Time integration

Numerical Flow Simulation

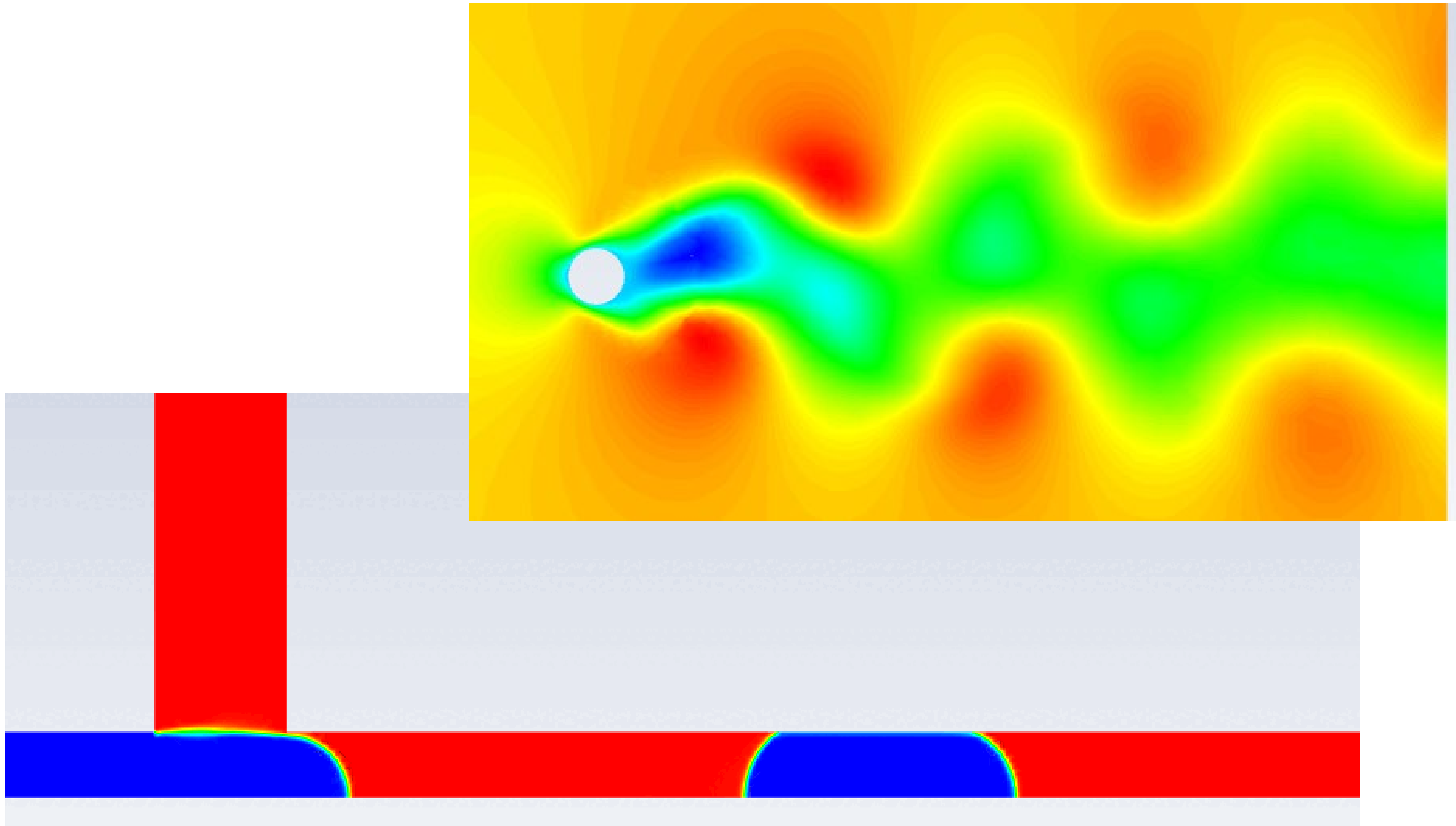
Edouard Boujo

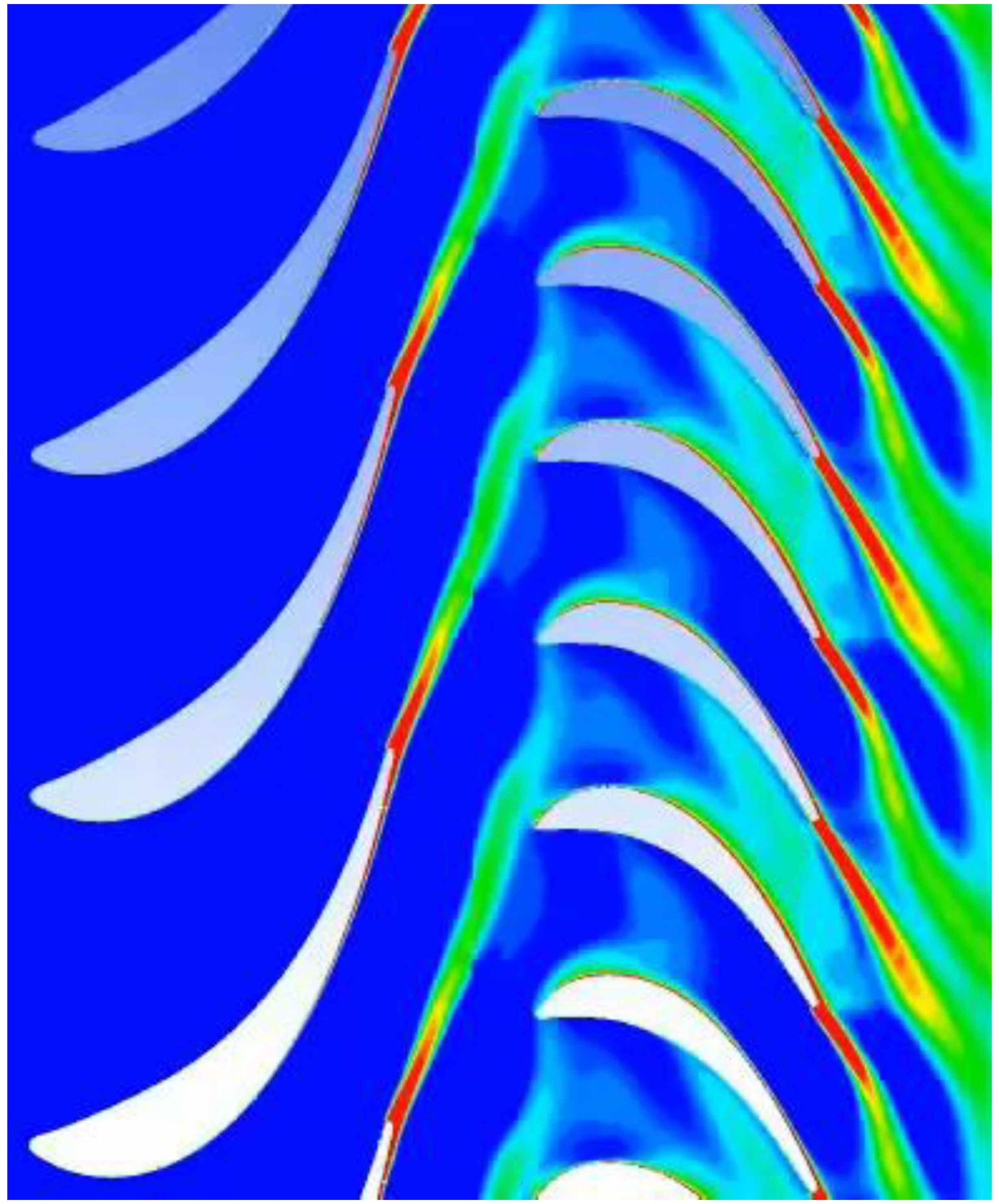
Fall 2022

École
polytechnique
fédérale
de Lausanne

Unsteady flows

- Ubiquitous in nature and engineering: instabilities, moving or deforming boundaries, natural convection, multiphase flows, turbulent flows…

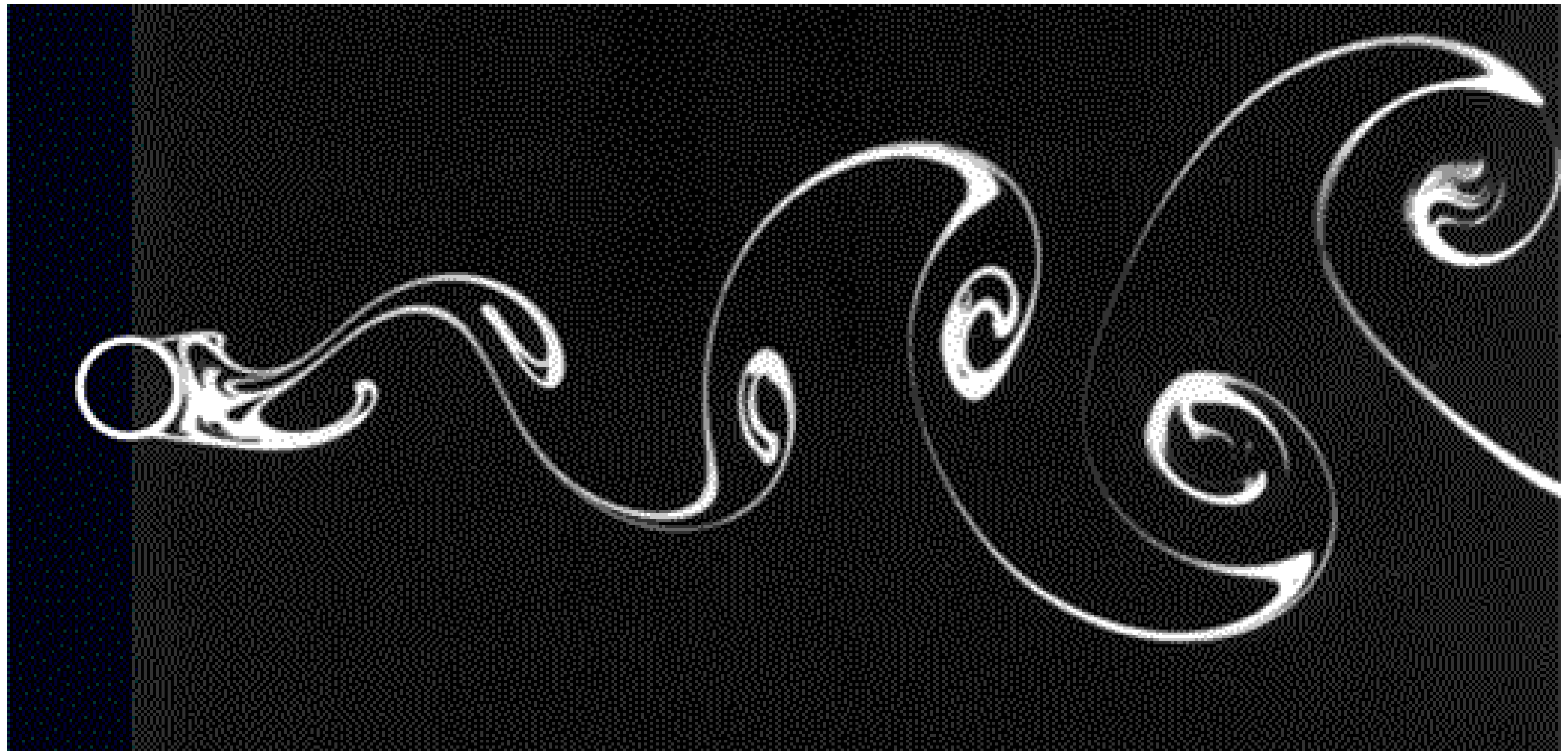




Unsteady flows

- Need to integrate the equations in time.

Steady flow: solution **u**(**x**), *p*(**x**)
depends on space only.



Unsteady flow: solution **u**(**x**,t), *p*(**x**,t)
depends on both space and time.

Simple "model" equations

- General conservation equation:

$$\boxed{\frac{\partial(\rho\phi)}{\partial t}} + \boxed{div(\rho\phi\mathbf{u})} = \boxed{div(\Gamma\, grad(\phi))} + \boxed{S}$$

unsteadiness convection diffusion source

- Steady/unsteady diffusion:

$$\boxed{div(\Gamma\, grad(\phi))} + S = 0$$

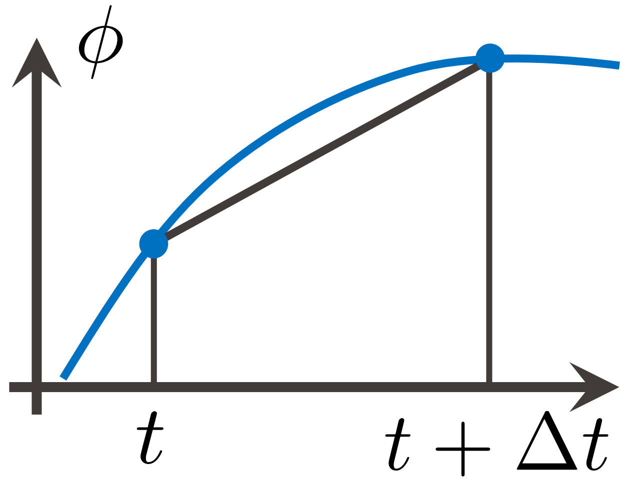
$$\boxed{\frac{\partial(\rho\phi)}{\partial t}} = div(\Gamma\, grad(\phi))$$

- Steady convection-diffusion, unsteady convection:

$$\boxed{div(\rho\phi\mathbf{u})} = div(\Gamma\, grad(\phi))$$

$$\boxed{\frac{\partial(\rho\phi)}{\partial t}} + div(\rho\phi\mathbf{u}) = 0$$

Reminder: ordinary differential equations (ODE)



- Unknown $\phi(t)$ depends on **time only**. The RHS may depend explicitly on time, and may be nonlinear in ϕ:

$$\frac{d\phi}{dt} = f(t, \phi) \qquad \phi(0) = \phi_0$$

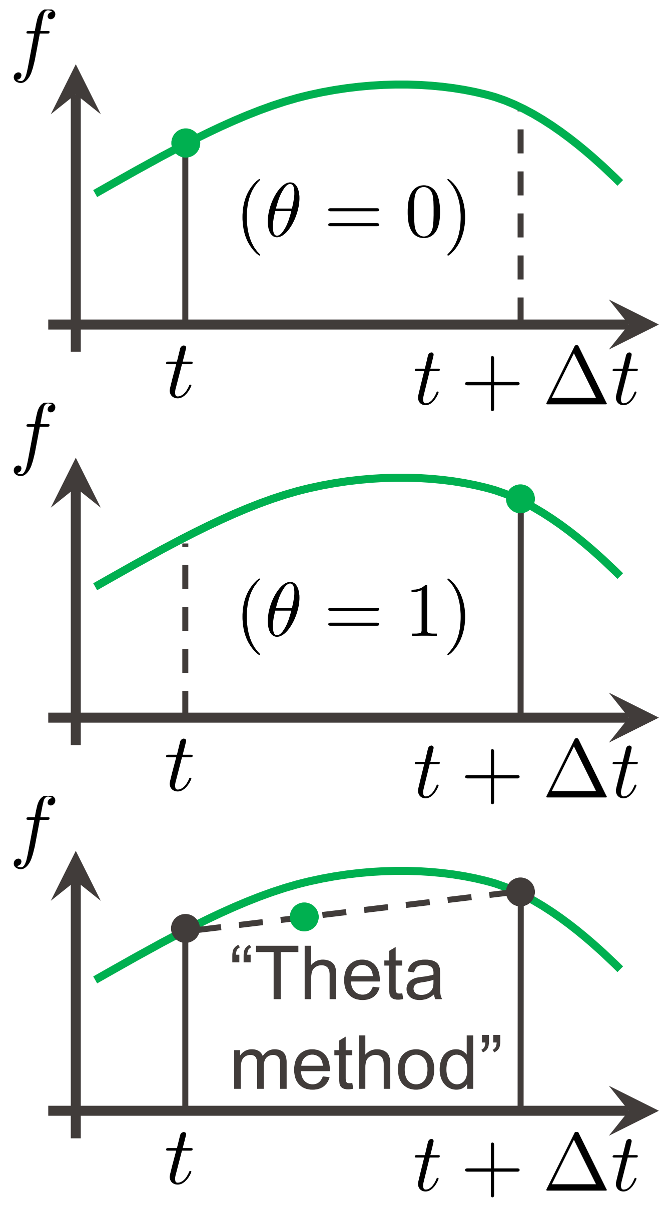
- Simplest methods: linear approximation of time derivative: $\displaystyle \frac{d\phi}{dt} \approx \frac{\phi^{n+1} - \phi^n}{\Delta t}$

