

An Overly Simplified and Brief Review of Differential Equation Solution Methods

We will be dealing with initial or boundary value problems. A typical initial value problem has the form

$$y' + y = 0 \quad y(0) = 1$$

A typical boundary value problem looks like

$$y'' + y = 0 \quad y(0) = 1 \quad y(1) = 2$$

1. Some Common Exact Solution Methods for Differential Equations

a) Integrating Factors

Equations can often be solved using a chain rule, i.e.

$$\begin{aligned} x \frac{dy}{dx} + 2y &= x^3 & x \left[x \frac{dy}{dx} + 2y \right] &= x^4 & x^2 \frac{dy}{dx} + 2xy &= x^4 \\ \frac{d}{dx} [x^2 y] &= x^4 & x^2 y &= \frac{x^5}{5} + C & y &= \frac{x^3}{5} + \frac{C}{x^2} \end{aligned}$$

The “x” which multiplies the second equation in the above list is an *integrating factor*. Integrating factors may be found by trial and error, or in the following general form:

$$\frac{dy}{dx} + p(x)y = q(x) \quad \exp \int p(x)dx \frac{dy}{dx} + p(x) \exp \int p(x)dx y = q(x) \exp \int p(x)dx$$

since this equation reduces to

$$\frac{d}{dx} \left[y \exp \int p(x)dx \right] = q(x) \exp \int p(x)dx$$

This last equation can be integrated and solved for y. Note that the equation could involve higher derivatives, providing that they occur in a combination which allows a chain rule to be formed.

b) Linear Equations

Linear equations of any order can be solved by the method outlined below. We will typically work with first or second order equations. **This method only works for linear equations with constant coefficients.** Taking the following second order equation as an example

$$y'' + ay' + by = c$$

Linear equations have a homogeneous and a particular solution. The total solution is the sum of these two, i.e.

$$y = y_H + y_P$$

The homogeneous solution satisfies the equation with no forcing

$$y_H'' + ay_H' + by_H = 0 \quad (1)$$

The particular solution is any solution of the original equation with forcing

$$y_P'' + ay_P' + by_P = c$$

The particular solution can be found by trial and error and is often a constant or other simple function. In the above example, guessing a constant works because the derivatives are zero, i.e.

$$0 + 0 + by_P = c \quad y_P = \frac{c}{b}$$

The homogeneous solution is usually found by guessing the following solution form

$$y_H \propto e^{kx}$$

Substitution into the original equation (1) gives the *characteristic equation*

$$k^2 e^{kx} + a k e^{kx} + b e^{kx} = 0 \quad k^2 + ak + b = 0$$

This equation is a polynomial of the same order as the differential equation. One linearly independent solution is given by each root of the equation. So, a third order differential equation (i.e. the highest derivative is a third derivative) will have a characteristic equation with three roots and hence three solutions. The above equation has the two roots

$$k = \frac{1}{2} \left[-a \pm \sqrt{a^2 - 4b} \right]$$

and two linearly independent solutions

$$y_H = \exp\left[\frac{1}{2} \left[-a + \sqrt{a^2 - 4b} \right] x\right], \exp\left[\frac{1}{2} \left[-a - \sqrt{a^2 - 4b} \right] x\right]$$

Therefore, the full homogeneous solution is

$$y_H = C_1 \exp\left[\frac{1}{2} \left[-a + \sqrt{a^2 - 4b} \right] x\right] + C_2 \exp\left[\frac{1}{2} \left[-a - \sqrt{a^2 - 4b} \right] x\right]$$

and the complete solution is the sum of the homogeneous and particular solutions, i.e.

$$y = \frac{c}{b} + C_1 \exp\left[\frac{1}{2} \left[-a + \sqrt{a^2 - 4b} \right] x\right] + C_2 \exp\left[\frac{1}{2} \left[-a - \sqrt{a^2 - 4b} \right] x\right]$$

The constants C_1 and C_2 are found by applying boundary conditions. In general, The above solution is given in terms of complex exponentials and these may be simplified to yield a combination of real exponential functions (i.e. of the form $\exp(x)$ and $\exp(-x)$) and trigonometric functions (i.e. of the form $\sin(x)$ and $\cos(x)$) or various combinations of these functions, depending on the values of a and b . If the discriminant is positive, i.e. if

$$a^2 - 4b > 0$$

then the arguments of the exponentials are real and the solutions are real exponentials. If the discriminant is negative then the solution takes the form

$$y = \frac{c}{b} + C_1 \exp\left[\frac{1}{2} \left[-a + i\sqrt{|a^2 - 4b|} \right] x\right] + C_2 \exp\left[\frac{1}{2} \left[-a - i\sqrt{|a^2 - 4b|} \right] x\right]$$

