SF2520 — Applied numerical methods

Lecture 4

Runge–Kutta methods
Multistep methods

Olof Runborg
Numerical analysis
Department of Mathematics, KTH

2023-09-11

Today's lecture

- Summary of last lecture
- High order methods
  - Runge–Kutta methods
  - Multi-step methods

Absolute stability

- An absolutely stable numerical solution satisfies

$$|\tilde{u}_n - u_n| \text{ is bounded for all } n, \text{ if } |\tilde{u}_0 - u_0| \text{ is small enough.}$$

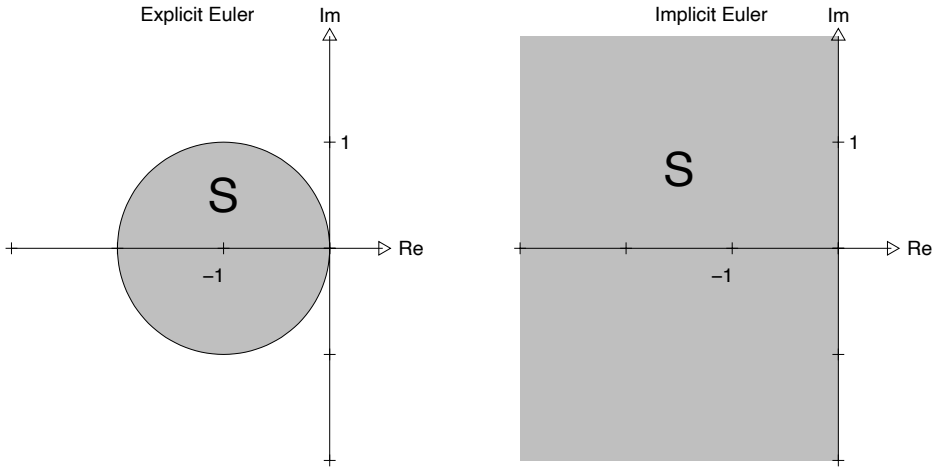
- Absolute stability necessary in practice, when exact solution is stable. Ensures that the effect of small perturbations remains small. (Important as small errors/perturbations are always introduced.)
- An absolutely stable solution behaves qualitatively correct.
- The fact that a method is absolutely stable for a certain time step does not mean that the solution is accurate. The time step must be chosen such that the method is both stable and accurate.

Stability regions

The region of absolute stability $\mathcal{S}$ for a method is defined via the "test problem"

$$y' = \lambda y, \qquad y(0) = y_0, \qquad \text{Real}(\lambda) \leq 0. \tag{1}$$

- Let w_n be a numerical solution to (1). Then $\mathcal{S}$ are the values $h\lambda \in \mathbb{C}$ for which w_n is bounded as $n \to \infty$.
- Every method has its own stability region $\mathcal{S}$.



- $\mathcal{S}$ characterizes the absolute stability also for general ODEs:

ODE	Stability condition
$y' = \lambda y + g$	$h\lambda \in \mathcal{S}$
$\boldsymbol{y}' = A\boldsymbol{y} + \boldsymbol{g}(t)$	$h\lambda_k \in \mathcal{S}$ for all eigenvalues λ_k of A
$\boldsymbol{y}' = A(t)\boldsymbol{y} + \boldsymbol{g}(t)$	$h\lambda_k(t) \in \mathcal{S}$ for all t, k
$\boldsymbol{y}' = \boldsymbol{F}(\boldsymbol{y})$	$h\lambda_k(\boldsymbol{u}_n) \in \mathcal{S}$ for all eigenvalues $\lambda_k(\boldsymbol{u}_n)$ of $J(\boldsymbol{u}_n)$, the Jacobian matrix of $\boldsymbol{F}$

Stiff ODE and implicit methods

- Explicit methods typically have a *bounded* $\mathcal{S}$ (with the origin on its boundary). This gives a stability limit for h. Since λ is fixed, we must have $h \leq h_{\text{stab}}$.
- Implicit methods typically have an *unbounded* $\mathcal{S}$. Then there is no stability limit. The method is "unconditionally stable".
  (There are examples of useful implicit methods with bounded $\mathcal{S}$.)
- In stiff problems explicit methods require much smaller timestep h than expected based on exact solution and LTE, because of stability requirement: $h \leq h_{\text{stab}} \ll h_{\text{acc}}$.
- Explicit methods become expensive as very many steps are needed, and "unnecessarily" accurate. Implicit methods advantageous.

