MATH 408: Computational Methods for Differential Equations

Basic Numerical Methods for IVPs

Greg Fasshauer

Department of Applied Mathematics & Statistics
Colorado School of Mines

Fall 2021

Outline

1 Basic Numerical Methods for IVPs

The General Idea

We consider our standard first-order IVP

$$u'(t) = f(t, u(t)), \quad t \geq t_0,$$

with initial condition (IC)

$$u(t_0) = \eta.$$

Our numerical methods will iteratively compute a sequence of approximations

$$(t_0, U^0),\ (t_1, U^1),\ (t_2, U^2),\ \ldots,\ (t^N, U^N)$$

for the solution u of the IVP, i.e.,

$$U^0 = u(t_0) = \eta \text{ (given IC)}, \quad U^n \approx u(t_n),\ n = 1, 2, \ldots, N.$$

Also, using k to denote the time step (i.e., $k = \Delta t$)

$$t_0 \text{ (given from IC)}, \quad t_{n+1} = t_n + k,\ n = 0, 1, 2, \ldots, N-1.$$

All of the following methods are obtained by applying FD approximations to the left-hand side of the ODE above.

Remark

To keep things simple, the following deal only with the single equation IVP from the previous slide.
However, using vector notation in MATLAB *all methods also have straightforward implementations for IVP* **systems**.

Remark

We will begin with the well-known (and very simple) Euler method, but keep in mind that all the basic methods—as well as fancy Runge–Kutta and multistep methods we will later study—are just generalizations of the basic framework we use for Euler's method.

Euler's Method

As mentioned in `Notes408_01_FiniteDifferences` (see also MATH 307), Euler's method (or the forward Euler method) results from approximating $u'(t_n)$ with a forward difference D_+, i.e.,

$$u'(t_n) \approx D_+ U^n = \frac{U^{n+1} - U^n}{k}$$

so that the ODE from the IVP gives us $\frac{U^{n+1}-U^n}{k} = f(t_n, U^n)$ or

$$\boxed{U^{n+1} = U^n + kf(t_n, U^n), \quad n = 0, 1, 2, \ldots, N-1}$$

with $U^0 = \eta$ and t_0 given by the IC and $t_{n+1} = t_n + k$ with time step k.

Remark

Euler's method is an explicit time marching method and can be implemented in a straightforward `for-loop`*.*

The Backward Euler Method

The backward Euler method appears to be very similar to Euler's method as it results from approximating $u'(t_n)$ with a backward difference D_-, i.e.,

$$u'(t_n) \approx D_- U^n = \frac{U^n - U^{n-1}}{k}$$

so that the ODE from the IVP gives us $\frac{U^n - U^{n-1}}{k} = f(t_n, U^n)$ or

$$\boxed{U^{n+1} = U^n + kf(t_{n+1}, U^{n+1}), \quad n = 0, 1, 2, \ldots, N-1}$$

with $U^0 = \eta$ and $t_1 = t_0 + k$ given via the IC.

Remark

The backward Euler method is an implicit time marching method and generally requires a nonlinear solver such as Newton's method or fixed point iteration to find U^{n+1} in each iteration.

Trapezoidal Method

By adding the forward and backward Euler methods we obtain the trapezoidal method:

$$\begin{aligned} U^{n+1} &= U^n + kf(t_n, U^n) \\ +U^{n+1} &= U^n + kf(t_{n+1}, U^{n+1}) \\ \hline 2U^{n+1} &= 2U^n + kf(t_n, U^n) + kf(t_{n+1}, U^{n+1}) \end{aligned}$$

or

$$\boxed{U^{n+1} = U^n + \frac{k}{2}\left(f(t_n, U^n) + f(t_{n+1}, U^{n+1})\right), \quad n = 0, 1, 2, \ldots, N-1}$$

with $U^0 = \eta$, t_0 and $t_1 = t_0 + k$ given via the IC.

Remark

The trapezoidal method is also an implicit time marching method. It has its name because it is closely related to the trapezoidal rule for numerical integration.

Leapfrog Method

By approximating $u'(t_n)$ with the symmetric difference D_0 we obtain an *explicit* two-step method known as the leapfrog or midpoint method:

$$u'(t_n) \approx D_0 U^n = \frac{U^{n+1} - U^{n-1}}{2k}$$

or

$$\boxed{U^{n+1} = U^{n-1} + 2kf(t_n, U^n), \quad n = 1, 2, \ldots, N-1}$$

with $U^0 = \eta$ and $t_1 = t_0 + k$ given by the IC. Note that U^1 is not known from the IC.

Remark

A two-step method needs extra care in getting started (since we're given only one IC). Note that this "midpoint method" is different from the (Runge–Kutta) "midpoint method" we implemented in MATH 307.

The BDF2 Method

The following backward differentiation formula (BDF) is an *implicit* two-step method that generalizes the backward Euler method. It is obtained by approximating $u'(t_n)$ with the one-sided difference D_2, i.e.,

$$u'(t_n) \approx D_2 U^n = \frac{1}{2k}\left(3U^n - 4U^{n-1} + U^{n-2}\right)$$

so that the ODE from the IVP gives us $\frac{3U^n - 4U^{n-1} + U^{n-2}}{2k} = f(t_n, U^n)$ or

$$\boxed{U^{n+1} = \frac{4}{3}U^n - \frac{1}{3}U^{n-1} + \frac{2}{3}kf(t_{n+1}, U^{n+1}), \quad n = 1, 2, \ldots, N-1}$$

with $U^0 = \eta$ given by the IC.

Remark

This two-step method not only needs extra care in getting started, but also requires a nonlinear solver since it is an implicit method.

References I


R. LeVeque.
Finite Difference Methods for Ordinary and Partial Differential Equations.
SIAM, Philadelphia, 2007.