To find the final solution, simply carry out the complex algebra, remembering that the complex exponential is defined to be

$$\exp(i\theta) = \cos \theta + i \sin \theta$$

Here, if $a=0$ then the solution will be a combination of sines and cosines. However, if a is non-zero then the solution will involve a mixture of exponential and trigonometric functions. Note

that if the roots are complex then they will occur as complex conjugate pairs, providing that the coefficients of the characteristic equation are real.

c) Degenerate Roots

There are two main issues that we will encounter which requires the above picture to be modified. The first occurs when the characteristic equation for the homogeneous problem has degenerate roots, i.e. the characteristic equation

$$a_0k^N + a_1k^{N-1} + \dots + a_N = 0$$

can be factored into the form

$$(k - \lambda_1)^p (b_0k^M + \dots + b_M) = 0$$

in which the root λ_1 occurs p times. The homogeneous solution then takes the form

$$y = C_1e^{\lambda_1x} + C_2xe^{\lambda_1x} + \dots + C_px^{p-1}e^{\lambda_1x} + D_1e^{\lambda_2x} + \dots + D_{M-1}e^{\lambda_Mx}$$

In other words, the linearly independent functions associated with the degenerate root are:

$$e^{\lambda_1x}, xe^{\lambda_1x}, x^2e^{\lambda_1x}, x^3e^{\lambda_1x}, \dots, x^pe^{\lambda_1x}$$

For example, the characteristic equation

$$(k - 2)^2(k - 3)(k - 4) = 0$$

Has the homogeneous solution

$$y = C_1e^{2x} + C_2xe^{2x} + D_1e^{3x} + D_2e^{4x}$$

d) More General Particular Solutions

The second complication occurs when what would appear to be a reasonable guess for the particular solution is also a solution of the homogeneous equation. This typically happens in a problem which has an exponential or a trigonometric function as a forcing term on the right hand side of the equation, such as

$$y'' - y = e^x$$

Here, the characteristic equation and the homogeneous solution are

$$k^2 - 1 = 0 \quad k = \pm 1 \quad y_H = C_1 e^x + C_2 e^{-x}$$

A reasonable guess for the particular solution is

$$y_p = Ae^x$$

This is “reasonable” because the exponential function value as well as the second derivative of the exponential both produce an exponential as the result, which (we guess) can be matched to the forcing term on the right hand side of the equation if the coefficient A is chosen correctly. However $Ae^{xp(x)}$ is a solution of the homogeneous equation and so it will always give zero for the left hand side of the equation.

Therefore the particular solution must be something else. In many problems, adding additional functions x , x^2 , x^3 , ... etc. will give a viable particular solution. For example,

$$y_p = Axe^x$$

has the second derivative

$$y_p'' = A(e^x + xe^x)' = A(e^x + e^x + xe^x) = A(2e^x + xe^x)$$

Substituting the function value and the second derivative into the original equation gives

$$y'' - y = A(2e^x + xe^x) - Axe^x = e^x$$

which simplifies to the form

$$2Ae^x = e^x$$

Therefore, the correct choice is $A=1/2$. The same simple method will often work with various exponential and trigonometric forcing terms on the right hand side of the equation (though quadratic and higher functions of x may be needed).

If a simple solution cannot be found quickly by direct observation then the more general *method*

of *undetermined coefficients* and the *method of variation of parameters* may be needed.

2. Some Simple Numerical Solution Methods for Differential Equations

a) Boundary Value Problems

Consider an equation of the form

$$y'' + by' + cy = d \quad y(0) = L \quad y(1) = R$$

where b, c, d, L and R are known constants. The first step involves approximating the derivatives in the equation. Since we will be working with simple equations, we will use the simplest approach, which is to central difference all derivatives using a finite difference method (see Tannehill *et al.* (1997) for other examples of differencing):

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{\Delta x^2} + b \frac{y_{i+1} - y_{i-1}}{2\Delta x} + cy_i = d$$

The grid-points are uniformly spaced with i ranging from 1 to N , and

$$x_i = x_{\min} + (i - 1)\Delta x \quad \Delta x = \frac{x_{\max} - x_{\min}}{N - 1}$$

