Modélisation et Simulation des Plasmas

G. Fuhr
Guillaume.Fuhr@univ.amu.fr

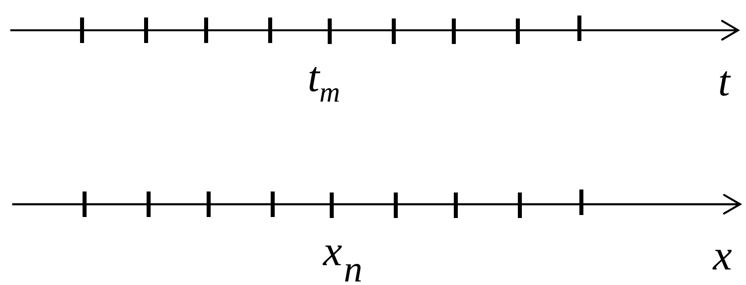
From Mathematical Models to Numerical Simulations

Continuum Model

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

Sommerfeld Wave Equation (c= wave speed). This radiation condition is sometimes used at open boundaries of ocean models.

Discrete Model



$$t_m = t_0 + m\Delta t\ , \quad m = 0, 1, \ldots M-1$$

$$x_n = x_0 + n\,\Delta x\ , \quad n = 0, 1, \ldots N-1$$

$$\frac{dw}{dx} \simeq \frac{\Delta w}{\Delta x}\ , \quad \frac{dw}{dt} \simeq \frac{\Delta w}{\Delta t}$$

p parameters

Differential Equation

$$L(p, w, x, t) = 0$$

"Differentiation"
"Integration"

Difference Equation

$$L_{mn}(p_{mn}, w_{mn}, x_n, t_m) = 0$$

System of Equations

$$\sum_{j=0}^{N-1} F_i(w_j) = B_i$$

Linear System of Equations

$$\sum_{j=0}^{N-1} A_{ij} w_j = B_i$$

"Solving linear equations"

Eigenvalue Problems

$$\overline{\overline{\mathbf{A}}}\mathbf{u} = \lambda\mathbf{u} \Leftrightarrow (\overline{\overline{\mathbf{A}}} - \lambda\overline{\overline{\mathbf{I}}})\mathbf{u} = \mathbf{0}$$

Non-trivial Solutions

$$\det(\overline{\overline{\mathbf{A}}} - \lambda\overline{\overline{\mathbf{I}}}) = 0$$

"Root finding"

Consistency/Accuracy and Stability => Convergence
(Lax equivalence theorem for well-posed linear problems)

Classification of Partial Differential Equations

(2D case, 2nd order)

Meaning of Hyperbolic, Parabolic and Elliptic

- The general 2nd order PDE in 2D:

$$A\phi_{xx} + B\phi_{xy} + C\phi_{yy} = F$$

  is analogous to the equation for a conic section:

$$Ax^2 + Bxy + Cy^2 = F$$

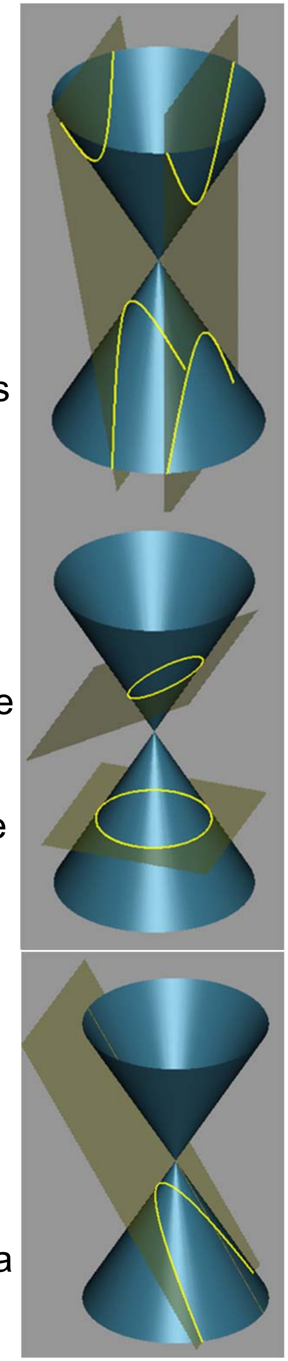
- Conic section:
  - Is the intersection of a right circular cone and a plane, which generates a group of plane curves, including the circle, ellipse, hyperbola, and parabola
  - One characterizes the type of conic sections using the discriminant $B^2\text{-}4AC$
- PDE:
  - $B^2\text{-}4AC > 0$ (Hyperbolic)
  - $B^2\text{-}4AC = 0$ (Parabolic)
  - $B^2\text{-}4AC < 0$ (Elliptic)

hyperbolas

ellipse

circle

parabola



Finite Differences - Basics

- Finite Difference Approximation idea directly borrowed from the definition of a derivative.

