DD2363/2023 – Lecture 9 Scalar initial value problems (ch.13)

Johan Hoffman

Intial value problem

Consider the following *ordinary differential equation* (ODE) for a scalar function $u : [0, T] \to R$, with derivative $\dot{u} = du/dt$,

$$\begin{aligned} \dot{u}(t) &= f(u(t), t), \quad 0 < t \leq T, \\ u(0) &= u_0, \end{aligned} \tag{13.1}$$

which we refer to as a *scalar initial value problem* (IVP), defined on the interval $I = [0, T]$ by the function $f : R \times R^+ \to R$, and the *initial condition* $u(0) = u_0$. Only in special cases can exact closed form solutions be found, instead approximation methods must be used in general.

Example 13.1. For $f(u(t), t) = \alpha u(t)$ and the initial condition $u(0) = 1$, the exact solution to the IVP (13.1) is the exponential function $u(t) = \exp(\alpha t)$, for all $\alpha \in R$ and $t > 0$.

Intial value problem

The variable $t \in [0, T]$ is often interpreted to be time, and numerical methods to solve the IVP (13.1) can be formulated based on the idea of *time stepping*, where successive approximations $U(t_n)$ are computed on a partition

$$0 = t_0 < t_1 < \ldots < t_N = T,$$

starting from $U(t_0) = u_0$. By interpolation over each subinterval, or *time step*, $I_n = [t_{n-1}, t_n]$ of length $k_n = t_n - t_{n-1}$, we construct an approximation $U(t)$ for any $t \in [0, T]$. To compute the solution at $t = t_n$, we can use a *forward difference approximation* of the derivative at $t = t_{n-1}$,

$$\dot{u}(t_{n-1}) \approx \frac{u(t_n) - u(t_{n-1})}{k_n},$$

$$u(t_n) \approx u(t_{n-1}) + k_n \dot{u}(t_{n-1}) = u(t_{n-1}) + k_n f(u(t_{n-1}), t_{n-1}).$$

Forward Euler method

This is the *forward Euler method* for successive approximation of $U_n = U(t_n)$, given by the update formula

$$U_n = U_{n-1} + k_n f(U_{n-1}, t_{n-1}).$$

The forward Euler method is *explicit*, meaning that U_n is directly computable from the previous approximation U_{n-1} in the time stepping algorithm. Therefore, the method is also referred to as the *explicit Euler method*.

Forward Euler method

ALGORITHM 13.1. f = explicit_euler_method(f, u0, t0, T, k).
Input: function **f**, initial data **u0**, initial time **t0**, final time **T**, time step **k**.
Output: approximation at final time **u**.

```
1: t = t0
2: while t < T do
3:     u = u0 + k*f(u0,t)
4:     u0 = u
5:     t = t + k
6: end while
7: return u
```

Backward Euler method

Alternatively, we can use a *backward difference approximation* of the derivative at $t = t_n$,

$$\dot{u}(t_n) \approx \frac{u(t_n) - u(t_{n-1})}{k_n},$$

which leads to the *backward Euler method*, or *implicit Euler method*,

$$u(t_n) \approx u(t_{n-1}) + k_n \dot{u}(t_n) = u(t_{n-1}) + k_n f(u(t_n), t_n),$$

with the update formula

$$U_n = U_{n-1} + k_n f(U_n, t_n).$$

Backward Euler method

The backward Euler method is *implicit*, meaning that U_n is not directly obtained from U_{n-1}, but needs to be computed from the algebraic equation

$$x = U_{n-1} + k_n f(x, t_n),$$

for example, by the fixed point iteration

$$x^{(k+1)} = U_{n-1} + k_n f(x^{(k)}, t_n).$$

Backward Euler method

ALGORITHM 13.2. f = implicit_euler_method(f, u0, t0, T, k).
Input: function **f**, initial data **u0**, initial time **t0**, final time **T**, time step **k**.
Output: approximation at final time **u**.

```
1: t = t0
2: while t < T do
3:     u = newtons_method(u - u0 - k*f(u,t), u0)
4:     u0 = u
5:     t = t + k
6: end while
7: return u
```

Time stepping methods as quadrature rules

$$u(t_n) = u(t_{n-1}) + \int_{t_{n-1}}^{t_n} f(u(t), t)\, dt,$$

from which we can construct various time stepping methods by using different quadrature rules to evaluate the integral in equation (13.3) at each time step I_n.

