AN IMPROVED NUMERICAL INTEGRATION METHOD FOR FLIGHT SIMULATION

R.M. Howe*
The University of Michigan
Ann Arbor, Michigan
Applied Dynamics International
Ann Arbor, Michigan

A89-48414

Abstract

In this paper a modified form of Euler integration is described which, when applied to the six-degree of freedom flight equations, retains and enhances many of the advantages of AB-2 integration and at the same time eliminates the disadvantages. The scheme is based on the Euler integration formula, but with the state-variable derivative represented at the midpoint of each integration step. In this case the conventional first-order Euler method actually becomes second order, with a very small accompanying error coefficient. To apply this method to the six-degree-of-freedom flight equations it is necessary to define velocity states at half-integer frame times and position states at integer frame times. It is shown through dynamic error analysis that the modified Euler method has an error coefficient which is one-tenth that associated with AB-2. The method also exhibits minimal output delay in response to transient inputs. The modified Euler method may also be useful in the integration of state and costate equations in real-time mechanization of Kalman filters for navigation and control systems.

1.Introduction

The ever increasing complexity of the math models used as a basis for real time flight simulation has continued to apply pressure on digital processor speed requirements for such simulations. More effective numerical integration algorithms can help relieve some of this pressure. The most popular method currently in use for flight simulation is the Adams-Bashforth second-order predictor method, usually referred to as AB-2. Its advantages include second-order accuracy with respect to integration step size, only one required pass through the state equations per integration step, and compatibility with real-time inputs. Disadvantages include stability problems associated with extraneous roots and response delays of one or two frames following transient inputs.

In the next section we consider AB-2 along with two other second-order integration methods suitable for real-time simulation of dynamic systems. The first is a two-pass real-time predictor-corrector algorithm and the second is a single-pass version of the same. We then introduce the modified-Euler method and describe its application to the flight equations. The accuracy of the various methods is compared by means of time-history plots of the dynamic error in simulating aircraft response to a control-surface input.

2. Some Real-time Second-order Integration Algorithms

Consider first the AB-2 predictor integration method applied to the state equation given by

$$\dot{X} = F[X, U(t)] \tag{1}$$

Here X is the state vector, U is the input vector, and h is the integration step size. The standard AB-2 predictor algorithm is the following:

$$X_{n+1} = X_n + h\,(1.5\,F_n - .5\,F_{n-1}) \tag{2}$$

where $X_n = X(nh)$ and

$$F_n = F(X_n, U_n), \quad F_{n-1} = F(X_{n-1}, U_{n-1}) \tag{3}$$

The AB-2 formula in Eq. (2) is derived from the area under a linear extrapolation from F_n to F_{n+1} based on F_n and F_{n-1}.

From Z transform theory it can be shown that the numerical integration formula in general has a transfer function for sinusoidal inputs which takes the form [1]

$$H_I^*(e^{j\omega h}) \cong \frac{1}{j\omega[1 + e_I(j\omega h)^k]}, \quad \omega h << 1 \tag{4}$$

where for AB-2 integration, $k = 2$ and $e_I = 5/12$. The term $e_I(j\omega h)^k$ represents the error in the integrator transfer function compared with the ideal continuous integrator transfer function, $1/j\omega$. Based on the integrator model of Eq. (4) the transfer function gain and phase error in simulating any order of dynamic system, when quasi-linearized, can easily be obtained [1]. It can also be shown that the fractional error in any characteristic root as a result of integration truncation errors is given by

$$e_\lambda = \frac{\lambda^* - \lambda}{\lambda} \cong -\,e_I(\lambda h)^k, \quad |\lambda h| << 1 \tag{5}$$

where λ is the continuous system root and λ^* is the equivalent root for the digital simulation. Thus it is apparent that the order k and error coefficient e_I for any given integration algorithm can be used to predict the dynamic errors that will be introduced into a simulation because of the finite step size h when using that algorithm. It should be noted that this methodology of error analysis is not applicable to multiple-pass integration methods, such as the Runge-Kutta algorithms, where different orders of integration algorithms are used in the various derivative evaluation passes that constitute a single integration step. It is applicable to all of the real-time methods considered in this paper.