$$\phi'(x_i) = \lim_{\Delta x \to 0} \frac{\phi(x_{i+1}) - \phi(x_i)}{\Delta x}$$

- Geometrical Interpretation
  - Quality of approximation improves as stencil points get closer to x_i
  - Central difference would be exact if ϕ was a second order polynomial and points were equally spaced

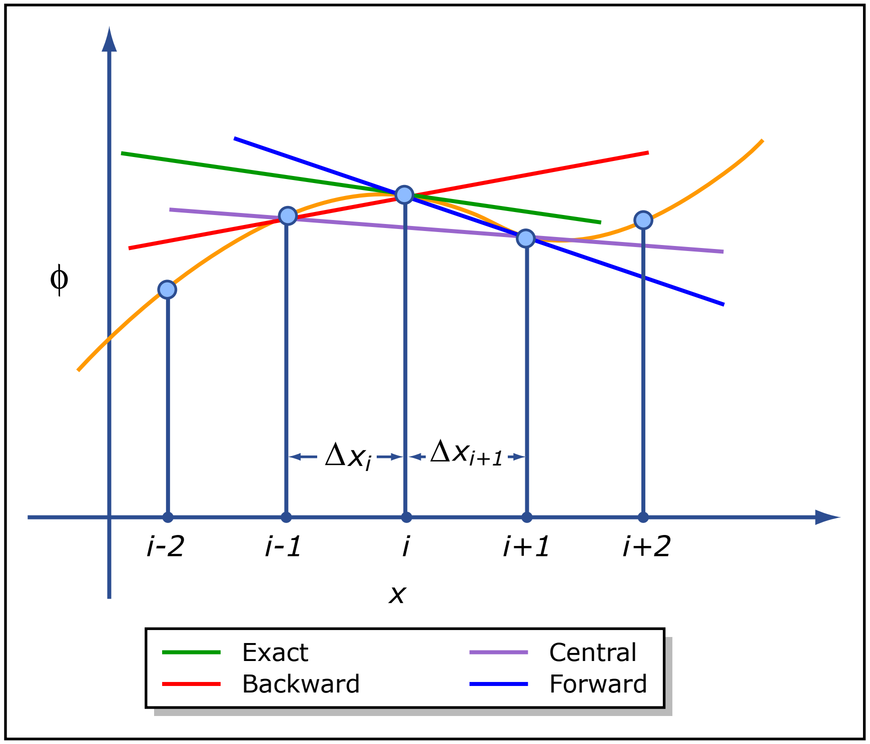


Image by MIT OpenCourseWare.

Forward Differences

Forward finite-divided-difference formulas:

First Derivative | **Error**

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_i)}{h} \qquad O(h)$$

$$f'(x_i) = \frac{-f(x_{i+2}) + 4f(x_{i+1}) - 3f(x_i)}{2h} \qquad O(h^2)$$

Second Derivative

$$f''(x_i) = \frac{f(x_{i+2}) - 2f(x_{i+1}) + f(x_i)}{h^2} \qquad O(h)$$

$$f''(x_i) = \frac{-f(x_{i+3}) + 4f(x_{i+2}) - 5f(x_{i+1}) + 2f(x_i)}{h^2} \qquad O(h^2)$$

Third Derivative

$$f'''(x_i) = \frac{f(x_{i+3}) - 3f(x_{i+2}) + 3f(x_{i+1}) - f(x_i)}{h^3} \qquad O(h)$$

$$f'''(x_i) = \frac{-3f(x_{i+4}) + 14f(x_{i+3}) - 24f(x_{i+2}) + 18f(x_{i+1}) - 5f(x_i)}{2h^3} \qquad O(h^2)$$

Backward Differences

Backward finite-divided-difference formulas:

First Derivative **Error**

$$f'(x_i) = \frac{f(x_i) - f(x_{i-1})}{h} \quad O(h)$$

$$f'(x_i) = \frac{3f(x_i) - 4f(x_{i-1}) + 3f(x_{i-2})}{2h} \quad O(h^2)$$

Second Derivative

$$f''(x_i) = \frac{f(x_i) - 2f(x_{i-1}) + f(x_{i-2})}{h^2} \quad O(h)$$

$$f''(x_i) = \frac{2f(x_i) - 5f(x_{i-1}) + 4f(x_{i-2}) - f(x_{i-3})}{h^2} \quad O(h^2)$$