The differenced form of the equation may be re-arranged to give

$$A_i y_{i-1} + B_i y_i + C_i y_{i+1} = D_i \quad (2)$$

where the coefficients for this particular example are given by

$$A_i = \frac{1}{\Delta x^2} - \frac{b}{2\Delta x} \quad B_i = -\frac{2}{\Delta x^2} + c \quad C_i = \frac{1}{\Delta x^2} + \frac{b}{2\Delta x} \quad D_i = d$$

Note that for this example, these coefficients are all constants. The above equation is inverted using a tri-diagonal solver or *Thomas algorithm*. First, we guess a recursion relation

$$y_i = R_i y_{i-1} + S_i \quad (3)$$

Note that the right boundary condition gives the following result

$$y(1) = y_N = R \quad \text{and} \quad y_N = R_N y_{N-1} + S_N \quad \Rightarrow \quad R_N = 0 \quad S_N = R$$

Now increment the recursion relation (3)

$$y_{i+1} = R_{i+1} y_i + S_{i+1}$$

Substitute this equation into the original difference equation (2) to get

$$A_i y_{i-1} + B_i y_i + C_i [R_{i+1} y_i + S_{i+1}] = D_i$$

and solve for y_i

$$y_i = -\frac{A_i}{B_i + C_i R_{i+1}} y_{i-1} + \frac{D_i - C_i S_{i+1}}{B_i + C_i R_{i+1}}$$

Comparing this equation with the original recursion relation (3) implies that

$$R_i = -\frac{A_i}{B_i + C_i R_{i+1}} \quad S_i = \frac{D_i - C_i S_{i+1}}{B_i + C_i R_{i+1}}$$

In summary, the Thomas algorithm proceeds as follows

1) specify the recursion coefficients at the right boundary using the boundary condition

$$R_N = 0 \quad S_N = R$$

2) Solve for the recursion coefficients by marching from the right to the left boundary

$$R_i = -\frac{A_i}{B_i + C_i R_{i+1}} \quad S_i = \frac{D_i - C_i S_{i+1}}{B_i + C_i R_{i+1}} \quad i = N, \dots, 2$$

3) Specify the function value at the left boundary using the boundary condition

$$y(0) = y_1 = L$$

4) Solve for the function values by marching from the left to the right boundary

$$y_i = R_i y_{i-1} + S_i \quad i = 2, \dots, N$$

Note that the recursion coefficients from step #2 must be stored in arrays so that they can be used

in step #4.

The above algorithm may be generalized to other equations. The recursion relation (3) could be given in terms of i and $i+1$ instead of i and $i-1$ (thereby reversing all the marching directions). The algorithm can be generalized to include multiple equations in multiple unknowns (which is called a *block tri-diagonal algorithm*). The algorithm can also be extended to higher bandwidths, for example the following penta-diagonal equation:

$$A_i y_{i-2} + B_i y_{i-1} + C_i y_i + D_i y_{i+1} + E_i y_{i+2} = F_i$$

All of these generalizations are straightforward and can be found by consistently applying the simple method given above.

b) Nonlinearities

Some equations may involve nonlinearities, which are nonlinear functions of the dependent variable which appear in the equation. Two examples of nonlinear equations are

$$y'' + byy' + cy = d$$

and

$$y'' + by' + cy^2 e^y = d$$

We will use a *Newton linearization* to solve these problems. The Newton linearization assumes that the function value y is close to some guessed value, i.e.

$$y = y^g + \Delta y = y^g + (y - y^g)$$

The function to be linearized is expanded in a Taylor series about the guessed value. For example, the non-linear function might be

$$F(y) = y^2 e^y$$

In general, the Taylor series expansion of this function is given by

$$F(y) = F(y^g + \Delta y) \approx F(y^g) + \Delta y F'(y^g) + \dots$$

For the example problem given above, this yields

$$F(y) = F(y^g + \Delta y) \approx y^{g2}e^{y^g} + (y - y^g)[2y^ge^{y^g} + y^{g2}e^{y^g}] + \dots$$

This produces an approximation for $F(y)$ which is linear in the unknown y . A simpler and more common example are the nonlinearities appearing in the following equation

$$y'' + byy' + cy^2 = d \quad (4)$$

Here, the linearization for the quadratic term is

$$F(y) = y^2 \approx y^{g2} + (y - y^g)[2y^g] + \dots \approx (2y^g)y - y^{g2} \quad (5)$$

