Chapter 11

Interlude: Linear Multistep Methods

Main concepts: Linear multistep methods, accuracy, root condition, stability.

In this chapter we introduce one of the two main classes of numerical integrators for dynamical systems: the linear multistep methods. The chapter contains an overview of the analytical issues relevant to these methods. In subsequent chapters, we will address the other major class: one step methods.

Assuming the solution has been computed for some number of time steps, the main idea of multistep methods is to use the last k step values to approximate the solution at the next step. A linear k-step method is defined as

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h \sum_{j=0}^{k} \beta_j f(y_{n+j}). \tag{11.1}$$

For a linear k-step method, we require that $\alpha_k \neq 0$ and either $\alpha_0 \neq 0$ or $\beta_0 \neq 0$. Furthermore, the coefficients in (11.1) are not uniquely defined, since multiplication through by a constant defines the same method. Usually the coefficients are normalized such that either $\alpha_k = 1$, or $\sum_j \beta_j = 1$.

When using (11.1) in a computer code, at time step $n+k-1$ it is assumed that the values y_{n+j}, $j = 0, \ldots, k-1$ are already computed, so y_{n+k} is the only unknown in the formula. If β_k is nonzero the method is implicit, otherwise it is explicit. Since the initial value problem (1.3) only specifies $y_0 = y(t_0)$, it is necessary to first generate data y_j, $j = 1, \ldots, k-1$ before the formula (11.1) can be applied. This is done, for example, by using forward Euler or another one-step method in the first $k-1$ steps.

11.1 Examples

Some examples of linear multistep methods are:

- The θ-method generalizes all linear one-step methods

$$y_{n+1} - y_n = h(1-\theta) f(y_n) + h\theta f(y_{n+1}). \tag{11.2}$$

  Here we have $\alpha_0 = -1$, $\alpha_1 = 1$, $\beta_0 = 1-\theta$ and $\beta_1 = \theta$. Important methods are forward Euler ($\theta = 0$), backward Euler ($\theta = 1$) and trapezoidal rule ($\theta = 1/2$). For any $\theta > 0$, this method is implicit.

- *Leapfrog* is an explicit two-step method ($k = 2$) given by $\alpha_0 = -1$, $\alpha_1 = 0$, $\alpha_2 = 1$ and $\beta_1 = 2$:

$$y_{n+2} - y_n = 2hf(y_{n+1}) \tag{11.3}$$

- The class of *Adams methods* have $\alpha_k = 1$, $\alpha_{k-1} = -1$ and $\alpha_j = 0$ for $j < k-1$. *Adams-Bashforth methods* are explicit, additionally satisfying $\beta_k = 0$. Examples of 1, 2 and 3-step methods are (using notation $f_n \equiv f(y_n)$):

$$y_{n+1} - y_n = hf_n \tag{11.4}$$

$$y_{n+2} - y_{n+1} = h\left(\frac{3}{2}f_{n+1} - \frac{1}{2}f_n\right) \tag{11.5}$$

$$y_{n+3} - y_{n+2} = h\left(\frac{23}{12}f_{n+2} - \frac{4}{3}f_{n+1} + \frac{5}{12}f_n\right) \tag{11.6}$$

  Adams-Moulton methods are implicit, with $\beta_k \neq 0$.

- The *Backward differentiation formulae (BDF)* are a class of linear multistep methods satisfying $\beta_j = 0$, $j < k$ and generalizing backward Euler. The two-step method (BDF-2) is

$$y_{n+2} - \frac{4}{3}y_{n+1} + \frac{1}{3}y_n = h\frac{2}{3}f(y_{n+2}). \tag{11.7}$$

11.2 Order of accuracy and convergence

Associated with the linear multistep method (11.1) are the polynomials

$$\rho(\zeta) = \sum_{j=0}^{k} \alpha_j \zeta^j, \qquad \sigma(\zeta) = \sum_{j=0}^{k} \beta_j \zeta^j. \tag{11.8}$$

These are important for understanding the dynamics of multistep methods, and will be used later.