Higher order methods

Explicit/Implicit Euler are the basic first order methods. To get higher order methods the local truncation error must be made smaller in terms of h.

Two ways to do this:

1. Make more function evaluations $f(t, y)$ in each step.
   This leads to Runge–Kutta methods.
   Example: Heun's method (2 evaluations, or "stages")

$$u_{n+1} = u_n + \frac{1}{2}h\Big(f(t_n, u_n) + f(t_{n+1}, u_n + hf(t_n, u_n))\Big).$$

2. Use several of the previously computed approximations.
   This leads to multi-step methods.
   Example: Midpoint method (2 previous values)

$$u_{n+1} = u_{n-1} + 2hf(t_n, u_n).$$

Runge–Kutta methods

Examples: (for $y' = f(t, y)$)

- Heun's method (2 stages, order 2)

$$\begin{aligned} k_1 &= f(t_n, u_n), \\ k_2 &= f(t_n + h, u_n + hk_1), \\ u_{n+1} &= u_n + h\left(\frac{1}{2}k_1 + \frac{1}{2}k_2\right). \end{aligned}$$

- RK4 (4 stages, order 4)

$$\begin{aligned} k_1 &= f(t_n, u_n), \\ k_2 &= f\left(t_n + \frac{1}{2}h, u_n + \frac{1}{2}hk_1\right), \\ k_3 &= f\left(t_n + \frac{1}{2}h, u_n + \frac{1}{2}hk_2\right), \\ k_4 &= f\left(t_n + h, u_n + hk_3\right), \\ u_{n+1} &= u_n + h\left(\frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4\right). \end{aligned}$$

Runge–Kutta methods

General form: (s stages, explicit RK)

$$
\begin{aligned}
k_1 &= f(t_n, u_n),\\
k_2 &= f(t_n + hc_2, u_n + ha_{21}k_1),\\
k_3 &= f(t_n + hc_3, u_n + ha_{31}k_1 + ha_{32}k_2),\\
&\vdots\\
k_s &= f(t_n + hc_s, u_n + ha_{s,1}k_1 + \cdots + ha_{s,s-1}k_{s-1}),\\
u_{n+1} &= u_n + h(b_1k_1 + b_2k_2 + \cdots + b_sk_s)
\end{aligned}
$$

Intuition: ($y' = f(t, y)$)

$$y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} y'(s)ds = y(t_n) + h\int_0^1 f(t_n + ht, y(t_n + ht))dt$$

- Approximate the integral by s-point quadrature method

$$y(t_{n+1}) \approx y(t_n) + h\sum_{j=1}^{s} b_j \underbrace{f(t_n + hc_j, y(t_n + hc_j))}_{\approx k_j}$$

- Approximate $y(t_n + hc_j)$ by an explicit method (note $k_j \approx y'(t_n + hc_j)$)

$$y(t_n + hc_j) \approx y(t_n) + ha_{j,1}k_1 + \cdots + ha_{j,j-1}k_{j-1}$$

Runge–Kutta methods

- Methods often summarized using "Butcher tableaus"

$$\begin{array}{c|cccc} c_1 & a_{11} & & & \\ c_2 & a_{21} & a_{22} & & \\ \vdots & \vdots & \vdots & \ddots & \\ c_s & a_{s,1} & a_{s,2} & \cdots & a_{s,s} \\ \hline & b_1 & b_2 & \cdots & b_s \end{array}$$

  where $a_{jj} = 0$ for explicit RK-methods, and $c_1 = 0$.