Third Derivative

$$f'''(x_i) = \frac{f(x_i) - 3f(x_{i-1}) + 3f(x_{i-2}) - f(x_{i-3})}{h^3} \quad O(h)$$

$$f'''(x_i) = \frac{5f(x_i) - 18f(x_{i-1}) + 24f(x_{i-2}) - 14f(x_{i-3}) + 3f(x_{i-4})}{2h^3} \quad O(h^2)$$

Centered Differences

Centered finite-divided-difference formulas:

Error

First Derivative

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h} \qquad O(h)$$

$$f'(x_i) = \frac{-f(x_{i+2}) + 8f(x_{i+1}) - 8f(x_{i-1}) + f(x_{i-2})}{12h} \qquad O(h^2)$$

Second Derivative

$$f''(x_i) = \frac{f(x_{i+1}) - 2f(x_i) + f(x_{i-1})}{h^2} \qquad O(h)$$

$$f''(x_i) = \frac{-f(x_{i+2}) + 16f(x_{i+1}) - 30f(x_i) + 16f(x_{i-1}) - f(x_{i-2})}{12h^2} \qquad O(h^2)$$

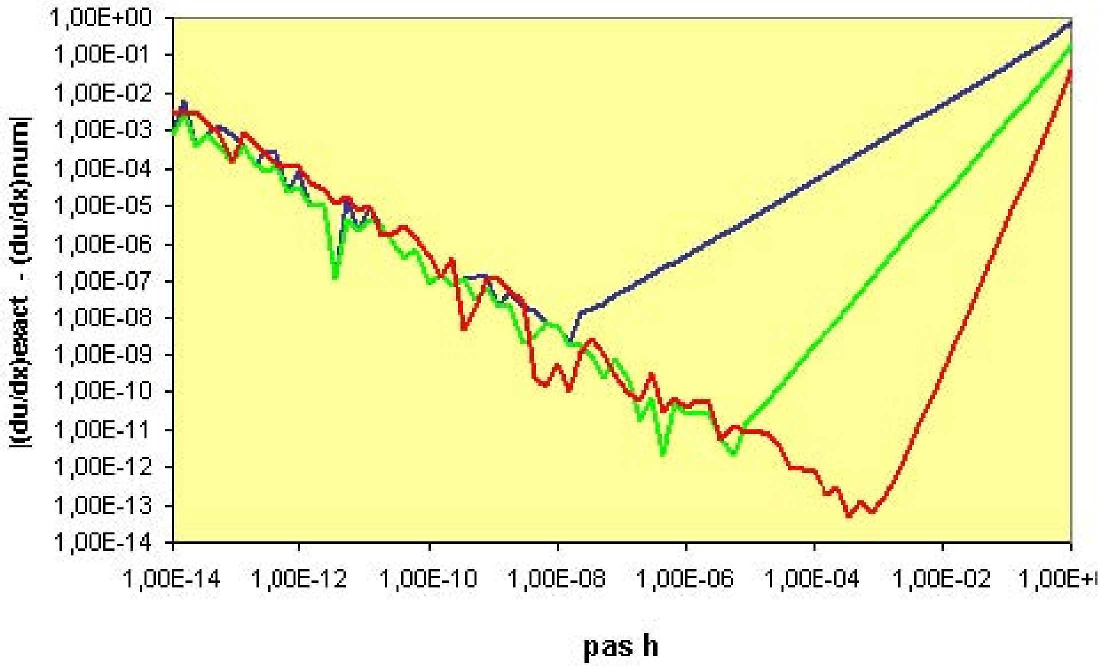
Third Derivative

$$f'''(x_i) = \frac{f(x_{i+2}) - 2f(x_{i+1}) + 2f(x_{i-1}) - f(x_{i-2})}{2h^3} \qquad O(h)$$

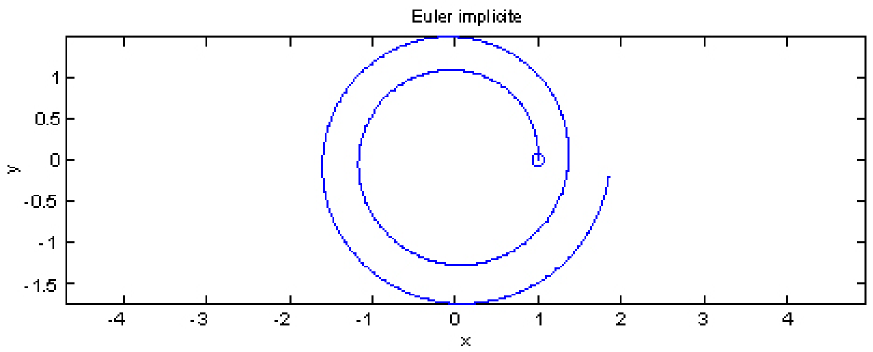
$$f'''(x_i) = \frac{-f(x_{i+3}) + 8f(x_{i+2}) - 13f(x_{i+1}) + 13f(x_{i-1}) - 8f(x_{i-2}) + f(x_{i-3})}{8h^3} \qquad O(h^2)$$