The *residual* of a linear multistep method at time t_{n+k} may be defined in a number of ways. We obtain it by substituting the exact solution $y(t)$ of (1.3) at times $y(t_{n+j})$, $j = 0, \ldots, k$ into (11.1), i.e.

$$r_n := \sum_{j=0}^{k} \alpha_j y(t_{n+j}) - h\sum_{j=0}^{k} \beta_j y'(t_{n+j}). \tag{11.9}$$

(This is actually the residual accumulated in the $(n+k-1)$th step, but for notational convenience we will denote it r_n.) A linear multistep method has maximal *order of accuracy* p if $r_n = \mathcal{O}(h^{p+1})$ for all sufficiently smooth f.

Write the Taylor series expansions of $y(t_{n+j})$ and $y'(t_{n+j})$ as

$$y(t_{n+j}) = \sum_{i=0}^{\infty} \frac{(jh)^i}{i!} y^{(i)}(t_n), \qquad y'(t_{n+j}) = \sum_{i=0}^{\infty} \frac{(jh)^i}{i!} y^{(i+1)}(t_n),$$

where in this chapter $y^{(i)}(t)$ means the ith derivative of $y(t)$. Substituting these into (11.9) and manipulating,

$$\begin{aligned} r_n &= \sum_{j=0}^{k} \alpha_j \sum_{i=0}^{\infty} \frac{(jh)^i}{i!} y^{(i)}(t_n) - h\sum_{j=0}^{k} \beta_j \sum_{i=0}^{\infty} \frac{(jh)^i}{i!} y^{(i+1)}(t_n) \\ &= \sum_{j=0}^{k} \alpha_j y(t_n) + \sum_{i=1}^{\infty}\sum_{j=0}^{k} \alpha_j \frac{(jh)^i}{i!} y^{(i)}(t_n) - \sum_{i=1}^{\infty}\sum_{j=0}^{k} \beta_j \frac{(jh)^i}{j(i-1)!} y^{(i)}(t_n) \\ &= \sum_{j=0}^{k} \alpha_j y(t_n) + \sum_{i=1}^{\infty} \frac{1}{i!} h^i y^{(i)}(t_n) \left[\sum_{j=0}^{k} \alpha_j j^i - i\sum_{j=0}^{k} \beta_j j^{i-1}\right]. \end{aligned}$$

Equivalent conditions for a linear multistep method to have order of accuracy p are:

- The coefficients α_j and β_j satisfy (where $0^0 = 1$)

$$\sum_{j=0}^{k} \alpha_j = 0 \qquad \text{and} \qquad \sum_{j=0}^{k} \alpha_j j^i = i \sum_{j=0}^{k} \beta_j j^{i-1} \quad \text{for} \quad i = 1, \ldots, p. \tag{11.10}$$

- The polynomials $\rho(\zeta)$ and $\sigma(\zeta)$ satisfy

$$\rho(e^z) - z\sigma(e^z) = \mathcal{O}(z^{p+1}). \tag{11.11}$$

- The polynomials $\rho(\zeta)$ and $\sigma(\zeta)$ satisfy

$$\frac{\rho(z)}{\log z} - \sigma(z) = \mathcal{O}((z-1)^p). \tag{11.12}$$

The first of these follows from the considerations above. For proofs of the second and third forms, see Hairer, Nørsett and Wanner (1993).
Examples. The method (11.3) has order 2. The methods (11.5) and (11.6) have orders 2 and 3, respectively. The method (11.7) has order 2.

11.2.1 The root condition, a counter-example

Earlier we saw that for Euler's method, convergence follows from the fact that the residual is $\mathcal{O}(h^2)$, leading to a global error of $\mathcal{O}(h)$ on a fixed interval. For linear multistep methods, first order accuracy is insufficient to ensure convergence. We will not prove convergence for this class of methods, but will simply state the convergence theorem and show where it can go wrong.

