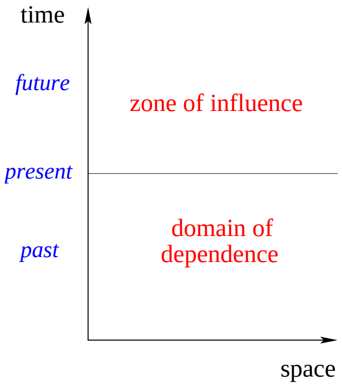
Time-stepping techniques

Unsteady flows are parabolic in time $\Rightarrow$ use 'time-stepping' methods to advance transient solutions step-by-step or to compute stationary solutions



Initial-boundary value problem $u = u(\mathbf{x}, t)$

$$\begin{cases} \frac{\partial u}{\partial t} + \mathcal{L}u = f & \text{in } \Omega \times (0,T) & \text{time-dependent PDE} \\ \mathcal{B}u = 0 & \text{on } \Gamma \times (0,T) & \text{boundary conditions} \\ u = u_0 & \text{in } \Omega \ \text{ at } t = 0 & \text{initial condition} \end{cases}$$

Time discretization $\boxed{0 = t^0 < t^1 < t^2 < \ldots < t^M = T}$ $u^0 \approx u_0$ in Ω

- Consider a short time interval (t^n, t^{n+1}), where $t^{n+1} = t^n + \Delta t$
- Given $u^n \approx u(t^n)$ use it as initial condition to compute $u^{n+1} \approx u(t^{n+1})$

Space-time discretization

Space discretization: finite differences / finite volumes / finite elements

Unknowns: $u_i(t)$ time-dependent nodal values / cell mean values

Time discretization: (i) before or (ii) after the discretization in space

The space and time variables are essentially decoupled and can be discretized independently to obtain a sequence of (nonlinear) algebraic systems

$$\boxed{A(u^{n+1}, u^n)u^{n+1} = b(u^n)} \qquad n = 0, 1, \ldots, M-1$$

Method of lines (MOL) $\mathcal{L} \to \mathcal{L}_h$ yields an ODE system for $u_i(t)$

$$\frac{du_h}{dt} + \mathcal{L}_h u_h = f_h \quad \text{on } (t^n, t^{n+1}) \qquad \text{semi-discretized equations}$$

FEM approximation $u_h(\mathbf{x}, t) = \sum_{j=1}^{N} u_j(t)\varphi_j(\mathbf{x}), \qquad u_i^n \approx u(\mathbf{x}_i, t^n)$

Galerkin method of lines

Weak formulation $\int_\Omega \left(\frac{\partial u}{\partial t} + \mathcal{L}u - f\right) v\, d\mathbf{x} = 0, \quad \forall v \in V, \quad \forall t \in (t^n, t^{n+1})$

$\frac{d}{dt}(u,v) + a(u,v) = l(v), \ \forall v \in V \quad \rightarrow \quad \frac{d}{dt}(u_h, v_h) + a(u_h, v_h) = l(v_h), \ \forall v_h \in V_h$

Differential-Algebraic Equations $\boxed{M_C \frac{du}{dt} + Au = b}$ $\quad t \in (t^n, t^{n+1})$

where $M_C = \{m_{ij}\}$ is the *mass matrix* and $u(t) = [u_1(t), \ldots, u_N(t)]^T$

Matrix coefficients $m_{ij} = (\varphi_i, \varphi_j), \quad a_{ij} = a(\varphi_i, \varphi_j), \quad b_i = l(\varphi_i)$

Mass lumping $M_C \rightarrow M_L = \mathrm{diag}\{m_i\}, \quad m_i = \sum_j m_{ij} = (\varphi_i, \sum_j \varphi_j) = \int_\Omega \varphi_i\, d\mathbf{x}$

due to the fact that $\sum_j \varphi_j \equiv 1$. In the 1D case *FDM=FVM=FEM + lumping*

Two-level time-stepping schemes

Lumped-mass discretization $\quad M_L \frac{du}{dt} + Au = b, \quad R(u,t) = M_L^{-1}[Au - b]$

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)

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

where $\Delta t = t^{n+1} - t^n$ is the time step and θ 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,	$\mathcal{O}(\Delta t)$