Euler methods as quadrature rules

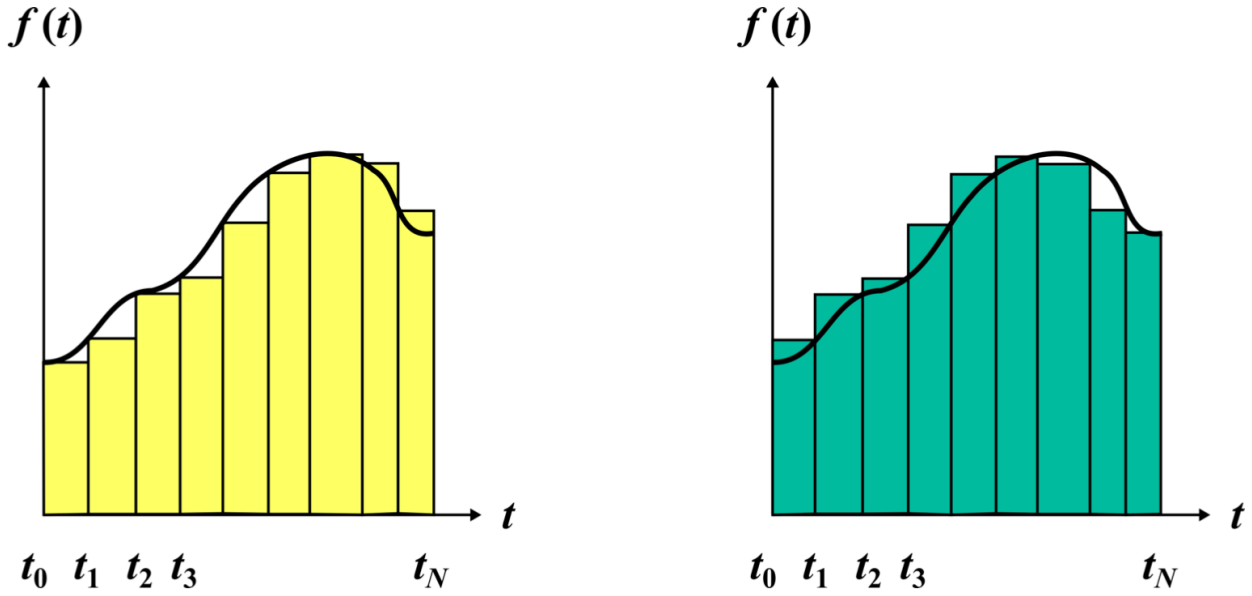


Figure 13.1. *Left (left) and right (right) Riemann sums which approximate the integral of* $f(t)$*, corresponding to the explicit and implicit Euler time stepping method for approximation of the IVP (13.1) with* $f(u(t), t) = f(t)$.

Midpoint and trapezoidal methods

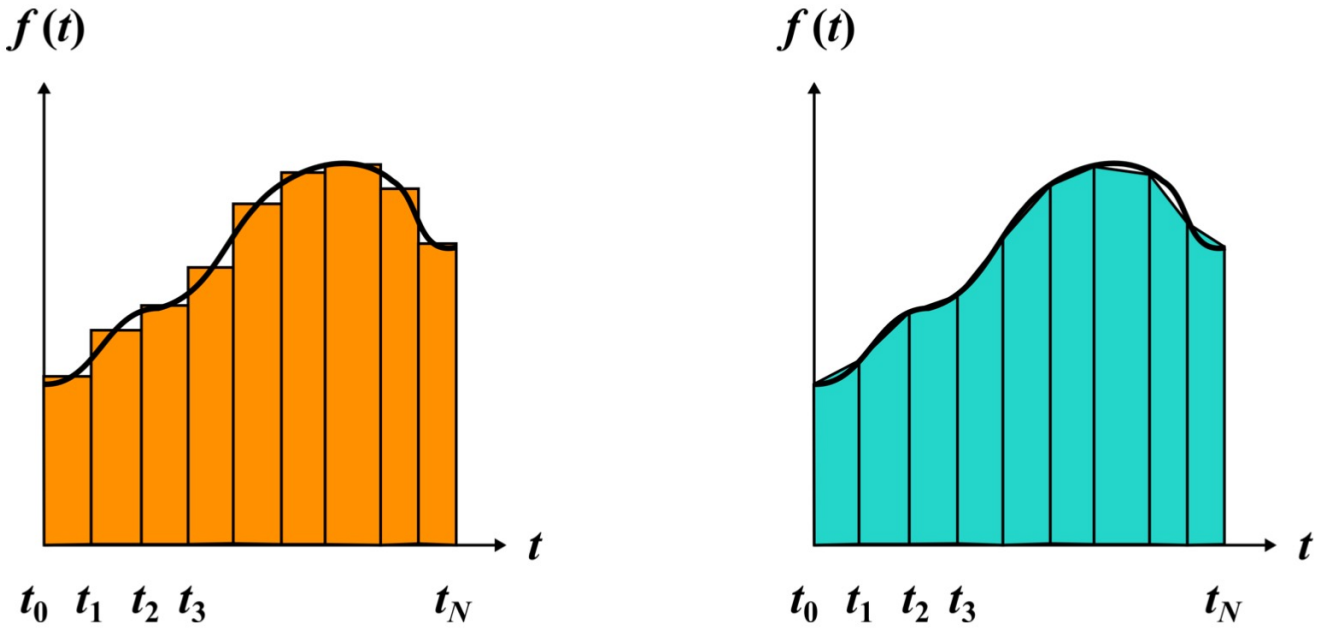


Figure 13.2. *The midpoint (left) and the trapezoidal (right) quadrature rules, corresponding to interpolation by a piecewise constant and piecewise linear function respectively.*

Theta method (F/B Euler, trapezoidal,...)

ALGORITHM 13.3. f = theta_method(f, u0, t0, T, k, theta).
Input: function **f**, initial data **u0**, **theta**, final time **T**, time step **k**.
Output: approximation at final time **u**.

```
1: t = t0
2: while t < T do
3:     u = newtons_method(u - u0 - k*((1-theta)*f(u) + theta*f(u0)), u0)
4:     u0 = u
5:     t = t + k
6: end while
7: return u
```

Piecewise polynomial approximation

To seek approximate solutions to differential equations in the form of piecewise polynomials is a powerful idea. Any piecewise polynomial can be expressed as a linear combination of basis functions for which we seek to determine the coordinates, for example, based on collocation or projection of the residual of the differential equation

$$R(U(t)) = f(U(t), t) - \dot{U}(t).$$

Example 13.3. The explicit Euler method is a collocation method with the collocation points being the time steps $t_k = t_n$, and using a forward difference approximation of the derivative. Analogously, the midpoint method is a collocation method with $t_k = (t_{n-1} + t_n)/2$, and a *central difference approximation* of the derivative at $t = t_k$,

$$\dot{u}(t_k) \approx \frac{u(t_n) - u(t_{n-1})}{k_n}.$$

Piecewise polynomial approximation

In a projection method we seek an approximation function $U \in V_N$ in a finite dimensional *trial space* V_N, for which the residual $R(U(t))$ is orthogonal to a finite dimensional *test space* $\hat{V}_N$, with respect to the L^2 inner product. To determine $U(t)$, the trial and test spaces need to have the same dimension, $\dim(V_N) = \dim(\hat{V}_N) = N$, since $\dim(V_N)$ is the number of degrees of freedom in the approximation $U(t)$, and $\dim(\hat{V}_N)$ is the number of equations.