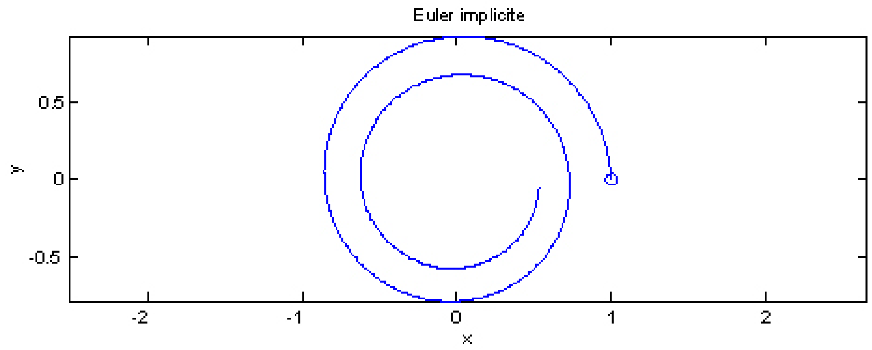
Exemple de calcul d'erreurs

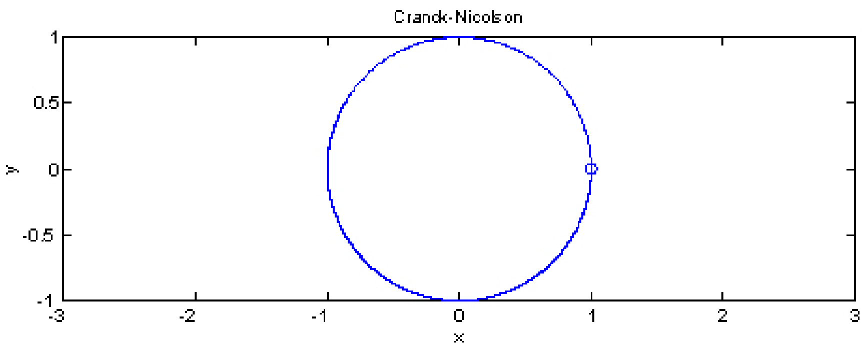
- Considérons la fonction $u(x)=e^{x,}$ en $x=0$. Les dérivées sont $d^n u/dx^n=e^x$ et valent 1 en 0. Nous donnons ci-dessous la valeur de quelques dérivées obtenues numériquement en $x=0$, par l'opérateur aux différences finies (DF) ci-dessous pour différents pas de discrétisation h :



- <u>Choix du type de méthode :</u>
  - Compromis précision/rapidité
  - Contraintes « matérielles » : puissance machiné, mémoire…
  - Contraintes de stabilité (limite sur le pas de temps dans le cas explicite et pas/peu dans le cas implicite)







Initial Value Problems: **Heun's method**
(which is also a "one-step" Predictor-Corrector scheme)

Initial Slope Estimate (Euler)

$$y_i' = f(x_i, y_i)$$

Predictor: Euler

$$y_{i+1}^0 = y_i + f(x_i, y_i)h$$

which allows to estimate the Endpoint Derivative/slope:

$$y_{i+1}' = f(x_{i+1}, y_{i+1}^0)$$

and so an Average Derivative Estimate:

$$\overline{y}' = \frac{y_i' + y_{i+1}'}{2} = \frac{f(x_i, y_i) + f(x_{i+1}, y_{i+1}^0)}{2}$$

Corrector

$$y_{i+1} = y_i + \frac{f(x_i, y_i) + f(x_{i+1}, y_{i+1}^0)}{2}h$$

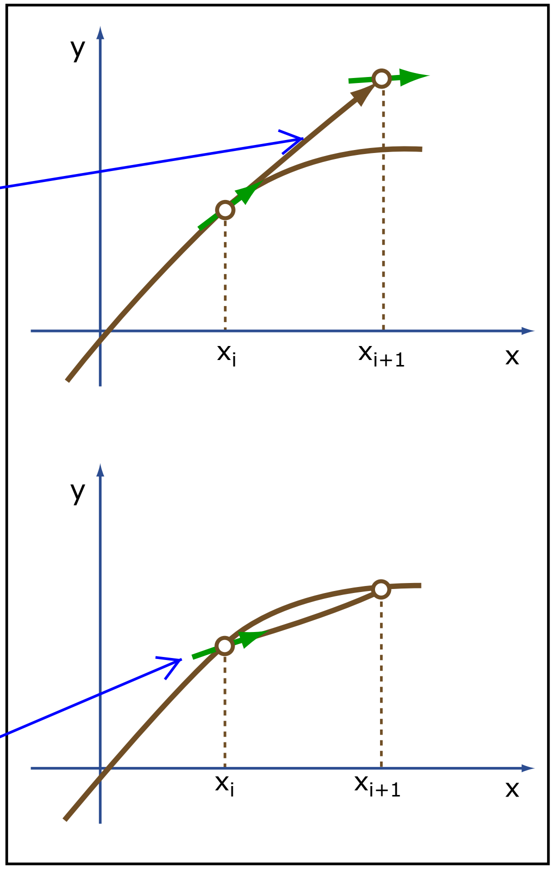


Image by MIT OpenCourseWare.