- Example

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

RK4: $\begin{array}{c|cccc} 0 & 0 & & & \\ \frac{1}{2} & \frac{1}{2} & 0 & & \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & \\ 1 & 0 & 0 & 1 & 0 \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$

Runge–Kutta methods – accuracy and convergence

- Runge–Kutta methods are one-step methods (although many *stages*). General theorem for one-step methods applies, so that if $\ell_n = O(h^{p+1})$ then the method is of order p.
- Accuracy determined both by the quadrature (b_j, c_j) and the explicit method $(c_j, a_{j,\ell})$.
- Finding $a_{j,\ell}$, b_j and c_j such that $\ell_n = O(h^{p+1})$ is possible using Taylor expansions, but complicated for p large.
- For $p \leq 4$ it is enough with $s = p$. Ex. Heun has $s = p = 2$ and RK4 has $s = p = 4$. (For higher order, s must be larger than p. Ex. $p = 5$ requires $s = 6$ and $p = 10$ requires $s = 17$.)
- For consistency, $\ell_n = O(h^2)$, one just needs

$$\sum_{j=1}^{s} b_j = 1.$$

- There are also implicit Runge–Kutta methods. Leads to a $d\,s \times d\,s$ nonlinear system of equations in each step. Then $p = 2s$ possible.

Example: Heun's method

- Apply to test problem $y' = \lambda y$, $y(0) = 1$,

$$k_1 = f(t_n, w_n) = \lambda w_n,$$

$$k_2 = f(t_n + h, w_n + hk_1) = \lambda(w_n + h\lambda w_n) = (\lambda + h\lambda^2)w_n,$$

$$w_{n+1} = w_n + h\left(\frac{1}{2}k_1 + \frac{1}{2}k_2\right) = \left(1 + h\lambda + \frac{1}{2}(h\lambda)^2\right) w_n.$$

- Hence

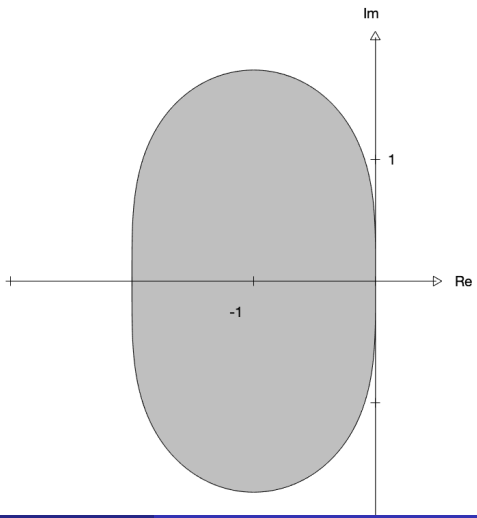
$$w_{n+1} = Q(h\lambda)w_n, \qquad Q(z) = 1 + z + \frac{1}{2}z^2.$$