The method (11.1) is said to satisfy the *root condition*, if all roots ζ of

$$\rho(\zeta) = 0,$$

lie on the unit disc ($|\zeta| \leq 1$), and any root of modulus one ($|\zeta| = 1$) has multiplicity one.

Furthermore, a linear multistep method is incomplete without a starting procedure to generate the first $k-1$ iterates $y_1, \ldots, y_{k-1}$.

Theorem 11.2.1 *Suppose a linear multistep method (11.1) is equipped with a starting procedure satisfying* $\lim_{h\to 0} y_j = y(t_0 + jh)$ *for* $j = 1, \ldots, k-1$. *Then the method converges to the exact solution of (1.3) on a fixed interval as* $h \to 0$ *if and only if it has order of accuracy* $p \geq 1$ *and satisfies the root condition.*

The proof of this theorem will not be handled in these notes. See, e.g., the monograph of Hairer, Nørsett, & Wanner (1993).

To illustrate the necessity of the root condition, consider the method

$$y_{n+3} + y_{n+2} - y_{n+1} - y_n = h\left(\frac{8}{3}f(y_{n+2}) + \frac{2}{3}f(y_{n+1}) + \frac{2}{3}f(y_n)\right). \tag{11.13}$$

Substituting $\rho(\zeta) = \zeta^3 + \zeta^2 - \zeta - 1$ and $\sigma(\zeta) = \frac{8}{3}\zeta^2 + \frac{2}{3}\zeta + \frac{2}{3}$ into (11.11) gives

$$\rho(e^z) - z\sigma(e^z) = \frac{1}{3}z^4 + \mathcal{O}(z^5),$$

so the method is third order accurate. Now applying the method to the easiest of all initial value problems

$$y' = 0, \quad y(0) = 1, \quad t \geq 0$$

yields the linear difference equation

$$y_{n+3} + y_{n+2} - y_{n+1} - y_n = 0. \tag{11.14}$$

If the roots ζ_1, ζ_2 and ζ_3 of the characteristic polynomial $\rho(\zeta) = 0$ were distinct, the exact solution of such a recursion would be

$$y_n = c_1\zeta_1^n + c_2\zeta_2^n + c_3\zeta_3^n.$$

If any root ζ_i had modulus greater than 1, then the recursion would satisfy $|y_n| \to \infty$ unless the corresponding constant c_i were identically 0. For the current case, $\rho(\zeta) = (\zeta-1)(\zeta+1)^2$, and there is a double root at -1. For a double root $\zeta_3 \equiv \zeta_2$, the solution of the recursion (11.14) becomes

$$y_n = c_1\zeta_1^n + c_2\zeta_2^n + c_3 n\zeta_2^n.$$

One still has $|y_n| \to \infty$ (but with linear growth), unless $c_3 = 0$.

The constants c_1, c_2 and c_3 are determined by the initial conditions necessary to start the multistep method. Suppose we take $y_0 = y_1 = y_2 = 1$, consistent with the exact solution. Then this yields

$$c_1 = 1, \quad c_2 = c_3 = 0,$$

and the solution is $y_n = 1$, for all n. The method is exact.

Suppose, however, that the initial conditions are perturbed slightly. We take $y_0 = 1 + \varepsilon$, $y_1 = y_2 = 1$. Then we find

$$c_1 = 1 + \frac{3}{4}\varepsilon, \quad c_2 = \frac{1}{4}\varepsilon, \quad c_3 = -\frac{1}{2}\varepsilon,$$

and the solution is unbounded. The numerical solution sequence is unstable to perturbations in the initial conditions. As a consequence, any errors incurred will destabilize the solution. The method works only for the trivial differential equation, $y' = 0$, and then only if the starting procedure is exact. It fails to converge for all other differential equations.