Heun can be set implicit, one can iterate => **Iterative Heun:**

$$y_{i+1}^{k+1} = y_i + \frac{f(x_i, y_i) + f(x_{i+1}, y_{i+1}^{k})}{2}h$$

Notes:
- Heun becomes Trapezoid rule if fully implicit scheme is used
- Heun's global error is of 2nd order: $O(h^2)$
- Convergence of iterative Heun not guaranteed + can be expensive with PDEs

Initial Value Problems: **Midpoint method**

First: uses Euler to obtain a Midpoint Estimate:

$$y_{i+1/2} = y_i + f(x_i, y_i)\frac{h}{2}$$

Then: uses this value to obtain a Midpoint Derivative Estimate:

$$y'_{i+1/2} = f(x_{i+1/2}, y_{i+1/2})$$

Assuming that this slope is representative of the whole interval => Midpoint Method recurrence:

$$\boxed{y_{i+1} = y_i + f(x_{i+1/2}, y_{i+1/2})h}$$

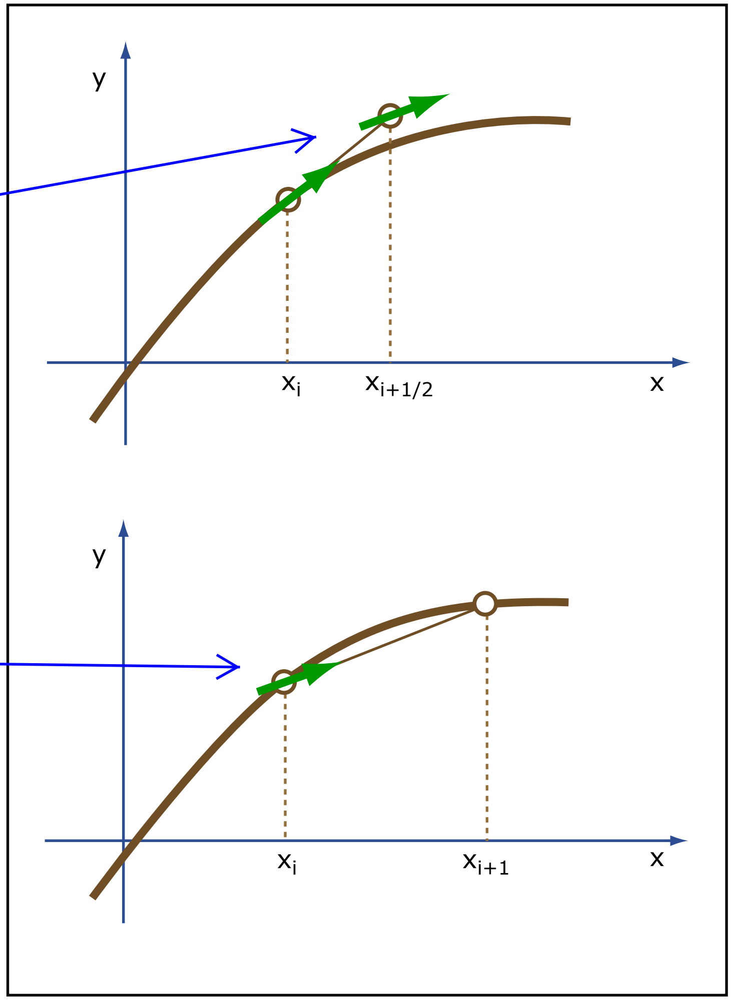


Image by MIT OpenCourseWare.

