Numerical Solutions to Ordinary Differential Equations: A Second-Order Method

Single Equations

Consider the single first-order ODE (either linear or nonlinear)

$$\frac{dy}{dt} = f(y,t) \tag{1}$$

with the initial condition

$$y(t_o) = y_o \tag{2}$$

We can again solve this numerically, using an Euler-like formula

$$y_i = y_{i-1} + (t_i - t_{i-1})\tilde{f} \tag{3}$$

where $\tilde{f}$ is an approximation to the derivative over the interval from t_{i-1} to t_i. For a second order method, the approximation to the derivative is simply an average of the values at t_{i-1} and t_i.

$$\tilde{f} = \frac{1}{2}\left[f(t_{i-1}, y_{i-1}) + f(t_i, y_i)\right] \tag{4}$$

The problem is that we don't know the value of y_i that appears on the right hand side of equation (4).

Second Order Method for Generic ODEs

In the general case (meaning that equation (1) is either linear or nonlinear in y), we can use the Euler method to approximate y_i for the purposes of equation (4).

$$y_i = y_{i-1} + (t_i - t_{i-1})f(t_{i-1}, y_{i-1}) \tag{5}$$

We then substitute this value into equation (4) and substitute the resulting value of $\tilde{f}$ into equation (3), yielding a second order method:

$$y_i = y_{i-1} + (t_i - t_{i-1})\tilde{f} = y_{i-1} + (t_i - t_{i-1})\frac{1}{2}\left[f(t_{i-1}, y_{i-1}) + f(t_i, y_{i-1} + (t_i - t_{i-1})f(t_{i-1}, y_{i-1}))\right] \tag{6}$$

This implementation of the second order method is a member of a class of methods known as predictor-corrector methods, because you use Euler's method to predict y_i and you use equation (6) to correct the value. Specifically, equation (6) is called Heun's method.

Second Order Method for Linear ODEs

In obtaining equation (6), we were forced to make an additional approximation, namely we had to use the Euler method for a preliminary estimate of y_i. If the ODE (eqn. (1)) is linear, then we do not have to make this approximation in order to solve the problem. Consider a linear ODE of the form

$$\frac{dy}{dt} = f(y,t) = a(t)y(t) + b(t) \tag{7}$$

with the initial condition of equation (2). We substitute this linear ODE into equation (4) obtaining

$$\tilde{f} = \frac{1}{2}\left[a_{i-1}y_{i-1} + b_{i-1} + a_i y_i + b_i\right] \tag{8}$$

where we have used the shorthand notation that $a_i = a(t_i)$, as we have done for y and b as well. We substitute equation (8) into equation (3)

$$y_i = y_{i-1} + (t_i - t_{i-1})\frac{1}{2}\left[a_{i-1}y_{i-1} + b_{i-1} + a_i y_i + b_i\right] \tag{9}$$

We solve for the unknown, y_i, obtaining

$$A_{i,i}y_i = -A_{i,i-1}y_{i-1} + B_i \tag{10}$$

where $A_{i,j}$ is the coefficient (in general a function of the independent variable t) in front of the j^{th} variable at the i^{th} time, t_i, and has the form

$$A_{i,j} = \begin{cases} 1 - \dfrac{(t_i - t_{i-1})}{2}a_j & \text{if } i = j \\ -1 - \dfrac{(t_i - t_{i-1})}{2}a_j & \text{if } i \neq j \end{cases} \tag{11}$$

$$B_i = \frac{(t_i - t_{i-1})}{2}(b_i + b_{i-1}) \tag{12}$$

This method is classified as an implicit method because the value of the unknown y_i appears on both the left hand side and right hand side of equation (9).

Option 1. Solve for y sequentially

We could proceed as we did in the Euler method, where we evaluate y at t_1, then use that result to generate y at t_2, etc. through repeated applications of equation (10). This is totally legitimate and will lead to the correct approximation of the solution. However, we will see in subsequent applications, that there is an alternate method for the solution of the ODE which is more attractive.