- Can evaluate the RHS at different times:

$$\frac{\phi^{n+1} - \phi^n}{\Delta t} = f^n \quad \Rightarrow \quad \phi^{n+1} = \phi^n + \Delta t \, f^n$$

$$\frac{\phi^{n+1} - \phi^n}{\Delta t} = f^{n+1} \quad \Rightarrow \quad \phi^{n+1} = \phi^n + \Delta t \, f^{n+1}$$

$$\frac{\phi^{n+1} - \phi^n}{\Delta t} = \theta f^{n+1} + (1-\theta) f^n \quad \Rightarrow \quad \phi^{n+1} = \phi^n + \Delta t \left(\boxed{\theta f^{n+1}} + (1-\theta) f^n \right)$$

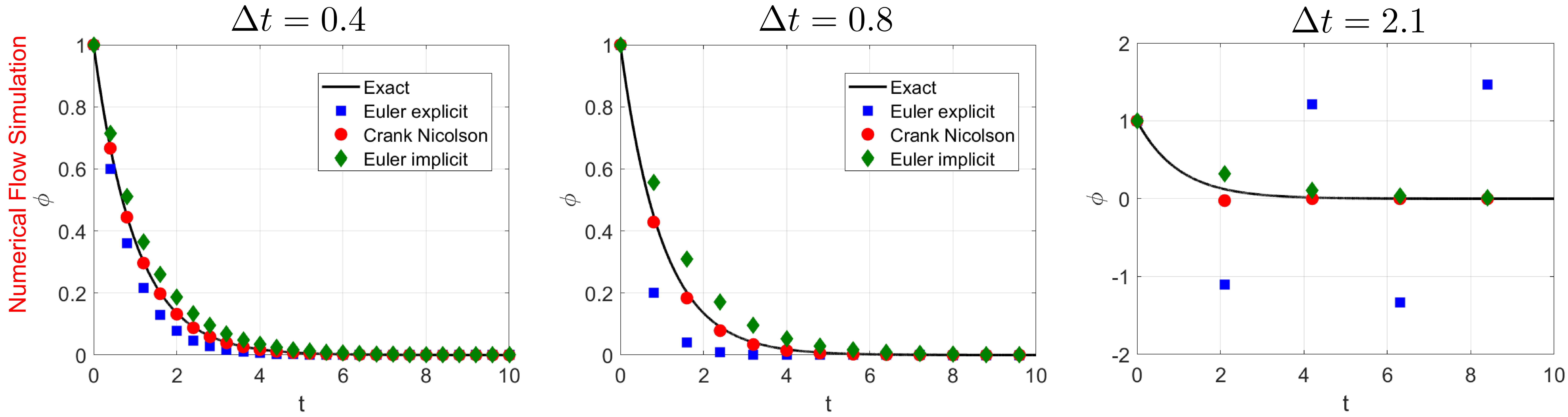


- If $\theta \neq 0$, must solve an implicit equation.

Reminder: ordinary differential equations (ODE)

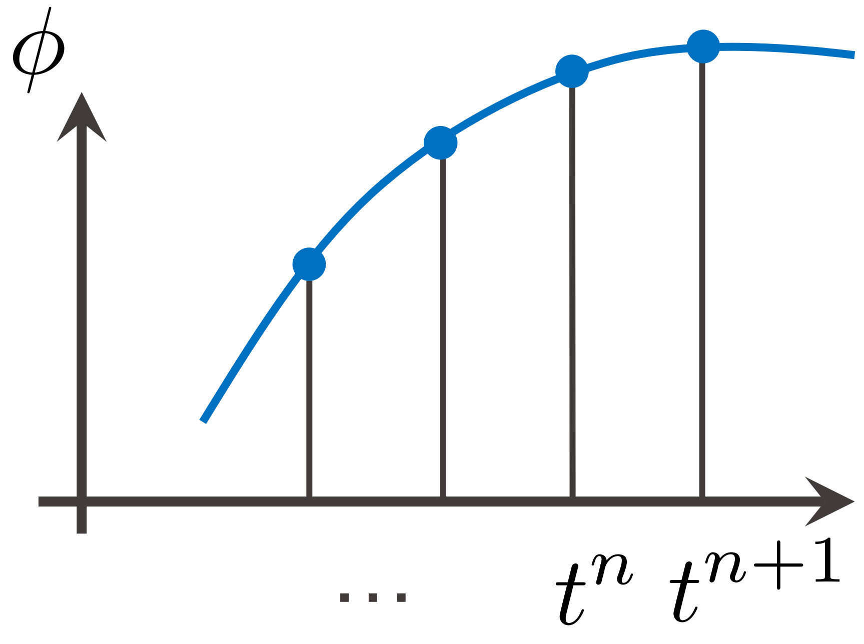
- **Explicit Euler** (“forward Euler”): 1^{st}-order accurate; can be unstable
- **Implicit Euler** (“backward Euler”): 1^{st}-order accurate; unconditionally stable
- **Crank-Nicolson** ($\theta = 1/2$): 2^{nd}-order accurate; unconditionally stable

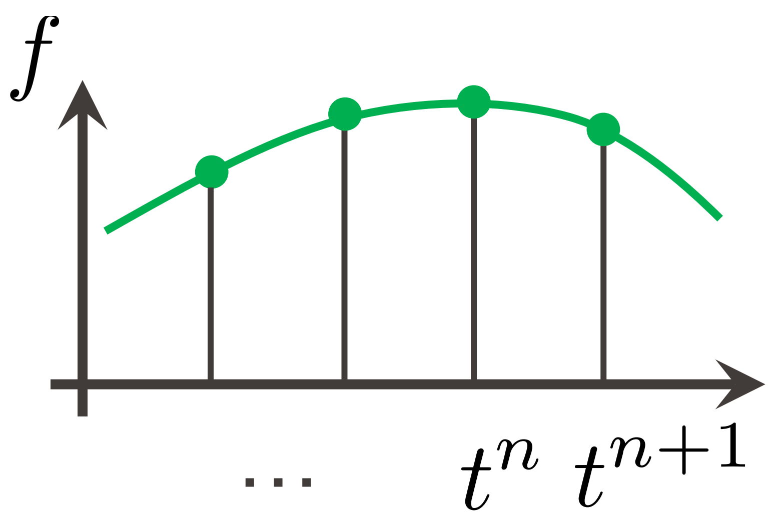
- Example: $\dfrac{d\phi}{dt} = -\phi \quad \longrightarrow \quad \phi(t) = \phi_0 e^{-t}$



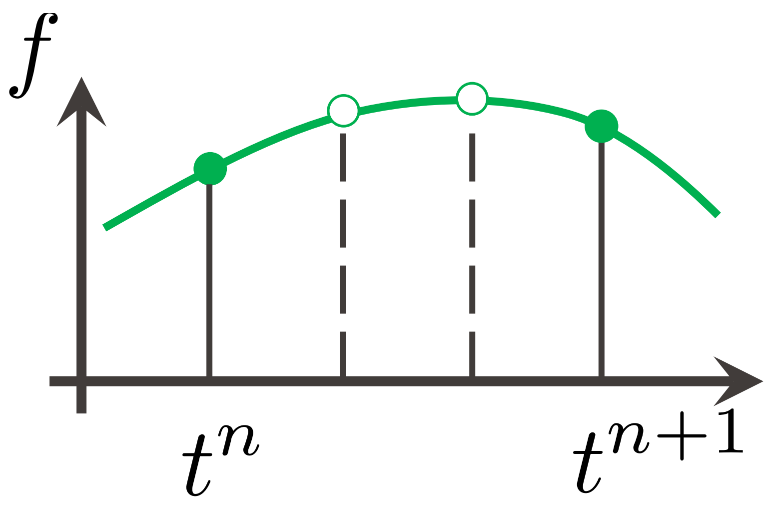
Reminder: ordinary differential equations (ODE)

- Other families of methods:
  1. **Linear multistep methods**: use more than 1 previous time step





  2. **Multistage methods**: evaluate f at intermediate stages between t and $t + \Delta t$



Reminder: ordinary differential equations (ODE)

1. Linear multistep methods: a *s*-step method uses the known solution at *s* previous time steps

$$\alpha_0\phi^{n+1}+\alpha_1\phi^{n}+\alpha_2\phi^{n-1}\ldots+\alpha_s\phi^{n+1-s} = \Delta t\left(\boxed{\beta_0 f^{n+1}}+\beta_1 f^{n}+\beta_2 f^{n-1}\ldots+\beta_s f^{n+1-s}\right)$$

- Implicit if $\beta_0 \neq 0$
- Adams family: LHS always $\dfrac{\phi^{n+1}-\phi^{n}}{\Delta t}$, i.e. $\alpha_0 = 1, \quad \alpha_1 = -1, \quad \alpha_j = 0 \ \forall j > 1$
  - Adams-Bashforth: explicit, accuracy *s*
  - Adams-Moulton: implicit, accuracy *s+1*
- Backward-differentiation formula (BDF) family: RHS always f^{n+1}, i.e. $\beta_0 = 1, \quad \beta_j = 0 \ \forall j > 1$
  - Implicit, accuracy *s*

Reminder: ordinary differential equations (ODE)

1. Linear multistep methods

$$\alpha_0\phi^{n+1}+\alpha_1\phi^{n}+\alpha_2\phi^{n-1}\ldots+\alpha_s\phi^{n+1-s}=\Delta t\left(\beta_0 f^{n+1}+\beta_1 f^{n}+\beta_2 f^{n-1}\ldots+\beta_s f^{n+1-s}\right)$$

Adams-Bashforth

s	accuracy	β_1	β_2	β_3	β_4	
1	1	1				(Explicit Euler)
2	2	3/2	-1/2			
3	3	23/12	-16/12	5/12		
4	4	55/24	-59/24	37/24	-9/24	

Adams-Moulton

s	accuracy	β_0	β_1	β_2	β_3	
0	1	1				(Implicit Euler)
1	2	1/2	1/2			(Crank-Nicolson)
2	3	5/12	8/12	-1/12		
3	4	9/24	19/24	-5/24	1/24	

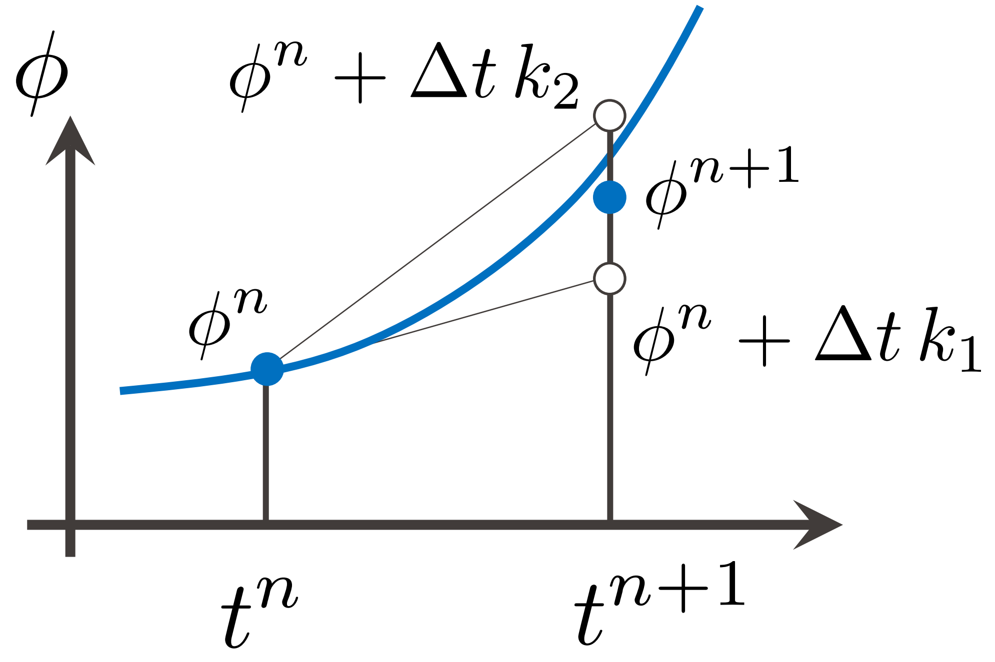
BDF

s	accuracy	α_0	α_1	α_2	α_3	α_4	β_0	
1	1	1	-1				1	(Implicit Euler)
2	2	1	-4/3	1/3			2/3	
3	3	1	-18/11	9/11	-2/11		6/11	
4	4	1	-48/25	36/25	-16/25	3/25	12/25	

Reminder: ordinary differential equations (ODE)

2. Multistage methods

- Most famous: explicit Runge-Kutta
  - RK2 (2nd-order accurate, 2 evaluations of the RHS per time step): estimate the new solution with the current slope
    → recompute the slope using this estimate → correct the estimate



$$\phi^{n+1} = \phi^n + \Delta t \frac{k_1 + k_2}{2}$$

$$k_1 = f(t^n, \phi^n)$$

$$k_2 = f\left(t^n + \Delta t, \phi^n + \Delta t\, k_1\right)$$

  - RK4 (4th-order accurate, 4 evaluations of the RHS per time step): same idea, with intermediate values at time interval midpoint

$$\phi^{n+1} = \phi^n + \Delta t \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}$$

$$k_1 = f(t^n, \phi^n)$$

$$k_2 = f\left(t^n + \frac{\Delta t}{2}, \phi^n + \frac{\Delta t}{2} k_1\right)$$

$$k_3 = f\left(t^n + \frac{\Delta t}{2}, \phi^n + \frac{\Delta t}{2} k_2\right)$$

$$k_4 = f\left(t^n + \Delta t, \phi^n + \Delta t\, k_3\right)$$

- More stable than explicit multistep methods, but more expensive per time step.

Partial differential equations (PDE)

- Now the unknown $\phi(\mathbf{x}, t)$ depends on both **time and space**.

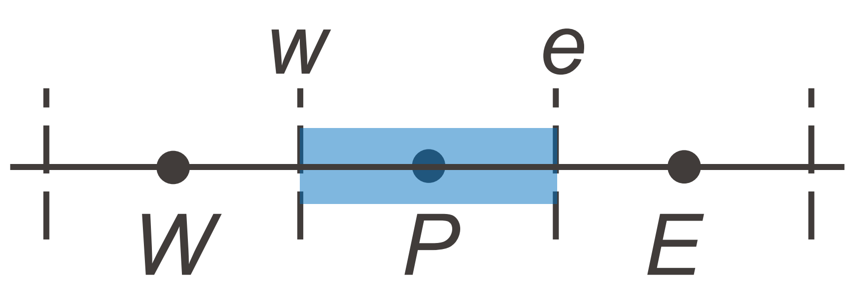
- The time-marching problem looks qualitatively similar, because spatial discretization turns the PDE into an ODE in time:

$$\left.\begin{aligned} \frac{\partial(\rho\phi)}{\partial t} + div(\rho\phi\mathbf{u}) = div(\Gamma\, grad(\phi)) + S \\ \phi(\mathbf{x}, t) \rightarrow \boldsymbol{\phi}(t) = (\phi_1(t),\, \phi_2(t)\ldots) \end{aligned}\right\} \quad \rightarrow \quad \frac{\partial \boldsymbol{\phi}}{\partial t} = \mathbf{A}(\boldsymbol{\phi})$$

- However, not so simple: small spatial errors at each time step can accumulate and compromise accuracy / stability.

