MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

COMPUTATIONAL METHODS IN
SYSTEMS AND CONTROL THEORY

# A Bluffer's Guide to Time Integration

C. Himpe

DISC Reading Group (Time Integration)

2021–03–10

# Motivation

$$\frac{\partial x}{\partial t} =: \dot{x} = f(t, x(t))$$

- Generally: $0 = f(t, x, \dot{x}, \ddot{x}, \dots)$
- Reminder: Higher order systems reduce to first order systems
- Here: Only explicit first-order form

# Initial Value Problems (IVP)

$$\dot{x}(t) = f(t, x(t)), \quad x(t_0) = x_0$$

- Opposed to: Boundary value probelms (BVP)
- Generally: discretize time and iterate
- Methods: Taylor series, numerical quadrature, finite differences

# Solvability

$$\exists! x$$

- Fundamental theorem of analysis
- Picard-Lindelöf theorem
- Peano theorem

# Taylor Series

$$x(t_k + h) = x(t_k) + h\ \dot{x}(t_k) + \frac{h^2}{2}\ddot{x}(t_k) + O(h^3)$$

- At time $t_k$, advance $h$.
- Essential for error anaylsis.
- Can be used to derive higher order methods.

# Numerical Quadrature

$$x(t_k + h) = \int_{t_k}^{t_k+h} f(s, x)\,\mathrm{d}s \approx h \cdot \sum_{n=1}^{N} \omega_n f(t_n, x_n)$$

- How to choose $h$?
- What are "good" sets triples $(\omega_n, t_n, x_n)$?
- What would the simple Riemann integration look like?

# Finite Difference I

$$\dot{x}(t_k) \approx \frac{x(t_k + h) - x(t_k)}{h} = f(t_k, x(t_k))$$

$$\Rightarrow x(t_k + h) = x(t_k) + h \cdot f(t_k, x(t_k))$$

- Explicit Euler Method.
- Explicit means $x(t+h)$ in terms of $x(t)$.
- Equivalent to left-hand Riemann sum.

# Finite Difference II

$$\dot{x}(t_k) \approx \frac{x(t_k) - x(t_k - h)}{h} = f(t_k, x(t_k))$$

$$\Rightarrow x(t_k) = x(t_k - h) + h \cdot f(t_k, x(t_k))$$

- Implicit Euler Method.
- Implicit means $x(t)$ in terms of $x(t)$.
- Equivalent to right-hand Riemann sum.

# Order

- Are we done here?
- Just use smaller time-steps?
- No: Error decay!
- Think: Polynomial interpolation!

# Single-Step Methods

$$x_{k+1} = x_k + h \sum_{n=1}^{S} b_n z_n$$

$$z_n = f\left(t_k + h\, c_n,\ x_k + h \sum_{m=1}^{S} a_{nm} z_m\right)$$

**Butcher Tableau:**

$$\begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \dots & a_{1S} \\ c_2 & a_{21} & a_{22} & \dots & a_{2S} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_S & a_{S1} & a_{S2} & \dots & a_{SS} \\ \hline & b_1 & b_2 & \dots & b_S \end{array}$$

# Prominent Methods

- **2nd Order:**
  - Collatz (modified Euler, explicit midpoint)
  - Heun (improved Euler, explicit trapezoidal)
  - Ralston (explicit, minimum local error)
  - Implicit midpoint (symplectic)
  - Crank-Nicholson (implicit trapezoidal)
- **3rd Order:**
  - Kutta (explicit)
  - Heun (explicit)
  - Ralston (explicit, minimum local error)
  - Simpson's rule (explicit)
- **4th Order:**
  - Classic Runge-Kutta (explicit)
  - 3/8-Rule (explicit)

# Multi-Step Methods

$$\sum_{\ell=0}^{R} v_\ell x_{k+l} = h \sum_{\ell=0}^{R} u_k f(t_{k+\ell}, x_{k+\ell})$$

- Needs single-step method for initial step(s).

# Prominent methods

- **Adams-Bashforth** (explicit)
  - AB2 used to be a standard method for engineers.
  - Maximum order $s$ for $s$ steps.
- **Adams-Moulton** (implicit)
  - Maximum order $s+1$ for $s$ steps.
  - Maximum order $2$ for A-stable methods.
- **Nyström** (explicit)
  - Maximum order $s$ for $s > 1$ steps.
- **Milne-Simpson** (implicit)
  - Minimum order $4$ for $s > 1$ steps.
- **Backward Differentiation** (implicit)
  - aka Gear's method.
  - Maximum order $6$.