As indicated in Eq. (1), the state variable derivative F will in general depend on the state X. Since the AB-2 algorithm in Eq. (2) involves the past derivative F_{n-1}, the next state X_{n+1} will depend on the past state X_{n-1} as well as the current state X_n. For this reason the AB-2 method introduces one extraneous state per integration. The characteristic roots

Professor of Aerospace Engineering
Associate Fellow, AIAA

corresponding to these extraneous states damp rapidly for small integration step sizes, in which case they do not contribute significant errors to the simulation. However, when the step size becomes large, the extraneous roots can cause instability. In particular, for a negative real root λ, instability occurs when $\lambda h < -1$. This means that when AB-2 integration is used, the step size must be kept less than the shortest time constant in the system being simulated. Stability charts in the complex λh plane show the allowable step sizes when the roots of the continuous system are complex [2].

Consider next the Adams-Moulton two-pass predictor-corrector algorithm. Here the AB-2 method is used on the first pass to compute an estimate, X'_{n+1}, for the n+1 state. From this state and the input U_{n+1} the estimated derivative F'_{n+1} is in turn calculated. The corrector pass then computes X_{n+1} using F_n and F'_{n+1} with the following formula:

$$X_{n+1} = X_n + .5h\,(F'_{n+1} + F_n) \tag{6}$$

If the estimate F'_{n+1} in Eq. (6) is repaced by the true derivative F_{n+1}, the formula represents implicit trapezoidal integration. It turns out that the explicit AM-2 method has the same asymptotic error coefficient, $e_I = -1/12$, as the implicit trapezoidal integration which it approximates. The order $k = 2$.

It should be noted that the AM-2 algorithm requires two passes through the state equations per integration step. At the beginning of the second pass, the input U_{n+1} is used in the calculation of the derivative estimate F'_{n+1}. But U_{n+1} will not be available in real time until the completion of the second pass. Hence AM-2 is not compatible with real-time inputs.

However, a second-order predictor-corrector algorithm suitable for real-time inputs can be constructed using the concept behind modified Euler integration. In modified Euler integration the state-variable derivative is represented at the midpoint of the integration step. Thus the integration formula becomes

$$X_{n+1} = X_n + h\,F_{n+1/2} \tag{7}$$

If the derivative F is a function of the state X, which is normally the case, then it is necessary to compute an estimate for the state $X_{n+1/2}$ in order to evaluate $F_{n+1/2}$. In real-time Runge-Kutta 2 this is accomplished using Euler integration with a step size of $h/2$. The resulting real-time RK-2 integrator error coefficient $e_I = 1/6$. In the second-order real-time predictor-corrector method, denoted here as RTAM-2, the estimate for $X_{n+1/2}$ is computed using a second-order predictor algorithm. This leads directly to the following difference equations [3]:

$$X'_{n+1/2} = X_n + h(\tfrac{5}{8}F_n - \tfrac{1}{8}F_{n-1}) \tag{8}$$

$$X_{n+1} = X_n + hF'_{n+1/2} \tag{9}$$

where

$$F'_{n+1/2} = F(X'_{n+1/2}, U_{n+1/2}) \tag{10}$$

The predictor formula in Eq. (8) for $X_{n+1/2}$ is derived from the area under a linear extrapolation based on F_n and F_{n-1}. The RTAM-2 given by Eqs. (8), (9) and (10) requires the input U_n at the start of the first pass and $U_{n+1/2}$ at the start of the second pass, both compatible with real time. Here the RTAM-2 integrator error coefficient $e_I = 1/24$, compared with -1/12 for standard AM-2. For a negative real root λ, instability occurs when $\lambda h < -2$. In the complex λh plane the overall stability boundary is slightly larger than the stability boundary for standard AM-2.

Thus the modified AM-2 integration denoted here as RTAM-2 can be used as a general, real-time integration method for simulating nonlinear systems. For small step sizes h it is twice as accurate as either implicit trapezoidal integration or standard AM-2 integration, neither of which are compatible with real-time inputs. It is ten times more accurate than AB-2 integration. It should be realized, however, that the RTAM-2 considered here is a two pass per step method. Thus it will normally take about twice as long for computer execution as AB-2. When this speed differential is taken into account, the modified AM-2 still exhibits 2.5 times the dynamic accuracy of AB-2 based on the approximate asymptotic formulas for small step size. In a real-time simula-tion the intermediate state $X'_{n+1/2}$ can be used as a real-time output which has full second-order predictor accuracy for the step size $h/2$. Use of both $X'_{n+1/2}$ and X_{n+1}, then, provides outputs at the sample rate of a single-pass method, even though a two-pass method is being utilized. Note also that the method uses two input samples per frame, U_n and $U_{n+1/2}$.