Unsteady 1D diffusion

$$\frac{\partial(\rho\phi)}{\partial t} = \frac{\partial}{\partial x}\left(\Gamma\frac{\partial \phi}{\partial x}\right)$$



- Assume the density is constant.
- Integration over CV, with CD scheme for diffusion term:

$$\rho\int_{x_w}^{x_e}\frac{\partial\phi}{\partial t}dx \approx \rho\frac{\partial\phi_P}{\partial t}\Delta x = \left(\Gamma\frac{\partial\phi}{\partial x}\right)_e - \left(\Gamma\frac{\partial\phi}{\partial x}\right)_w \approx \Gamma_e\frac{\phi_E-\phi_P}{\delta x_{PE}} - \Gamma_w\frac{\phi_P-\phi_W}{\delta x_{WP}}$$

- Linear approximation of the time derivative, theta method for the RHS:

$$\rho\frac{\phi_P^{n+1}-\phi_P^n}{\Delta t}\Delta x = \theta\left[\Gamma_e\frac{\phi_E^{n+1}-\phi_P^{n+1}}{\delta x_{PE}} - \Gamma_w\frac{\phi_P^{n+1}-\phi_W^{n+1}}{\delta x_{WP}}\right] + (1-\theta)\left[\Gamma_e\frac{\phi_E^{n}-\phi_P^{n}}{\delta x_{PE}} - \Gamma_w\frac{\phi_P^{n}-\phi_W^{n}}{\delta x_{WP}}\right]$$

- Algebraic equation for $\boldsymbol{\phi}^{n+1}$: $\quad a_P\phi_P^{n+1} = a_W\phi_W^{n+1} + a_E\phi_E^{n+1} + b(\boldsymbol{\phi}^n)$

- Assemble the system and march in time:

$$\mathbf{A}\boldsymbol{\phi}^{n+1} = \mathbf{b}(\boldsymbol{\phi}^n) \quad \rightarrow \quad \boldsymbol{\phi}^{n+1} = \mathbf{A}^{-1}\mathbf{b}(\boldsymbol{\phi}^n)$$

Unsteady 1D diffusion

- Algebraic equation: $$a_P \phi_P^{n+1} = a_W \phi_W^{n+1} + a_E \phi_E^{n+1} + b(\boldsymbol{\phi}^n)$$

$$a_P = \frac{\rho \Delta x}{\Delta t} + \frac{\theta\,\Gamma_e}{\delta x_{PE}} + \frac{\theta\,\Gamma_w}{\delta x_{WP}} \qquad a_E = \frac{\theta\,\Gamma_e}{\delta x_{PE}} \qquad a_W = \frac{\theta\,\Gamma_w}{\delta x_{WP}}$$

$$b = \left(\frac{\rho \Delta x}{\Delta t} - \frac{(1-\theta)\Gamma_w}{\delta x_{WP}} - \frac{(1-\theta)\Gamma_e}{\delta x_{PE}}\right)\phi_P^n + \left(\frac{(1-\theta)\Gamma_e}{\delta x_{PE}}\right)\phi_E^n + \left(\frac{(1-\theta)\Gamma_w}{\delta x_{WP}}\right)\phi_W^n$$

- **Explicit Euler** ($\theta = 0$):

$$\boxed{a_P = \frac{\rho \Delta x}{\Delta t}, \quad a_W = 0, \quad a_E = 0}$$

Each CV decoupled from neighbors.
No need to solve a system!

$$b = \boxed{\left(\frac{\rho \Delta x}{\Delta t} - \frac{\Gamma_w}{\delta x_{WP}} - \frac{\Gamma_e}{\delta x_{PE}}\right)}\phi_P^n + \left(\frac{\Gamma_e}{\delta x_{PE}}\right)\phi_E^n + \left(\frac{\Gamma_w}{\delta x_{WP}}\right)\phi_W^n$$

ϕ_W^{n+1}	ϕ_P^{n+1}	ϕ_E^{n+1}
ϕ_W^n	ϕ_P^n	ϕ_E^n

Boundedness criterion: all coefficients should have the same sign. Here, if uniform grid and constant Γ: $$\Delta t < \frac{\rho (\Delta x)^2}{2\Gamma}$$

Unsteady 1D diffusion

Each CV coupled to neighbors.
Must solve a linear system at each iteration!

- **Crank-Nicolson** ($\theta = 1/2$):

$$\boxed{a_P = \frac{\rho\Delta x}{\Delta t} + \frac{\Gamma_e}{2\delta x_{PE}} + \frac{\Gamma_w}{2\delta x_{WP}}, \quad a_W = \frac{\Gamma_w}{2\delta x_{WP}}, \quad a_E = \frac{\Gamma_e}{2\delta x_{PE}}}$$

$$b = \boxed{\left(\frac{\rho\Delta x}{\Delta t} - \frac{\Gamma_w}{2\delta x_{WP}} - \frac{\Gamma_e}{2\delta x_{PE}}\right)} \phi_P^n + \left(\frac{\Gamma_e}{2\delta x_{PE}}\right)\phi_E^n + \left(\frac{\Gamma_w}{2\delta x_{WP}}\right)\phi_W^n$$

Similar boundedness criterion: $\Delta t < \dfrac{\rho(\Delta x)^2}{\Gamma}$

- **Implicit Euler** ($\theta = 1$):

$$\boxed{a_P = \frac{\rho\Delta x}{\Delta t} + \frac{\Gamma_e}{\delta x_{PE}} + \frac{\Gamma_w}{\delta x_{WP}}, \quad a_W = \frac{\Gamma_w}{\delta x_{WP}}, \quad a_E = \frac{\Gamma_e}{\delta x_{PE}}}$$

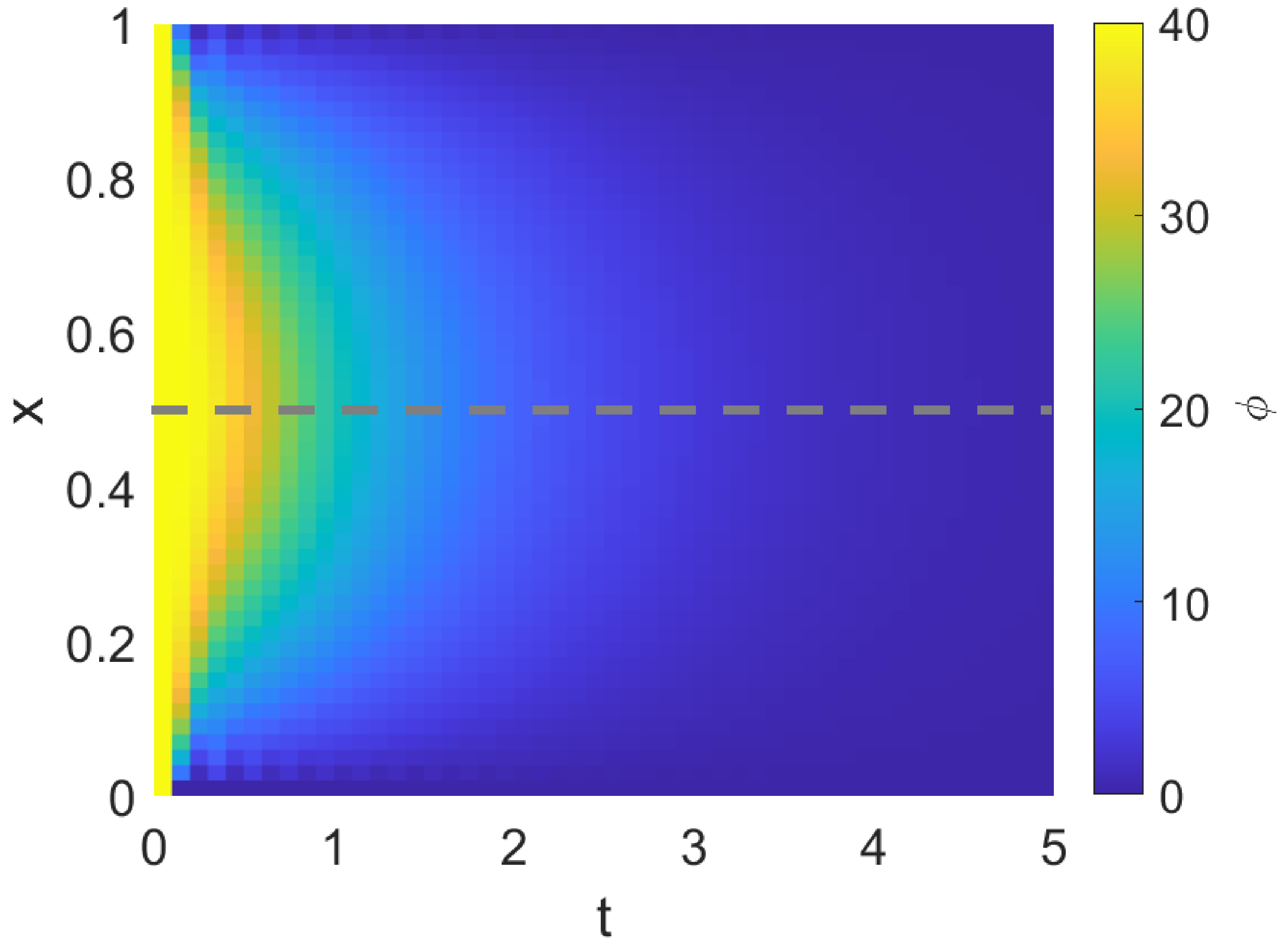
$$b = \left(\frac{\rho\Delta x}{\Delta t}\right)\phi_P^n$$

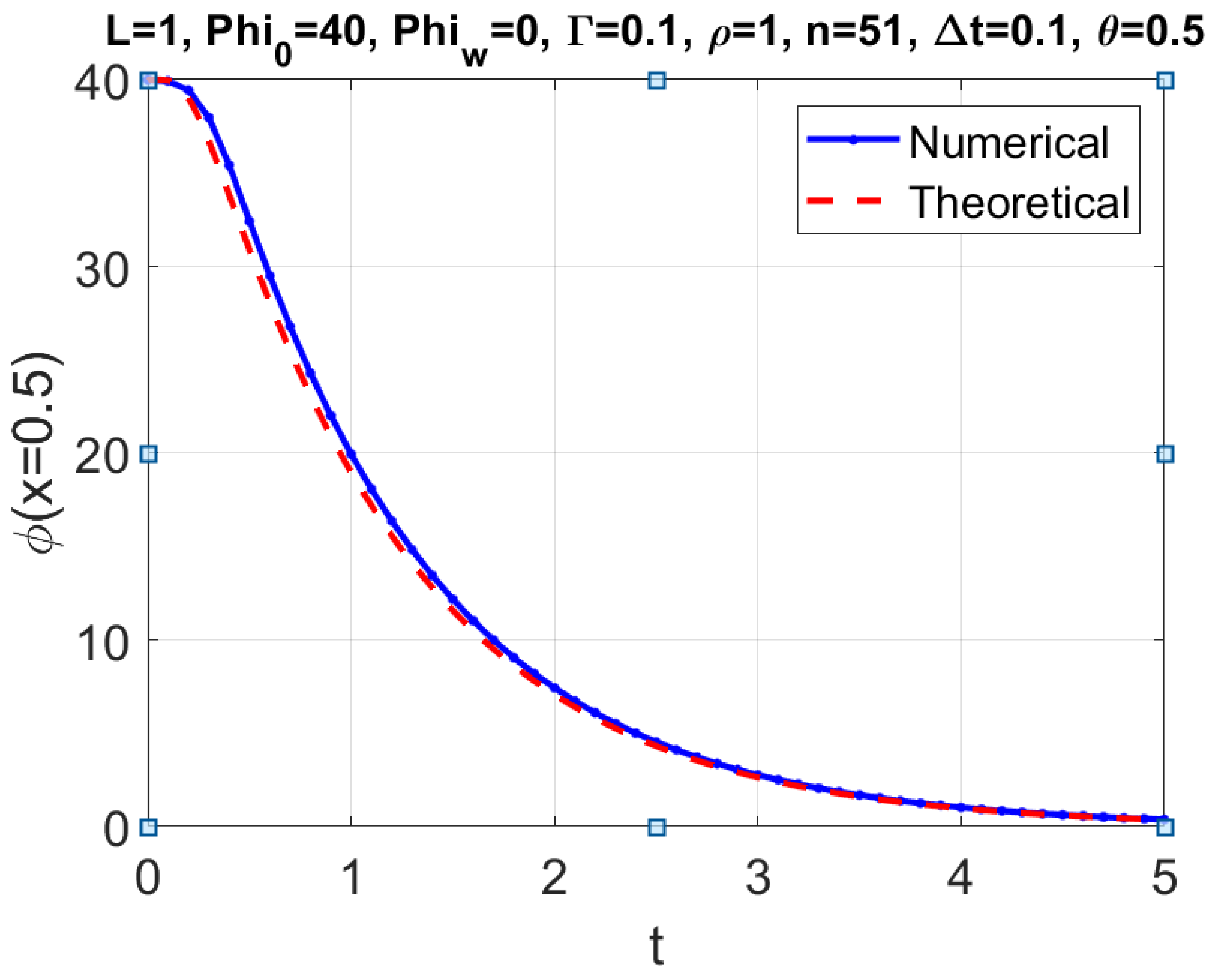
All coefficients always positive.

Unsteady 1D diffusion

- Example

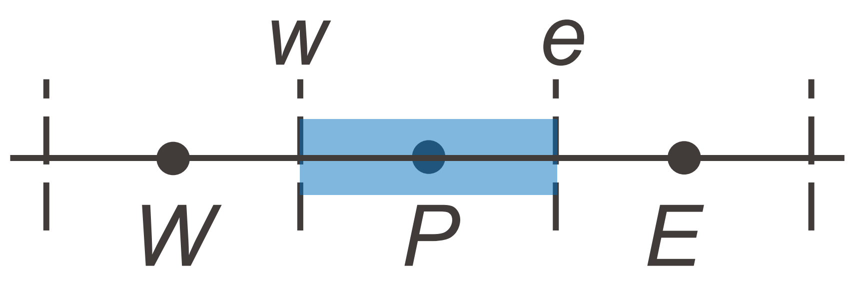
ρ=1, Γ=0.1





Unsteady 1D convection

$$\frac{\partial(\rho\phi)}{\partial t} + \frac{\partial(\rho\phi u)}{\partial x} = 0$$



- Assume the density and velocity are known.
- Integration over CV, with UD scheme for convection term ($u > 0$):

$$\rho \int_{x_w}^{x_e} \frac{\partial\phi}{\partial t} dx + (\rho\phi u)_e - (\rho\phi u)_w \approx \rho \frac{\partial\phi_P}{\partial t} \Delta x + (\rho u)_e \phi_P - (\rho u)_w \phi_W = 0$$

- Linear approximation of the time derivative, theta method for the RHS:

$$\rho \frac{\phi_P^{n+1} - \phi_P^n}{\Delta t} \Delta x + \theta \left[(\rho u)_e \phi_P^{n+1} - (\rho u)_w \phi_W^{n+1}\right] + (1-\theta)\left[(\rho u)_e \phi_P^n - (\rho u)_w \phi_W^n\right] = 0$$

- Algebraic equation for $\boldsymbol{\phi}^{n+1}$: $a_P \phi_P^{n+1} = a_W \phi_W^{n+1} + b(\boldsymbol{\phi}^n)$

- Assemble the system and march in time:

$$\mathbf{A}\boldsymbol{\phi}^{n+1} = \mathbf{b}(\boldsymbol{\phi}^n) \quad \rightarrow \quad \boldsymbol{\phi}^{n+1} = \mathbf{A}^{-1}\mathbf{b}(\boldsymbol{\phi}^n)$$

Unsteady 1D convection

- Algebraic equation: $$a_P \phi_P^{n+1} = a_W \phi_W^{n+1} + b(\boldsymbol{\phi}^n)$$

$$a_P = \left(\frac{\rho_P \Delta x}{\Delta t} + \theta \left(\rho u\right)_e \right), \quad a_W = \theta \left(\rho u\right)_w$$

$$b = \left[\frac{\rho_P \Delta x}{\Delta t} - (1-\theta) \left(\rho u\right)_e \right] \phi_P^n + (1-\theta) \left(\rho u\right)_w \phi_W^n$$

- **Explicit Euler** ($\theta = 0$):

$$a_P = \frac{\rho_P \Delta x}{\Delta t}, \quad \boxed{a_W = 0}$$

Each CV decoupled from neighbors.
No need to solve a system.

$$b = \boxed{\left[\frac{\rho_P \Delta x}{\Delta t} - \left(\rho u\right)_e \right]} \phi_P^n + \left(\rho u\right)_w \phi_W^n$$

Boundedness criterion: all coefficients should have the same sign. Here, if constant density:

$$\Delta t < \frac{\Delta x}{u}$$

Courant–Friedrichs–Lewy (CFL) condition

Unsteady 1D convection

- **Crank-Nicolson** $(\theta = 1/2)$:

$$a_P = \left(\frac{\rho_P \Delta x}{\Delta t} + \frac{1}{2}(\rho u)_e\right), \quad a_W = \frac{1}{2}(\rho u)_w$$

$$b = \boxed{\left[\frac{\rho_P \Delta x}{\Delta t} - \frac{1}{2}(\rho u)_e\right]} \phi_P^n + \frac{1}{2}(\rho u)_w \phi_W^n$$