- Since, by induction

$$w_n = Q(h\lambda)^n w_0,$$

we have

$$\mathcal{S} = \{z \in \mathbb{C} \ : \ |Q(z)| \leq 1\}.$$

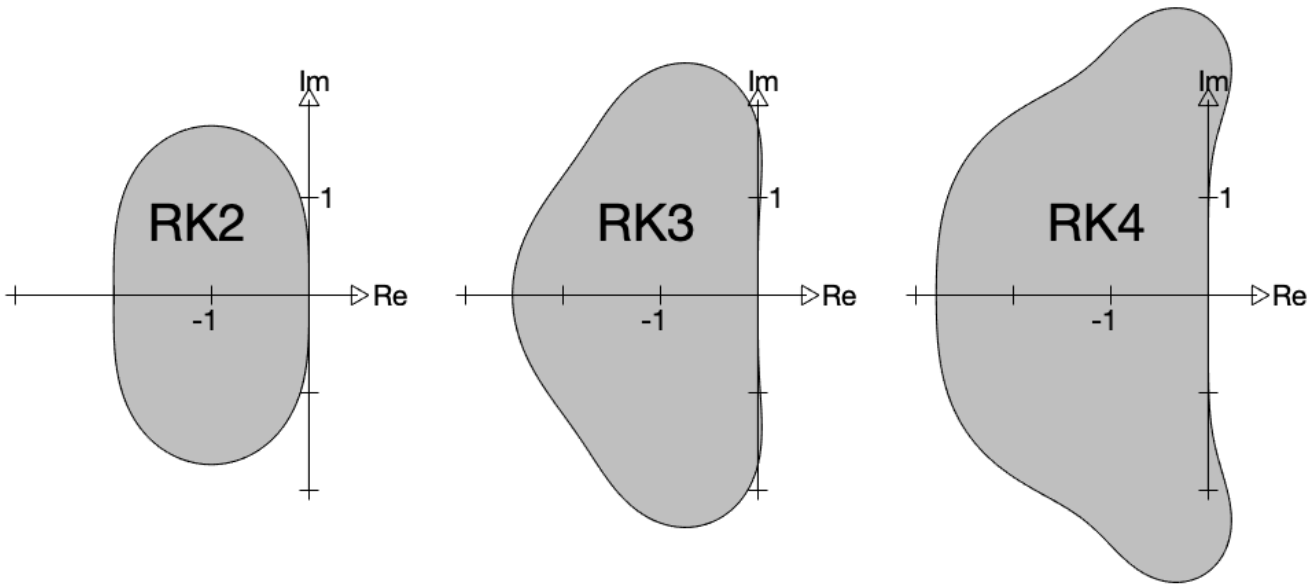


Runge–Kutta methods and absolute stability, cont.

- Applying an explicit R–K method with s stages to the test problem $y' = \lambda y$ always gives an iteration of the type

$$w_{n+1} = Q(h\lambda)w_n, \qquad \text{where } Q \text{ is a polynomial of degree } s.$$

- Stability region given by $\mathcal{S} = \{z \in \mathbb{C} \ : \ |Q(z)| \leq 1\}$.
  Note: Q polynomial $\Rightarrow$ Q unbounded $\Rightarrow$ $\mathcal{S}$ bounded



- In a p-th order method, $Q(z) = e^z + O(z^{p+1})$, since

$$y(t_{n+1}) = e^{h\lambda}y(t_n) \qquad \text{and} \qquad y(t_{n+1}) = Q(h\lambda)y(t_n) + \ell_{n+1}.$$

Adaptive Runge–Kutta methods

- In adaptive Runge–Kutta methods, the stages chosen such that different quadratures (sets of b_j) give different order of accuracy (= embedded methods). (Requires few extra function evaluations.)
- **Example: Dormand–Prince** (as used in Matlab's `ode45` command)
  - Based on a Runge–Kutta-metod with 7 stages.
  - First 6 stages give a 5th order approximation:

$$u_{n+1,5} = u_n + h\left(\frac{35}{384}k_1 + \frac{500}{1113}k_3 + \frac{125}{192}k_4 - \frac{2187}{6784}k_5 + \frac{11}{84}k_6\right).$$

  - Then let $k_7 = f(t_n + h, u_{n+1,5})$. (Note: same as k_1 in next step!)
  - With k_7 a 4th order approximation is given as

$$u_{n+1,4} = u_n + h\left(\frac{5179}{57600}k_1 + \frac{7571}{16695}k_3 + \frac{393}{640}k_4 - \frac{92097}{339200}k_5 + \frac{187}{2100}k_6 + \frac{1}{40}k_7\right)$$

  - Local truncation error estimated by difference $|u_{n+1,5} - u_{n+1,4}|$.
  - Requires 6 function evaluations per step. (Optimal for $p = 5$.)

Linear multistep methods

Example: Adams–Bashforth 3 (for $y' = f(t, y)$)

$$u_{n+1} = u_n + \frac{h}{12}(23f_n - 16f_{n-1} + 5f_{n-2}), \qquad f_n = f(t_n, u_n).$$

Uses 3 old values to compute the next. Need u_0, u_1, u_2 to start.