The term yy' is expanded using a two-variable Taylor series

$$F(u, v) = F(u^g + \Delta u, v^g + \Delta v) \approx F(u^g, v^g) + \Delta u F_u(u^g, v^g) + \Delta v F_v(u^g, v^g) + \dots$$

where

$$u = y \quad v = y' \quad F(u, v) = uv = yy'$$

The linearization for this particular example is

$$yy' = uv \approx u^g v^g + \Delta u [v^g] + \Delta v [u^g] + \dots \approx u^g v^g + (u - u^g)[v^g] + (v - v^g)[u^g]$$

or

$$yy' \approx y^g y'^g + (y - y^g)[y'^g] + (y' - y'^g)[y^g]$$

$$yy' \approx yy'^g + y^g y' - y^g y'^g \quad (6)$$

The two linearizations (5) and (6) are substituted into the original equation (4) to give

$$y'' + b[yy'^g + y^g y' - y^g y'^g] + c[(2y^g)y - y^{g2}] = d$$

The unknowns involving y are finite differenced to give

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{\Delta x^2} + b \left[y_i y_i'{}^g + y_i^g \frac{y_{i+1} - y_{i-1}}{2\Delta x} - y_i^g y_i'{}^g \right] + c \left[(2y_i^g) y_i - y_i^{g2} \right] = d$$

The equation is then put in the form

$$A_i y_{i-1} + B_i y_i + C_i y_{i+1} = D_i \quad (2')$$

Note that the coefficients will involve the guessed function value and it's derivative, which is

$$y_i'{}^g = \frac{y_{i+1}^g - y_{i-1}^g}{2\Delta x}$$

The equation is solved as before, with an additional iteration loop added to the Thomas algorithm in order to update the guessed function value. In other words, a sensible initial guess is chosen for y_i^g at each point on the grid. With the guessed function value given, the coefficients in equations (2,2') are known at all points on the grid. The equation is then inverted using the Thomas algorithm (which is the same algorithm used for the linear equation). The new solution is used to update the initial guess, i.e.

$$y_i^g = y_i$$

The above process is repeated until the error at every point on the grid falls below some tolerance:

$$\text{error} = |y_i - y_i^g| < \epsilon$$

While it may not be clear that this is the case, the Newton linearization discussed above is equivalent to the Newton method for solving nonlinear equations which is discussed in basic calculus.

c) Runge–Kutta Methods

Runge–Kutta schemes are predictor–corrector schemes developed by eliminating higher order error terms in a Taylor series expansion going from the i to the $i+1$ grid point. There are a wide range of such schemes. For our purposes, a reasonably accurate and easy to implement Runge–Kutta scheme is the Ralston method. The differential equation is assumed to have the form

$$\frac{dy}{dx} = f(y)$$

Note that the right hand side is taken to be a function of only the dependent variable y , this need not be the case (though we will not review the more general case here). The function value at

the i 'th grid-point is used to predict an intermediate function value using

$$y^{(1)} = y_i + \frac{3\Delta x}{4}f(y_i)$$

This intermediate function value should be discarded after use (i.e. it should not be output as one of the solution values). The function value at i and the intermediate function value are used to generate a corrected function value, which in the Ralston scheme is simply the solution at $i+1$:

$$y_{i+1} = \frac{5}{9}y_i + \frac{4}{9}y^{(1)} + \frac{2\Delta x}{3}f(y^{(1)})$$

Note that the right hand side, i.e. $f(y)$, can be linear or nonlinear, the solution algorithm is the same in either case.

In summary, the algorithm proceeds as follows:

- 1) Set the function value at the initial grid-point using the initial condition

$$y_1 = y_i = I$$

- 2) Solve for the intermediate function value

$$y^{(1)} = y_i + \frac{3\Delta x}{4}f(y_i)$$

- 3) Solve for the function value at the next grid-point

$$y_{i+1} = \frac{5}{9}y_i + \frac{4}{9}y^{(1)} + \frac{2\Delta x}{3}f(y^{(1)})$$

- 4) Replace the function value at the old grid-point with the function value at the new grid-point and proceed to the next grid-point, i.e. repeat steps #2 through #4.

