MA5233 Computational Mathematics

Lecture 17: Implicit Runge-Kutta Methods

Simon Etter

National University of Singapore

2019/2020

Implicit Runge-Kutta Methods

Implicit Runge-Kutta methods

Variation on the explicit Runge-Kutta methods introduced in Lecture 16.

The following example introduces a model problem where explicit methods perform poorly.

Example

Consider the ODE

$$y(0) = 1, \qquad \dot{y}(t) = -y(t) \quad \text{for } t \geq 0.$$

Solution is given by $y(t) = \exp(-t)$.

Numerical observations: (see `example()` in the accompanying Julia file)

- Midpoint method diverges if the ratio $\frac{T}{m}$ becomes too large.
- Same holds true for Euler method.

Implicit Runge-Kutta Methods

Stability analysis

For the simple ODE $\dot{y} = -y$, we can compute explicit formulae for a single Euler / midpoint step.

- Euler: $$\begin{aligned}\tilde{y}(\tfrac{T}{m}) &= y(0) + f\big(y(0)\big)\,\tfrac{T}{m} = y(0) - y(0)\,\tfrac{T}{m} \\ &= (1 - \tfrac{T}{m})\,y(0).\end{aligned}$$
- Midpoint: $$\begin{aligned}\tilde{y}(\tfrac{T}{m}) &= y(0) + f\Big(y(0) + f\big(y(0)\big)\,\tfrac{T}{2m}\Big)\,\tfrac{T}{m} \\ &= y(0) - \Big(y(0) - y(0)\,\tfrac{T}{2m}\Big)\,\tfrac{T}{m} \\ &= \Big(1 - \tfrac{T}{m} + \tfrac{T^2}{2m^2}\Big)\,y(0).\end{aligned}$$

Observation: numerical solution after k steps is given by

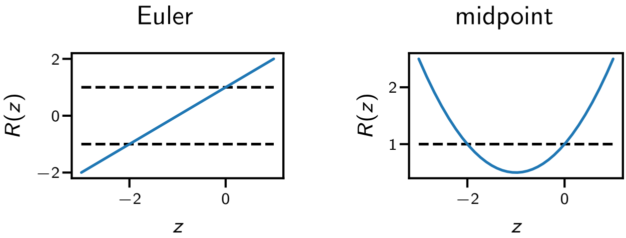
$$\tilde{y}(\tfrac{T}{m}\,k) = R(-\tfrac{T}{m})^k\,y(0) \quad \text{where} \quad R(z) = \begin{cases} 1 + z & \text{(Euler)}, \\ 1 + z + \frac{z^2}{2} & \text{(midpoint)}. \end{cases}$$

Hence, $\lim_{k\to\infty} \tilde{y}(\frac{T}{m}k) = 0$ if and only if $|R(-\frac{T}{m})| < 1$.

Implicit Runge-Kutta Methods

Stability analysis (continued)

Plot of stability functions $R(z)$:



Conclusion: $\tilde{y}(\frac{T}{m}k) = R(-\frac{T}{m})^k \, y(0) \to 0$ if and only if $\frac{T}{m} < 2$.

This is precisely what we observed experimentally.

Implicit Runge-Kutta Methods

Discussion

In previous lecture, we have seen the estimate $|\tilde{y}(T) - y(T)| = \mathcal{O}\big(m^{-p}\big)$ with $p = 1$ (Euler) and $p = 2$ (midpoint).
This is an asymptotic estimate for the limit $m \to \infty$.

The above stability analysis sheds some light on the behaviour of Runge-Kutta methods in the preasymptotic regime $m \ll \infty$.

See also `convergence()` in accompanying Julia file.

Implicit Runge-Kutta Methods

Linearisation of ODEs

Discussion so far was specific to the ODE $\dot{y} = -y$, but the conclusions are relevant for generic ODEs $\dot{y} = f(y)$ as long as there is an *attractive fixed-point*, i.e. a y_f such that $f(y_f) = 0$ and $\nabla f(y_f)$ has at least one eigenvalue λ with $\mathrm{Re}(\lambda) < 0$.