As an example we compute the solution of $y' = -y$, $y(0) = 1$, $t \in [0, 1.5]$ using (11.13) for $h = 1/10$, $1/100$ and $1/1000$. The instability gets *worse* as h decreases.

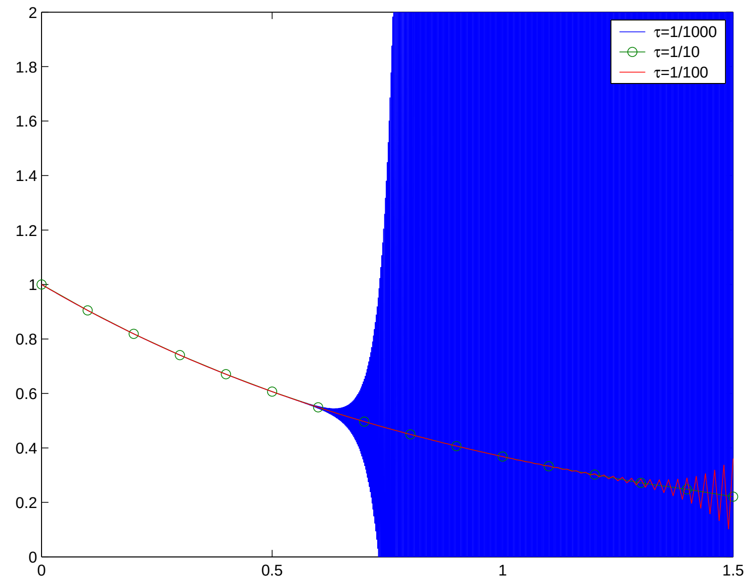

Figure 11.1: Solution of $y' = -y$, $y(0) = 1$ with (11.13) for various h.

Theorem 11.2.2 *The maximum order of a k-step method satisfying the root condition is $p = k$ for explicit methods and, for implicit methods, $p = k + 1$ for odd k and $p = k + 2$ for even k.*

11.3 Stability

As we will see later in the course, an important criterion for distinguishing between different methods is their ability to preserve the stability of a stable equilibrium. To test this, we check under what conditions the numerical solution converges to zero when we apply (11.1) to the scalar linear test problem $y' = \lambda y$:

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h\lambda \sum_{j=0}^{k} \beta_j y_{n+j}.$$

Letting $z = h\lambda$ we write

$$\sum_{j=0}^{k} (\alpha_j - z\beta_j) y_{n+j} = 0.$$

For any z this is a linear difference equation with characteristic polynomial

$$\sum_{j=0}^{k} (\alpha_j - z\beta_j)\zeta^j = 0 = \rho(\zeta) - z\sigma(\zeta).$$

The *stability region* $\mathcal{S}$ of a linear multistep method is the set of all points $z \in \mathbb{C}$ such that all roots ζ of the polynomial equation $\rho(\zeta) - z\sigma(\zeta) = 0$ lie on the unit disc $|\zeta| \leq 1$, and those with modulus one are simple.

On the boundary of the stability region $\mathcal{S}$, precisely one root has modulus one, say $\zeta = e^{\mathrm{i}\theta}$. Therefore an explicit representation for the boundary of $\mathcal{S}$ is easily derived:

$$\partial\mathcal{S} = \left\{ z = \frac{\rho(e^{\mathrm{i}\theta})}{\sigma(e^{\mathrm{i}\theta})},\ \theta[-\pi, \pi] \right\}.$$

Figure 11.2 shows plots of the stability regions for the Adams-Bashforth methods of orders $p = 1$, 2 and 3.

A linear multistep method is called *A-stable* (or *unconditionally stable*) if the stability domain $\mathcal{S}$ contains the entire left half-plane

$$\{z \in \mathbb{C} : \operatorname{Re} z \leq 0\} \subset \mathcal{S}.$$

Theorem 11.3.1 *An A-stable linear multistep method has order $p \leq 2$.*

This restriction on the maximum order of a linear multistep method was an important result in numerical analysis, proved by G. Dahlquist.