Similar boundedness criterion: $\quad \Delta t < 2\frac{\Delta x}{u}$

- **Implicit Euler** $(\theta = 1)$:

$$a_P = \left(\frac{\rho_P \Delta x}{\Delta t} + (\rho u)_e\right), \quad a_W = (\rho u)_w$$

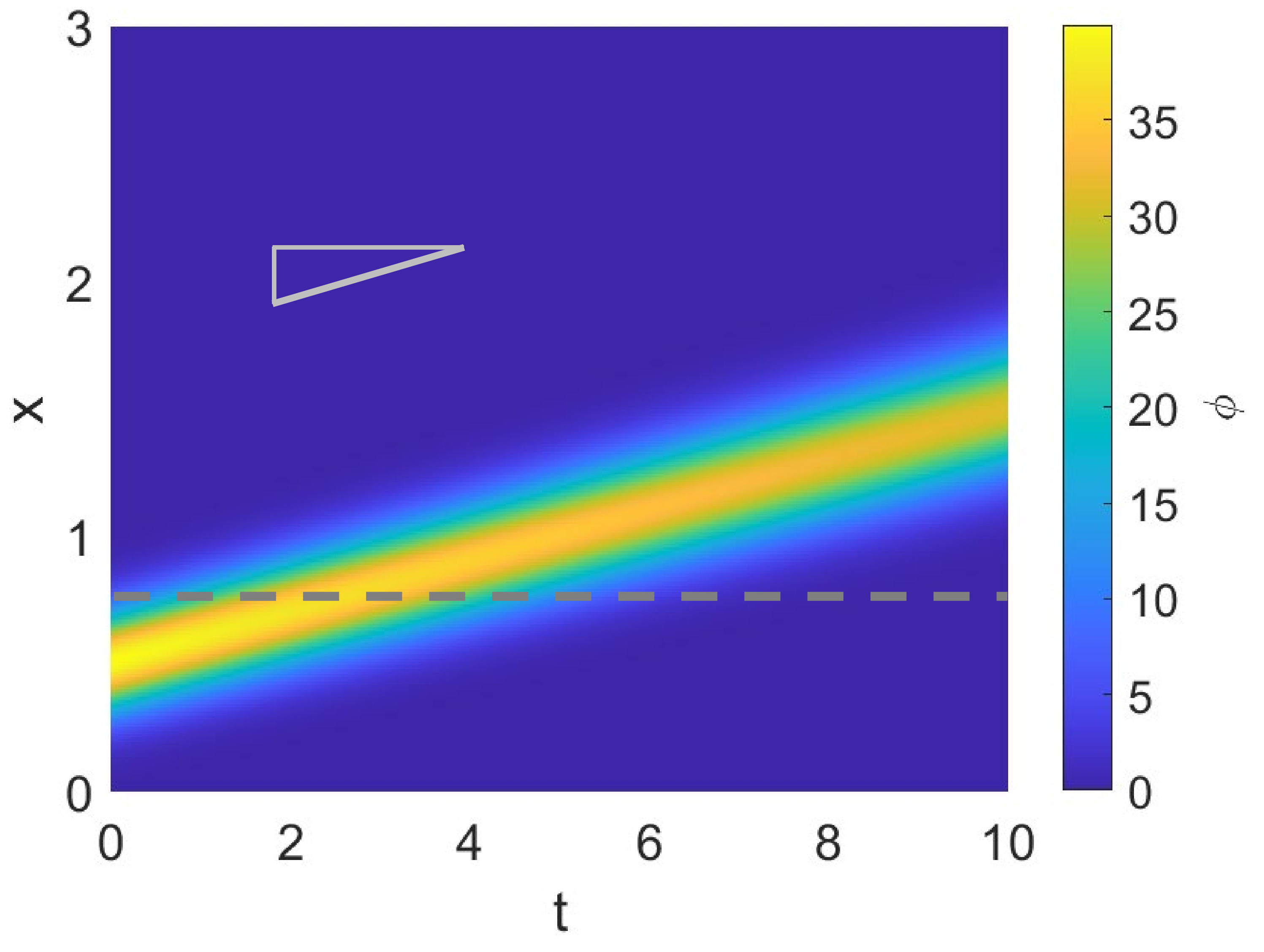
$$b = \left(\frac{\rho_P \Delta x}{\Delta t}\right) \phi_P^n$$

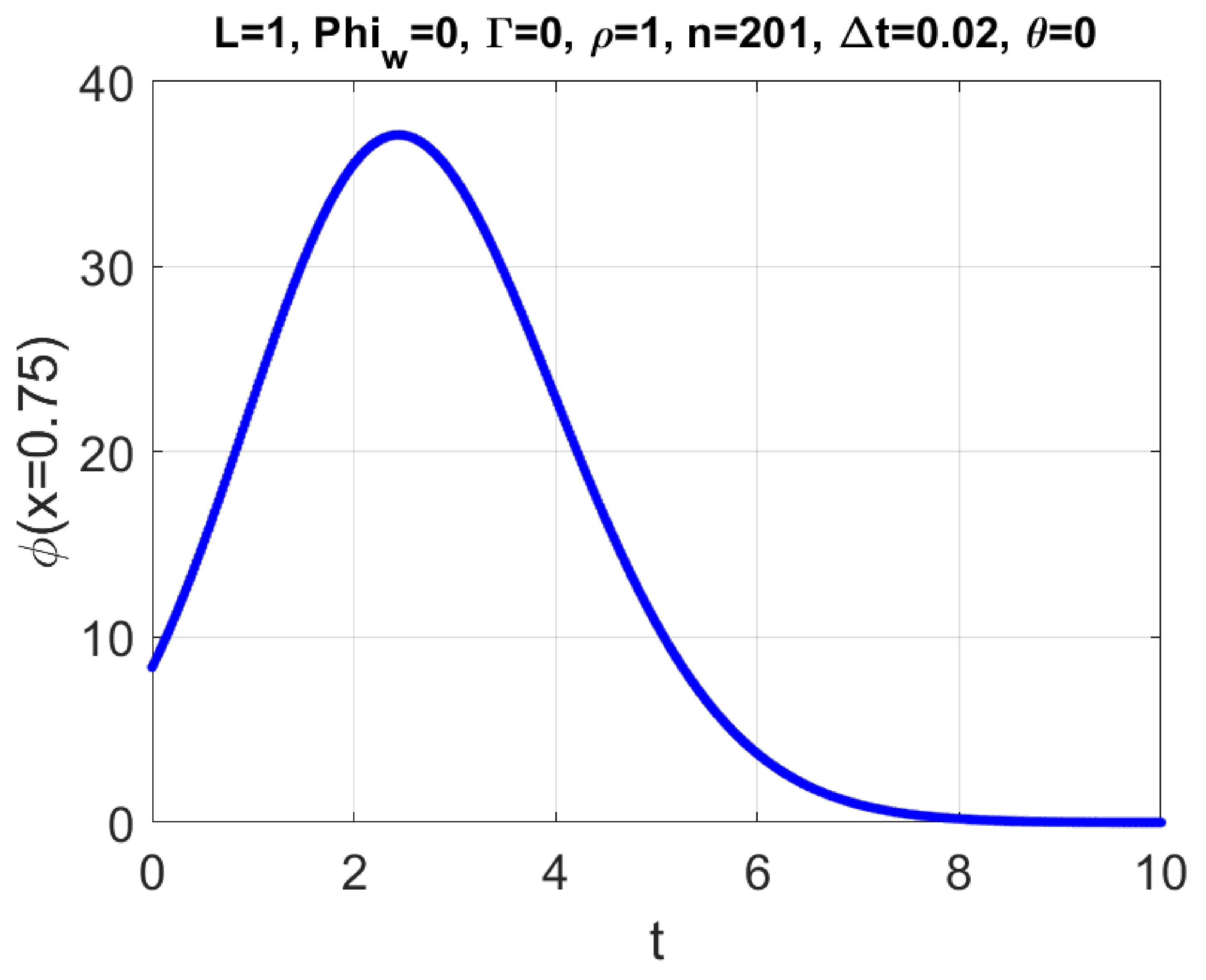
All coefficients always positive.

Unsteady 1D convection

- Example

ρ=1, u=0.1



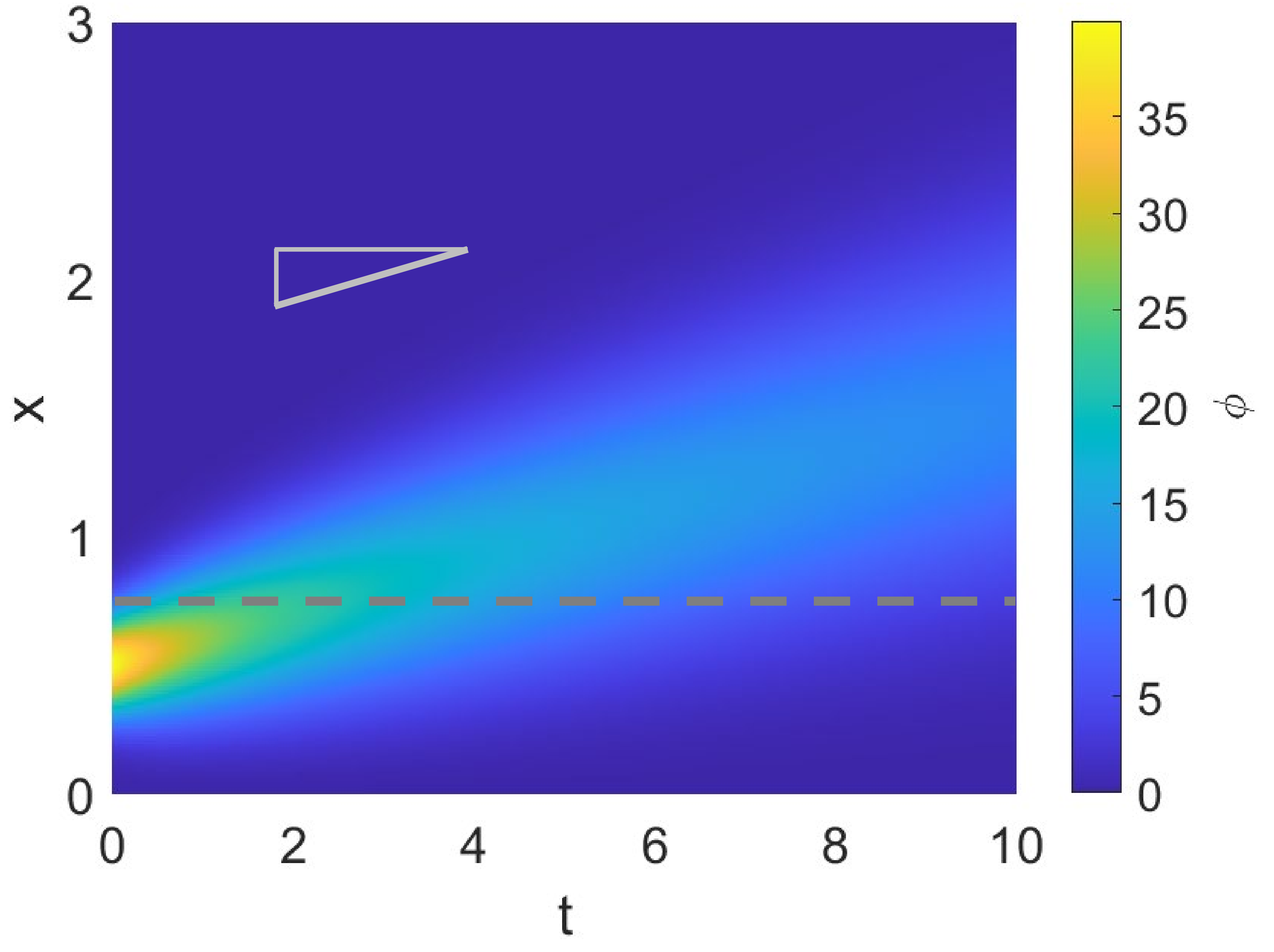


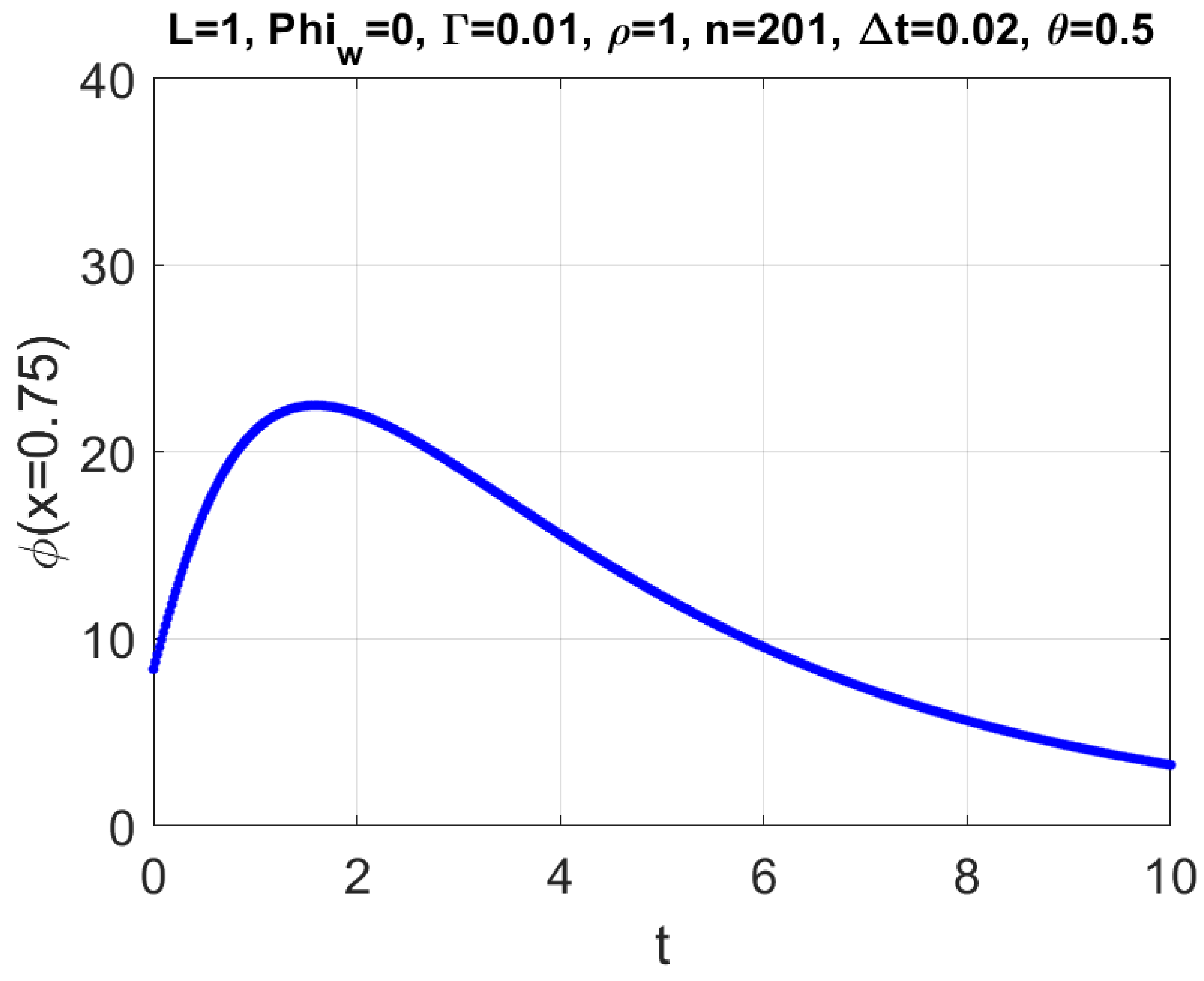
Numerical Flow Simulation

Unsteady 1D convection-diffusion

- Example

ρ=1, Γ=0.01, u=0.1





Numerical Flow Simulation

Boundedness criterion: physical interpretation

- For any partially explicit theta method ($\theta < 1$), the time step must satisfy:

for pure diffusion: $\Delta t < O\left(\dfrac{\rho(\Delta x)^2}{\Gamma}\right)$,

for pure convection: $\Delta t < O\left(\dfrac{\Delta x}{u}\right)$.