- **General form:** (q steps)

$$\begin{aligned} u_{n+1} = {} & \alpha_0 u_n + \alpha_1 u_{n-1} + \cdots + \alpha_{q-1} u_{n-q+1} \\ & + h\left(\beta_{-1} f_{n+1} + \beta_0 f_n + \cdots + \beta_{q-1} f_{n-q+1}\right) \end{aligned}$$

- Initial data needed for u_0, u_1, …, u_{q-1}. Typically obtained from explicit one-step method (e.g. R–K methods).
- If $\beta_{-1} = 0$ the method is explicit, otherwise implicit.
- Very high order of accuracy can be reached in a stable way.
- Multistep methods only need one new function evaluation/time step, but need more memory since old values must be stored.
- Common families of methods: Adams and BDF.

Adams methods

- Adams methods only use previous values of f, (not u)

$$u_{n+1} = u_n + h\left(\beta_{-1} f_{n+1} + \beta_0 f_n + \cdots + \beta_{q-1} f_{n+1-q}\right).$$

- Explicit ones called Adams–Bashforth (order q).
  Implicit ones called Adams–Moulton (order $q+1$).
- Derivation (Adams–Bashforth):
  - Exact solution satisfies

$$y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} y'(t)dt = y(t_n) + \int_{t_n}^{t_{n+1}} f(y(t))dt.$$

  - How to compute integral? Note $f(y(t))$ only approximated in the previous time points t_{n-q+1}, t_{n-q},..., t_n.
  - Let p be $q-1$ degree polynomial that approximates $p(t) \approx f(y(t))$ by interpolation of f_n in past points

$$p(t_{n-k}) = f_{n-k}, \qquad k = 0, \ldots, q-1.$$

  - Then set

$$u_{n+1} = u_n + \int_{t_n}^{t_{n+1}} p(t)dt =: u_n + h\left(\beta_0 f_n + \cdots + \beta_{q-1} f_{n-q+1}\right).$$

Backward Differentiation Formulae (BDF)

- BDF methods only use previous values of u, (not f)

$$u_{n+1} = \alpha_0 u_n + \alpha_1 u_{n-1} + \cdots + \alpha_{q-1} u_{n-q+1} + h\beta_{-1} f_{n+1}.$$

- Implicit q-step methods of order q.
- Often used for stiff ODEs (e.g. parabolic PDEs).
- Derivation
  - Let p be a degree q polynomial interpolating the *solution* u_n in the next and past points,

$$p(t_{n+1-k}) = u_{n+1-k}, \qquad k = 0, \ldots, q+1.$$

  - Find u_{n+1} such that $p'(t_{n+1}) = f(u_{n+1})$ $(= f_{n+1})$.
  - Leads to

$$u_{n+1} = \alpha_0 u_n + \alpha_1 u_{n-1} + \cdots + \alpha_{q-1} u_{n-q+1} + h\beta_{-1} f_{n+1},$$

    for some α_j and β_{-1}.
- **Example: BDF 2**

$$u_{n+1} = \frac{4}{3} u_n - \frac{1}{3} u_{n-1} + h\frac{2}{3} f_{n+1}.$$

Linear multistep methods, summary of methods.

- Adams–Bashforth (explicit, order q)

$$u_{n+1} = u_n + h\left(\beta_0 f_n + \cdots + \beta_{q-1} f_{n-q+1}\right).$$

  - **Example: Adams–Bashforth 3** (order 3)

$$u_{n+1} = u_n + \frac{h}{12}(23f_n - 16f_{n-1} + 5f_{n-2}).$$

- Adams–Moulton (implicit, order $q+1$)

$$u_{n+1} = u_n + h\left(\beta_{-1} f_{n+1} + \beta_0 f_n + \cdots + \beta_{q-1} f_{n-q+1}\right).$$