Piecewise polynomial approximation

Example 13.4. Consider the approximation space V_N of continuous piecewise linear functions that satisfy the initial condition $U(0) = u_0$. A function $U \in V_N$ has one degree of freedom for each time step $I_n = [t_{n-1}, t_n]$, which is the value $U(t_n)$, since $U(t_{n-1})$ is determined at the previous time step I_{n-1}. The test space $\hat{V}_N$ of piecewise constant functions also has one degree of freedom per time step, with the global basis being the characteristic functions for each time step I_n. Hence, the orthogonality of the residual to the test space $\hat{V}_N$ is expressed as

$$\int_{I_n} R(U(t))\, dt = 0, \quad n = 1, ..., N,$$

or equivalently,

$$U_n = U_{n-1} + \int_{I_n} f(U(t), t)\, dt, \quad n = 1, ..., N$$

Piecewise polynomial approximation

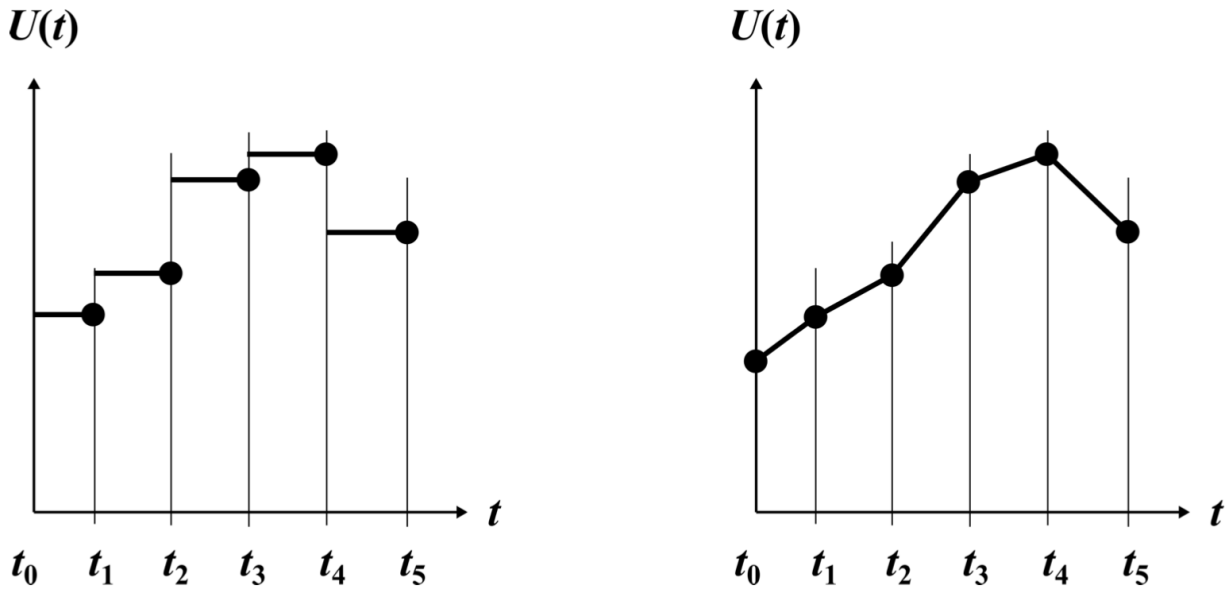


Figure 13.3. *Examples of a discontinuous piecewise constant polynomial determined by its value at the right endpoint of each subinterval (left), and a continuous piecewise linear polynomial determined by its value at the nodes of the partition (right).*

Predictor-corrector method

Explicit time stepping has the benefits of being simple and fast, since it just takes the form of an update formula for each component of the approximation U_n. To construct explicit methods that are more accurate than the explicit Euler method, we can use the idea of a *predictor-corrector method*, which consists of the following two steps:

1. a *prediction step*, which computes a *predicted solution* $\tilde{U}_n$ by an explicit method based on the solution at the previous time step U_{n-1}, and

2. a *correction step* where the *corrected solution* U_n is computed by an implicit method, but with U_n replaced by $\tilde{U}_n$ to make also the implicit method into an explicit update formula.

Example 13.5. *Heun's method* is a predictor-corrector method which uses the explicit Euler method for the prediction step, and the trapezoidal method for the corrector step.

Runge-Kutta methods

A generalization of the two-stage predictor-corrector method is the family of *Runge-Kutta* (RK) methods, where the approximation U_n at the new time step is computed in s stages as a weighted sum, at a number of nodes $c_i \in I_n$ with weights $b_i \in R$, for $i = 1, ..., s$,

$$U_n = U_{n-1} + k_n \sum_{i=1}^{s} b_i f_i,$$

where

$$f_i = f(U_{n-1} + k_n \sum_{j=1}^{i-1} a_{ij} f_j, t_{n-1} + c_i k_n), \quad i = 1, ..., s.$$

The Runge-Kutta matrix $A = (a_{ij})$ is lower triangular, with $a_{ij} = 0$ for $j \geq i$, which allows for direct evaluation of each f_i in sequence. If the Runge-Kutta matrix A is full, a matrix equation needs to be solved in each time step, which leads to an implicit RK method. To specify the type of RK method, a *Butcher tableau* is used, where the coefficients a_{ij}, b_i and c_i are stored.