- Physically:

$$\frac{\partial(\rho\phi)}{\partial t} = \frac{\partial}{\partial x}\left(\Gamma \frac{\partial \phi}{\partial x}\right)$$

$$\sim \frac{\rho\phi}{T} \qquad \sim \frac{\Gamma\phi}{L^2}$$

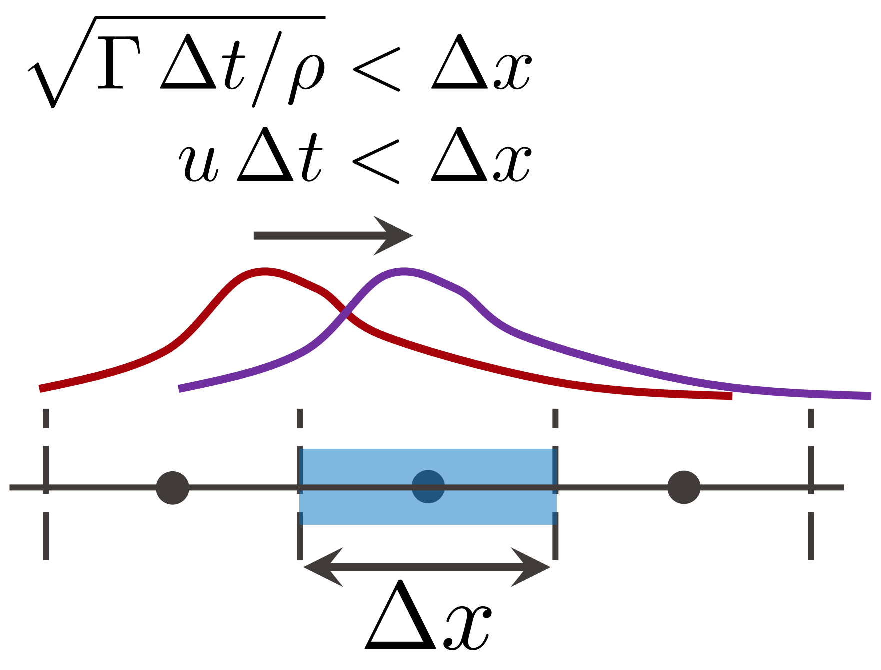
Characteristic diffusive length: $L \sim \sqrt{\dfrac{\Gamma T}{\rho}}$

$$\frac{\partial(\rho\phi)}{\partial t} + \frac{\partial(\rho\phi u)}{\partial x} = 0$$

$$\sim \frac{\rho\phi}{T} \qquad \sim \frac{\rho\phi u}{L}$$

Char. convective length: $L \sim u\,T$

- The time step Δt should be smaller than the time needed for the diffusive / convective process to travel over a distance Δx (one CV).



Boundedness criterion: physical interpretation

- For any partially explicit theta method ($\theta < 1$), the time step must satisfy:

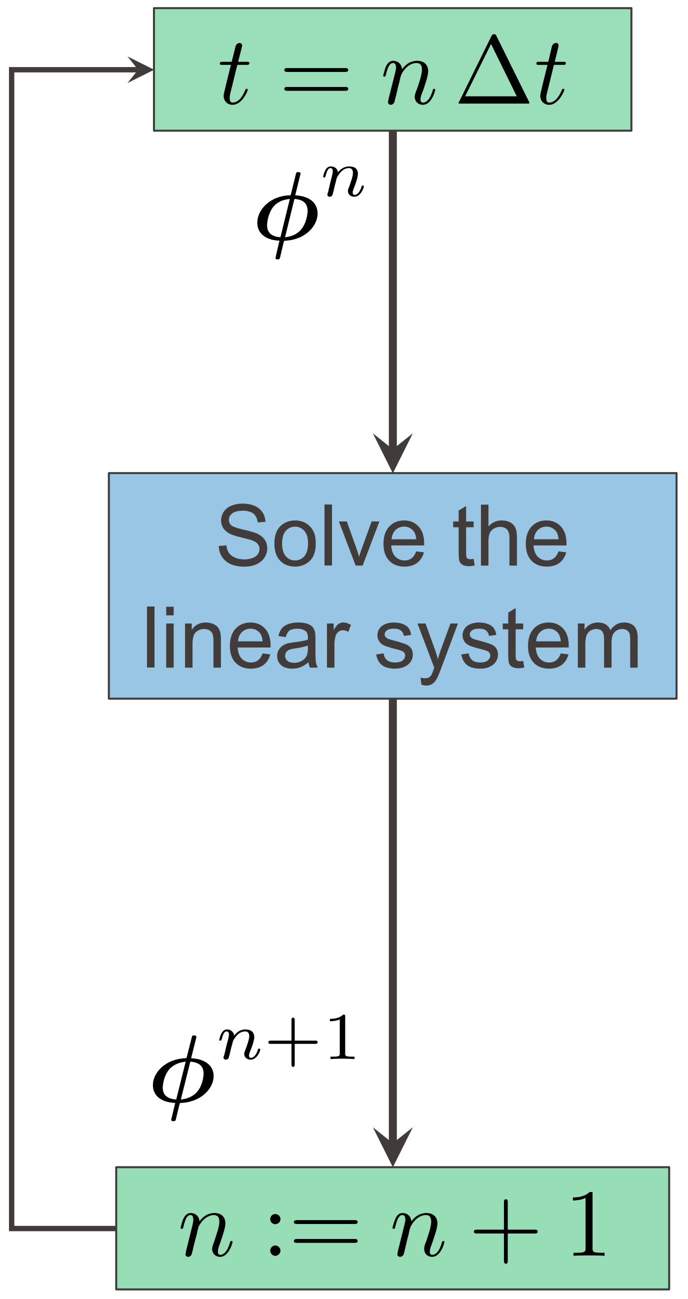
for pure diffusion: $\Delta t < O\left(\dfrac{\rho(\Delta x)^2}{\Gamma}\right)$, | for pure convection: $\Delta t < O\left(\dfrac{\Delta x}{u}\right)$.

- Note: when refining the mesh, the time step must also be reduced
→ simulations become more expensive for two reasons.

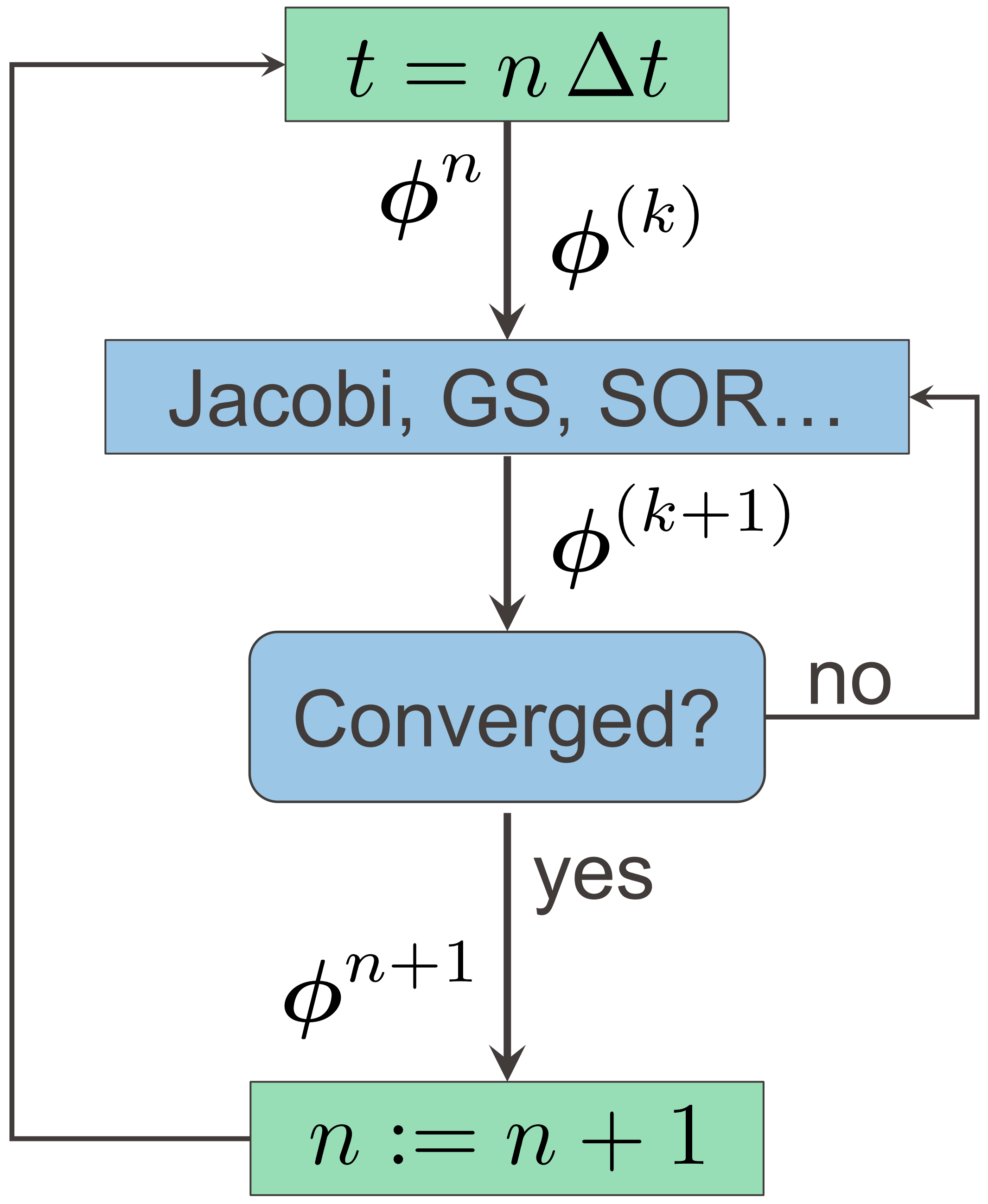
Schematic algorithms for unsteady simulation

- At **each time step**, must solve an algebraic system of equations.

Linear eq., direct method



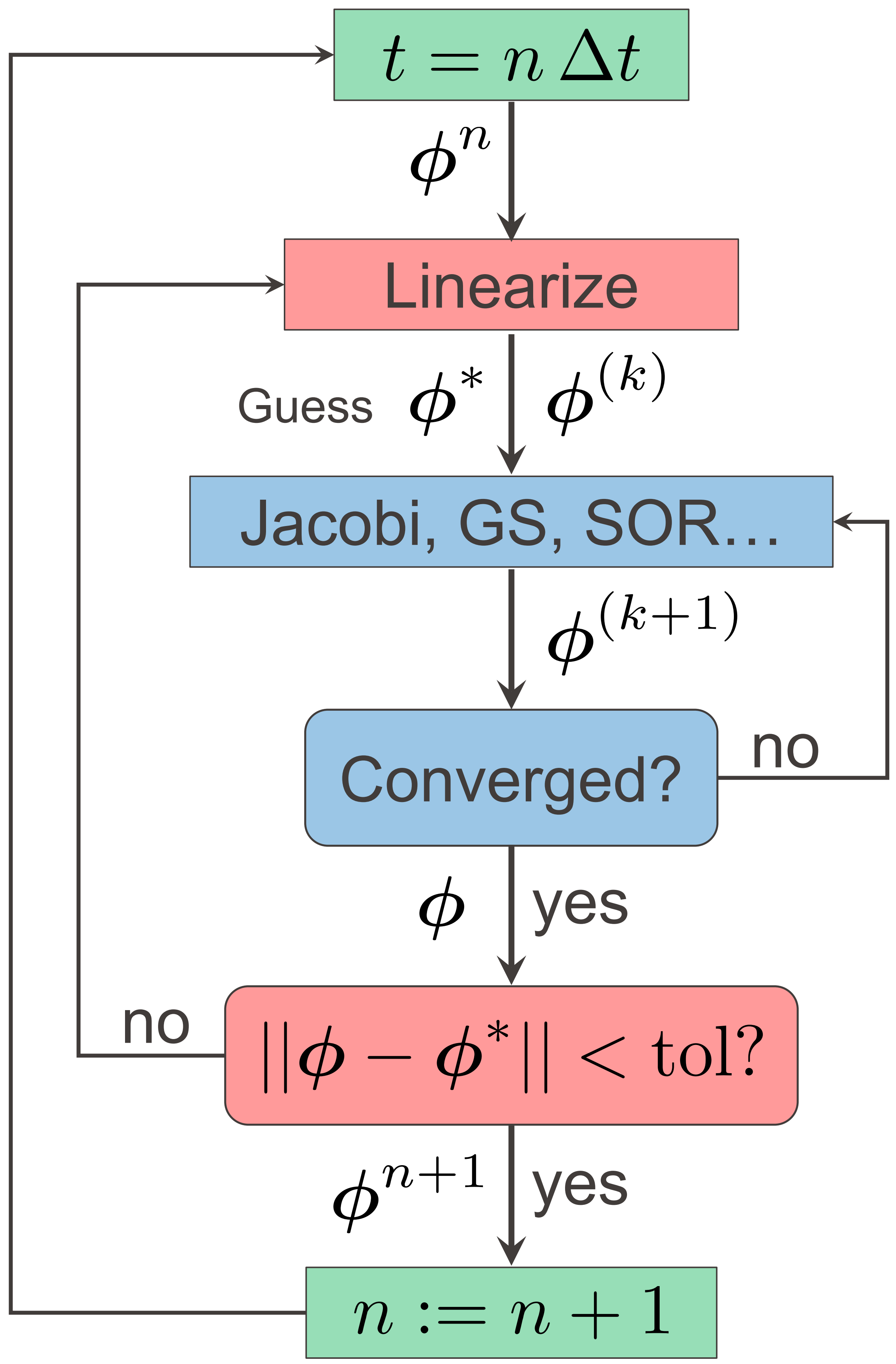
Linear eq., iterative method



Schematic algorithms for unsteady simulation

- So far (today), we have only dealt with **linear** governing equations.
  If the equations are **nonlinear**, one must linearize at **each time step.**

Nonlinear eq., iterative method



Schematic algorithms for unsteady simulation

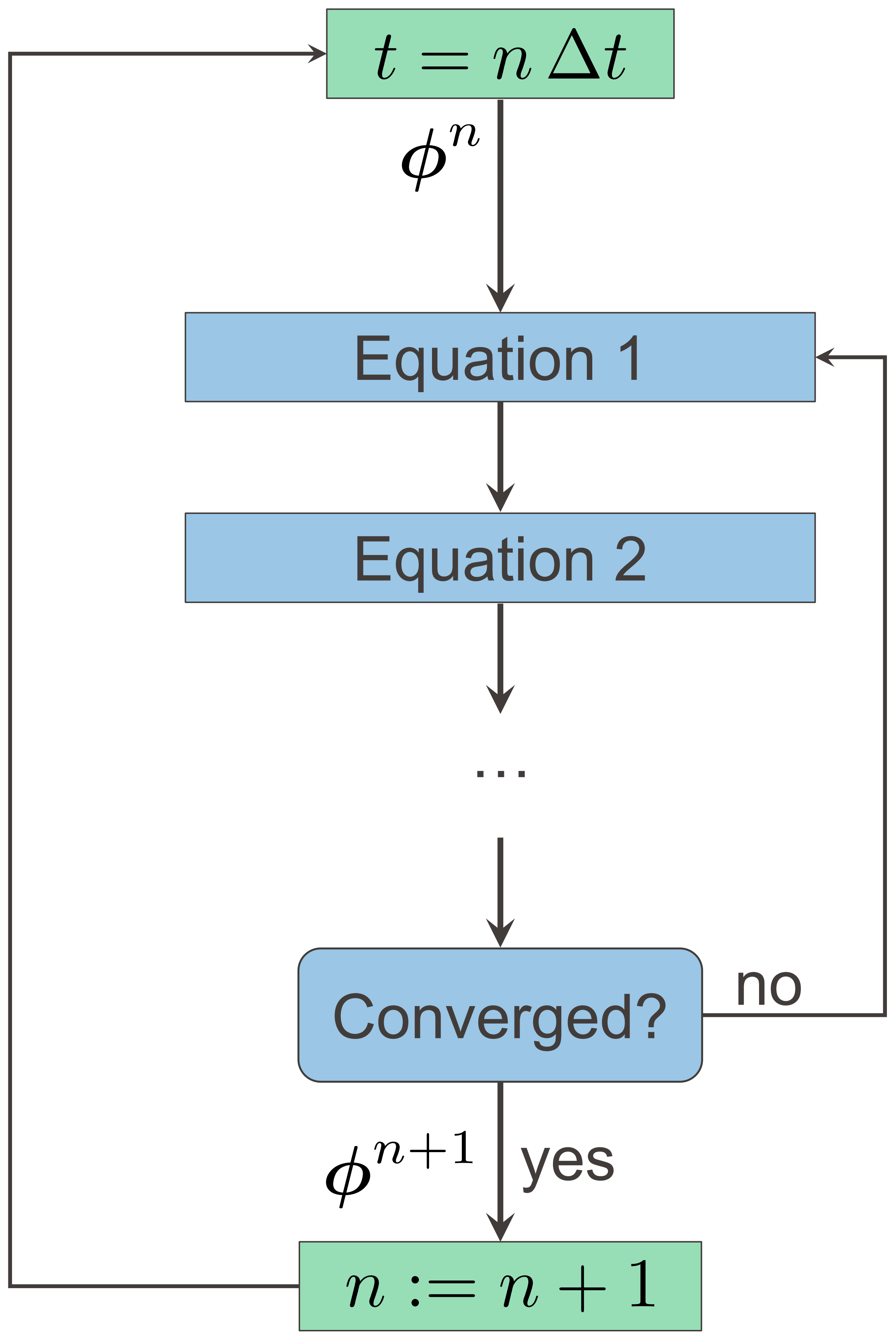
- Reminder: so far (today), we have only dealt with the solution of a **single** governing equation. If **several** equations are solved, 2 approaches (see also week 5 about the Navier-Stokes equations):

1. **Coupled** approach: all equations solved **simultaneously** → one single large system (similar to previous slides). Requires more memory.

