

CS341 SYSTEMS BIOLOGY I

LECTURE 15

NUMERICAL METHODS FOR SOLVING DIFFERENTIAL EQUATIONS (CONT)

So far, we have implemented Runge-Kutta methods with error $\mathcal{O}(\Delta t)$ (forward Euler) and $\mathcal{O}(\Delta t^2)$ (explicit trapezoidal) and $\mathcal{O}(\Delta t^4)$ (RK4). The chief idea behind a Runge-Kutta method is that we estimate the value at the end of one time step by evaluating the function at multiple places within that time step. We increase accuracy by increasing the number of times the ODE function is evaluated (the 4th-order method has 4 function evaluations per time step). But could we use just one function evaluation per time step and also achieve high accuracy? Yes, there is a class of methods called *linear multi-step methods* that use a linear combination of function evaluations from previous time steps. The more time steps we use, the higher the accuracy. Today, we will look at a family of linear multi-step methods called Adams-Bashforth methods. We begin with the second order method and then briefly describe the 4th order method.

SECOND-ORDER ADAMS-BASHFORTH METHOD

The idea behind the second-order Adams-Bashforth method (which I will call AB2), is the same as that behind the trapezoidal rule for numerical quadrature. But instead of using the first and last points of the current time step to determine the slope of the top of the trapezoid, we use the first point in the previous time step and the first point in the current time step and project the line. See Figure 1 for an illustration.

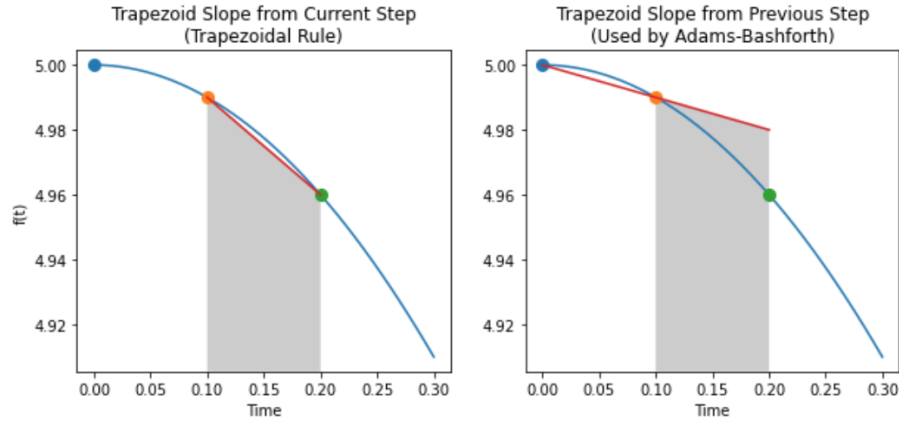


FIGURE 1. On the left, we illustrate the trapezoidal rule used in numerical quadrature to estimate the area under the curve between $t=0.1$ and $t=0.2$. It uses the slope of the line from $f(0.1)$ to $f(0.2)$. On the right, we illustrate the trapezoid used by the second order Adams-Bashforth method. In this method, we use the slope of the line from $f(0)$ to $f(0.1)$ to estimate the area under the curve from $t=0.1$ to $t=0.2$. The estimate on the right looks worse than the estimate on the left. And if I actually knew my function over time (i.e. if I were doing numerical quadrature, rather than solving an ODE), I would certainly choose the method on the left. But when solving an ODE, we know the values at previous time steps, so why not take advantage of that?

Let's work out the math. We use information at $t_0 = 0$ and $t_1 = 0.1$ to estimate the area under the curve from t_1 to $t_2 = 0.2$. $\Delta t = 0.1$. The slope of the Adams-Bashforth line is

$$s = \frac{f(t_1) - f(t_0)}{\Delta t}$$

So the trapezoid's height at t_1 is $f(t_1)$ and at t_2 is $f(t_1) + \Delta t * s$. The area under the trapezoid is therefore the average height of the two points times the width, i.e.

$$\begin{aligned} \Delta t \frac{f(t_1) + (f(t_1) + \Delta t * s)}{2} &= \Delta t \frac{f(t_1) + f(t_1) + \Delta t * \frac{f(t_1) - f(t_0)}{\Delta t}}{2} \\ &= \Delta t \frac{f(t_1) + f(t_1) + f(t_1) - f(t_0)}{2} \\ &= \frac{\Delta t}{2} (3f(t_1) - f(t_0)) \end{aligned}$$

This leads us to the formula for the second order Adams-Bashforth method for solving differential equations. It uses the estimated solution at t and $t - \Delta t$ to estimate the solution at $t + \Delta t$.

$$\begin{aligned}f_b &= f(t - \Delta t, \hat{y}(t - \Delta t)) \\f_a &= f(t, \hat{y}(t)) \\ \hat{y}(t + \Delta t) &= \hat{y}(t) + \frac{\Delta t}{2} (3f_a - f_b)\end{aligned}$$

Cool! This is a second-order method and, as long as we are smart about saving f_a , every time we move forward a step, we need to call the function f just once. This is less computation than the explicit trapezoidal method.

But there is one more important detail. How do we get started? If we need estimates from more than one step, how do we compute them? The answer is to use a 1-step method of the same order. So we use an explicit trapezoidal step to compute the first step after the initial value. Then we have the initial value and the explicit trapezoidal step. Then, we compute all the remaining steps with the Adams-Bashforth method, as in Figure 2.