The final integration method considered in this section is a single-pass version of the RTAM-2 algorithm described above. The method computes the state-variable derivative F only at integer frame times rather than both half-integer and integer frame times, as in Eqs. (8), (9) and (10). Values of the state X at half-integer frame times are computed using the modified-Euler algorithm, while values of X at integer frame times are computed using a second-order predictor. The difference equations are the following [3]:

$$X_{n+1/2} = X_{n-1/2} + hF'_n \tag{11}$$

$$X'_{n+1} = X_{n+1/2} + h(\tfrac{7}{8}F'_n - \tfrac{3}{8}F'_{n-1}) \tag{12}$$

where

$$F'_n = F(X'_n, U_n) \tag{13}$$

The predictor formula in Eq. (12) is derived from a linear extrapolation based on F'_n and F'_{n-1}. The SPRTAM-2 given by Eqs. (11), (12) and (13) requires the input U_n at the start of the nth integration step, which is compatible with real time. From Z transform theory we can show that the SPRTAM-2 integrator error coefficient $e_I = 1/24$, the same as that obtained previously for the two-pass RTAM-2. Since it executes twice as fast (one pass per integration step versus two for RTAM-2) and the dynamic errors are proportional to h^2, it will in general be four times as accurate.

The price we pay for the accuracy increase in the single-pass predictor-corrector method is reduced stability. For a negative real root λ, instability occurs for $\lambda h < -4/7$, compared with $\lambda h < -2$ for RTAM-2 (actually, $\lambda h < -1$ if one takes into account the doubled execution time for the two-pass RTAM-2). In the complex λh plane the overall stability boundary for SPRTAM-2 is therefore somewhat smaller than the boundary for either standard AM-2 or the RTAM-2 introduced here. It is also somewhat smaller than the stabilty boundary for AB-2 integration. In any event we should remember that the integrator error coefficient e_I in Eq. (4) and the characteristic root error in Eq. (5) are both based on approximate formulas that assume the integration step size is small in comparison with

the reciprocal frequencies or eigenvalues, respectively. For the moderate step sizes normally used in flight simulation, final comparison of integration methods should be based on actual example simulations, as we shall see in a following section.

3. The Modified Euler Integration Method

Application of the modified-Euler integration method to the nonlinear flight equations can be understood by considering the following two vector state equations for the velocity vector V and the displacement vector D:

$$\dot{V} = A[D,V,U(t)], \quad \dot{D} = V \tag{14}$$

To apply the modified-Euler method we represent the discrete velocity state V at half-integer frame times, denoted by $V_{n-1/2}$. The acceleration A and discrete displacement state D are represented at integer frame times, denoted by A_n and D_n, respectively. Then the modified-Euler difference equations become

$$V_{n+1/2} = V_{n-1/2} + hA_n, \quad D_{n+1} = D_n + hV_{n+1/2} \tag{15}$$

where

$$A_n = A(D_n, V_n', U_n) \tag{16}$$

Here V_n' represents an estimate of V at the nth frame. To obtain this estimate we resort to the same predictor formula used above in Eq. (12) for the SPRTAM-2 method. Thus we let

$$V_n' = V_{n-1/2} + h(\tfrac{7}{8}A_{n-1} - \tfrac{3}{8}A_{n-2}) \tag{17}$$

Unlike the SPRTAM-2 method, however, we note here that the displacement state D_n in Eq. (16) is not a predictor-derived estimate but results directly from the modified-Euler integration algorithm itself in Eq. (15). This results in considerably improved stability, particularly for dynamic systems having quasi-linear charcteristic roots that are near the imaginary axis. In fact if A does not depend on V and depends linearly on D, as would be the case for the pure imaginary roots of an undamped dynamic system, it is easy to show that the modified-Euler method presented here exhibits exactly zero damping, regardless of the integration step size h [4].