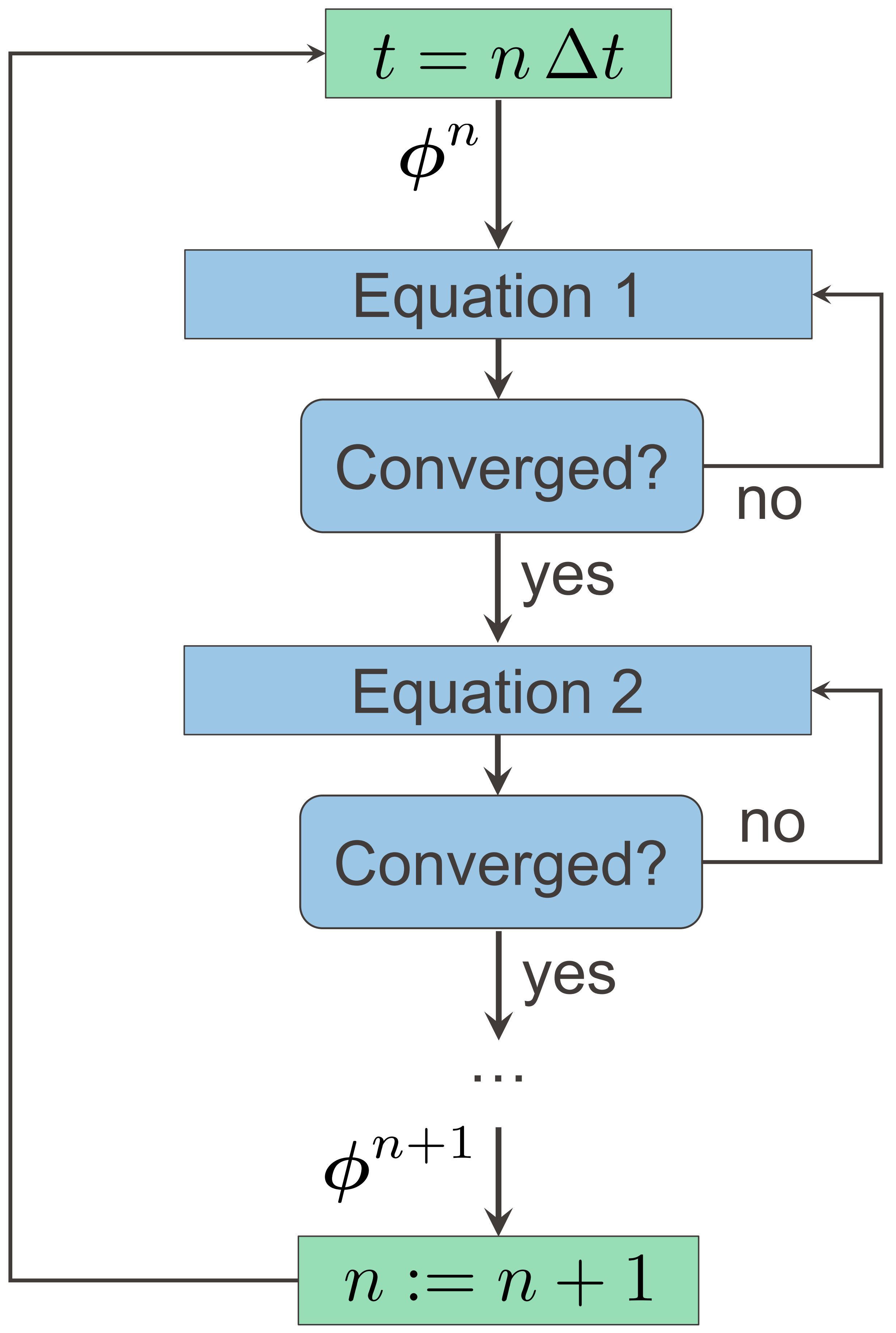
2. **Segregated** approach: equations solved **separately**. Two options:
   a) **iterative** scheme: solve each equation once, then repeat until global convergence;
   b) **non-iterative** scheme: iterate each equation until individual convergence, then proceed to next equation.

Schematic algorithms for unsteady simulation

Segregated approach, iterative scheme



Segregated approach, non-iterative scheme



Summary and guidelines

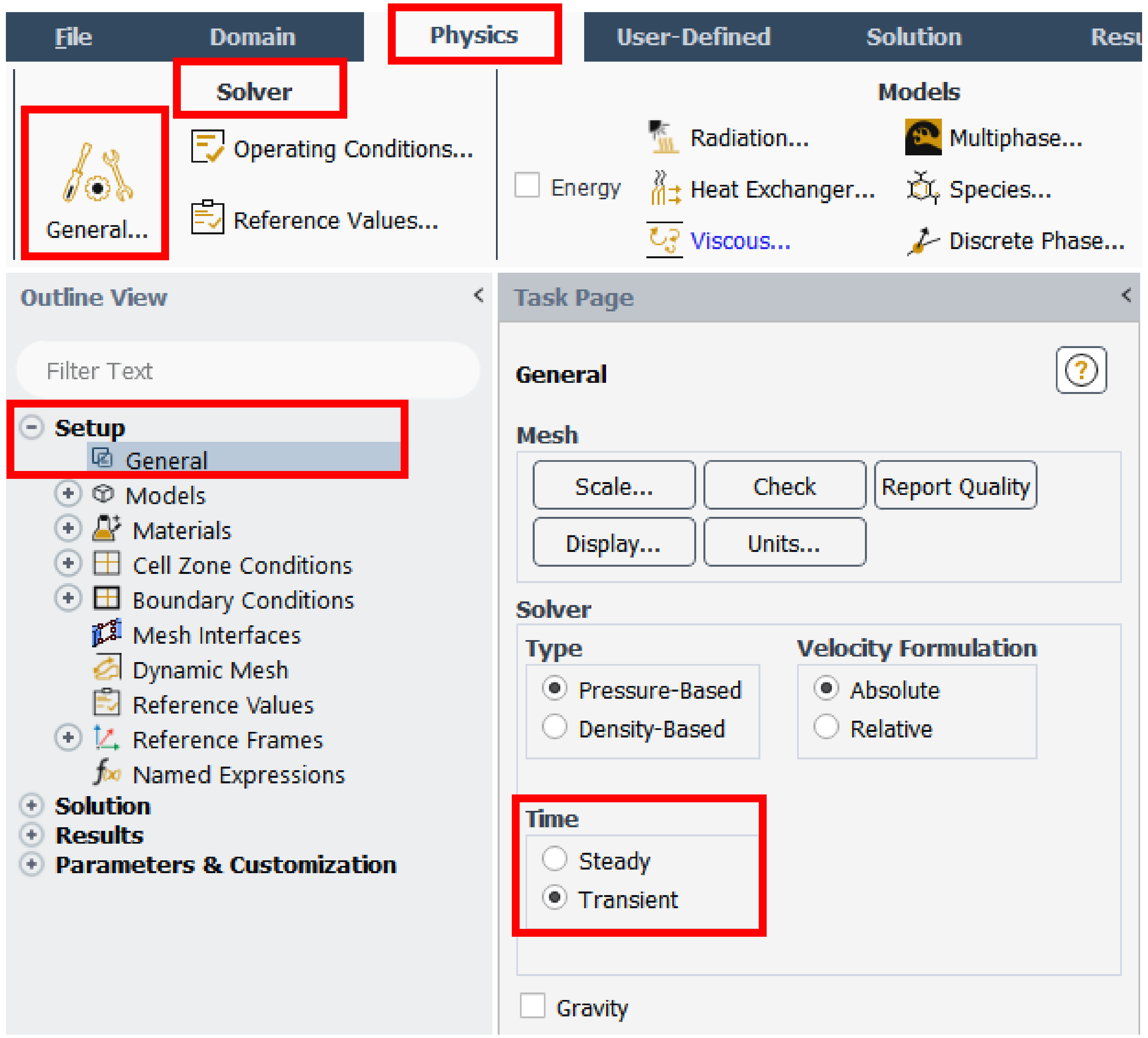
- Unsteady simulations are much more **expensive** than steady ones (→ ask yourself if they are really needed).
- They also produce **a lot of data** (→ be careful with what you store).

- **Explicit** schemes: strong stability **limitation on the time step**.
- **Implicit** schemes: unconditionally stable (ODEs), BUT the time step must be small enough to:
  - avoid unphysical oscillations (PDEs),
  - resolve time-dependent features **accurately** (diffusion, convection, buoyancy…),
  - at the very least, converge the solution at each time step (→ check the residuals).

- The **initial condition** may be very important.

Fluent specifics

- Enable transient solver:

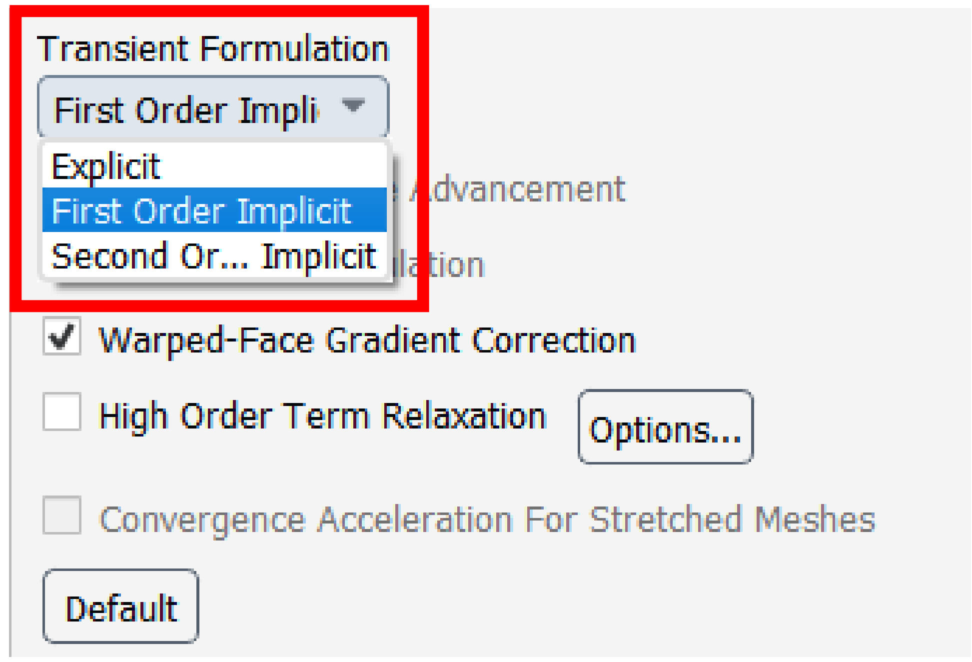


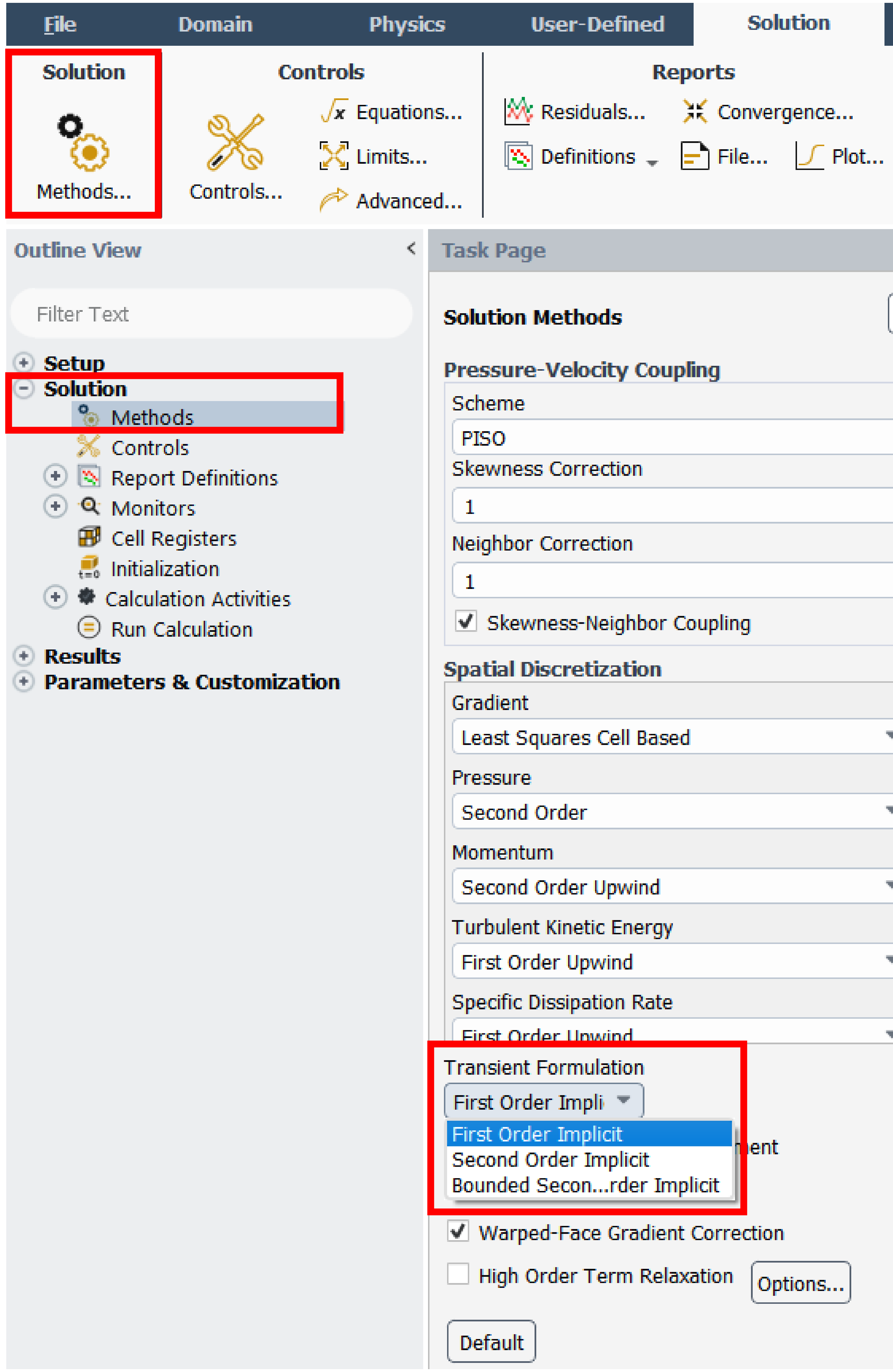
Numerical Flow Simulation

Fluent specifics

- **Pressure**-based solver: **implicit** schemes.
  - 1^{st} order (implicit Euler): more stable but less accurate,
  - 2^{nd} order (BDF2): more accurate but may produce oscillations,
  - Bounded 2^{nd} order (weighted BDF2): eliminates oscillations.