Option 2. Solve for all y simultaneously

Consider that we divide the independent variable, t, into n intervals, each of size

$$\Delta t = \frac{(t_f - t_o)}{n} \tag{13}$$

where is the initial time from the initial condition in equation (2) and t_f is the final time, beyond which we are no longer interested in the solution of the ODE. Our approximate solution will be evaluated at n+1 points, the initial condition and the n subsequent values of t. If we designate the solution of the ODE at each of these points as y_i for $i = 1$ to n+1, then we can write the following set equations:

$$y_1 = y_o \qquad \text{for } i = 1$$

$$A_{i,i} y_i = -A_{i,i-1} y_{i-1} + B_i \qquad \text{for } i = 2 \text{ to } n+1 \tag{10}$$

This is a system of linear algebraic equations. It can be written in matrix form as:

$$\underline{\underline{A}}\,\underline{y} = \underline{B} \tag{14}$$

where the vector $\underline{B}$ is principly defined by equation (12), except for the first entry which is the initial condition,

$$\underline{B} = \begin{bmatrix} y_o \\ B_2 \\ \vdots \\ B_{n+1} \end{bmatrix} \tag{15}$$

and where the matrix $\underline{\underline{A}}$ is principly defined by equation (11), except for the first row which is the initial condition,

$$\underline{\underline{A}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ A_{2,1} & A_{2,2} & 0 & 0 \\ 0 & \ddots & \ddots & 0 \\ 0 & 0 & A_{n+1,n} & A_{n+1,n+1} \end{bmatrix} \tag{16}$$

Thus, we have transformed the numerical solution of a linear ODE into the solution of a system of linear algebraic equations, which we know how to solve.

The transformation of ODEs to linear algebraic equations through the discretization of the independent and dependent variables is a commonly encountered transformation. It is one that is used ubiquitously through-out the solution of PDEs and Integral Equations. Therefore, it was educational to introduce the concept here, even though we would likely never use this methodology to solve a single linear ODE as we did here.

Second Order Method for a System of Generic first order ODEs

A general system of first order ODEs can be expressed as

$$\underline{g}(\underline{y}, \frac{d\underline{y}}{dt}, t) = 0 \tag{17}$$

with the initial conditions

$$\underline{y}(t_o) = \underline{y}_o \tag{18}$$

We will assume for the time being that we can rearrange this general form of equation (17) into a form where the derivatives can be isolated on the left hand side of the equation, such that we can write

$$\frac{d\underline{y}}{dt} = \underline{f}(\underline{y}, t) \tag{19}$$

In the general case (the equations are either linear or nonlinear in $\underline{y}$, we can write the straightforward multi-equation analog of Heun's method. We again use Euler's method to predict the value of the function,

$$\underline{y}_i^p = \underline{y}_{i-1} + (t_i - t_{i-1})\underline{\tilde{f}} = \underline{y}_{i-1} + (t_i - t_{i-1})\underline{f}(\underline{y}_{i-1}, t_{i-1}) \tag{20}$$

followed by the correction step

$$\underline{y}_i = \underline{y}_{i-1} + (t_i - t_{i-1})\underline{\tilde{f}} = \underline{y}_{i-1} + (t_i - t_{i-1})\frac{1}{2}\left[\underline{f}(\underline{y}_{i-1}, t_{i-1}) + \underline{f}(\underline{y}_i^p, t_i)\right] \tag{21}$$

As we did in the single equation case, we sequentially solve for each $\underline{y}_i$ based on $\underline{y}_{i-1}$.

Second Order Method for a System of Linear ODEs

We can also repeat the derivation for a more accurate second order method if every equation in the system of ODEs is linear. In fact, if the equations are linear, we don't even need to make the approximation that we can isolate the derivatives on the left hand side of the equation, as we did above in the general solution. If the equations are linear, the general system

$$\underline{g}(\underline{y}, \frac{d\underline{y}}{dt}, t) = 0 \tag{17}$$

can be written as

$$\underline{\underline{C}}(t)\frac{d\underline{y}}{dt} = \underline{\underline{A}}(t)\underline{y}(t) + \underline{B}(t) \tag{22}$$

Equation (22) would be identical to equation (19) if $\underline{\underline{C}}(t)$ were the identity matrix. In solving a system of mass and energy balances, this is frequently the case.

The derivation of the method, now becomes more complicated. We need three indices on the elements of $\underline{\underline{A}}(t)$. The first index indicates the equation. The second index indicates the variable. The third index indicates the time step, once we have performed the discretization of time necessary to obtain the numerical solution. We will use the notation $A_{i,j}^{(k)}$ to designate the coefficient (the time functionality is now implicit) of the j^{th} variable in the i^{th} equation at discretized time t_k. Similarly, $\underline{\underline{A}}^{(k)}$ represents the entire matrix of coefficients at discretized time t_k.