We note that the predictor formula can be used to compute the displacement $D_{n+3/2}'$ in accordance with the formula

$$D_{n+3/2}' = D_n + h(\tfrac{7}{8}V_{n+1/2} - \tfrac{3}{8}V_{n-1/2}) \tag{18}$$

Thus at the end of the nth integration frame in a real-time simulation we can output the displacement state D_{n+1} as needed in real time and a prediction of the state, $D_{n+3/2}'$, one half step ahead of real time. This could be quite advantageous in compensating for other delays in an overall real-time simulation, such as the half-frame delay associated with the dynamics of zero-order DAC (digital-to-analog) extrapolators.

The nonlinear dependence of the acceleration A on the velocity V in Eq. (14) can often be expressed in terms of $V\partial A/\partial V$, where $\partial A/\partial V$ is not a function of V, or at worst is only slightly dependent on V. For example if A represents dQ/dt, the time derivative of pitch rate Q in the flight equations, then $\partial A/\partial Q$ is proportional to the aerodynamic stability dericvative C_{M_Q}, i.e., the dimensionless pitching moment due to dimensionless pitch rate. C_{MQ} is normally independent of Q, although it may be dependent on other variables such as Mach number. Also, the overall $\partial A/\partial Q$ in this case will be independent of Q. Letting V be a scalar which represents the angular velocity Q, we can rewrite Eq. (14) as follows:

$$\dot{V} = C_0[D,U(t)] + C_1[D,U(t)]V \tag{19}$$

where $C_0 + C_1V = A$ and $C_1 = \partial A/\partial V$. Now, when mechanizing the modified-Euler difference equations (15) and (16) we can compute V_n', the estimate of V at the nth frame, by the formula

$$V_n' = \tfrac{1}{2}(V_{n+1/2} + V_{n-1/2}) \tag{20}$$

From Eqs. (19) and (20) the difference equation (15) then becomes

$$V_{n-1/2} = V_{n-1/2} + h[C_0(D_n,U_n) + C_1(D_n,U_n)\frac{V_{n+1/2}+V_{n-1/2}}{2}] \tag{21}$$

With respect to the velocity state V this equation clearly represents implicit trapezoidal integration. However it can be solved to obtain the following explicit formula for $V_{n+1/2}$:

$$V_{n+1/2} = \frac{(1+hC_1/2)V_{n-1/2} + hC_0}{1 - hC_1/2} \tag{22}$$

This formulation, i.e., the use of trapezoidal integration for the damping term, expands very substantially the stability region in the λh plane compared with the use of the predictor formula of Eq. (17) for the damping term [5]. It can also reduce appreciably the dynamic errors following transient inputs. The extra required computation is modest and consists mainly of an additional division.

In deriving Eq. (22) we have assumed that V ia a scalar, whereas V will in general be a vector. In this case $\partial A/\partial V$ will be a matrix, which must be inverted to obtain the explicit formula for $V_{n+1/2}$. Fortunately, the critical terms in this matrix in the case of the flight equations are the diagonal terms, in which case simple formulas similar to Eq. (22) involving only the diagonal terms can be derived. In particular, if P, Q, and R represent angular velocity components in roll, pitch, and yaw, respectively, difference equations similar to (22) can be written for $P_{n+1/2}$, $Q_{n+1/2}$, and $R_{n+1/2}$ where C_1 in each equation is proportional to the stability derivatives C_{ℓ_P}, C_{M_Q}, and C_{N_R} respectively.

4. ExampleSolutions of Flight Equations

In this section we compare the performance of the various real-time integration algorithms described in the previous two sections in the solution of actual flight equations. Since the largest characteristic roots for the rigid airframe are normally those associated with the short period pitching motion, we will only consider symmetric flight, i.e., the longitudinal equations of motion, in our example simulation. The conclusions regarding dynamic errors can safely extrapolated to the full six-degree-of-freedom case. The scalar rotational flight equations are invariably written with respect to aircraft body axes. However, the translational equations can be written with respect to either body or flight path axes [6]. Here we will use flight path axes, since they seem to be somewhat more suitable for the modified-Euler algotithm. In this case the longitudinal equations of motion can be written as follows:

$$\dot{V}_p = \frac{F_{wx}}{mV_p} \tag{23}$$

$$\dot{\alpha} = Q + \frac{F_{wz}}{mV_p} \tag{24}$$

$$\dot{Q} = \frac{M}{I_{yy}} \tag{25}$$

$$\dot{\Theta} = Q \tag{26}$$

$$\dot{H} = V_p \sin(\Theta - \alpha) \tag{27}$$

Here V_p is the total aircraft velocity, α is the angle of attack, Q is the pitch rate, Θ is the pitch angle, and H is the altitude; F_{wx} and F_{wz} are the external force components along the x and z flight-path axes, respectively, and M is the moment about the y body axis; finally, m and I_{yy} represent, respectively, the aircraft mass and pitch-axis moment of inertia. The following formulas were used to represent the external forces and moment:

$$F_{wx} = -qS(C_{D_0} + C_{D_{C_L^2}} C_L^2) - g\sin(\Theta - \alpha) + \frac{T}{m}\cos\alpha \tag{28}$$

$$F_{wz} = -qS(C_L + C_{L_{\delta_e}}\delta_e) + g\cos(\Theta - \alpha) - \frac{T}{m}\sin\alpha \tag{29}$$

$$M = qcS(C_{M_0} + C_{M_\alpha}\alpha + C_{M_Q}\frac{c}{2V_p}Q + C_{M_{\dot{\alpha}}}\frac{c}{2V_p}\dot{\alpha} + C_{M_{\delta_e}}\delta_e) \tag{30}$$

where

$$q = \text{dynamic pressure} = \frac{1}{2}\rho V_p^2 \tag{31}$$

and

$$C_L = \text{lift coefficient} = C_{L_0} + C_{L_\alpha}\alpha \tag{32}$$

In these equations S is the aircraft wing area, g is the gravity acceleration, T is powerplant thrust, δ_e is elevator displacement, and c is the mean aerodynamic chord. The various C's represent aerodynamic coefficients and stability derivatives in accordance with the subscripts. In a full flight-envelope simulation these will be nonlinear functions of other variables such as V_p (through Mach number dependence), α, δ_e, and h. The actual difference equations used to solve (23) through (32) using modified Euler integration are presented in the Appendix.

As a specific example we consider a business jet flying at 40,000 feet at a speed of Mach 0.7 [7]. For the above flight condition the undamped natural frequency of the short-period mode is about 3 rad/sec and the damping ratio is 0.4. We consider the aircraft response to two different input functions. One is a step elevator displacement of -0.01 radians at the initial time $t = 0$. The second is the input function shown in Figure 1, which is a step elevator displacement with a one second rise time. Use of this input function tends to reduce the large transient errors caused by step inputs when predictor integration algorithms are used. It is also probably more representative of an actual pilot input. Figures 2 and 3 show the aircraft pitch rate and pitch angle response, as generated by the simulation using RK-4 integration with a step size $h = 0.05$ seconds. With this step size the RK-4 simulation is sufficiently accurate to serve as an ideal solution against which the real-time second-order algorithms described in this paper can be checked.

The predictor algorithms considered in this paper all depend on the past as well as the present value of each state-variable

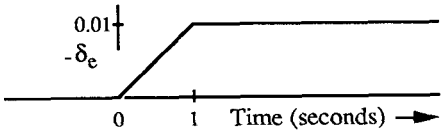

Figure 1. Finite rise-time elevator step input.

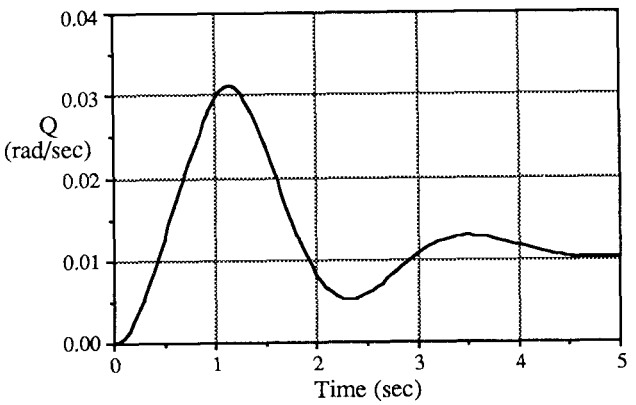

Figure 2. Pitch-rate response to input of Figure 1.