Multistep method

A linear multistep method takes the general form,

$$\sum_{k=0}^{s} a_k U_{n+k} = k_n \sum_{k=0}^{s} b_k f(U_{n+k}, t_{n+k}), \tag{13.4}$$

which is explicit if $b_s = 0$, else if $b_s \neq 0$ the method is implicit.

Example 13.6. The *backward differential formula* (BDF) is a family of linear implicit multistep methods of the form (13.4), with $b_k = 0$ for $k = 0, \ldots, s-1$, and with a_k determined so that the derivative $\dot{u}(t)$ is approximated by the derivative of the polynomial interpolant $\pi_s u(t)$ constructed from the nodes t_{n+k}, for $k = 0, \ldots, s$.

Stability of initial value problems

The concept of *stability* of an initial value problem refers to how small perturbations propagate in time, and in the context of time stepping methods, how local approximation errors accumulate from one time step to the next. Specifically, for the initial value problem (13.1) we are interested in the stability of *equilibrium points* u^*, defined by the condition that

$$f(u^*, t) = 0, \quad \forall t \geq 0.$$

Stability of linear model problem

To analyze the stability of an equilibrium point $u^* \in R$, we start from the linear model problem

$$\dot{u}(t) + \lambda u(t) = g(t), \quad u(0) = u_0, \tag{13.5}$$

for which we obtain the solution through multiplication by the integrating factor $\exp(\lambda t)$,

$$\begin{aligned} \dot{u}(t)\exp(\lambda t) + u(t)\lambda\exp(\lambda t) &= g(t)\exp(\lambda t), \\ \frac{d}{dt}(u(t)\exp(\lambda t)) &= g(t)\exp(\lambda t), \end{aligned}$$

and by integration from 0 to t, we get the solution

$$u(t) = u_0\exp(-\lambda t) + \int_0^t g(\tau)\exp(-\lambda(t-\tau))\,d\tau. \tag{13.6}$$

Stability of linear model problem

First assume that the model problem is homogeneous, that is, $g(t) = 0$. Then by equation (13.5) there is one equilibrium point $u^* = 0$, which we choose as the initial condition $u_0 = u^*$. We also introduce a problem with perturbed initial condition $u^* + \epsilon$ and solution $v(t)$,

$$\dot{u}(t) + \lambda u(t) = 0, \quad u(0) = u^*, \tag{13.7}$$

$$\dot{v}(t) + \lambda v(t) = 0, \quad v(0) = u^* + \epsilon, \tag{13.8}$$

with $\epsilon \neq 0$ a small perturbation. To study the propagation of the perturbation $\varphi(t) = v(t) - u(t)$, we subtract equation (13.7) from equation (13.8), to get the *perturbation equation*

$$\dot{\varphi}(t) + \lambda \varphi(t) = 0, \quad \varphi(0) = \epsilon,$$

with solution

$$\varphi(t) = \exp(-\lambda t)\epsilon.$$

The equilibrium point $u^* = 0$ is stable if $\lambda > 0$ and unstable if $\lambda < 0$, since then $\varphi(t)$ either decays to zero or grows exponentially. If $\lambda = 0$, then $u^* + \epsilon$ is a new equilibrium point.

Stability of linear model problem

For a complex number $z \in C$, we still have that

$$\frac{d}{dt}\exp(zt) = z\exp(zt),$$

and hence for $\lambda = a + ib$ a complex number, the solution to the perturbation equation is

$$\varphi(t) = \exp(-\lambda t)\epsilon = \exp(-at)\exp(-ibt)\epsilon = \exp(-at)(\cos(bt) - i\sin(bt))\epsilon.$$

We here again distinguish between three cases. If the real part $a < 0$ the equilibrium point u^* is unstable, because perturbations grow at exponential rate, whereas if $a > 0$ then u^* is stable since perturbations decay exponentially. For the pure imaginary case $a = 0$, perturbations oscillate around u^* with the same amplitude over time, in the sense that $|\exp(-ibt)| = 1$.

Linear stability analysis of general IVP

For the general nonlinear initial value problem (13.1), the stability of an equilibrium point u^* can be analyzed in a *linearized* perturbation equation, derived by truncating a Taylor expansion of the right hand side of equation (13.1). Assuming that the function $f(u(t), t)$ is continuously differentiable, and that $|v(t) - u(t)|$ is small, we drop the higher order terms which leads to the approximation

$$f(v(t), t) - f(u(t), t) \approx f'(u(t), t)(v(t) - u(t)),$$

where the derivative $f'(u(t), t)$ is taken with respect to the first argument of the function. By *linearization* at the equilibrium point $u(t) = u^*$, we get the *linearized perturbation equation*

$$\dot{\varphi}(t) - f'(u^*, t)\varphi(t) = 0, \quad \varphi(0) = \epsilon. \tag{13.9}$$

Linear stability analysis of general IVP

Such a *linear stability analysis* is based on the sign of the real part of $f'(u^*, t)$, analogous to the linear model problem (13.5). As long as the perturbation $\varphi(t)$ is sufficiently small, the stability properties of the linearized perturbation equation (13.9) will be representative for a nonlinear problem (13.1). Alternatively, we can use the mean value theorem and the chain rule to express the linearization in the form of an averaged linearized derivative,

$$\bar{f}'(v, u, t) = \int_0^1 f'(w(u, v; s), t)\, ds, \tag{13.10}$$

where $w(u, v; s) = sv(t) + (1 - s)u(t)$ is a convex combination of $v(t)$ and $u(t)$, and

$$f(v(t), t) - f(u(t), t) = \int_0^1 \frac{d}{ds} f(w(u, v; s), t)\, ds = \bar{f}'(v, u, t)\varphi(t). \tag{13.11}$$

Linear stability analysis of general IVP

Example 13.7. Consider the nonlinear initial value problem

$$\dot{u}(t) = (1 - u(t))u(t), \quad u(0) = u_0,$$

with two equilibrium points $u_1^* = 0$ and $u_2^* = 1$. Since $f(u,t) = (1-u)u$, the derivative with respect to u is $f'(u,t) = 1 - 2u$, so that

$$f'(u_1^*, t) = 1, \quad f'(u_2^*, t) = -1.$$

Hence, the equilibrium point u_1^* is unstable, and u_2^* is stable.

