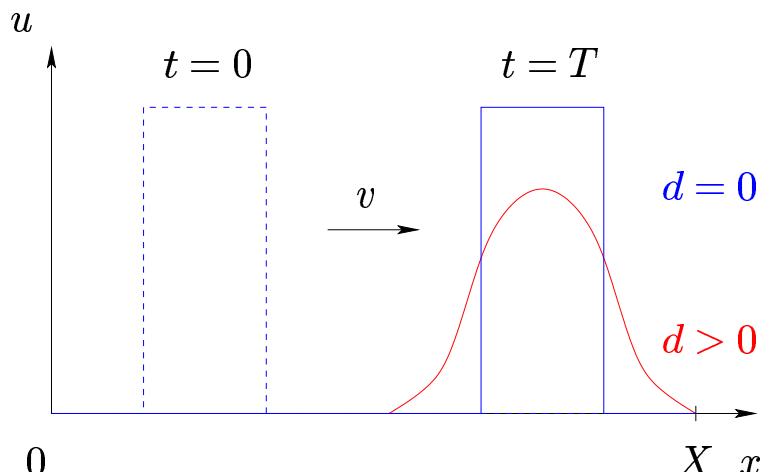
Lagrangian representation  $\frac{du(x(t), t)}{dt} = d \frac{\partial^2 u}{\partial x^2}$  pure diffusion equation

where  $\frac{d}{dt}$  is the substantial derivative along the characteristic lines  $\frac{dx(t)}{dt} = v$

Initial profile is convected at speed  $v$   
and smeared by diffusion if  $d > 0$

$$d = 0 \quad \Rightarrow \quad \frac{du(x(t), t)}{dt} = 0$$

For the pure convection equation  $u$  is  
constant along the characteristics



# Example: 1D convection-diffusion equation

Uniform space-time mesh

$$x_i = i\Delta x, \quad \Delta x = \frac{X}{N}, \quad i = 0, \dots, N$$

$$t^n = n\Delta t, \quad \Delta t = \frac{T}{M}, \quad n = 0, \dots, M$$

$$u^n \rightarrow u^{n+1}, \quad u_i^0 = u_0(x_i)$$

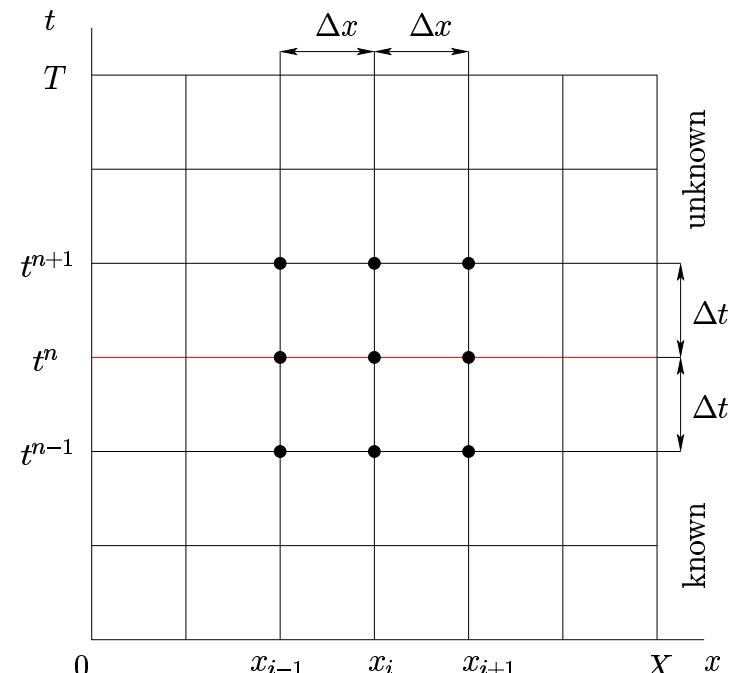
Fully discretized equation  $0 \leq \theta \leq 1$

$$u_i^{n+1} = u_i^n + [\theta f_h^{n+1} + (1 - \theta) f_h^n] \Delta t$$

Central difference / lumped-mass FEM

$$\begin{aligned} \frac{u_i^{n+1} - u_i^n}{\Delta t} &= \theta \left[ -v \frac{u_{i+1}^{n+1} - u_{i-1}^{n+1}}{2\Delta x} + d \frac{u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}}{(\Delta x)^2} \right] \\ &\quad + (1 - \theta) \left[ -v \frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x} + d \frac{u_{i-1}^n - 2u_i^n + u_{i+1}^n}{(\Delta x)^2} \right] \end{aligned}$$

*a sequence of tridiagonal linear systems*  $i = 1, \dots, N, \quad n = 0, \dots, M - 1$



# Example: 1D convection-diffusion equation

Standard  $\theta$ -scheme (two-level)

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + v \frac{u_{i+1}^{n+\theta} - u_{i-1}^{n+\theta}}{2\Delta x} = d \frac{u_{i-1}^{n+\theta} - 2u_i^{n+\theta} + u_{i+1}^{n+\theta}}{(\Delta x)^2}$$

$$u_i^{n+\theta} = \theta u_i^{n+1} + (1 - \theta) u_i^n, \quad 0 \leq \theta \leq 1$$

Forward Euler ( $\theta = 0$ )  $u_i^{n+1} = h(u_{i-1}^n, u_i^n, u_{i+1}^n)$

Backward Euler ( $\theta = 1$ )  $u_i^{n+1} = h(\textcolor{red}{u_{i-1}^{n+1}}, u_i^n, \textcolor{red}{u_{i+1}^{n+1}})$

Crank-Nicolson ( $\theta = \frac{1}{2}$ )  $u_i^{n+1} = h(\textcolor{red}{u_{i-1}^{n+1}}, u_{i-1}^n, u_i^n, u_{i+1}^n, \textcolor{red}{u_{i+1}^{n+1}})$

Leapfrog time-stepping  $u_i^{n+1} = u_i^{n-1} + 2\Delta t f_h^n$

$$\frac{u_i^{n+1} - u_i^{n-1}}{2\Delta t} + v \frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x} = d \frac{u_{i-1}^n - 2u_i^n + u_{i+1}^n}{(\Delta x)^2}$$

(explicit, three-level)  $u_i^{n+1} = h(u_{i-1}^n, u_i^{n-1}, u_i^n, u_{i+1}^n)$