"Proof". For y close to y_f, we obtain

$$\begin{aligned}\frac{d}{dt}\big(y(t) - y_f\big) &= f\big(y(t)\big) \\ &= f(y_f) + \nabla f(y_f)\,\big(y(t) - y_f\big) + \mathcal{O}\big(\|y(t) - y_f\|^2\big) \\ &= \nabla f(y_f)\,(y - y_f) + \mathcal{O}\big(\|y(t) - y_f\|^2\big).\end{aligned}$$

Assume $\nabla f(y_f)$ has eigendecomposition $\nabla f(y_f) = V \Lambda V^{-1}$.
Ignoring the $\mathcal{O}$-term and introducing $y(t) - y_f = V w(t)$, we obtain

$$V \dot{w}(t) = \frac{d}{dt} V w(t) = \nabla f(y_f)\, V w(t) = V \Lambda w(t) \quad \Longleftrightarrow \quad \dot{w} = \Lambda w.$$

Implicit Runge-Kutta Methods

"Proof" (continued).

Solution to $\dot{w} = \Lambda w$ is given by $w_i(t) = \exp(\lambda_i\, t)\, w_i(0)$.

We observe $\lim_{t\to\infty} w_i(t) = 0$ if $\mathrm{Re}(\lambda_i) < 0$.

Repeating the above stability analysis, one can show

$$\tilde{w}_i(\tfrac{T}{m}\, k) = R(\lambda_i \tfrac{T}{m})^k\, w_i(0) \quad \text{where} \quad R(z) = \begin{cases} 1 + z & \text{(Euler)}, \\ 1 + z + \frac{z^2}{2} & \text{(midpoint)}. \end{cases}$$

Same conclusion as before:

$$\lim_{k\to\infty} \tilde{w}_i(\tfrac{T}{m} k) = 0 \text{ if and only if } |R(\lambda_i \tfrac{T}{m})| < 1.$$

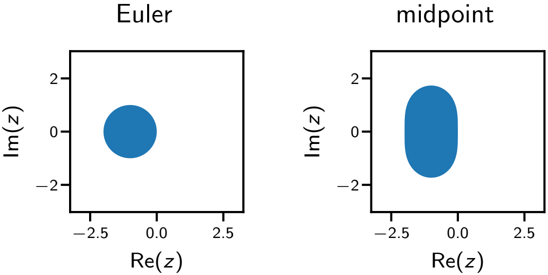
However, unlike before we are now also interested in the behaviour of $R(z)$ for complex z if λ_i is complex.

In particular, we are interested in the *stability domain*

$$\big\{ z \in \mathbb{C} \mid |R(z)| < 1 \big\}.$$

Implicit Runge-Kutta Methods

Stability domains $\{z \in \mathbb{C} \mid |R(z)| < 1\}$



Implicit Runge-Kutta Methods

Example

Consider the ODE $\ddot{x} = -x$, or equivalently

$$\dot{y} = \begin{pmatrix} \dot{y}_1 \\ \dot{y}_2 \end{pmatrix} = \begin{pmatrix} y_2 \\ -y_1 \end{pmatrix} = f(y) \qquad \text{with} \qquad \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} x \\ \dot{x} \end{pmatrix}.$$

We have $\nabla f = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$ with eigenvalues $\lambda = \pm\iota$.

Knowledge of the eigenvalues can be used to predict rate of divergence of Euler and midpoint methods. See `harmonic_oscillator()`.

Implicit Runge-Kutta Methods

Stability functions of explicit Runge-Kutta methods

Consider Runge-Kutta scheme with Butcher tableau $\left(\begin{array}{c|c} x & V \\ \hline & w^T \end{array}\right)$

applied to the ODE $\dot{y} = \lambda y$.
Solution after a single step from 0 to t is given by

$$\tilde{y}(t) = y(0) + w^T \mathbf{f}\, t, \quad \mathbf{f} = \lambda \Big(y(0) + \lambda V\, \mathbf{f}\, t \Big).$$

Simplifying these formulae yields ($\mathbf{1}$ denotes vectors of all-ones)

$$\tilde{y}(t) = \Big(1 + \lambda t\, w^T \big(I - \lambda t\, V \big)\, \mathbf{1} \Big) y(0).$$

Hence, stability function is given by

$$R(z) = 1 + z\, w^T \big(I - z\, V \big)^{-1} \mathbf{1}.$$

Implicit Runge-Kutta Methods

Example: midpoint method

Butcher tableau:

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

Stability function:

$$\begin{aligned} R(z) &= 1 + z \begin{pmatrix} 0 & 1 \end{pmatrix} \left(I - z \begin{pmatrix} 0 & 0 \\ \frac{1}{2} & 0 \end{pmatrix} \right)^{-1} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\ &= 1 + z \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -\frac{z}{2} & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\ &= 1 + z \left(1 + \frac{z}{2}\right) \\ &= 1 + z + \frac{z^2}{2}. \end{aligned}$$

Implicit Runge-Kutta Methods

Stability function (recap)

$$R(z) = 1 + z\, w^T \left(I - z\, V\right)^{-1} \mathbf{1}$$

Stability domains of explicit Runge-Kutta methods

- For explicit methods, V is strictly lower triangular.
- One can show: V strictly lower triangular $\implies$ $R(z)$ is a polynomial.
- $\lim_{|z|\to\infty} |R(z)| = \infty$ for all polynomials $R(z)$.
- Hence stability domain is bounded.

Conclusion: for all explicit methods, there is a stability constraint $\frac{T}{m} < \Delta t_{\max}(f)$ on the time step $\frac{T}{m}$.

Discussion

It depends on application whether time-step constraint is a problem.

Next slide provides a typical example of when time-step constraints are indeed a problem.

Implicit Runge-Kutta Methods

Example: separation of time scales

$$\begin{pmatrix} \dot{y}_1 \\ \dot{y}_2 \end{pmatrix} = \begin{pmatrix} 1000\iota\, y_1 \\ 1 \end{pmatrix}, \qquad \begin{pmatrix} y_1(0) \\ y_2(0) \end{pmatrix} = \begin{pmatrix} 0.001 \\ 0 \end{pmatrix}.$$

Behaviour of solution:

- $y_1(t) = 0.001\, e^{1000\iota t}$: high-freq. oscillation with small amplitude.
- $y_2(t) = t$: linear motion.

Why time-step constraint is problematic:

- Oscillations in y_1 dictate largest admissible time step but contribute very little to final solution due to small amplitude.
- We would be happy not to resolve the oscillations of y_1 as long as doing so does not affect the linear motion in y_2.
- In explicit methods, this is not possible: choosing a time-step much larger than the oscillation period $\frac{2\pi}{1000}$ will result in exponential blow-up in $\tilde{y}_1$.

Implicit Runge-Kutta Methods

Example (continued)

A situation as described on previous slide frequently arises in molecular dynamics simulations (`https://youtu.be/GClPr5Qpd5A`).

Consider a two-atom molecule (e.g. oxygen O_2):

- Bond between the atoms introduces a high-frequency oscillation within the molecule.
- Centre of mass moves very slow compared to this oscillation.
- Step size is determined by oscillation frequency even though we mostly care about translation and rotation of molecule.
- Common approach to avoid step-size constraint: replace stretchable with rigid bond.

Remark

We will encounter another example of "separation of time-scales" when we discuss numerical methods for parabolic PDEs.

Implicit Runge-Kutta Methods

Stability function (recap): $R(z) = 1 + z\, w^T \left(I - z\, V\right)^{-1} \mathbf{1}$

Implicit Runge-Kutta methods

Stability constraints can be avoided if we allow nonzeros in the upper triangle of V, since then $R(z)$ becomes a rational function which may be bounded for $|z| \to \infty$.

Examples

- Implicit Euler method: $\tilde{y}(t) = y(0) + f\left(\tilde{y}(t)\right) t$.

$$\left(\begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array}\right), \qquad R(z) = 1 + \frac{z}{1-z} = \frac{1}{1-z}.$$

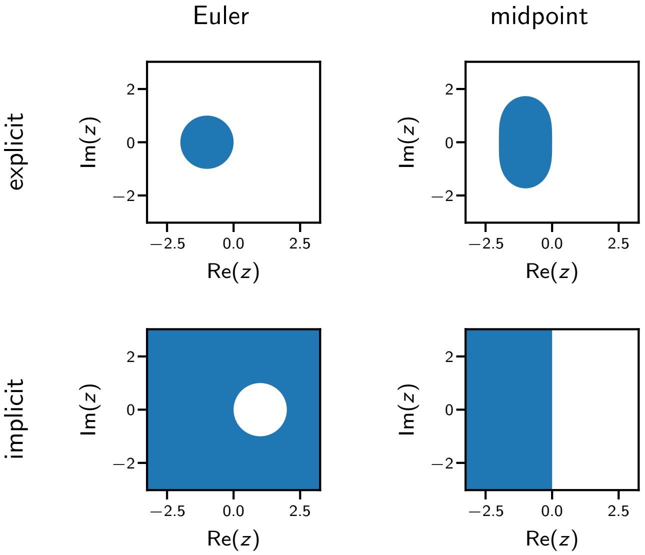
- Implicit midpoint method:

$$\tilde{y}(t) = y(0) + f\left(\tilde{y}(\tfrac{t}{2})\right) t, \qquad \tilde{y}(\tfrac{t}{2}) = y(0) + f\left(\tilde{y}(\tfrac{t}{2})\right) \tfrac{t}{2}.$$

$$\left(\begin{array}{c|c} \frac{1}{2} & \frac{1}{2} \\ \hline & 1 \end{array}\right), \qquad R(z) = 1 + \frac{z}{1-\frac{z}{2}} = \frac{2+z}{2-z}.$$

Implicit Runge-Kutta Methods

Stability domains $\{z \in \mathbb{C} \mid |R(z)| < 1\}$



Implicit Runge-Kutta Methods

Discussion: explicit vs. implicit methods

Main drawback of implicit methods:

must solve equations of the form $y = F(y)$.

Not much of a problem if $F(y)$ is linear and fast solvers are available.

If not, solving $y = F(y)$ may be as expensive as taking many small explicit time-steps.

Conclusion: which algorithm is best is highly problem-specific.

Remark

$y = F(y)$ may be a non-linear equation!

Time permitting, we will discuss algorithms for solving such problems later in this module.

Implicit Runge-Kutta Methods

Discussion: Euler vs s-stage Runge-Kutta

Error e_m of Runge-Kutta methods converges as $e_m = \mathcal{O}\big(m^{-p}\big)$ for some $p > 1$ compared to $e_m - \mathcal{O}\big(m^{-1}\big)$ for Euler.

However, this faster convergence comes at the price of more computations per step.

Can we design methods which are as cheap as Euler and as fast as RK?

Observation

RK methods sample $f(y)$ at a number of points $y \in [y(0), y(t)]$ to learn about higher-order derivatives of f and then use this knowledge to predict $y(t)$.

However, for all time steps t_k other than the first, we already have samples $f\big(\tilde{y}(t_k)\big)$.

Multistep methods use these samples to predict $\tilde{y}(t_{k+1})$.

More details on next slide.

Implicit Runge-Kutta Methods

Terminology

Runge-Kutta methods are also called *single-step methods* because they only look at a single "step" $y_k := y(t_k)$ to predict y_{k+1}.
Multistep methods are called multi-step because they look at several steps $y_{k-n}, \ldots, y_k$ to predict y_{k+1}.

Explicit multistep methods (Adams-Bashford)

$$\tilde{y}_{k+1} = \tilde{y}_k + \Delta t \left(c_0\, f(\tilde{y}_k) + \ldots + c_n\, f(\tilde{y}_{k-n}) \right).$$

Implicit multistep methods (Adams-Multon)

$$\tilde{y}_{k+1} = \tilde{y}_k + \Delta t \left(c_{-1}\, f(\tilde{y}_{k+1}) + c_0\, f(\tilde{y}_k) + \ldots + c_n\, f(\tilde{y}_{k-n}) \right).$$

Implicit Runge-Kutta Methods

Discussion

- Theory of multistep methods is similar to theory for RK methods.
- Multistep methods are useful if evaluation of $f(y)$ is expensive. However, even in that case they are at most a constant factor faster than Runge-Kutta methods.
- Multistep methods require another method to compute $\tilde{y}_1, \dots, \tilde{y}_n$.

References and further reading

- `DifferentialEquations.jl` (http://docs.juliadiffeq.org/)
  Very powerful library of ODE solvers. Looking through the documentation gives a good idea of what people use in practice.

  Off-topic, but fascinating: http://tutorials.juliadiffeq.org/html/type_handling/02-uncertainties.html

- E. Suli and D. F. Mayers. *An Introduction to Numerical Analysis.* Cambridge University Press (2003), doi:10.1017/CBO9780511801181
  Can be accessed online for free via the library website!