Fully discretized problem

Consistent-mass discretization $\quad M_C \frac{du}{dt} + Au = b, \quad R(u,t) = M_C^{-1}[Au - b]$

$$[M_C + \theta \Delta t A] u^{n+1} = [M_C - (1-\theta)\Delta t A] u^n + \Delta t\, b^{n+\theta}$$

where $\quad b^{n+\theta} = \theta b^{n+1} + (1-\theta) b^n, \quad 0 \le \theta \le 1, \quad n = 0, \ldots, M-1$

In general, time discretization is performed using numerical methods for ODEs

Initial value problem $\quad \begin{cases} \frac{du(t)}{dt} = f(t, u(t)) \\ u(t^n) = u^n \end{cases} \quad$ on $\ (t^n, t^{n+1})$

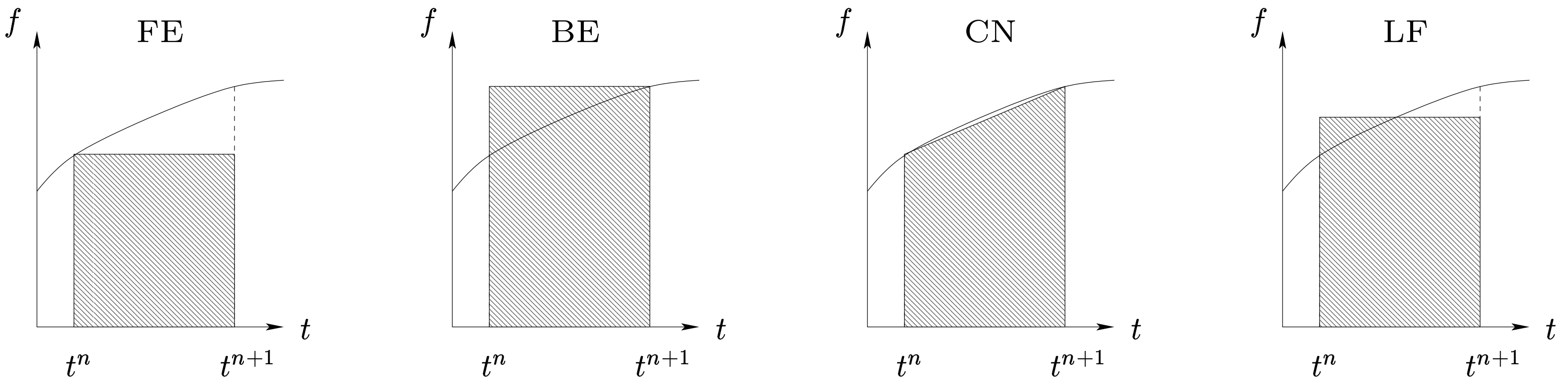
Exact integration $\quad \int_{t^n}^{t^{n+1}} \frac{du}{dt}\, dt = u^{n+1} - u^n = \int_{t^n}^{t^{n+1}} f(t, u(t))\, dt$

$u^{n+1} = u^n + f(\tau, u(\tau)) \Delta t, \quad \tau \in (t^n, t^{n+1})$ by the mean value theorem

Idea: evaluate the integral **numerically** using a suitable quadrature rule

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

Properties of time-stepping schemes

Time discretization $t^n = n\Delta t, \quad \Delta t = \frac{T}{M} \quad \Rightarrow \quad M = \frac{T}{\Delta t}$

Accumulation of truncation errors $n = 0, \ldots, M-1$

$$\epsilon_\tau^{\text{loc}} = \mathcal{O}(\Delta t)^p \qquad \Rightarrow \qquad \epsilon_\tau^{\text{glob}} = M\epsilon_\tau^{\text{loc}} = \mathcal{O}(\Delta t)^{p-1}$$

Remark. The order of a time-stepping method (i.e., the asymptotic rate at which the error is reduced as $\Delta t \to 0$) is not the sole indicator of accuracy

The optimal choice of the time-stepping scheme depends on its purpose:

- to obtain a time-accurate discretization of a highly dynamic flow problem (evolution details are essential and must be captured) or
- to march the numerical solution to a steady state starting with some reasonable initial guess (intermediate results are immaterial)