$$y_i = y_{i+1}$$

Note that the more general Ralston scheme is

$$\frac{dy}{dx} = f(x, y)$$

The predictor step is given by

$$x^{(1)} = x_i + \frac{3\Delta x}{4} \quad y^{(1)} = y_i + \frac{3\Delta x}{4}f(x_i, y_i)$$

and the corrector step is

$$y_{i+1} = \frac{5}{9}y_i + \frac{4}{9}y^{(1)} + \frac{2\Delta x}{3}f(x^{(1)}, y^{(1)})$$

d) Systems of Equations Using Runge–Kutta Methods

We will often be solving second order differential equations, for example

$$\frac{d^2y}{dx^2} + y \frac{dy}{dx} + y^2 = 0 \quad y(0) = 0 \quad y'(0) = 1$$

To solve this equation as an initial value problem using a Runge–Kutta method requires writing the equation as a system of 1st order equations. For the example, first let

$$g = \frac{dy}{dx}$$

which produces the system

$$\frac{dy}{dx} = g \quad \frac{dg}{dx} = -yg - y^2$$

with the initial conditions

$$y(0) = 0 \quad g(0) = 1$$

Note that these equations have the form

$$\frac{dy}{dx} = f(y)$$

However, y and f are now column vectors, i.e.

$$\frac{dy}{dx} = \frac{d}{dx} \begin{bmatrix} y \\ g \end{bmatrix} = \begin{bmatrix} -yg - y^2 \\ g \end{bmatrix}$$

The Runge–Kutta method given previously, i.e. the predictor step

$$y^{(1)} = y_i + \frac{3\Delta x}{4}f(y_i)$$

and the corrector step

$$y_{i+1} = \frac{5}{9}y_i + \frac{4}{9}y^{(1)} + \frac{2\Delta x}{3}f(y^{(1)})$$

are applied to each component individually, i.e. all the y 's and f 's in the above equation are now column vectors. This requires an additional loop to cycle through each of the components of the column vector (or, for simple problems, the individual components are calculated separately in each step of the algorithm).

e) The Crank–Nicholson Method

Runge–Kutta methods are easy to implement for single and multiple equations. However, they suffer from stability limitations. Using an implicit method usually helps getting around these difficulties, and there are a variety of implicit Runge–Kutta methods which do this.

Here we will present a simple and easy–to–implement implicit method: the *Crank–Nicholson method*. Consider the following equation

$$y' = f(x, y) \quad y(0) = I$$

or, to be more specific, let's say

$$\frac{dy}{dx} = y^2 + \cos(x) \quad y(0) = I$$

The Crank–Nicholson method advances the function value from $i-1$ to i by central differencing the equation about the $i-1/2$ point, i.e.

$$\frac{y_i - y_{i-1}}{\Delta x} = \frac{1}{2}[y_i^2 + y_{i-1}^2] + \frac{1}{2}[\cos(x_i) + \cos(x_{i-1})]$$

The term on the left hand side is a second order accurate central difference about $i-1/2$. The averaging operator used on the right hand side of the equation is also second order accurate about $i-1/2$. Note that the averaging must be applied consistently to all terms in the equation.

The nonlinearity in the equation at the i grid–point must be linearized, which gives

$$\frac{y_i - y_{i-1}}{\Delta x} = \frac{1}{2}[2y_i^g y_i - y_i^{g2} + y_{i-1}^2] + \frac{1}{2}[\cos(x_i) + \cos(x_{i-1})]$$

where y_i^g is a guessed value of y which must be iteratively updated at the i 'th grid–point. In other

words, the value of y^g must be iterated to convergence at the i 'th grid point before the solution proceeds from the i to the $i+1$ grid point.

The equation is now solved for y :

$$y_i = \left[\frac{1}{\Delta x} - y_i^g \right]^{-1} \left[\frac{1}{\Delta x} y_{i-1} + \frac{1}{2} (y_{i-1}^2 - y_i^{g2}) + \frac{1}{2} (\cos(x_i) + \cos(x_{i-1})) \right] \quad (7)$$

In summary, the algorithm proceeds as follows:

1) Set the function value at the initial point using the initial condition

$$y_1 = y_{i-1} = I$$

2) Define the guessed value at the i 'th point using the value at the preceding point

$$y_i^g = y_{i-1}$$

3) Solve for y_i

$$y_i = \left[\frac{1}{\Delta x} - y_i^g \right]^{-1} \left[\frac{1}{\Delta x} y_{i-1} + \frac{1}{2} (y_{i-1}^2 - y_i^{g2}) + \frac{1}{2} (\cos(x_i) + \cos(x_{i-1})) \right]$$

4) Update the guessed value

$$y_i^g = y_i$$

5) Repeat steps #3 and #4 until the solution converges (at the i 'th point!)