  - **Example: Adams–Moulton 3** (order 4)

$$u_{n+1} = u_n + \frac{h}{24}(9f_{n+1} + 19f_n - 5f_{n-1} + f_{n-2}).$$

- BDF (implicit order q)

$$u_{n+1} = \alpha_0 u_n + \alpha_1 u_{n-1} + \cdots + \alpha_{q-1} u_{n-q+1} + h\beta_{-1} f_{n+1}.$$

  - **Example: BDF 2** (order 2)

$$u_{n+1} = \frac{4}{3}u_n - \frac{1}{3}u_{n-1} + h\frac{2}{3}f_{n+1}.$$

Linear multistep methods, accuracy and convergence

- Local truncation error ℓ_n as before ($y_n = y(t_n)$ is exact solution)

$$y_{n+1} = \alpha_0 y_n + \alpha_1 y_{n-1} + \cdots + \alpha_{q-1} y_{n-q+1} + h\left(\beta_{-1} f(y_{n+1}) + \beta_0 f(y_n) + \cdots + \beta_{q-1} f(y_{n-q+1})\right) + \ell_{n+1},$$

- Also define the polynomial

$$\rho(z) := \alpha_{q-1} + \alpha_{q-2} z + \cdots + \alpha_0 z^{q-1} - z^q.$$

Theorem (Dahlquist – global error)

Suppose f is Lipschitz and $\ell_n = O(h^{p+1})$. If, in addition, the initial data error in $u_0, \ldots, u_{q-1}$ is $O(h^p)$ and ρ satisfies the root condition:

All roots z_j of ρ satisfies either $|z_j| < 1$ or, if z_j is a simple root, $|z_j| \leq 1$,

then

$$\max_{0 \leq n \leq N} |y_n - u_n| \leq C h^p, \qquad \text{(Nh = T fixed, and C independent of h).}$$

- Note: For one-step methods $\rho(z) = 1 - z$ (since $\alpha_0 = 1$ then).

Linear multistep methods, remarks.

Remarks

- Convergence theory more complicated than for one-step methods. Consistent methods ($\ell_n = O(h^2)$) may fail to converge.
- Root condition needed to guarantee convergence (aka as "zero stability"). (Automatically satisfied for one-step methods.)
- Adams–Bashforth and Adams–Moulton are convergent for all q. (They have $\rho(z) = z^{q-1} - z^q$.)
- BDF are only convergent for $q \leq 6$.
- Shortcut available to determine order. Introduce polynomial

$$\sigma(z) := \beta_{q-1} + \beta_{q-2} z + \cdots + \beta_0 z^q + \beta_{-1} z^{q+1}.$$

  Then $\ell_n = O(h^{p+1})$ iff $\rho(e^h) + h\sigma(e^h) = O(h^{p+1})$.
- Adaptivity more difficult since changes in time step will change coefficients α_j and β_j.

Linear multistep methods and absolute stability

- Absolute stability analysis leads to study of difference equations.
- For a general multistep method stability governed by the polynomial

$$P(z) = \rho(z) + h\lambda\sigma(z),$$

with ρ, σ defined above.

- Let $z_j(h\lambda)$ be the roots of P. Absolute stability region $\mathcal{S}$ consists of those values $h\lambda \in \mathbb{C}^-$ for which

$$|z_j(h\lambda)| < 1, \qquad j = 1, \ldots, q.$$

(Or $|z_j(h\lambda)| = 1$ if root is simple.)

- Note that the root condition is the same condition for ρ. This means that root condition (zero stability) equivalent to $0 \in \mathcal{S}$.