For stiff problems in which the stiff components have eigenvalues near the real axis, A-stability is too strong a requirement. Instead a weaker concept is introduced:

The linear multistep method (11.1) is A(α)-stable, for $\alpha \in (0, \pi/2)$, if the stability domain contains a wedge in the left half-plane:

$$\{z \in \mathbb{C} : |\arg(z) - \pi| < \alpha\} \subset \mathcal{S}.$$

The important point is that for λ lying within the wedge of stability, the method is unconditionally stable (norm-nonincreasing for any h).

Figure 11.3 shows plots of the stability regions for the Backward differentiation formulae (BDF) of orders $p = 1, \ldots, 6$. The first and second order methods are A-stable. The rest are A(α)-stable with α (approximately):

Order, p	α
3	86.03°
4	73.35°
5	51.84°
6	17.84°

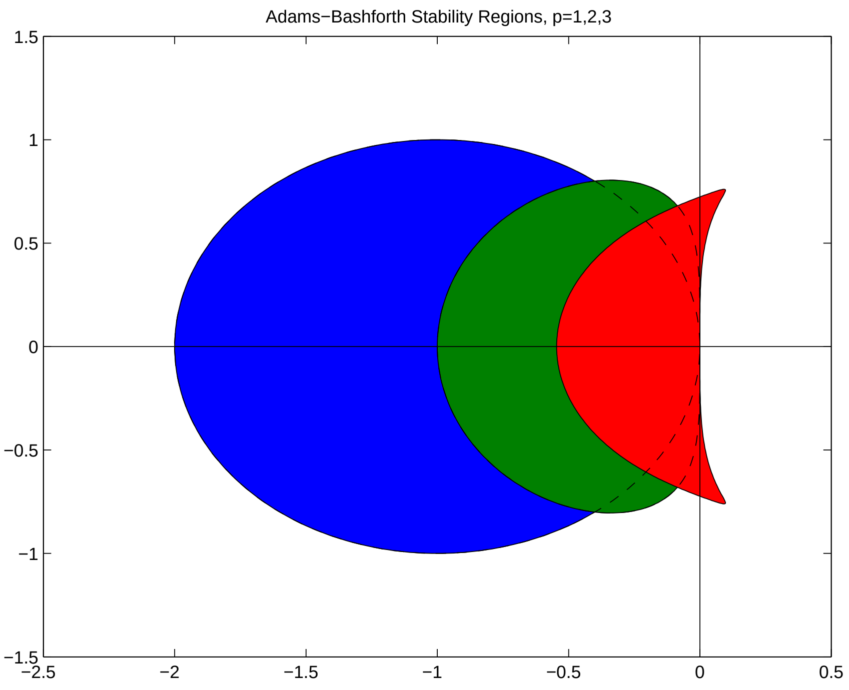

Figure 11.2: Stability regions of the Adams-Bashforth methods of orders $p = 1$ (blue), 2 (green) and 3 (red).

11.4 Implementation issues

11.4.1 Starting process.

As mentioned briefly in section 11.2, a linear k-step method needs a starting procedure to generate the first $k-1$ solution values. This is typically done using either $k-1$ steps of a one-step (Runge-Kutta) method, or applying successively ℓ-step methods with $\ell = 1, 2, \ldots$. For example, to start a 4-step Adams-Bashforth scheme, the methods (11.4), (11.5), (11.6) could be applied in succession, for $n = 0$. For problems with discontinuous f, it is necessary to apply the starting procedure after each discontinuity.