The computational cost of explicit and implicit schemes differs considerably

Explicit vs. implicit time discretization

Pros and cons of explicit schemes

- $\oplus$ easy to implement and parallelize, low cost per time step
- $\oplus$ a good starting point for the development of CFD software
- $\ominus$ small time steps are required for stability reasons, especially if the velocity and/or mesh size are varying strongly
- $\ominus$ 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

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

Example: 1D convection-diffusion equation

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

$$f(t, u(t)) = -v\frac{\partial u}{\partial x} + d\frac{\partial^2 u}{\partial x^2}$$

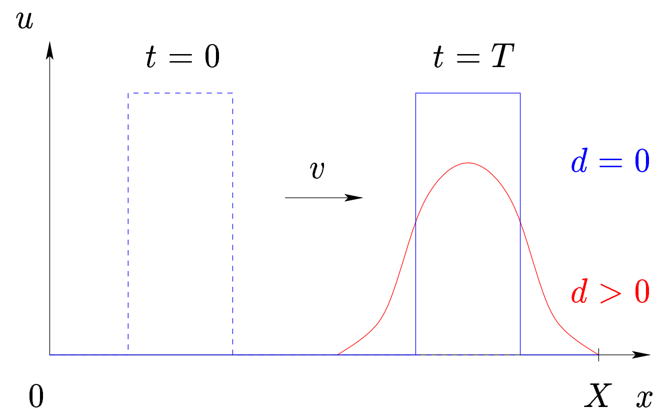
Lagrangian representation $\qquad \dfrac{du(x(t),t)}{dt} = d\dfrac{\partial^2 u}{\partial x^2} \qquad$ pure diffusion equation

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



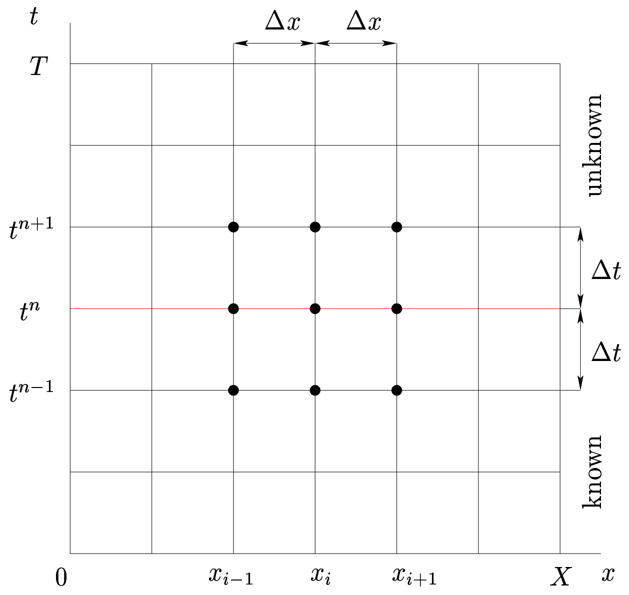
Example: 1D convection-diffusion equation

Uniform space-time mesh

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

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

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



Fully discretized equation $\qquad 0 \le \theta \le 1$

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

Central difference / lumped-mass FEM

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

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

Example: 1D convection-diffusion equation

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

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

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

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

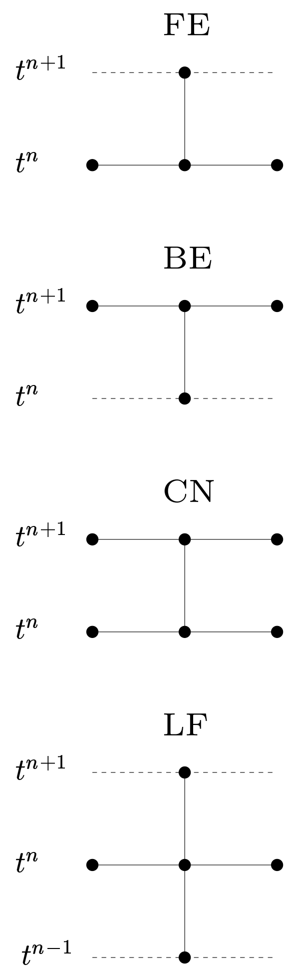
Backward Euler $(\theta=1)$ $\quad u_i^{n+1}=h(u_{i-1}^{n+1},u_i^n,u_{i+1}^{n+1})$