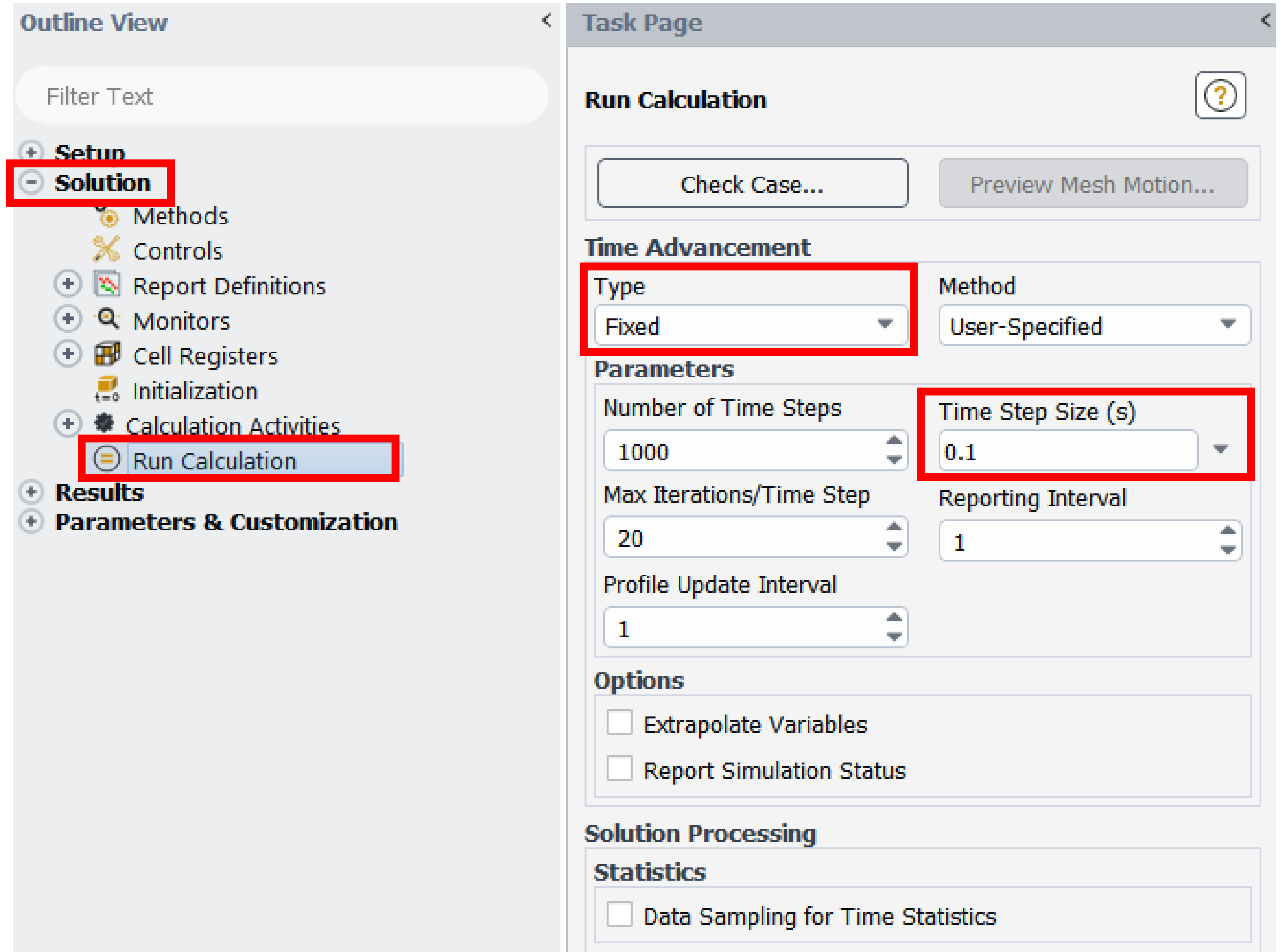
- **Density**-based solver: **implicit** (1^{st} / 2^{nd} order) and **explicit** (Runge-Kutta) schemes.

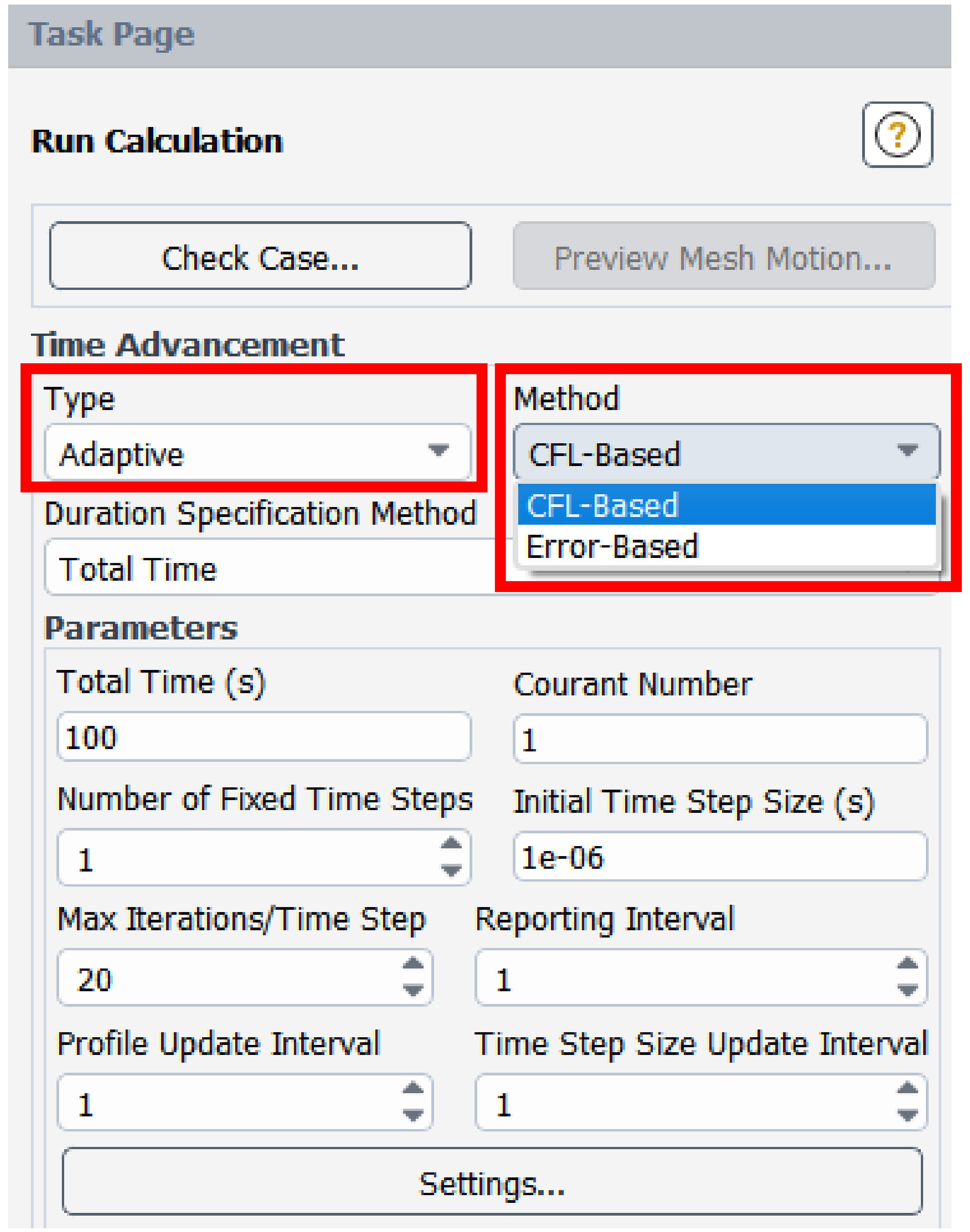




Fluent specifics

- Specify the **time step**:
  - **Fixed**: constant value
  - **Adaptive**: Fluent adjusts the time step to respect a target CFL number or a target truncation error (temporal error).

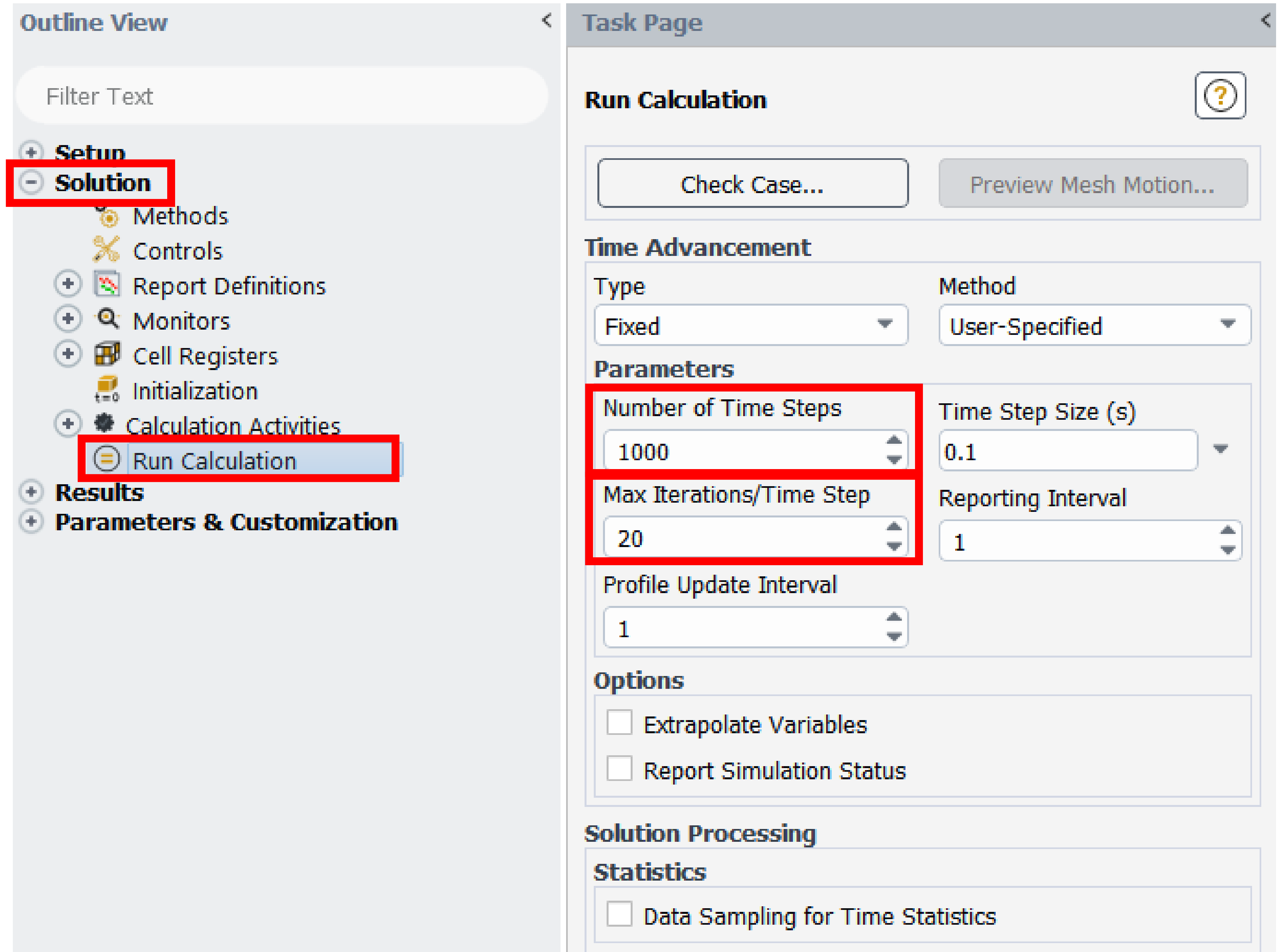


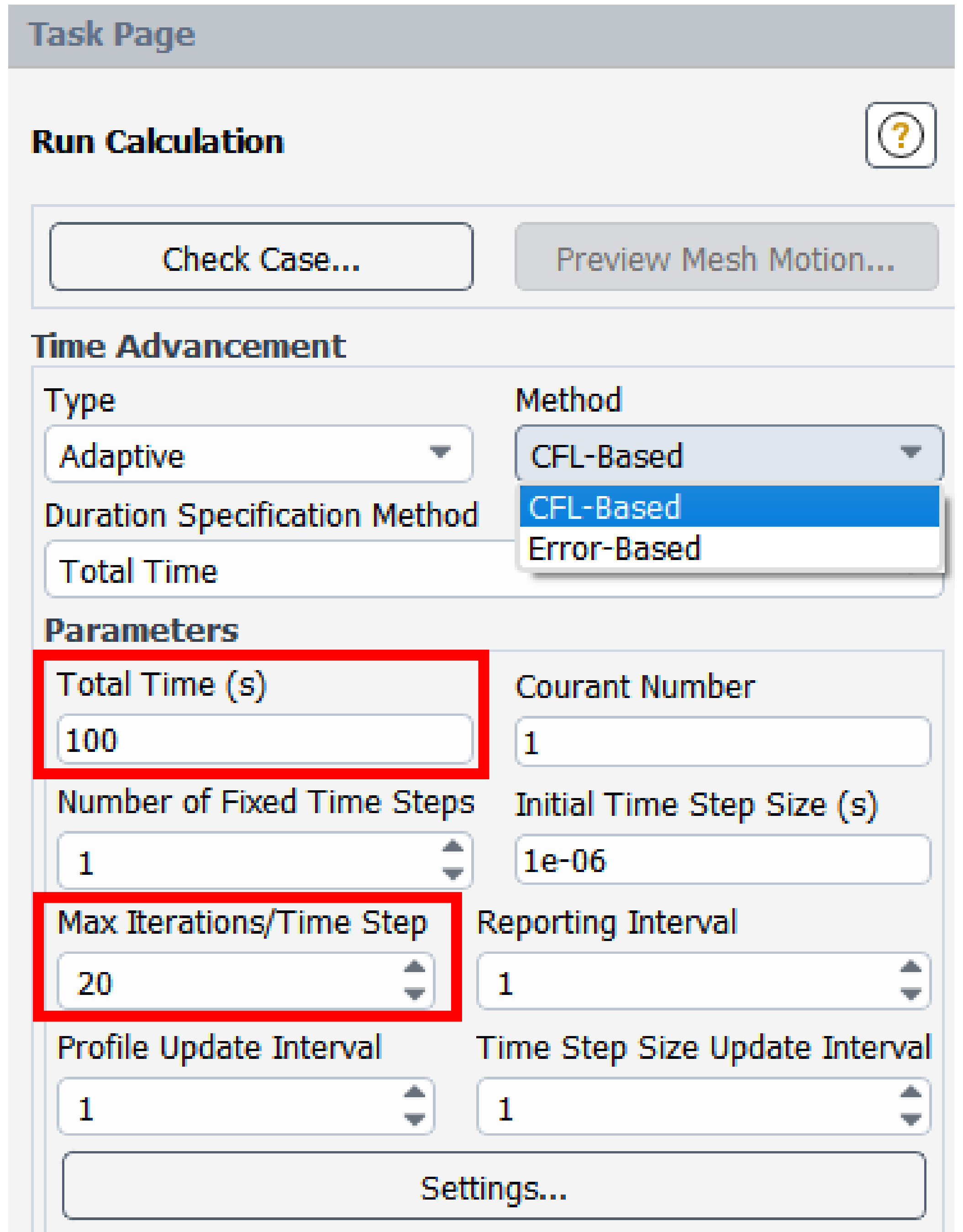


Fluent specifics

- Specify the **maximum number of iterations** allowed in each time step. Avoid too large values (better to reduce the time step).