Stability of time stepping methods

The stability of approximate solutions computed by a time stepping method is determined by the underlying IVP (13.1) and the parameters of the method. For accurate approximation, the stability properties of the IVP must be reflected in the time stepping method. Specifically, if the IVP is stable the time stepping method should be stable. To analyze the stability of time stepping methods, it is the convention to use the following stable linear model problem,

$$\begin{aligned} \dot{u}(t) &= \lambda u(t), \quad 0 < t \leq T, \\ u(0) &= 1, \end{aligned} \tag{13.12}$$

with solution $u(t) = \exp(\lambda t)$, where λ is a complex number with negative real part $\mathrm{Re}(\lambda) < 0$.

Stability of time stepping methods

If a time stepping method with constant time step k converges to zero as $t \to \infty$, as the exact solution does, the method is said to be *A-stable*. For example, with $U_0 = 1$, approximations by the explicit Euler method, the implicit Euler method and the trapezoidal method take the forms,

$$
\begin{aligned}
U_n &= (1 + k_n\lambda)U_{n-1} = (1 + k\lambda)^n, \\
U_n &= \frac{1}{1 - k\lambda}U_{n-1} = (1 - k\lambda)^{-n}, \\
U_n &= \frac{1 + 0.5k\lambda}{1 - 0.5k\lambda}U_{n-1} = \left(\frac{1 + 0.5k\lambda}{1 - 0.5k\lambda}\right)^n .
\end{aligned}
$$

Stability of time stepping methods

We obtain the following condition on $k\lambda$ for the explicit Euler method to be A-stable,

$$|1 + k\lambda| < 1 \Leftrightarrow |-1 - k\lambda| < 1,$$

a unit circle in the complex plane centered at $k\lambda = -1$, for the implicit Euler method,

$$|(1 - k\lambda)^{-1}| < 1 \Leftrightarrow |1 - k\lambda| > 1,$$

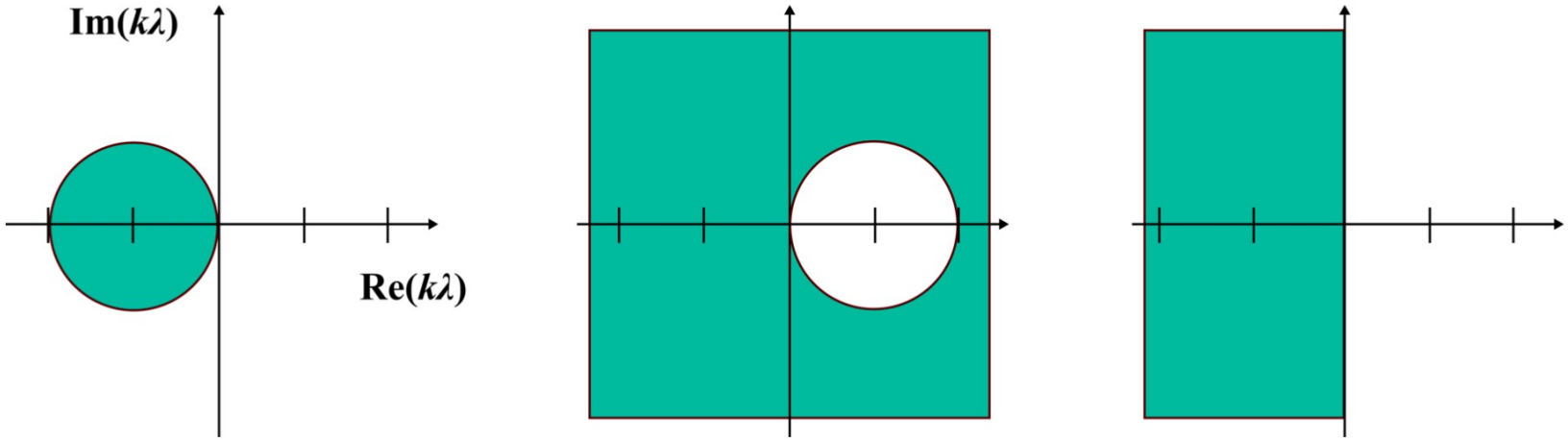
the exterior of a unit circle in the complex plane, and for the trapezoidal method

$$\left|\frac{1 + 0.5k\lambda}{1 - 0.5k\lambda}\right| < 1 \Leftrightarrow |1 + 0.5k\lambda| < |1 - 0.5k\lambda| \Leftrightarrow \mathrm{Re}(0.5k\lambda) < 0.$$

In Figure 13.4 we visualize these conditions as *stability regions* in the complex plane. Since $\mathrm{Re}(\lambda) < 0$ and $k > 0$, the explicit Euler method is *conditionally stable*, provided $|1 + k\lambda| < 1$, whereas the implicit Euler method and the trapezoidal method are *unconditionally stable*.

Stability regions

In Figure 13.4 we visualize these conditions as *stability regions* in the complex plane. Since $\mathrm{Re}(\lambda) < 0$ and $k > 0$, the explicit Euler method is *conditionally stable*, provided $|1 + k\lambda| < 1$, whereas the implicit Euler method and the trapezoidal method are *unconditionally stable*.