# General Linear Methods (GLM)

$$x_k = h \sum_{\ell=1}^{S} a_{k\ell} f(t_\ell, x_\ell) + \sum_{\ell=1}^{R} u_{k\ell} x_\ell^{n-1}$$

$$z_k = h \sum_{\ell=1}^{S} b_{k\ell} f(t_\ell, x_\ell) + \sum_{\ell=1}^{R} v_{k\ell} x_\ell^{n-1}$$

- Single-Step methods ($R = 1$)
- Multi-Step methods ($S = 1$)

**Apply GLM method to Dahlquist's test equation:**

$$\dot{y} = ay$$

$$\overset{\text{ex. Euler}}{\rightarrow} y_k = (1 + ha)^k y_0 \overset{z := ha}{\rightarrow} R(z) := 1 + z$$

$$\rightarrow A = \{z \in \mathbb{C} : |R(z)| < 1\}$$

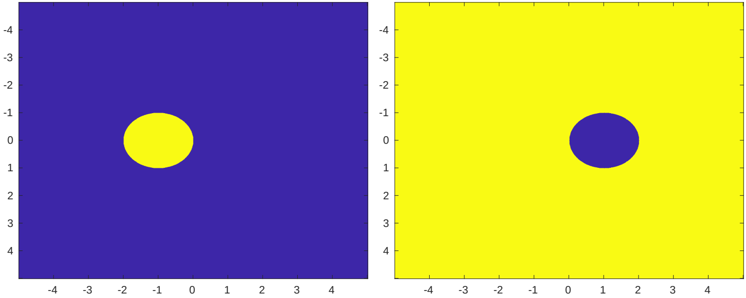


Figure: Stability regions (yellow) for explicit and implicit Euler methods.

# Core Properties

- **Truncation Error**
  - Local: Error after a single step
  - Global: Error after many steps
- **Consistency**
  - Consistent if $|\frac{x(t_k+h)-x(t_k)}{h} - x_{k+1}| \overset{h \to 0}{\to} 0$
  - Discretization error
- **Convergence**
  - Convergence if $\max(|x_k - x(t_k)|) \overset{h \to 0}{\to} 0$
  - Approximation error
- **Stability**
  - Numerical errors are not magnified.
  - Dahlquist theorem: Convergence = Consistency & Stability
- **Complexity**
  - Linear problems: flops
  - Nonlinear problems: vector field evaluations (here: stages & steps)

# Stability Alphabet

- **A-Stability**
  $R(z) < 1 \; \forall \; \mathrm{Re}(z) \leq 0$
- **B-Stability**
  $\|x_k - z_k\| \leq \|x_{k-1} - z_{k-1}\|$
- **D-Stability / 0-Stability**
  $\|x_k\| \leq M\|x_0\|$
- **G-Stability**
  $\|x_k - z_k\|_G^2 \leq \|x_{k-1} - z_{k-1}\|_G^2$
- **I-Stability:**
  $R(z) < 1 \; \forall \; \mathrm{Re}(z) = 0$
- **L-Stability:**
  A-stability & $\lim_{z \to \infty} R(z) = 0$ (stiffly accurate)
- **$\mathrm{L}_2$-Stability / Strong Stability**
  $\|x_{k+1}\| \leq \|x_k\|$
- and more: **Energy-Stability**, **C-Stability**, **P-Stability**, **Q-Stability**, ...

# Other Properties

- **Natural Continuous Extension**
  Existence of polynomials to interpolate in between time steps.
- **Minimum Error**
  Methods with free parameter optimized to yield minimum local error.
- **(Pseudo-)Symplectic / (Pseudo-)Energy-Preserving**
  $R(z)R(-z) \equiv 1$
- **Stiffly Accurate**
  $\lim_{z\to\infty} R(z) = 0.$
- and more **Passive**, **Positive**, **Lossless**, ...

# Adaptive & Embedded Methods

- **Adaptivity:**
  Adjust time-step based on error of two methods of consecutive order.
- **Embedding:**
  Use same sampling points and weights for both methods.

$$
\begin{array}{c|cccc}
c_1 & a_{11} & a_{12} & \dots & a_{1S} \\
c_2 & a_{21} & a_{22} & \dots & a_{2S} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
c_S & a_{S1} & a_{S2} & \dots & a_{SS} \\
\hline
 & b_1 & b_2 & \dots & b_S \\
 & \tilde{b}_1 & \tilde{b}_2 & \dots & \tilde{b}_S
\end{array}
$$

**Examples:**

- Euler
- Heun-Euler (1,2)
- Runge-Kutta-Fehlberg (RKF45)
- Block, Vectorized (7,8) pair (BV78)

$$\dot{x}(t) = \underbrace{Ax(t)}_{\text{linear, stiff}} + \underbrace{f(t, x(t))}_{\text{nonlinear, nonstiff}}$$

- Solve linear part implicitly, nonlinear part explicitly.
- Same order implicit Runge-Kutta and explicit Runge-Kutta.
- Extra constraints for overall Runge-Kutta compliance.
- Gas networks: First-order IMEX[1] (non-RK[2]).