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

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

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

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



Fractional-step $\theta-$scheme

Given the parameters $\theta \in (0,1)$, $\theta' = 1 - 2\theta$, and $\alpha \in [0,1]$ subdivide the time interval (t^n, t^{n+1}) into three substeps and update the solution as follows

Step 1. $u^{n+\theta} = u^n + [\alpha f(t^{n+\theta}, u^{n+\theta}) + (1-\alpha) f(t^n, u^n)]\theta\Delta t$

Step 2. $u^{n+1-\theta} = u^{n+\theta} + [(1-\alpha) f(t^{n+1-\theta}, u^{n+1-\theta}) + \alpha f(t^{n+\theta}, u^{n+\theta})]\theta'\Delta t$

Step 3. $u^{n+1} = u^{n+1-\theta} + [\alpha f(t^{n+1}, u^{n+1}) + (1-\alpha) f(t^{n+1-\theta}, u^{n+1-\theta})]\theta\Delta t$

Properties of this time-stepping method

- second-order accurate in the special case $\theta = 1 - \frac{\sqrt{2}}{2}$
- coefficient matrices are the same for all substeps if $\alpha = \frac{1-2\theta}{1-\theta}$
- combines the advantages of Crank-Nicolson and backward Euler

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 Δt 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

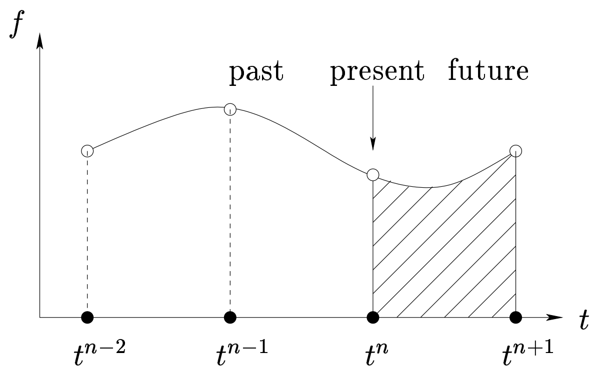
Adams methods	$t^{n+1}, \ldots, t^{n-m}, \quad m = 0, 1, \ldots$
Runge-Kutta methods	$t^{n+\alpha} \in [t^n, t^{n+1}], \quad \alpha \in [0, 1]$

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-Bashforth methods (explicit)

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

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

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

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

Runge-Kutta methods

Multipredictor-multicorrector algorithms of order p

$p=2$	$\tilde{u}^{n+1/2} = u^n + \frac{\Delta t}{2} f(t^n, u^n)$	forward Euler / predictor
	$u^{n+1} = u^n + \Delta t f(t^{n+1/2}, \tilde{u}^{n+1/2})$	midpoint rule / corrector
$p=4$	$\tilde{u}^{n+1/2} = u^n + \frac{\Delta t}{2} f(t^n, u^n)$	forward Euler / predictor
	$\hat{u}^{n+1/2} = u^n + \frac{\Delta t}{2} f(t^{n+1/2}, \tilde{u}^{n+1/2})$	backward Euler / corrector
	$\tilde{u}^{n+1} = u^n + \Delta t f(t^{n+1/2}, \hat{u}^{n+1/2})$	midpoint rule / predictor
	$u^{n+1} = u^n + \frac{\Delta t}{6}[f(t^n, u^n) + 2f(t^{n+1/2}, \tilde{u}^{n+1/2})$	Simpson rule
	$+ 2f(t^{n+1/2}, \hat{u}^{n+1/2}) + f(t^{n+1}, \tilde{u}^{n+1})]$	corrector

Remark. There exist ‘embedded’ Runge-Kutta methods which perform extra steps in order to estimate the error and adjust Δt in an adaptive fashion

General comments

Pros and cons of Runge-Kutta methods

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

Adaptive time-stepping strategy Δt Δt Δt Δt Δt Δt Δt Δt

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*

Automatic time step control

Objective: make sure that $||u - u_{\Delta t}|| \approx TOL$ (prescribed tolerance)