A priori error analysis

We discovered that the stability of the explicit Euler method depends on the time step k, and the next question is how the accuracy is influenced by k. To compare this with the stability analysis, we choose the same model problem (13.12), but now with $\lambda < 0$ a real number, for simplicity. Consider the first time step of the explicit Euler method to solve the model problem (13.12),

$$U_1 = u_0 + k\lambda u_0 = (1 + k\lambda)u_0,$$

with a constant time step length k and a final time $T = kN$, with N the number of time steps. By Taylor's formula, the exact solution of equation (13.12) at $t = t_1$ is given by

$$u(t_1) = u_0 + k\dot{u}(t_0) + \frac{1}{2}k^2\ddot{u}(s_1) = (1 + k\lambda)u_0 + \frac{1}{2}k^2\ddot{u}(s_1),$$

with $s_1 \in [t_0, t_1]$, assuming that $u \in C^2([0, T])$. Hence, we can express the local error over the first time step I_1 as

$$e(t_1) = u(t_1) - U_1 = \frac{1}{2}k^2\ddot{u}(s_1).$$

A priori error analysis

Similarly, we express the error at $t = t_2$ as the accumulation of the local errors over the time step I_2 and the previous time step I_1,

$$e(t_2) = u(t_2) - U_2 = u(t_2) - (1 + k\lambda)(u(t_1) - e(t_1)) = \frac{1}{2}k^2\ddot{u}(s_2) + (1 + k\lambda)\frac{1}{2}k^2\ddot{u}(s_1),$$

with $s_n \in [t_{n-1}, t_n]$. The error at the final time T can then be estimated as

$$e(T) = \frac{k^2}{2}(\ddot{u}(s_N) + ... + (1 + k\lambda)^{N-1}\ddot{u}(s_1)) \leq \frac{k^2}{2}\|\ddot{u}\|_\infty \sum_{n=0}^{N-1}(1 + k\lambda)^n.$$

In the limit $N \to \infty$ the geometric series converges to $1/k|\lambda|$, and hence we conclude that whereas the local error over one time step is $\mathcal{O}(k^2)$, the global error at final time is $\mathcal{O}(k)$. Therefore, by reducing the time step length by a constant factor we expect the global error to be reduced by the same factor.

A priori error analysis

We refer to this as an *a priori* error analysis, since the error estimate is expressed in terms of the regularity of the exact solution, before computation, by which we can estimate the error in the time stepping method for a problem where

$$\|\ddot{u}\|_\infty \leq C,$$

with $C > 0$ a constant independent of the time step length k.

For the model problem (13.12), the exact solution is $u(t) = \exp(\lambda t)$, with the second order derivative $\ddot{u}(t) = \lambda^2 \exp(\lambda t)$, and hence the local error at time step n is bounded by

$$\max_{s_n \in I_n} |\frac{1}{2} k^2 \ddot{u}(s_n)| = \frac{1}{2} k^2 \lambda^2 \exp(\lambda t_n).$$

Stiffness of an IVP

Our first observation is that the local errors decrease exponentially over time, so that the local error for the final time step N is much smaller than for the first time step. Since the local error is proportional to the square of the time step, a natural strategy would be to choose shorter time steps in the beginning of the time stepping algorithm and longer time steps in the end. But from the stability analysis we know that the time step for the explicit Euler method is bounded by

$$k_n < 2/|\lambda|,$$

independent of the time t_n. Therefore, we are forced to take short time steps due to the stability of the method, even though from an accuracy point of view we could take much larger time steps. This is a feature of a *stiff problem*, for which an unconditionally stable method is more suitable, such as the implicit Euler method or the trapezoidal method, that allows for arbitrary long time steps that can be optimized with respect to the desired accuracy in the simulation.

The error equation

Let $U(t)$ be a continuous piecewise polynomial approximate solution to the linear model problem (13.5), computed by a time stepping method, which satisfies

$$\dot{U}(t) + \lambda U(t) = G(t), \quad U(0) = U_0, \tag{13.13}$$

with $G(t) \approx g(t)$ and $U_0 \approx u_0$. Analogous to equation (7.5), by subtracting equation (13.13) from equation (13.5), we can derive an equation for the error $e(t) = u(t) - U(t)$,

$$\dot{e}(t) + \lambda e(t) = g(t) - G(t) = g(t) - \dot{U}(t) - \lambda U(t) = R(U(t)), \tag{13.14}$$

with the initial error $e(0) = u_0 - U_0$, and the residual of equation (13.5),

$$R(U(t)) = g(t) - \dot{U}(t) - \lambda U(t).$$

The solution to the error equation (13.14) at time t is the sum of the propagated initial error and the accumulation of residual errors,

$$e(t) = e(0)\exp(-\lambda t) + \int_0^t R(U(\tau))\exp(-\lambda(t-\tau))\,d\tau. \tag{13.15}$$

The adjoint problem

We now introduce a *dual*, or *adjoint*, equation to the linear model problem (13.5),

$$-\dot{\varphi}(t) + \lambda\varphi(t) = \psi(t), \quad \varphi(T) = \varphi_T. \tag{13.16}$$

A change of variables $s = T - t$ gives that $-\dot{\varphi}(t) = \dot{\varphi}(s)$, so that in the new variable s, which runs backwards in time, the adjoint problem takes the form

$$\dot{\varphi}(s) + \lambda\varphi(s) = \psi(s), \quad \varphi(0) = \varphi_T,$$

with solution

$$\varphi(s) = \varphi_T \exp(-\lambda s) + \int_0^s \psi(r) \exp(-\lambda(s-r))\, dr.$$