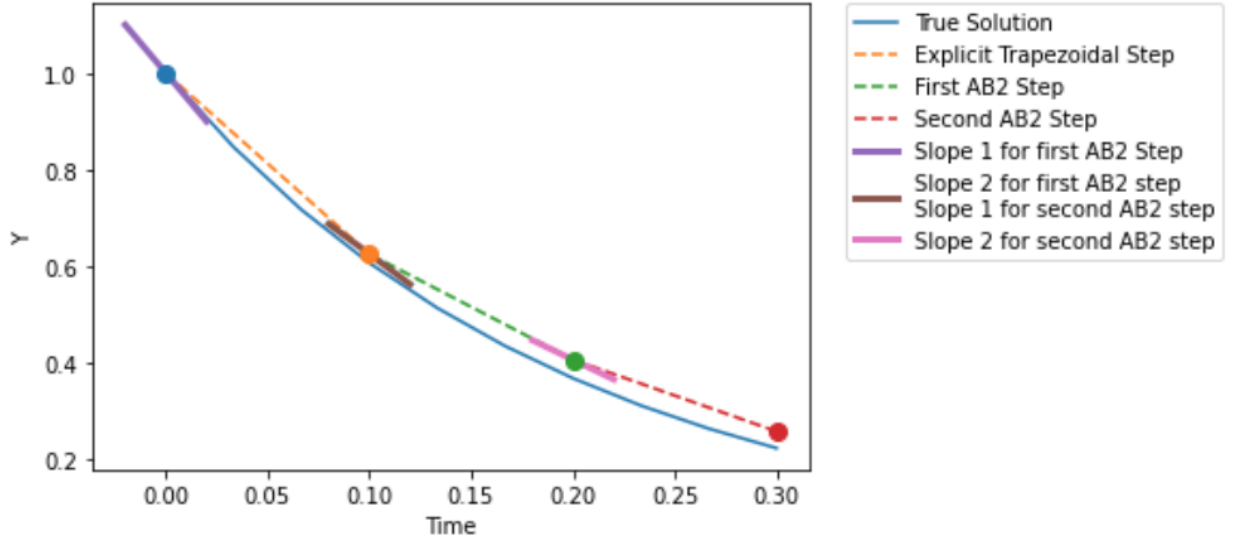


FIGURE 2. The second-order Adams-Bashforth is illustrated for the first 3 steps. The blue line is the true solution and the blue dot is the initial condition. The method begins with an explicit trapezoidal step (orange dashed line to orange dot). Then it uses the slopes of the purple and brown lines to compute the slope for the first AB2 step. The first AB2 step is shown by the green dashed line, and it ends at the green dot. Next, the slopes of the brown and pink lines are used to compute the slope for the second AB2 step. The second AB2 step is shown with the red dashed line and it ends with the red dot.

The error from AB2 ($\mathcal{O}(\Delta t^2)$) will generally be comparable to the error from explicit trapezoidal ($\mathcal{O}(\Delta t^4)$) and smaller than the error from forward Euler ($\mathcal{O}(\Delta t)$). (See Figure 3.)

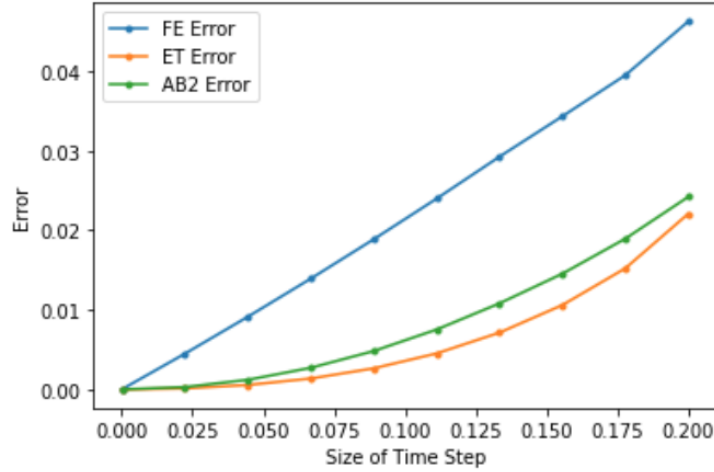


FIGURE 3. For the test equation with 2 ODE's with degradation rates of 2 and 5, integrated from $y_0=(1,1)$ from $t=0$ to $t=4$, we solved the equations with the forward Euler (FE), explicit trapezoidal (ET), and AB2 methods for time steps ranging from $1e-4$ to 0.2 . The error from each simulation was computed as the mean over time of the Euclidean distance between the estimated solution and the true solution at each time step.

FOURTH-ORDER ADAMS-BASHFORTH METHOD

(We are skipping the 3rd just so we can have a method to compare to RK4. There is nothing wrong with the 3rd order Adams-Bashforth method)

The fourth-order Adams-Bashforth method uses a polynomial passing through three points to estimate the error under the curve. As with RK4, the area under the polynomial is computed by a weighted average of the slopes.

$$\begin{aligned}
 f_d &= f(t - 3\Delta t, \hat{y}(t - 3\Delta t)) \\
 f_c &= f(t - 2\Delta t, \hat{y}(t - 2\Delta t)) \\
 f_b &= f(t - \Delta t, \hat{y}(t - \Delta t)) \\
 f_a &= f(t, \hat{y}(t)) \\
 \hat{y}(t + \Delta t) &= \hat{y}(t) + \frac{\Delta t}{24} (55f_a - 59f_b + 37f_c - 9f_d)
 \end{aligned}$$

We use RK4 to compute the first 4 steps (well, the initial condition plus the first 3 steps), the proceed with Adams-Bashforth steps. If we are smart about how we code it, we again

need only one evaluation of f at each time step (that at time t itself). This is $\frac{1}{4}$ the number of function evaluations than RK4. So it could be better. The only problem is that its region of stability is pretty small.