Local truncation error

1. $u_{\Delta t} = u + \Delta t^2 e(u) + \mathcal{O}(\Delta t)^4$
2. $u_{m\Delta t} = u + m^2 \Delta t^2 e(u) + \mathcal{O}(\Delta t)^4$

Heuristic error analysis

$$e(u) \approx \frac{u_{m\Delta t} - u_{\Delta t}}{\Delta t^2 (m^2 - 1)}$$

Remark. It is tacitly assumed that the error at $t = t^n$ is equal to zero.

Estimate of the relative error

$$||u - u_{\Delta t_*}|| \approx \left(\frac{\Delta t_*}{\Delta t}\right)^2 \frac{||u_{\Delta t} - u_{m\Delta t}||}{m^2 - 1} = TOL$$

Adaptive time stepping

$$\Delta t_*^2 = TOL \frac{\Delta t^2 (m^2 - 1)}{||u_{\Delta t} - u_{m\Delta t}||}$$

Richardson extrapolation: eliminate the leading term

$$u = \frac{m^2 u_{\Delta t} - u_{m\Delta t}}{m^2 - 1} + \mathcal{O}(\Delta t)^4 \qquad \text{fourth-order accurate}$$

Practical implementation

Automatic time step control can be executed as follows

Given the old solution u^n do:

1. Make one large time step of size $m\Delta t$ to compute $u_{m\Delta t}$
2. Make m small substeps of size Δt to compute $u_{\Delta t}$
3. Evaluate the relative solution changes $||u_{\Delta t} - u_{m\Delta t}||$
4. Calculate the 'optimal' value Δt_* for the next time step
5. If $\Delta t_* \ll \Delta t$, reset the solution and go back to step 1
6. Set $u^{n+1} = u_{\Delta t}$ or perform Richardson extrapolation

⊖ Note that the cost per time step increases substantially ($u_{m\Delta t}$ may be as expensive to obtain as $u_{\Delta t}$ due to slow convergence at large time steps).

⊕ On the other hand, adaptive time-stepping enhances the robustness of the code, the overall efficiency and the credibility of simulation results.

Evolutionary PID controller

Another simple mechanism for capturing the dynamics of the flow:

- Monitor the relative changes $e_n = \frac{||u^{n+1}-u^n||}{||u^{n+1}||}$ of an indicator variable u
- If $e_n > \epsilon$ reject the solution and repeat the time step using $\Delta t_* = \frac{\epsilon}{e_n}\Delta t_n$
- Adjust the time step smoothly so as to approach the prescribed tolerance

$$\Delta t_{n+1} = \left(\frac{e_{n-1}}{e_n}\right)^{k_P} \left(\frac{TOL}{e_n}\right)^{k_I} \left(\frac{e_{n-1}^2}{e_n e_{n-2}}\right)^{k_D} \Delta t_n$$

- Limit the growth and reduction of the time step so that

$$\Delta t_{\min} \le \Delta t_{n+1} \le \Delta t_{\max}, \qquad l \le \frac{\Delta t_{n+1}}{\Delta t_n} \le L$$

Empirical PID parameters $k_P = 0.075, \quad k_I = 0.175, \quad k_D = 0.01$

Pseudo time-stepping

Solutions to boundary value problems of the form $\mathcal{L}u = f$ represent the steady-state limit of the associated time-dependent problem

$$\frac{\partial u}{\partial t} + \mathcal{L}u = f, \qquad u(\mathbf{x}, 0) = u_0(\mathbf{x}) \quad \text{in } \Omega$$

where u_0 is an 'arbitrary' initial condition. Therefore, the numerical solution can be 'marched' to the steady state using a *pseudo time-stepping* technique.

- can be interpreted as an iterative solver for the stationary problem
- the artificial time step represents an adjustable relaxation parameter
- evolution details are immaterial $\Rightarrow$ Δt should be as large as possible
- the unconditionally stable backward Euler method is to be recommended
- explicit schemes can be used in conjunction with local time-stepping
- it is worthwhile to perform an adaptive time step control (e.g., PID)