The adjoint problem

Then changing back to the variable $t = T - s$ gives that

$$\varphi(t) = \varphi_T \exp(\lambda(t - T)) + \int_0^{T-t} \psi(r) \exp(-\lambda(T - t - r))\, dr,$$

and finally changing the integration variable to $\tau = T - r$, so that $d\tau = -dr$, leads to

$$\varphi(t) = \varphi_T \exp(-\lambda(T - t)) + \int_t^T \psi(T - \tau) \exp(-\lambda(\tau - t))\, d\tau.$$

Specifically, at $t = 0$ we have

$$\varphi(0) = \varphi_T \exp(-\lambda T) + \int_0^T \psi(T - \tau) \exp(-\lambda\tau)\, d\tau.$$

A posteriori error estimation – linear IVP

By the adjoint equation (13.16) we can state the error in the approximation (13.13) with respect to the solution of the linear model problem (13.5), as a sum of the integrated error over the entire time interval weighted by $\psi(t)$, and the error at final time weighted by φ_T, expressed in terms of the initial error, the residual and the adjoint solution,

$$\int_0^T e(t)\psi(t)\,dt + e(T)\varphi_T = e(0)\varphi(0) + \int_0^T R(U(t))\varphi(t)\,dt. \tag{13.17}$$

For $\varphi_T = 1$ and $\psi = 0$, the solution to the adjoint problem is $\varphi(t) = \exp(-\lambda(T-t))$, from which we recover equation (13.15), in the form

$$e(T) = e(0)\exp(-\lambda T) + \int_0^T R(U(t))\exp(-\lambda(T-t))\,dt,$$

but without using the solution formula for the error equation (13.14).

A posteriori error estimation – general IVP

This type of *a posteriori* error analysis can be extended to the general initial value problem (13.1), with the linearized adjoint problem

$$-\dot{\varphi}(t) - \bar{f}'(u, U, t)\varphi(t) = \psi(t), \quad \varphi(T) = \varphi_T, \tag{13.18}$$

and the averaged derivative defined by equation (13.10).

$$\bar{f}'(v, u, t) = \int_0^1 f'(w(u, v; s), t)\, ds,$$

where $w(u, v; s) = sv(t) + (1 - s)u(t)$ is a convex combination of $v(t)$ and $u(t)$, and

$$f(v(t), t) - f(u(t), t) = \int_0^1 \frac{d}{ds} f(w(u, v; s), t)\, ds = \bar{f}'(v, u, t)\varphi(t).$$

A posteriori error estimation – general IVP

To derive the a posteriori error estimate (13.17) for the general initial value problem, we multiply the adjoint equation (13.18) by the error and integrate over the time interval $I = [0, T]$,

$$\begin{aligned}
\int_0^T e(t)\psi(t)\,dt &= \int_0^T e(t)(-\dot{\varphi}(t) - \bar{f}'(u, U, t)\varphi(t))\,dt \\
&= e(0)\varphi(0) - e(T)\varphi(T) + \int_0^T (\dot{e}(t) - \bar{f}'(u, U, t)e(t))\varphi(t)\,dt \\
&= e(0)\varphi(0) - e(T)\varphi(T) + \int_0^T (\dot{u} - f(u, t)) - (\dot{U}(t) - f(U, t))\varphi(t)\,dt \\
&= e(0)\varphi(0) - e(T)\varphi(T) + \int_0^T R(U(t))\varphi(t)\,dt,
\end{aligned}$$

by equation (13.11), from which we obtain the error representation (13.17), with the residual

$$R(U(t)) = f(U, t) - \dot{U}(t).$$

Adaptive time stepping methods

The *a posteriori* error estimate takes the form of an integral over the domain $I = [0, T]$, which can be split into a sum of integrals over the N subinterval $I_i = [t_{i-1}, t_i]$,

$$\int_0^T e(t)\psi(t)\,dt + e(T)\varphi_T = e(0)\varphi(0) + \sum_{i=1}^{N} \int_{I_i} R(U(t))\varphi(t)\,dt = e(0)\varphi(0) + \sum_{i=1}^{N} \mathcal{E}_i,$$

where $\mathcal{E}_i$ is the *error indicator* for subinterval I_i. The error indicators can be used to formulate *adaptive* time stepping algorithms, where, for example, in each iteration of the adaptive algorithm we bisect all subintervals I_i for which

$$\mathcal{E}_i > \frac{1}{N} \sum_{j=1}^{N} \mathcal{E}_j.$$

Stochastic differential equation

The initial value problem (13.1) can be expressed in *differential form* as

$$du(t) = f(u(t), t)\, dt, \quad u(0) = u_0,$$

with the solution in integral form

$$u(t) = u(0) + \int_0^t f(u(s), s)\, ds.$$

In a *stochastic differential equation* (SDE), we also have a stochastic part of the function,

$$f(u(t), t) = \mu(u(t), t) + \sigma(u(t), t)\xi_t,$$

where $\{\xi_t\}$ is a *white noise* stochastic process, defined by $dW_t = \xi_t\, dt$, where W_t is the Wiener process, with the property that $dW_t \sim N(0, dt)$.

Stochastic differential equation

Hence, we seek a continuous stochastic process X_t, such that

$$dX_t = \mu(X_t, t)\,dt + \sigma(X_t, t)\,dW_t, \quad P(X_0 = u_0) = 1,$$

or in integral form

$$X_t = X_0 + \int_0^t \mu(X_s, s)\,ds + \int_0^t \sigma(X_s, s)\,dW_s,$$

which is the sum of one deterministic integral and one stochastic integral. We can define the stochastic integral as the limit of a Riemann sum, but it is important to specify what type of Riemann sum. Or in the context of an initial value problem, what time stepping method that is used. In the case we use an explicit Euler time stepping algorithm, the limit of the corresponding Riemann sums is the *Itô integral*.