6) Reset the function value at the old grid point

$$y_{i-1} = y_i$$

7) Return to step #2 (note that there is no need to explicitly update the guessed value, since it should already be equal to the function value at convergence)

Note that if a system of equations is being solved then a matrix inversion is required at the i 'th grid point. This will be discussed later. Also note that the general form of the algorithm is

$$\frac{dy}{dx} = f(x, y)$$

The differencing about $i-1/2$ gives

$$\frac{y_i - y_{i-1}}{\Delta x} = \frac{1}{2} [f(x_i, y_i) + f(x_{i-1}, y_{i-1})]$$

Linearization of the right hand side at the i 'th grid point gives

$$\frac{y_i - y_{i-1}}{\Delta x} = \frac{1}{2} [f(x_i, y_i^g + y_i - y_i^g) + f(x_{i-1}, y_{i-1})]$$

or

$$\frac{y_i - y_{i-1}}{\Delta x} = \frac{1}{2} \left[f(x_i, y_i^g) + (y_i - y_i^g) \frac{\partial f}{\partial y}(x_i, y_i^g) + \dots + f(x_{i-1}, y_{i-1}) \right]$$

in which case

$$y_i = \frac{1}{2} \left[\frac{1}{\Delta x} - \frac{1}{2} \frac{\partial f}{\partial y}(x_i, y_i^g) \right]^{-1} \left[2 \frac{y_{i-1}}{\Delta x} + f(x_i, y_i^g) - y_i^g \frac{\partial f}{\partial y}(x_i, y_i^g) + f(x_{i-1}, y_{i-1}) \right]$$

f) Systems of Equations Using the Crank–Nicholson Method

Recall that a second order differential equation such as

$$\frac{d^2y}{dx^2} + y \frac{dy}{dx} + y^2 = 0 \quad y(0) = 0 \quad y'(0) = 1$$

can be written as a system of first order equations of the form

$$\frac{dy}{dx} = f$$

Taken component by component, these equations are

$$\frac{dy_1}{dx} = f_1(y_1, y_2) \quad \frac{dy_2}{dx} = f_2(y_1, y_2)$$

These two equations are differenced about the $i-1/2$ grid–point, to give

$$\frac{y_{1,i} - y_{1,i-1}}{\Delta x} = \frac{1}{2} [f_1(x_i, y_{1,i}, y_{2,i}) + f_1(x_{i-1}, y_{1,i-1}, y_{2,i-1})]$$

and

$$\frac{y_{2,i} - y_{2,i-1}}{\Delta x} = \frac{1}{2} [f_2(x_i, y_{1,i}, y_{2,i}) + f_2(x_{i-1}, y_{1,i-1}, y_{2,i-1})]$$

Both equations are Newton linearized (but now in two variables instead of one), giving

$$\frac{y_{1,i} - y_{1,i-1}}{\Delta x} = \frac{1}{2} \left[f_1(x_i, y_{1,i}^g, y_{2,i}^g) + (y_{1,i} - y_{1,i}^g) \frac{\partial f_1}{\partial y_1}(x_i, y_{1,i}^g, y_{2,i}^g) + (y_{2,i} - y_{2,i}^g) \frac{\partial f_1}{\partial y_2}(x_i, y_{1,i}^g, y_{2,i}^g) + f_1(x_{i-1}, y_{1,i-1}, y_{2,i-1}) \right]$$

and

$$\frac{y_{2,i} - y_{2,i-1}}{\Delta x} = \frac{1}{2} \left[f_2(x_i, y_{1,i}^g, y_{2,i}^g) + (y_{1,i} - y_{1,i}^g) \frac{\partial f_2}{\partial y_1}(x_i, y_{1,i}^g, y_{2,i}^g) + (y_{2,i} - y_{2,i}^g) \frac{\partial f_2}{\partial y_2}(x_i, y_{1,i}^g, y_{2,i}^g) + f_2(x_{i-1}, y_{1,i-1}, y_{2,i-1}) \right]$$

This produces a linear system of two equations in the two unknowns, y_1 and y_2 , at the i 'th grid-point. The solutions for y_1 and y_2 , can be found using a 2x2 matrix inversion formula, which results in a system of equations similar in form to eqn. (7). Remember that the Newton linearization must be iterated to convergence at each i 'th grid-point before proceeding to the next grid point. Other than the matrix inversion, the structure of the algorithm is the same as that discussed previously.

References

Tannehill, J.C., Anderson, D.A., and Pletcher, R.H., 1997, *Computational Fluid Mechanics and Heat Transfer*, 2nd Ed., Taylor & Francis, 792pp.