Linear multistep methods and absolute stability

- **Example:** Applying AB3 to test problem $y' = \lambda y$ gives

$$w_{n+1} = w_n + \frac{h}{12}(23\lambda w_n - 16\lambda w_{n-1} + 5\lambda w_{n-2}),$$

  which can be rewritten as

$$w_{n+1} + r_0(h\lambda)w_n + r_1(h\lambda)w_{n-1} + r_2(h\lambda)w_{n-2} = 0,$$

  where $r_0(s) = -(1 + \beta_0 s)$ and for $j \geq 1$ we have $r_j(s) = -\beta_j s$.
- For fixed $h\lambda$ this is a difference equation, whose solution is determined by its characteristic polynomial P,

$$P(z) = z^3 + r_0 z^2 + r_1 z + r_2.$$

- If z_1, z_2 and z_3 are simple roots of P, then

$$w_n = c_1 z_1^n + c_2 z_2^n + c_3 z_3^n,$$

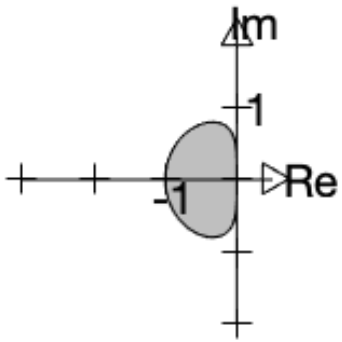
  where c_j are determined by initial data. (See Appendix A.2 in Edsberg.)
- Absolute stability requires $|z_j| < 1$ for all j. Then $|w_n| \to 0$.
- Note: r_j, c_j and w_j all depend on $h\lambda$. Also note that

$$P(z) = z^3 - z^2 - \beta_0 h\lambda z^2 - \beta_1 h\lambda z - \beta_2 h\lambda = \rho(z) - h\lambda\sigma(z).$$

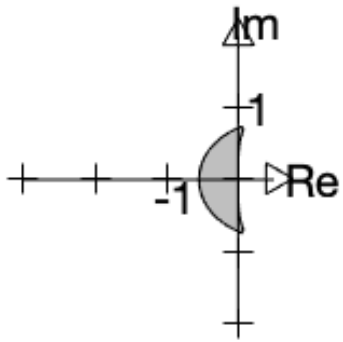
Linear multistep methods and absolute stability

Stability regions, Adams methods

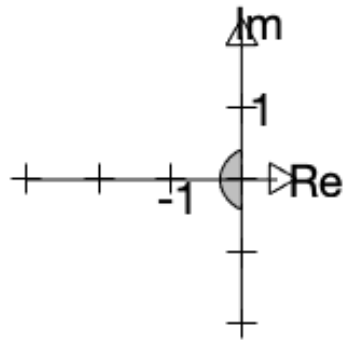
AB2



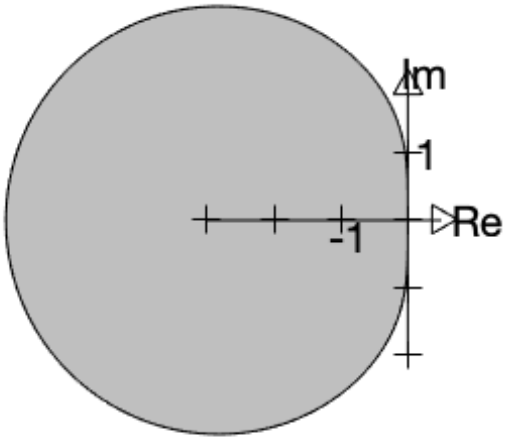
AB3



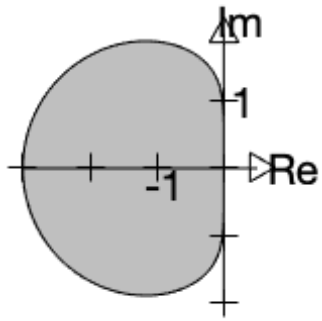
AB4



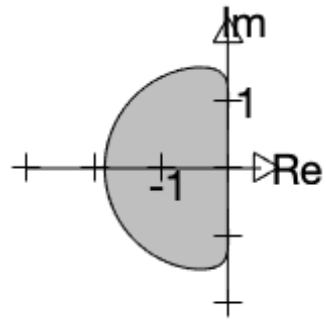
AM2



AM3



AM4



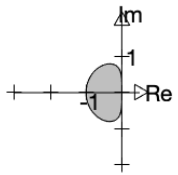
Linear multistep methods and absolute stability

- Stability regions of the implicit Adams–Moulton methods are *bounded* but much larger than those of Adams–Bashforth.
- Adams–Bashforth and Adams–Moulton usually used together in "predictor-corrector" pair: The u_{n+1} value is predicted with explicit AB and corrected with AM. (Fixed point iteration possible.) Used for non-stiff problems to increase stability region of AB.