Comments:

- Midpoint superior to Euler since it uses a centered FD for the first derivative
- Midpoint's global error is of 2nd order: $O(h^2)$

Two-level methods for time-integration of (spatially discretized) PDEs

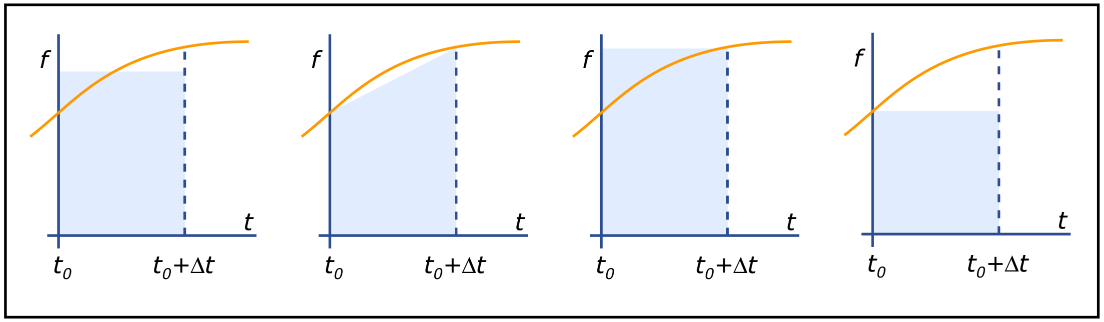
- Four simple schemes to estimate the time integral by approximate quadrature

$$\frac{d\phi}{dt} = f(t,\phi)\ ; \quad \text{with} \ \ \phi(t_0) = \phi_0 \quad \Leftrightarrow \quad \int_{t_n}^{t_{n+1}} \frac{d\phi}{dt}\, dt = \boxed{\phi^{n+1} - \phi^n = \int_{t_n}^{t_{n+1}} f(t,\phi)\, dt}$$

  - Explicit or Forward Euler: $\phi^{n+1} - \phi^n = f(t_n, \phi^n)\, \Delta t$
  - Implicit or backward Euler: $\phi^{n+1} - \phi^n = f(t_{n+1}, \phi^{n+1})\, \Delta t$
  - Midpoint rule (basis for the leapfrog method): $\phi^{n+1} - \phi^n = f(t_{n+1/2}, \phi^{n+1/2})\, \Delta t$
  - Trapezoid rule (basis for Crank-Nicholson method): $\phi^{n+1} - \phi^n = \frac{1}{2}\left[f(t_n, \phi^n) + f(t_{n+1}, \phi^{n+1}) \right] \Delta t$

Reminder on global error order:

- Euler methods are of order 1
- Midpoint rule and Trapezoid rule are of order 2
- Order n = truncation error cancels if true solution is polynomial of order n



Graphs showing the approximation of the time integral of f(t) using the midpoint rule, trapezoidal rule, implicit Euler, and explicit Euler methods. Image by MIT OpenCourseWare.

- Some comments
  - All of these methods are two-level methods (involve two times and are at best 2nd order)
  - All excepted forward Euler are implicit methods
  - Trapezoid rule often yields solutions that oscillates, but implicit Euler tends to behave well

Runge-Kutta Methods

Summary of General Taylor Series Method

$$x_n = a + nh\ , \quad n = 0, 1, \ldots N$$

$$y(x_{n+1}) = y_{n+1} = y_n + hT_k(x_n, y_n) + \frac{h^{k+1}}{(k+1)!} y^{(k+1)}(\xi)$$

where:

$$T_k(x_n, y_n) = f(x_n, y_n) + \frac{h}{2!} f'(x_n, y_n) + \cdots \frac{h^{k-1}}{k!} f^{(k-1)}(x_n, y_n)$$

$$E = \frac{h^{k+1} f^{(k)}(\xi, y(\xi))}{(k+1)!} = \frac{h^{k+1} y^{(k+1)}(\xi)}{(k+1)!}\ , \quad x_n < \xi x_n + h$$

Example: $k = 1$

Euler's method

$$y_{n+1} = y_n + hf(x_n, y_n)$$

$$E = \frac{h^2}{2!} y''(\xi)$$

Note: expensive to compute higher-order derivatives of $f(x,y)$, especially for spatially discretized PDEs => other schemes needed

Aim of Runge-Kutta Methods:

- Achieve accuracy of Taylor Series method without requiring evaluation of higher derivatives of $f(x,y)$
- Obtain higher derivatives using only the values of the RHS (first time derivative)
- Utilize points between t_n and t_{n+1} only

Initial Value Problems - Time Integrations
Derivation of 2nd order Runge-Kutta Methods

Taylor Series Recursion:

$$\begin{aligned} y(x_{n+1}) &= y(x_n) + hf(x_n, y_n) + \frac{h^2}{2}(f_x + ff_y)_n \\ &+ \frac{h^3}{6}(f_{xx} + 2ff_{xy} + f_{yy}f^2 + f_xf_y + f_y^2 f)_n + O(h^4) \end{aligned}$$

Runge-Kutta Recursion:

$$\boxed{\begin{aligned} y_{n+1} &= y_n + ak_1 + bk_2 \\ k_1 &= hf(x_n, y_n) \\ k_2 &= hf(x_n + \alpha h, y_n + \beta k_1) \end{aligned}}$$

Expand k_2 in a Taylor series:

$$\begin{aligned} \frac{k_2}{h} &= f(x_n + \alpha h, y_n + \beta k_1) \\ &= f(x_n, y_n) + \alpha h f_x + \beta k_1 f_y \\ &\quad + \frac{\alpha^2 h^2}{2} f_{xx} + \alpha h \beta k_1 f_{xy} + \frac{\beta^2}{2} {k_1}^2 f_{yy}) + O(h^4) \end{aligned}$$

Set a,b,α,β to match Taylor series as much as possible.

