(Note that n is used to index discrete steps in time and k is used to denote discrete iterations within a time step).

1 Backwards Euler

The Backwards Euler method is:

$$x_{n+1} = x_n + \Delta t \left[g\left(x_{n+1}\right) \right] \tag{1}$$

Or for a system of ODEs:

$$\mathbf{X}_{n+1} = \mathbf{X}_n + \Delta t \left[G\left(\mathbf{X}_{n+1}\right) \right] \tag{2}$$

2 Newton's Method

Newton's method is given as:

$$x^{k+1} = x^k - \frac{f(x^k)}{f'(x^k)} \tag{3}$$

For a system of equations Newton's method becomes

$$\mathbf{X}^{k+1} = \mathbf{X}^k - J^{-1}(\mathbf{X}^k) F(\mathbf{X}^k) \tag{4}$$

Where J is the Jacobian matrix:

$$J(\mathbf{X}^k) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{5}$$

In general the product $J^{-1}(\mathbf{X}^k)F(\mathbf{X}^k)$ may be found by solving

$$J(\mathbf{X}^k)\mathbf{S}^k = -F(\mathbf{X}^k) \tag{6}$$

For $\mathbf{S}^k$. Then the Newton iteration is:

$$\mathbf{X}^{k+1} = \mathbf{X}^k + \mathbf{S}^k \tag{7}$$

As the partial derivatives in the Jacobian may not be able to be determined analytically a finite difference method can be utilised. For example, a central difference method:

$$\frac{\partial f_n}{\partial x_n} \approx \frac{f_n(x_0, \ldots, x_{n-1}, x_n + h) - f_n(x_0, \ldots, x_{n-1}, x_n - h)}{2h} \tag{8}$$

3 Combining the Backwards Euler Method and Newton's Method

Rearranging the backwards Euler equation:

$$\mathbf{X}_{n+1} = \mathbf{X}_n + \Delta t \left[G\left(\mathbf{X}_{n+1}\right) \right] \tag{9a}$$

$$F(\mathbf{X}_{n+1}) = \mathbf{X}_{n+1} - \mathbf{X}_n - \Delta t \left[G\left(\mathbf{X}_{n+1}\right) \right] = 0 \tag{9b}$$

When solving using Newton's method the X_n term in the backwards Euler method at the start of the iteration process and will be referred to as X_0.

Substituting into Newton's method:

$$\mathbf{X}^{k+1} = \mathbf{X}^k - J^{-1}(\mathbf{X}^k) F(\mathbf{X}^k) \tag{10a}$$

$$\mathbf{X}^{k+1} = \mathbf{X}^k - J^{-1}(\mathbf{X}^k) \left(\mathbf{X}^k - \mathbf{X}_0 - \Delta t \left[G(\mathbf{X}^k) \right] \right) \tag{10b}$$

The start index is denoted by $k = 0$ and the starting term X^0 may be found by using the forward Euler method

$$\mathbf{X}^0 = \mathbf{X}_{n-1} + \Delta t \left[G(\mathbf{X}_{n-1})\right] \tag{11a}$$

with X_{n-1} the result of the previous step.

4 Solution with Aircraft State-Space Equations

In matrix form the aircraft state-space equations are given as:

$$\dot{\mathbf{X}} = A\mathbf{X} + B\mathbf{U} \tag{12}$$

The backwards Euler method is then:

$$F(\mathbf{X}_{n+1}) = \mathbf{X}_{n+1} - \mathbf{X}_n - \Delta t \left[G\left(\mathbf{X}_{n+1}\right)\right] = 0 \tag{13a}$$
$$F(\mathbf{X}_{n+1}) = \mathbf{X}_{n+1} - \mathbf{X}_n - \Delta t \left[A\mathbf{X}_{n+1} + B\mathbf{U}\right] = 0 \tag{13b}$$

So that Newton's method is:

$$\mathbf{X}^{k+1} = \mathbf{X}^k - J^{-1}(\mathbf{X}^k)\left(\mathbf{X}^k - \mathbf{X}_0 - \Delta t\left[A\mathbf{X}^k + B\mathbf{U}\right]\right) \tag{14}$$

For the SPO model the Jacobian is:

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix} \tag{15}$$

Expanding $F(\mathbf{X}_{n+1})$:

$$F(\mathbf{X}_{n+1}) = \mathbf{X}^k - \mathbf{X}_0 - \Delta t\left[A\mathbf{X}^k + B\mathbf{U}\right] = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} {x_1}^k \\ {x_2}^k \end{bmatrix} - \begin{bmatrix} x_{01} \\ x_{02} \end{bmatrix} - \Delta t \left[\begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}\begin{bmatrix} {x_1}^k \\ {x_2}^k \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} u\right] \tag{16}$$

So that:

$$f_1 = {x_1}^k - x_{01} - \Delta t\left[a_1{x_1}^k + a_2{x_2}^k + b_1u\right] \tag{17a}$$
$$f_2 = {x_2}^k - x_{02} - \Delta t\left[a_3{x_1}^k + a_4{x_2}^k + b_2u\right] \tag{17b}$$

$$\frac{\partial f_1}{\partial x_1} = \frac{f_1(x_2, x_1 + h) - f_1(x_2, x_1 - h)}{2h} \tag{18a}$$
$$\frac{\partial f_1}{\partial x_2} = \frac{f_1(x_1, x_2 + h) - f_1(x_1, x_2 - h)}{2h} \tag{18b}$$
$$\frac{\partial f_2}{\partial x_1} = \frac{f_2(x_2, x_1 + h) - f_2(x_2, x_1 - h)}{2h} \tag{18c}$$
$$\frac{\partial f_2}{\partial x_2} = \frac{f_2(x_1, x_2 + h) - f_2(x_1, x_2 - h)}{2h} \tag{18d}$$

h should be chosen to be small e.g. $h = 1e - 6$.

Given the inverse of a 2x2 matrix can be simply found, J^{-1} is:

$$J^{-1} = \frac{1}{\frac{\partial f_1}{\partial x_1}\frac{\partial f_2}{\partial x_2} - \frac{\partial f_1}{\partial x_2}\frac{\partial f_2}{\partial x_1}} \begin{bmatrix} \frac{\partial f_2}{\partial x_2} & -\frac{\partial f_1}{\partial x_2} \\ -\frac{\partial f_2}{\partial x_1} & \frac{\partial f_1}{\partial x_1} \end{bmatrix} \tag{19}$$

For when u is dependent on the output of the state-space system (e.g feedback loop with PID controller), u should be recalculated at each iteration step in Newton's method.