If $\underline{\underline{C}}$ is a constant matrix, we can discretize equation (22) as

$$\underline{\underline{C}}\frac{\underline{y}^{(k)} - \underline{y}^{(k-1)}}{(t_k - t_{k-1})} = \frac{1}{2}\left[\underline{\underline{A}}^{(k-1)}\underline{y}^{(k-1)} + \underline{B}^{(k-1)} + \underline{\underline{A}}^{(k)}\underline{y}^{(k)} + \underline{B}^{(k)}\right] \tag{23}$$

Once again, you can proceed to solve this problem sequentially or simultaneously.

Option 1. Solve for $\underline{y}$ sequentially

We can rearrange equation (23) to isolate our vector of unknowns $\underline{y}^{(k)}$

$$\left[\frac{1}{(t_k - t_{k-1})}\underline{\underline{C}} - \frac{1}{2}\underline{\underline{A}}^{(k)}\right]\underline{y}^{(k)} = \left[\frac{1}{(t_k - t_{k-1})}\underline{\underline{C}} - \frac{1}{2}\underline{\underline{A}}^{(k-1)}\right]\underline{y}^{(k-1)} + \underline{B}^{(k)} + \underline{B}^{(k-1)} \quad (24)$$

Everything on the right-hand side of equation (24) is known. You invert and solve for $\underline{y}^{(k)}$.

Option 2. Solve for all $\underline{y}^{(k)}$ simultaneously

We could write equation (24) for all values of k from 1 to n+1 (for a discretization of t involving n intervals). We then have a system of m ODEs, we now have a system of m(n+1) linear algebraic equations. We could invert this larger matrix if we so chose. In solving all of the equations simultaneously, we would have to create a single matrix of unknowns. We have some freedom as to how we choose to order our unknowns. Two possible arrangements for a system with m equations and n time intervals are shown below.

$$\underline{Y} = \begin{bmatrix} y_1^{(1)} \\ y_1^{(2)} \\ \vdots \\ y_1^{(n+1)} \\ y_2^{(1)} \\ y_2^{(2)} \\ \vdots \\ y_2^{(n+1)} \\ \vdots \\ y_m^{(1)} \\ y_m^{(2)} \\ \vdots \\ y_m^{(n+1)} \end{bmatrix} \qquad \underline{Y} = \begin{bmatrix} \underline{y}^{(1)} \\ \underline{y}^{(2)} \\ \vdots \\ \underline{y}^{(n)} \\ \underline{y}^{(n+1)} \end{bmatrix} \quad (25)$$

In the first arrangement, the unknowns are grouped by variable. In the second arrangement, the unknowns are grouped by time. The latter form is preferable for two reasons. First, it reduces the bandwidth of the resulting matrix, which equates to a quicker computation. Second, it also facilitates the mathematical description of the system. Here we assume that the vector of unknowns takes the form of the second arrangement.

We then can write a system of m(n+1) linear algebraic equations of the form

$$\underline{\underline{A}}^{*}\,\underline{Y} = \underline{B}^{*} \tag{26}$$

where, in order to present a compact description of the matrix and vector in equation (26), we first rewrite equation (24) as

$$\underline{\underline{M}}_{k,k}\,\underline{y}^{(k)} = -\underline{\underline{M}}_{k,k-1}\underline{y}^{(k-1)} + \underline{B}_k^{*} \tag{27}$$

where the matrix, $\underline{\underline{M}}_{k,k}$, and the vector, $\underline{B}_k^{*}$, are defined by comparison to equation (24).

The matrix and vector used in equation (26) are related to the smaller matrices and vectors used in equation (27) by the following equations:

$$\underline{\underline{A}}^{*} = \begin{bmatrix} \underline{\underline{I}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{M}}_{2,1} & \underline{\underline{M}}_{2,2} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{M}}_{3,2} & \underline{\underline{M}}_{3,3} & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} & \ddots & \ddots & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{M}}_{n+1,n} & \underline{\underline{M}}_{n+1,n+1} \end{bmatrix} \tag{28}$$

and

$$\underline{B}^{*} = \begin{bmatrix} \underline{y}_o \\ \underline{B}_2^{*} \\ \underline{B}_3^{*} \\ \vdots \\ \underline{B}_{n+1}^{*} \end{bmatrix} \tag{29}$$

Thus, we have shown how we can transform the numerical solution of a system of linear first-order ODEs into the solution of a system of linear algebraic equations.