[1] S. Grundel and L. Jansen. **Efficient simulation of transient gas networks using IMEX integration schemes and MOR methods**. In 54th IEEE Conference on Decision and Control: 4579–4584, 2015. doi:10.1109/CDC.2015.7402934

[2] U.M. Ascher, S.J. Ruuth, and R.J. Spiteri. **Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations**. Applied Numerical Mathematics, 25(2–3): 151–167, 1997. doi:10.1016/S0168-9274(97)00056-1

# Method Zoo

- **Diagonally Implicit Runge-Kutta method (DIRK)**
  Efficient implicit methods (see also SDIRK, PDIRK).
- **Predictor-Corrector Methods:**
  Use explicit method (predictor) to solve implicit method (corrector).
- **Total Variation Diminishing (TVD) Runge-Kutta:**
  Treat Gibbs phenomenon of hyperbolic PDEs (see also [W]ENO).
- **Gauss-Legendre Methods:**
  Implicit Runge-Kutta of maximal order $2s$ for $s$ stages.
- **Radau Methods:**
  Implicit Runge-Kutta of order $2s-1$ for $s$ stages for stiff problems.
- **Lobatto Methods:**
  Implicit Runge-Kutta of order $2s-2$ for $s$ stages for DAEs.
- **Runge-Kutta-Chebyshev (RKC) Methods:**
  Enhanced stability methods (see also DUMKA and ROCK).

# Special Mention

## SSPX2

- 2nd-order explicit Runge-Kutta.
- Optimal strong-stability-preserving ($\mathcal{C} = 1$).
- Low-storage algorithm (only two “registers” required).
- Stability-enhanced (more stages $\rightarrow$ large stability region).
- Default solver of `emgr`.

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

# Special Mention

## SSPX2

- 2nd-order explicit Runge-Kutta.
- Optimal strong-stability-preserving ($\mathcal{C} = 1$).
- Low-storage algorithm (only two "registers" required).
- Stability-enhanced (more stages $\rightarrow$ large stability region).
- Default solver of `emgr`.

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{2} & 0 \\ \hline & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{array}$$

# Special Mention

## SSPX2

- 2nd-order explicit Runge-Kutta.
- Optimal strong-stability-preserving ($\mathcal{C} = 1$).
- Low-storage algorithm (only two "registers" required).
- Stability-enhanced (more stages → large stability region).
- Default solver of `emgr`.

$$
\begin{array}{c|cccc}
0 & 0 & 0 & 0 & 0 \\
\frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 \\
\frac{2}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 \\
1 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 \\
\hline
 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4}
\end{array}
$$

# Actual GLMs

- Two-Step Runge-Kutta Methods
- Two-Step SSP Methods
- Accelerated Runge-Kutta Methods
- Improved Runge-Kutta Methods
- Pseudo Runge-Kutta Methods
- Almost Runge-Kutta Methods

# Generalizing GLMs

- **Rosenbrock methods:**
  Use derivatives of vector field $f$.
- **Obreshkov methods:**
  Use derivatives of trajectory $x$.
- **Second-Order methods:**
  Semi-implicit Euler, Leapfrog, Verlet, Central finite differences, ...
- **Modified Runge-Kutta methods:**
  Use alternative means to combine samples $(b_i)$.

# Interesting References


- A.-M. Wazwaz: **A comparison of modified Runge-Kutta formulas based on a variety of means**; International Journal of Computer Mathematics 50(1–2): 105–112, 1994. doi:10.1080/00207169408804245
- D. Fränken, K. Ochs; **Passive Integration Methods: Fundamental Theory**; AEU - International Journal of Electronics and Communications 55(3): 153–163, 2001. doi:10.1078/1434-8411-00024
- D. Fränken, K. Ochs: **Numerical Stability Properties of Passive Runge-Kutta Methods**; IEEE International Symposium on Circuits and Systems: 473–476, 2001. doi:10.1109/ISCAS.2001.921350
- D.I. Ketcheson: **Highly Efficient Strong Stability-Preserving Runge–Kutta Methods with Low-Storage Implementations**; SIAM J. Sci. Comput. 30(4): 2113–2136, 2008. doi:10.1137/07070485X
- E. Celledoni, R.I. McLachlan, D.I. McLaren, B. Owren, G.R.W. Quispel, W.M. Wright: **Energy-Preserving Runge-Kutta Methods**; ESAIM: M2AN 43: 645–649, 2009. doi:10.1051/m2an/2009020
- S. Boscarino, L. Pareschi, G. Russo: **A Unified IMEX Runge-Kutta Approach for Hyperbolic Systems with Multiscale Relaxation**; SIAM J. Numer. Anal. 55(4): 2085–2109, 2017. doi:10.1137/M1111449

# General References


- E. Hairer, S.P. Norsett, G. Wanner: **Solving Ordinary Differential Equations I**; Springer Series in Computational Mathematics, vol. 8, 1993. `doi:10.1007/978-3-540-78862-1`
- E. Hairer, G. Wanner: **Solving Ordinary Differential Equations II**; Springer Series in Computational Mathematics, vol. 14, 1996. `doi:10.1007/978-3-642-05221-7`
- L.F. Shampine: **Numerical Solution of Ordinary Differential Equations**; 1994. `doi:10.1201/9780203745328`
- P. Deuflhard, F. Bornemann: **Scientific Computing with Ordinary Differential Equations**; Texts in Applied Mathematics, vol. 42, 2012. `doi:10.1007/978-0-387-21582-2`
- J.C. Butcher: **Numerical Methods for Ordinary Differential Equations**; 2003. `doi:10.1002/9781119121534`
- **John Butcher's Tutorials**: `https://www.math.auckland.ac.nz/~butcher/ODE-book-2008/Tutorials/`