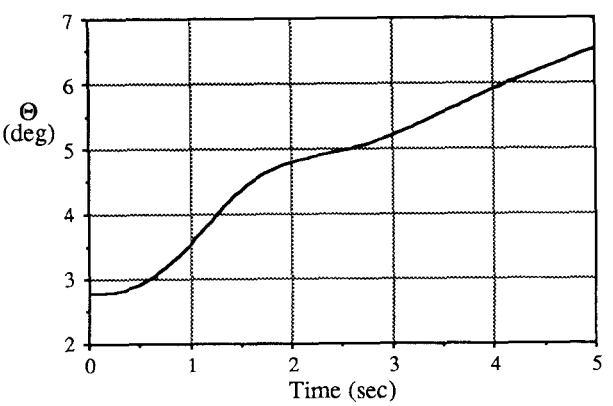

Figure 3. Pitch angle response to input of Figure 1.

derivative. This causes an initial startup ambiguity, since at the initial time t = 0 the past states and hence their derivatives are not known. They could be computed numerically prior to initiation of the simulation by integrating backwards one step using, for example, an RK algorithm. However, the usual method in real time simulation is to employ Euler simulation for the first step. Unfortunately, this can also cause startup transients which can mask the dynamic errors we are looking for in the second-order algorithms considered here. To circumvent this problem we have chosen to use a real-time RK algorithm for the first integration step. This consists of an Euler half-step followed by an RK-2 full step which uses the derivative as computed from the half-step result [1]. Subsequent integration steps use the particular second-order method being studied. In Figure 4 the error in pitch angle is plotted versus time with data points from the simulation using AB-2, RTAM-2, SPRTAM-2 and modified Euler integration, as described in this paper. For each algorithm the step size h =

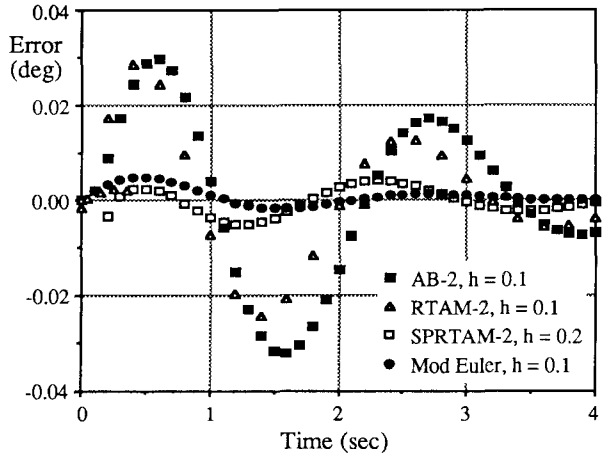

Figure 4. Pitch angle error versus time for step input in elev. displacement.

0.1 (10 integration steps per second), except that in the case of the RTAM-2 method we have used $h = 0.2$. This is because the RTAM-2 is a two-pass method which requires approximately twice the processor time per overall integration step in comparison with single-pass methods. From the figure it is evident that the modified-Euler algorithm performs slightly better than the SPRTAM-2 algorithm, with the AB-2 and RTAM-2 algorithms exhibiting considerably larger errors. For smaller step sizes the RTAM-2 method becomes significantly more accurate than AB-2, finally approaching an asymptotic limit equal to 0.4 times the AB-2 error, as noted earlier in Section 2. For smaller step sizes the SPRTAM-2 and modified-Euler methods continue to show an order of magnitude advantage over AB-2.

Next we consider the finite rise-time step input. In this case, in order to have the example be representative of an ongoing simulation, we have delayed three integration steps before applying the elevator input function of Figure 1. In Figure 5 the error in pitch angle is plotted versus time with data points from the simulation using AB-2, SPRTAM-2 and modified Euler integration. Again the step size $h = 0.1$. Once more the superior accuracy of the modified-Euler method is apparent. It is interesting to note the transient error introduced not only by the input slope discontinuity at $t = 0.3$, but also the additional transient error introduced by the second input slope discontinuity at $t = 1.3$.