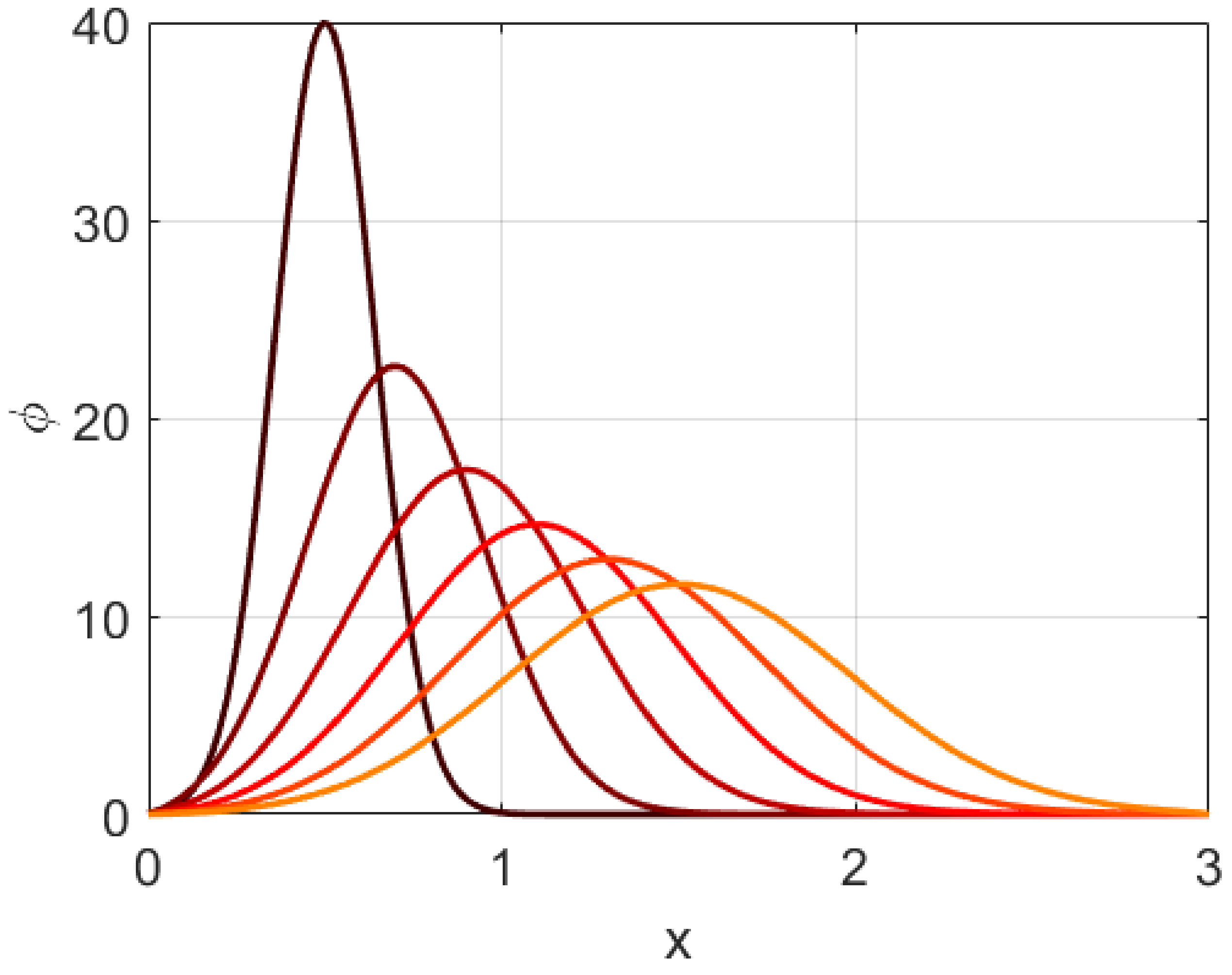
- Specify the **number of time steps**, or the **physical time** to be simulated.



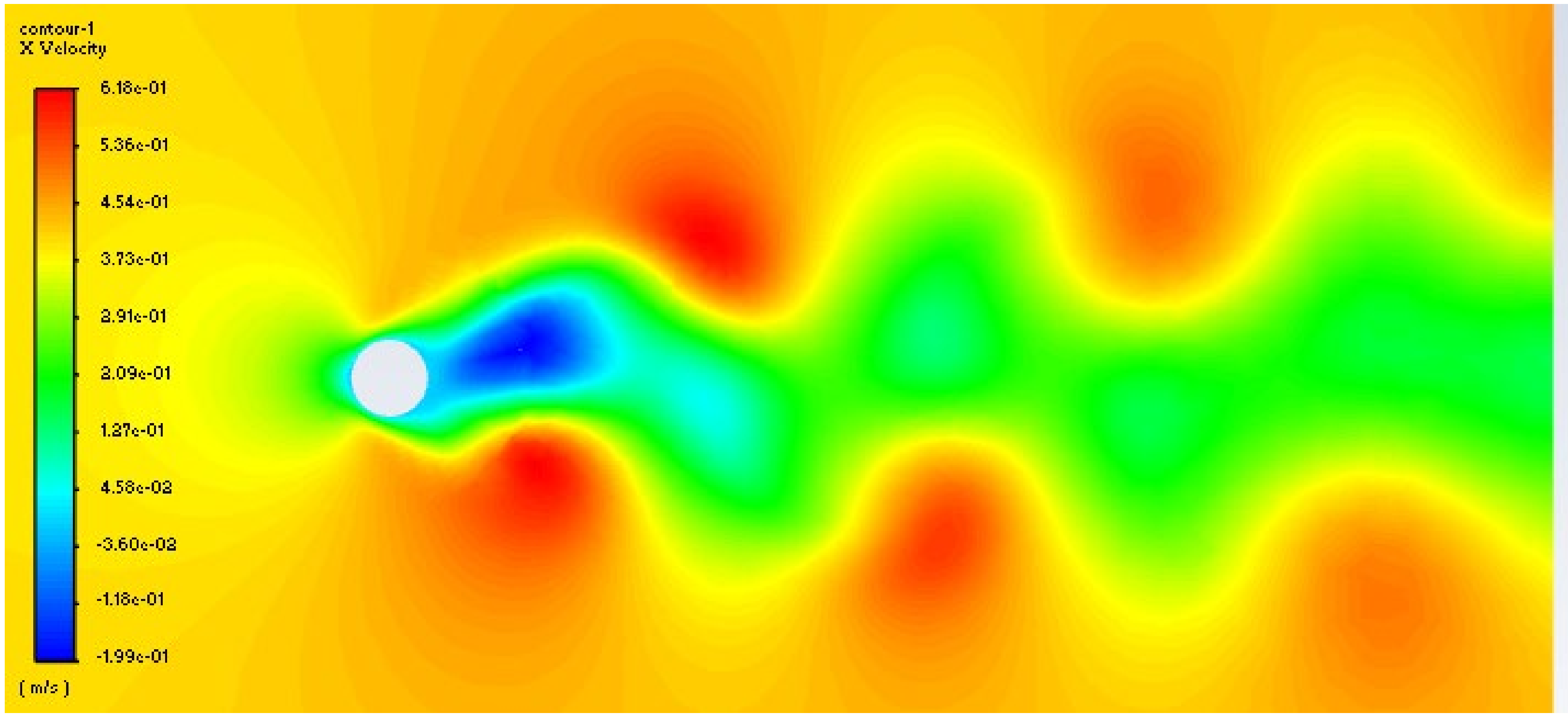


This week's exercise and tutorial

- Matlab exercise: convection-diffusion



- Fluent tutorial: vortex shedding



Numerical Flow Simulation

Appendix: pseudo-transient simulation (1/4)

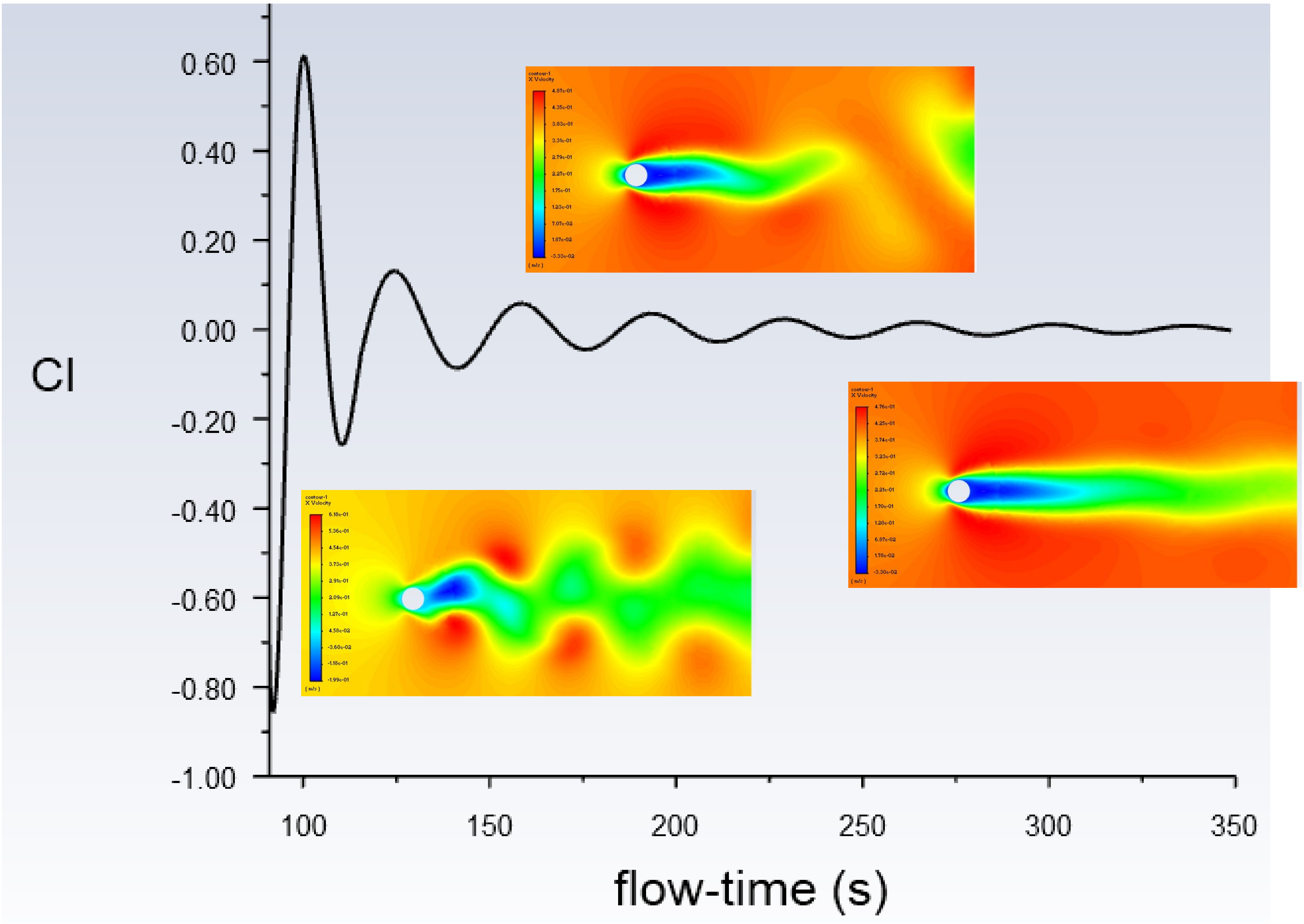
- For **steady** problems, most natural approach is to use a **steady** solver (weeks 2-5).
- In principle, one may also use an **unsteady** solver, simulate the physical process, and wait until reaching a steady state. However, this naive approach may be very slow because the flow may settle only very slowly (depending on the initial condition, the physical parameters, etc.).

Example: flow past a 2D cylinder, sudden reduction of *Re* from 100 (unsteady flow) to 40 (steady flow).

With an **unsteady** solver, need several vortex-shedding periods to reach the steady regime. Here:

- convective time (cylinder diameter divided by freestream velocity) = 1 s,
- physical time to reach steady state = $O(10^3)$ s,
- constant time step = 0.1 s

→ **$O(10^4)$ iterations.**

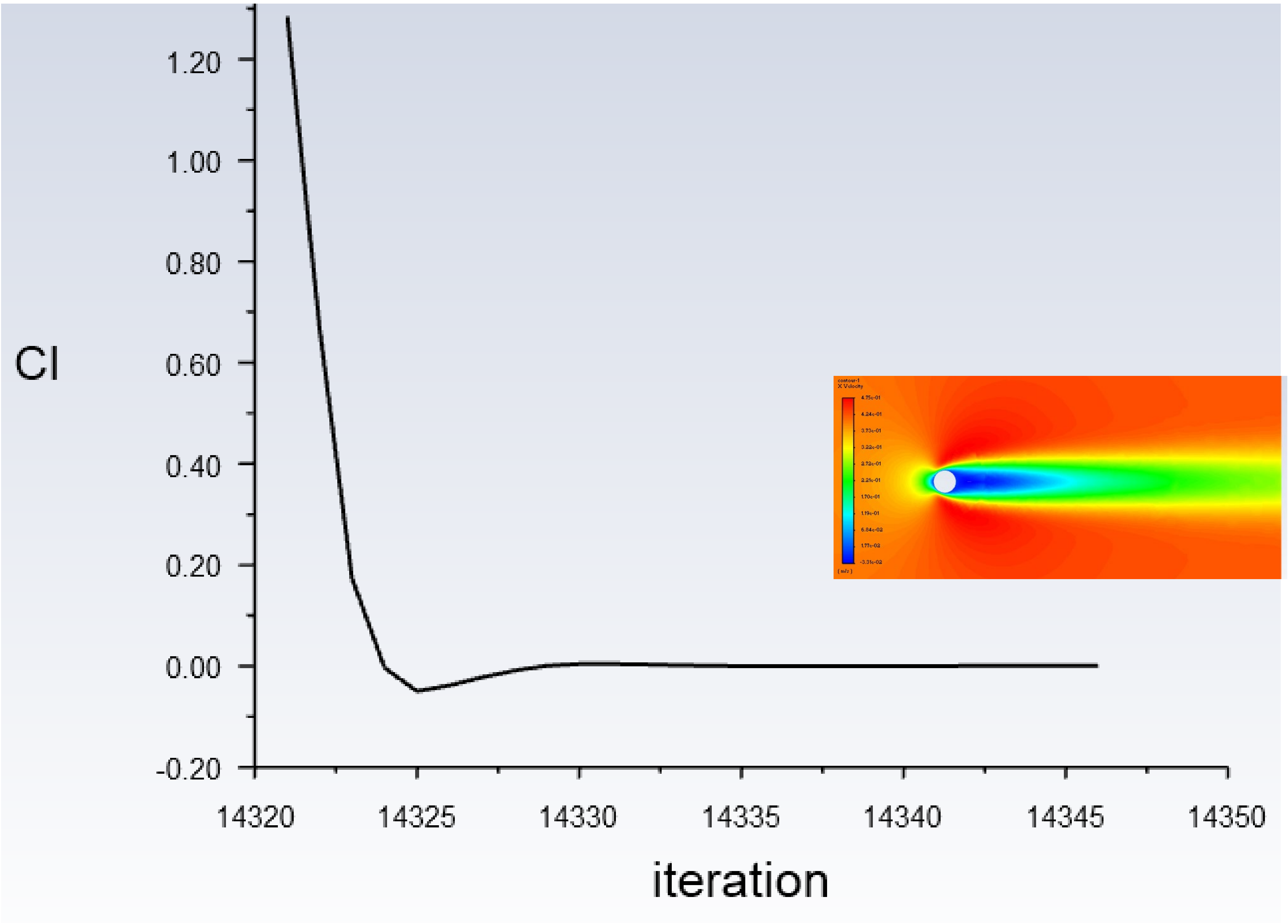


Appendix: pseudo-transient simulation (2/4)

- For **steady** problems, most natural approach is to use a **steady** solver (weeks 2-5).
- In principle, one may also use an **unsteady** solver, simulate the physical process, and wait until reaching a steady state. However, this naive approach may be very slow because the flow may settle only very slowly (depending on the initial condition, the physical parameters, etc.).

Example: flow past a 2D cylinder, sudden reduction of *Re* from 100 (unsteady flow) to 40 (steady flow).

With a **steady** solver: need only **30 iterations.**



Appendix: pseudo-transient simulation (3/4)

- One exception: so-called “pseudo-transient” simulation. An **unsteady** solver is used, but the time step does not correspond to a physical time step. Compare the following:
  - **Unsteady solver** (time integration; this week):
    $$\boxed{\left(\tilde{a}_P + \frac{\rho_P \Delta x}{\Delta t}\right)} \phi_P^{n+1} = \sum \tilde{a}_{nb} \phi_{nb}^{n+1} + \tilde{b}(\boldsymbol{\phi}^n) + \boxed{\frac{\rho_P \Delta x}{\Delta t}} \phi_P^n$$
  - **Steady solver** (iterative method, under-relaxation; week 3):
    $$\boxed{\left(\frac{\tilde{a}_P}{\alpha}\right)} \phi_P^{(k+1)} = \sum \tilde{a}_{nb} \phi_{nb}^{(k+1)} + \tilde{b}(\boldsymbol{\phi}^{(k)}) + \boxed{\left(\frac{1-\alpha}{\alpha}\right) \tilde{a}_P} \phi_P^{(k)}$$
- Clear analogy: can identify $$\frac{\rho_P \Delta x}{\Delta t} = \left(\frac{1-\alpha}{\alpha}\right) \tilde{a}_P$$
- Interpretation: pseudo-transient calculation equivalent to under-relaxed iterative steady calculation. The pseudo time step may be local (i.e. space-dependent).
- Useful in some cases, when stability problems in steady calculation.

Appendix: pseudo-transient simulation (4/4)

- **In Fluent:** available for pressure-based coupled solver, and density-based implicit solver. Can specify the pseudo time step, or leave the default automatic calculation. In both case, it is actually a global (not local) value.