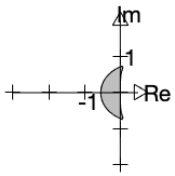
Linear multistep methods and absolute stability

Stability regions, predictor corrector

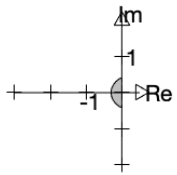
AB2



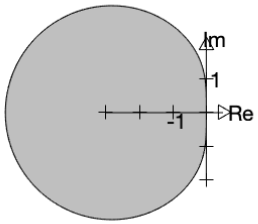
AB3



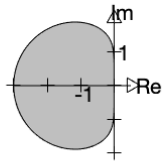
AB4



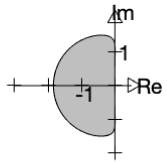
AM2



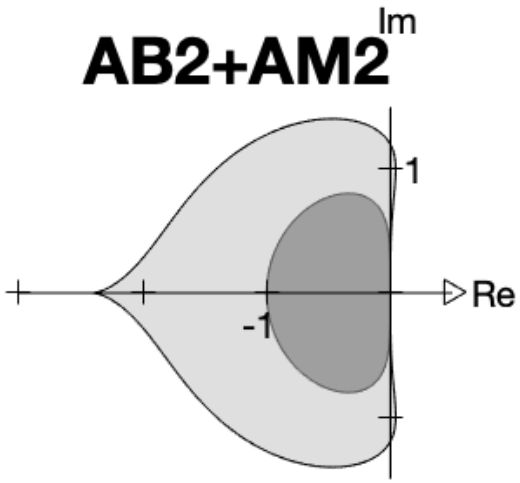
AM3



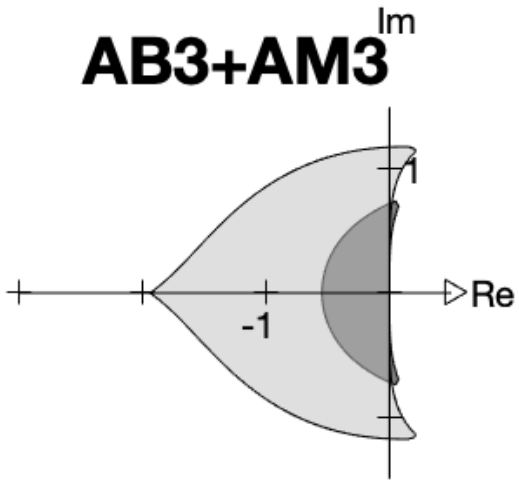
AM4



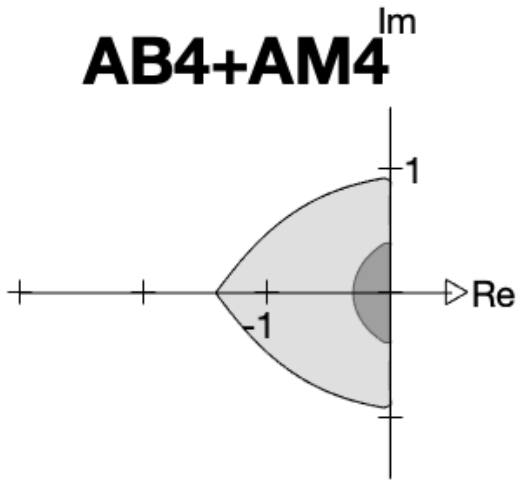
AB2+AM2



AB3+AM3



AB4+AM4



Linear multistep methods and absolute stability

Stability regions, BDF