11.4.2 Variable stepsize implementation.

The order conditions (11.10)–(11.12) only hold for constant stepsize h. If the stepsize varies over the interval $[t_n, t_{n+k}]$, i.e. $h_j = t_{n+j+1} - t_{n+j}$, $j = 0, \ldots, k-1$, with distinct h_j, then the coefficients for a given order become dependent on the h_j. In general the stepsize ratio $q_j = h_{j+1}/h_j$ will be restricted to some range $0 < q_{\min} < q_j < q_{\max}$, by the root condition. There are various techniques for implementing variable stepsize multistep methods. For more information, see Hairer, Nørsett & Wanner (1993).

In most cases it is not desirable to store all solutions $y_0, \ldots, y_N$, as this eats up computer memory. Instead only the minimum number of ‘back-values’ necessary for the implementation is kept. The solution is written to disk as often as desired. For this reason, variable stepsize implementations of multistep methods are problematic. Either the coefficients have to be modified to suit the local stepsize schedule, or the past solution values have to be interpolated onto a uniform grid.

An exception is a variable stepsize implementation in which the stepsize is allowed to change only by a factor of 2. In this case, provided the stepsize is not allowed to double too often, one

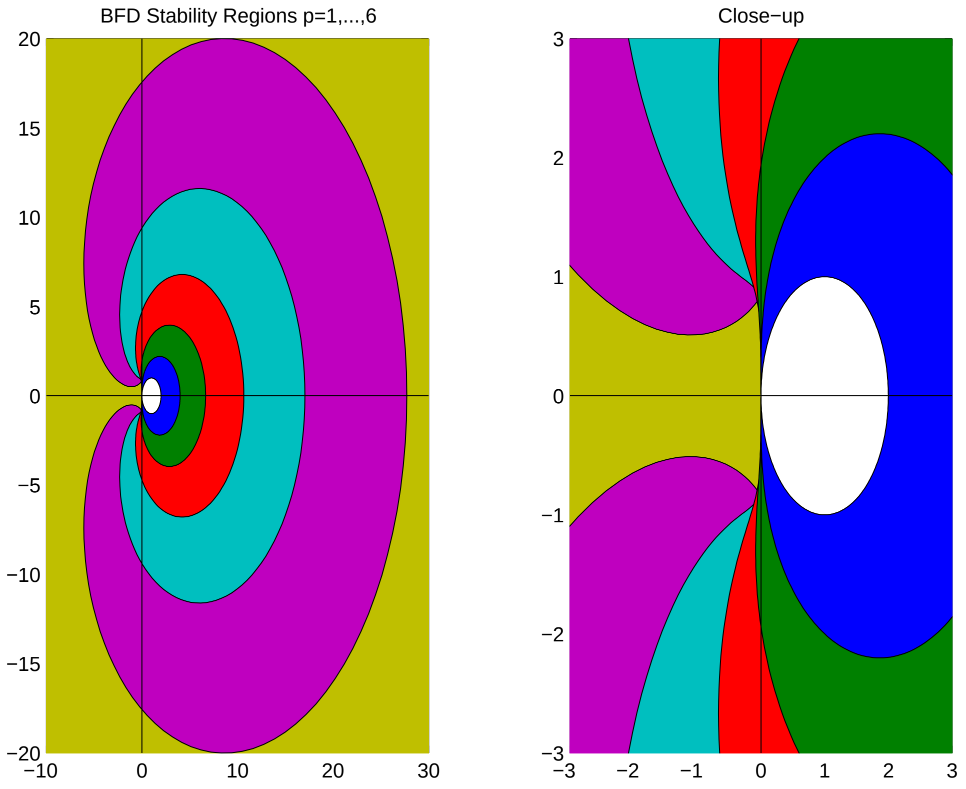

Figure 11.3: Stability regions of the BDF methods of orders $p = 1, \ldots, 6$. For BDF-1 (blue) $\mathcal{S}$ is everything outside the white region; for BDF-2 (green) it is everything outside the blue region; etc.

can get by with storing the $2k$ vectors $y_{n-k}, \ldots, y_{n+k-1}$. When the stepsize is doubled, the values $y_{n+k-2}, y_{n+k-4}, \ldots$ are used in formula (11.1) with h replaced by $2h$. When the stepsize is halved, a pth order interpolation can be used to obtain the vectors $y_{n+k-1/2}, y_{n+k-3/2}, \ldots$ needed to apply (11.1) with a stepsize of $h/2$.