Stochastic differential equation

The explicit Euler method for a stochastic differential equation takes the form:

start from $x_0 = u_0$, then compute $x_n = x(t_n)$ as

$$x_n = x_{n-1} + \mu(x_{n-1}, t_{n-1})\Delta t_n + \sigma(x_{n-1}, t_{n-1})\,\Delta W_n,$$

with the time step size $\Delta t_n = t_n - t_{n-1}$, and the increment of the Wiener process

$$\Delta W_n = W_{t_n} - W_{t_{n-1}} = Z\sqrt{\Delta t_n}, \quad Z \sim N(0,1).$$

This method is the *Euler-Maruyama method*, and the result of the method is the approximation of one observation, or *path*, $x(t)$ of the stochastic process X_t. To estimate statistics of the process, such as the expected value, we need to compute multiple paths which we can then analyze.

Diffusion process

The SDE describes a Markov process X_t for which the change dX_t over a small time interval dt is normally distributed with expected value $\mu(X_t, t)\, dt$ and variance $\sigma(X_t, t)^2\, dt$, independent of previous states X_s for $s < t$. We refer to X_t as a *diffusion process* with $\mu(X_t, t)$ the *drift*, and

$$D(X_t, t) = \sigma^2(X_t, t)/2$$

the *diffusion coefficient*. The diffusion process X_t has a probability density function $u(x, t)$, which satisfies the deterministic *Fokker-Planck equation*

$$\frac{\partial}{\partial t}u(x,t) + \frac{\partial}{\partial x}(\mu(x,t)u(x,t)) - \frac{\partial^2}{\partial x^2}(D(x,t)u(x,t)) = 0,$$

a partial differential equation which requires initial and boundary condition to have a unique solution.

Brownian motion

Example 13.8 (Brownian motion). *Brownian motion* is described by the Wiener process,

$$dX_t = dW_t,$$

a stochastic differential equation with $\mu(X_t, t) = 0$ and $\sigma(X_t, t) = 1$. If we take the expected value of the SDE, we find that $E[dX_t] = E[dW_t] = 0$, and since we can switch the order of differentiation and integration, $E[dX_t] = dE[X_t]$, the expected value is constant $E[X_t] = E[X_0]$. The probability density function $u(x, t)$ satisfies the Fokker-Planck equation

$$\frac{\partial}{\partial t}u(x,t) - D\frac{\partial^2}{\partial x^2}(u(x,t)) = 0,$$

which is a diffusion equation with diffusion coefficient $D = 1/2$, and solution

$$u(x,t) = \frac{1}{\sqrt{4\pi Dt}}\int_{-\infty}^{\infty} \exp\left(-\frac{(x-y)^2}{4Dt}u(y,0)\right)\, dy.$$

Brownian motion in R^d is defined by independent Brownian motion in each dimension.

Brownian motion

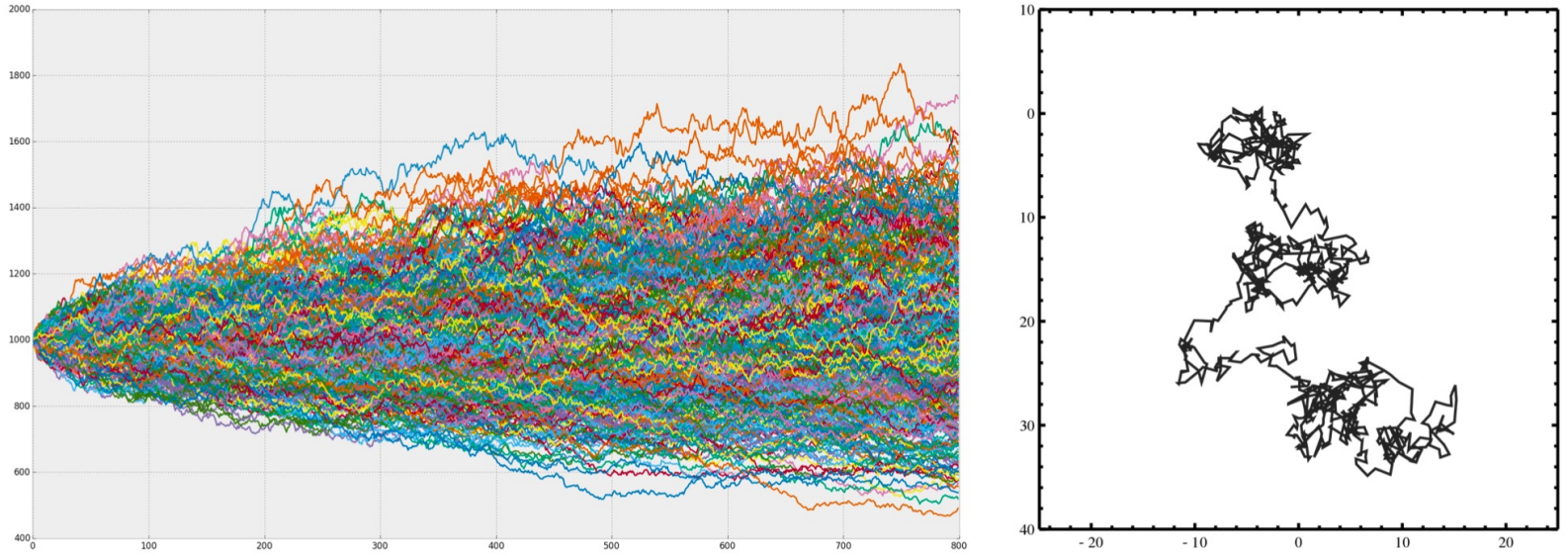


Figure 13.5. *An ensemble of simulated trajectories of Brownian motion (left), and one trajectory of a two dimensional Brownian motion (right).*