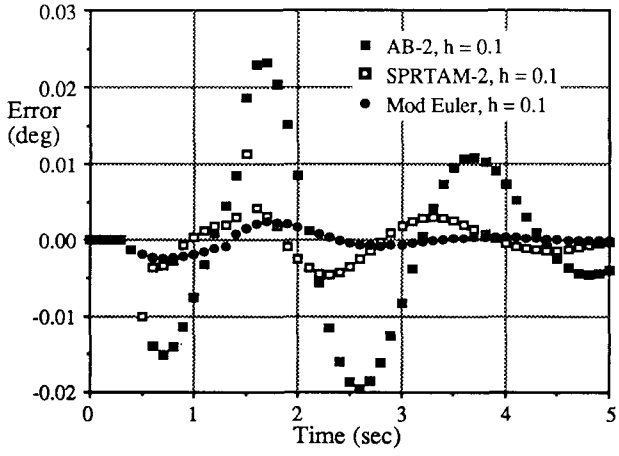

Figure 5. Pitch angle error for the input function of Figure 1, delayed in time by 1 second.

In this paper we have only considered second-order integraion algorithms. This is because it is usually not worthwhile to consider higher-order methods for the moderate dynamic accuracy (the order of one percent) normally considered adequate for real-time flight simulation. Higher order methods sutable for real-time simulation also tend to be less stable.

5. Conclusions

We have shown that the modified form of Euler integration described in this paper, when applied to the six-degree-of-freedom flight equations, gives results that are substantially more accurate than the AB-2 method which is usually employed. This has been demonstrated by considering the asymptotic formulas for characteristic root and transfer function errors, and by comparing actual time-domain errors in the response to control surface input functions. Following a transient input, the modified-Euler method produces an output response in the next integration step, whereas AB-2 integration exhibits an additional one-step delay in displacement response. The inherent nature of the modified-Euler algorithm permits it to produce outputs at half-integer steps. This feature can be used to produce an accurate half-frame lead in real-time output displacement. The modified Euler method may also be useful in the integration of state and costate equations in real-time mechanization of Kalman filters for navigation and control systems.

We have also shown that two other second-order integration methods, a real-time AM-2 predictor-corrector and a single-pass version of the same, represent more accurate alternatives to AB-2 integration for flight simulation.

Appendix

The flight equations used for the numerical results presented in this paper are given by Eqs. (23) through (32). Application of the AB-2, RTAM-2 and SPRTAM-2 integration algorithms to these equations is straightforward. Application of the modified Euler method requires additional explanation, which is the purpose of this appendix. When the six-degree-of-freedom flight equations are represented entirely in body axes, the six velocity states consist of U, V, W, the components of airframe velocity, and P, Q, R, the components of airframe angular velocity, along the body axes, x, y, z, respectively. The six displacement states consist of three position coordinates, normally latitude, longitude, and altitude, and, when Euler angles are used, three angular position coordinates, Φ, Θ, and Ψ, . Alternatively, four quaternions, e_1, e_2, e_3, and e_4, can be used as angular position states, from which direction cosines and Euler angles can be computed. To prevent accumulation of errors due to the the redundant fourth quaternion state, constraint terms which maintain $e_1^2 + e_2^2 + e_3^2 + e_4^2 = 1$ are added to the right side of each quaternion state equation.

When the translational equations of motion for the six-degree-of-freedom flight equations are derived using flight-path axes, the velocity states are represented by V_p, the total aircraft velocity, α, the angle of attack, and β, the sideslip angle [6]. When only the nonlinear symmetric (longitudinal) equations of motion are considered, the state equations become (23) through (27), where the equation for horizontal position has not been included. Based on the way in which modified-Euler integration was introduced in Section 3, the velocity states V_p, α and Q would be represented at half-integer frames, with the position states Θ and H represented at integer frames. For the nth integration frame this results in the computation of the $n+1/2$ velocity state from the n-1/2 velocity state, followed by computation of the $n+1$ position state from the n position state using the $n+1/2$ velocity state just obtained. However, from Eq. (24) it is apparent that it would be better to represent the angle of attack α at integrer frames, even though it is derived from a velocity state equation. This is because the dominant term on the right side of Eq. (24) affecting the high-speed dynamics is the pitch-rate Q, which is represented at half-integer frames. The other term in Eq. (24), F_{wz}/mV_p, is the negative of the flight-path-axis pitch rate, and is generally much smaller in magnitude than Q. This is the reason for representing α at integer frames in the modified Euler flight equations. When the lateral equations of motion are considered in a full six-degree-of-freedom simulation, the same rationale is used to represent the velocity state β at integer rather than half-integer frames when using modified Euler integration.