11.4.3 Error estimation.

To estimate the error for the purpose of adaptivity, one technique makes use of a second linear multistep method, also of order p (but perhaps not a k-step method). Denote the coefficients of this process with $\tilde{\alpha}_j$ and $\tilde{\beta}_j$. According to formula (11.11), these methods must satisfy, assuming the values $y_n, \ldots, y_{n+k-1}$ are exact,

$$y(t_{n+k}) - y_{n+k} = ch^{p+1}y^{(p+1)}(t_{n+k}) + \mathcal{O}(h^{p+2}),$$
$$y(t_{n+k}) - \tilde{y}_{n+k} = \tilde{c}h^{p+1}y^{(p+1)}(t_{n+k}) + \mathcal{O}(h^{p+2}),$$

for appropriate $c, \tilde{c} > 0$. Assuming we have chosen the second linear multistep method such that $c \neq \tilde{c}$, we can combine the above and eliminate $y^{(p+1)}(t_{n+k})$ to get the local error estimate

$$y(t_{n+k}) - y_{n+k} = \frac{c}{c - \tilde{c}}(\tilde{y}_{n+k} - y_{n+k}) + \mathcal{O}(h^{p+2}).$$

For efficiency, it is best to use an explicit method such as an Adams-Bashforth for the error estimator $\tilde{y}_{n+k}$. Since this value is thrown away after estimating the error, it cannot alter stability to use an explicit method.

11.4.4 Nonlinear systems.

For implicit methods, the nonlinear system to be solved in each timestep is

$$\alpha_k y_{n+k} = h\beta_k f(y_{n+k}) + \left[\sum_{j=0}^{k-1} -\alpha_j y_{n+j} + h\beta_j f_{n+j}\right],$$

where the terms in brackets are explicitly known at time step $(n+k-1)$. This system has the same form as discussed Appendix A and is of dimension d. Furthermore, the back values $y_n, \ldots y_{n+k-1}$ can be used with a high order extrapolation formula to generate a good initial guess for (modified) Newton iteration, so the nonlinear solves are fairly efficient.

11.5 Exercises

1. Determine the stability region $\mathcal{S}$ for the Leapfrog method

$$y_{n+1} = y_{n-1} + 2hf(t_n, y_n).$$

2. Derive a 4th order Adams-Bashforth method using one of (11.10)–(11.12). Does it satisfy the root condition? Determine its stability region.

3. Consider the Lorenz equations:

$$\begin{aligned} y_1' &= \sigma(y_2 - y_1) \\ y_2' &= ry_1 - y_2 - y_1 y_3 \\ y_3' &= y_1 y_2 - by_3 \end{aligned}$$

Choose the parameters to be $\sigma = 10$, $b = 8/3$, $r = 28$ and the initial condition $y_0 = (0, 1, 0)^T$. Integrate this system on the interval $t \in [0, 100]$. Use the 4th order Adams-Bashforth method you derived in the previous problem. Start the integration with 3 steps of Euler's method.

   (a) Plot the solution in 3D using the built-in matlab function `plot3`. Use the rotation tool in the figure window to observe the Lorenz attractor.

   (b) Experiment with different stepsizes. Can you observe 4th order convergence? If not, try integrating over a shorter interval, (say $t \in [0, 10]$). **Hint:** It may be helpful to plot the time history of one component (say $y_2(t)$) of the solution, for several different stepsizes, on the same axes.

   (c) Repeat the computation using forward Euler on the whole interval $t \in [0, 100]$. Do you see any advantage to using the fourth order method for solving this problem on a long time interval?