Substitute k_1 and k_2 in Runge Kutta

$$\begin{aligned} y_{n+1} &= y_n + (a+b)hf + bh^2(\alpha f_x + \beta f f_y) \\ &+ bh^3(\frac{\alpha^2}{2} f_{xx} + \alpha\beta f f_{xy} + \frac{\beta^2}{2} f^2 f_{yy}) + O(h^4) \end{aligned}$$

Match **2nd order Taylor series**

$$\left.\begin{aligned} a+b &= 1 \\ b\alpha &= 1/2 \\ b\beta &= 1/2 \end{aligned}\right\} \Leftarrow a = b = 0.5\ , \quad \alpha = \beta = 1$$

We have three equations and 4 unknowns =>

- There is an infinite number of Runge-Kutta methods of 2nd order
- These different 2nd order RK methods give different results if solution is not quadratic
- Usually, number of k's (recursion size) gives the order of the RK method.

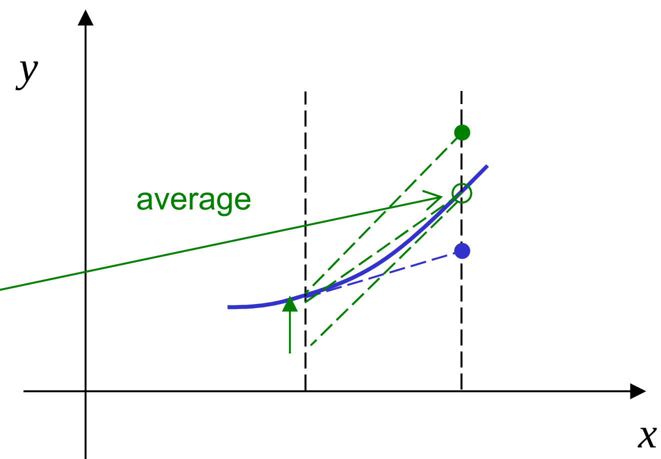
4^{th} order Runge-Kutta Methods
(Most Popular, there is an ∞ number of them, as for 2^{nd} order)

Initial Value Problem:

$$\begin{cases} y' &= f(x,y) \\ y(x_0) &= y_0 \\ x_n &= x_0 + nh \end{cases}$$

2^{nd} Order Runge-Kutta (Heun's version)

$$\boxed{\begin{aligned} y_{n+1} &= y_n + \tfrac{1}{2}(k_1 + k_2) \\ k_1 &= hf(x_n, y_n) \\ k_2 &= hf(x_n + h, y_n + k_1) \end{aligned}}$$



Predictor-corrector method

Second-order RK methods

b = ½, a = ½ : Heun's method

b= 1, a = 0 : Midpoint method

b =2/3, a = 1/3 : Ralston's Method

4^{th} Order Runge-Kutta

$$\boxed{\begin{aligned} y_{n+1} &= y_n + \tfrac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\ k_1 &= hf(x_n, y_n) \\ k_2 &= hf(x_n + \tfrac{h}{2}, y_n + \tfrac{k_1}{2}) \\ k_3 &= hf(x_n + \tfrac{h}{2}, y_n + \tfrac{k_2}{2}) \\ k_4 &= hf(x_n + h, y_n + k_3) \end{aligned}}$$

The k's are different estimates of the slope

Example: Taylor Table for the Adams-Moulton 3-steps (4 time-nodes) Method

Denoting $h \equiv \Delta t$, $\phi \equiv u$, $\dfrac{du}{dt} = u' = f(t,u)$ and $u'_n = f(t_n, u^n)$, one obtains for $K = 2$:

$$u^{n+1} - u^n = \sum_{k=-K}^{1} \beta_k f(t_{n+k}, u^{n+k})\, \Delta t = h\left[\beta_1 f(t_{n+1}, u^{n+1}) + \beta_0 f(t_n, u^n) + \beta_{-1} f(t_{n-1}, u^{n-1}) + \beta_{-2} f(t_{n-2}, u^{n-2})\right]$$

Taylor Table:

- The first row (Taylor series) + the last 5 rows (Taylor series for each term) must sum to zero
- This can be satisfied up to the 5th column (4th order term)
- Hence, the AM method with 4-time levels is 4th order accurate

	u_n	$h \cdot u'_n$	$h^2 \cdot u''_n$	$h^3 \cdot u'''_n$	$h^4 \cdot u''''_n$
u_{n+1}	1	1	$\frac{1}{2}$	$\frac{1}{6}$	$\frac{1}{24}$
$-u_n$	-1				
$-h\beta_1 u'_{n+1}$		$-\beta_1$	$-\beta_1$	$-\beta_1 \frac{1}{2}$	$-\beta_1 \frac{1}{6}$
$-h\beta_0 u'_n$		$-\beta_0$			
$-h\beta_{-1} u'_{n-1}$		$-\beta_{-1}$	β_{-1}	$-\beta_{-1} \frac{1}{2}$	$\beta_{-1} \frac{1}{6}$
$-h\beta_{-2} u'_{n-2}$		$-(-2)^0 \beta_{-2}$	$-(-2)^1 \beta_{-2}$	$-(-2)^2 \beta_{-2} \frac{1}{2}$	$-(-2)^3 \beta_{-2} \frac{1}{6}$

solving for the β_k's $\Rightarrow \beta_1 = 9/24, \quad \beta_0 = 19/24, \quad \beta_{-1} = -5/24$ and $\beta_{-2} = 1/24$

Example of Adams Methods for Time-Integration

Explicit Methods. (Adams-Bashforth, with ABn meaning nth order AB)

u_{n+1}	$= u_n + hu'_n$	Euler
u_{n+1}	$= u_{n-1} + 2hu'_n$	Leapfrog
u_{n+1}	$= u_n + \frac{1}{2}h\left[3u'_n - u'_{n-1}\right]$	AB2
u_{n+1}	$= u_n + \frac{h}{12}\left[23u'_n - 16u'_{n-1} + 5u'_{n-2}\right]$	AB3

Implicit Methods. (Adams-Moulton, with AMn meaning nth order AM)

u_{n+1}	$= u_n + hu'_{n+1}$	Implicit Euler
u_{n+1}	$= u_n + \frac{1}{2}h\left[u'_n + u'_{n+1}\right]$	Trapezoidal (AM2)
u_{n+1}	$= \frac{1}{3}\left[4u_n - u_{n-1} + 2hu'_{n+1}\right]$	2nd-order Backward
u_{n+1}	$= u_n + \frac{h}{12}\left[5u'_{n+1} + 8u'_n - u'_{n-1}\right]$	AM3

Practical Time-Integration Methods for CFD

- High-resolution CFD requires large discrete state vector sizes to store the spatial information
- This means that up to two times (one on each side of the current time step) have often been utilized (3 time-nodes): $u^{n+1} - u^n = h\left[\beta_1 f(t_{n+1}, u^{n+1}) + \beta_0 f(t_n, u^n) + \beta_{-1} f(t_{n-1}, u^{n-1})\right]$
- Rewriting this equations in a way such that differences wrt. the Euler's method are easily seen, one obtains ($\theta = 0$ for explicit schemes):

$$\boxed{(1+\xi)\, u^{n+1} = \left[(1+2\xi)\, u^n - \xi\, u^{n-1}\right] + h\left[\theta\, f(t_{n+1}, u^{n+1}) + (1-\theta+\varphi)\, f(t_n, u^n) - \varphi f(t_{n-1}, u^{n-1})\right]}$$

θ	ξ	φ	Method	Order
0	0	0	Euler	1
1	0	0	Implicit Euler	1
1/2	0	0	Trapezoidal or AM2	2
1	1/2	0	2nd-order Backward	2
3/4	0	−1/4	Adams type	2
1/3	−1/2	−1/3	Lees	2
1/2	−1/2	−1/2	Two-step trapezoidal	2
5/9	−1/6	−2/9	A-contractive	2
0	−1/2	0	Leapfrog	2
0	0	1/2	AB2	2
0	−5/6	−1/3	Most accurate explicit	3
1/3	−1/6	0	Third-order implicit	3
5/12	0	1/12	AM3	3
1/6	−1/2	−1/6	Milne	4

Comparaison des différents schémas

- Résolution de l'équation :

$$\frac{du}{dt} = u \ \textbf{avec} \ u(t=0) = 1$$

- *Du bas en haut et donc du moins performant au plus performant :*
  - *Euler,*
  - *Adams-Bashforth 1,*
  - *Milne 1,*
  - *Adams-Bashforth 2,*
  - *prédicteur-correcteur et trapèze,*
  - *Milne 3,*
  - *Adams-Bashforth 3,*
  - *Milne 3,*
  - *Runge-Kutta d'ordre 4*