With this as background, we now list the actual difference equations used in each modified-Euler integration step when solving the longitudinal flight equations.

Variables available at the start of the nth integration frame:

$$V_{p_{n-1/2}}, V'_{p_n}, Q_{n-1/2}, Q'_n, \alpha_n, \alpha'_{n+1/2}, \Theta_n, \Theta'_{n+1/2}, H_n, H'_{n+1/2}, \delta_{e_n}, \delta_{e_{n-1}}, \dot{V}_{p_{n-1}}, \dot{Q}_{n-1}, (F_{wz}/mV_p)_{n-1/2}$$

Difference equations for the nth integration frame (in order of execution):

$$q_n = .5\rho_n V'^2_{p_n}, \quad C_{L_n} = C_{L_0} + C_{L_\alpha}\alpha_n, \quad C_{L_{n+1/2}} = C_{L_0} + C_{L_\alpha}\alpha_{n+1/2} \tag{A.1}$$

$$\dot{V}_{p_n} = -[(q_n S/m_n)(C_{D_0} + C_{D_{C_L^2}} C_{L_n}^2) - g\sin(\Theta_n - \alpha_n) + (T_n/m_n)\cos(\alpha_n)]/V'_{p_n} \tag{A.2}$$

$$Q_n = q_n S c/I_{yy_n}[C_{M_0} + C_{M_\alpha}\alpha_n + (.5c/V'_{p_n})((C_{M_Q} + C_{M_{\dot\alpha}})Q_n + C_{M_{\dot\alpha}}(1.5(F_{wz}/mV_p)_{n-1/2} - .5(F_{wz}/V_p)_{n-3/2}) + C_{M_{\delta_e}}\delta_{e_n}] \tag{A.3}$$

$$V_{p_{n+1/2}} = V_{p_{n-1/2}} + h\dot{V}_{p_n} \tag{A.4}$$

$$V'_{p_{n+1}} = V_{p_{n+1/2}} + h(.875\dot{V}_{p_n} - .375\dot{V}_{p_{n-1}}) \tag{A.5}$$

$$Q_{n+1/2} = Q_{n-1/2} + h\dot{Q}_n \tag{A.6}$$

$$Q'_{n+1} = Q_{n+1/2} + h(.875\dot{Q}_n - .375\dot{Q}_{n-1}) \tag{A.7}$$

$$(F_{wz}/mV_p)_{n+1/2} = -q_n(S/m_n)[C_{L_{n+1/2}} + C_{L_{\delta_e}}(1.5\delta_{e_n} - .5\delta_{e_{n-1}})] + g\cos(\Theta'_{n+1/2} - \alpha'_{n+1/2}) - (T_n/m_n)\sin(\alpha'_{n+1/2}) \tag{A.8}$$

$$\alpha_{n+1} = \alpha_n + h[Q_{n+1/2} + (F_{wz}/V_p)_{n+1/2}] \tag{A.9}$$

$$\alpha'_{n+3/2} = \alpha_{n+1} + h\{.875[Q_{n+1/2} + (F_{wz}/V_p)_{n+1/2}] - .375[Q_{n-1/2} + (F_{wz}/V_p)_{n-1/2}]\} \tag{A.10}$$

$$\Theta_{n+1} = \Theta_n + hQ_{n+1/2} \tag{A.11}$$

$$\Theta'_{n+3/2} = \Theta_{n+1} + h(.875Q_{n+1/2} - .375Q_{n-1/2}) \tag{A.12}$$

$$H_{n+1} = H_n + hV_{p_{n+1/2}}\sin[.5(\Theta_{n+1} + \Theta_n) - .5(\alpha_{n+1} + \alpha_n)] \tag{A.13}$$

Some comments regarding the above equations are in order. As noted previously, in an actual full flight-envelope simulation the aerodynamic coefficients which appear as constants in the equations will in fact be nonlinear functions of variables such as angle of attack, Mach number, control-surface dispacement, etc. The calculation of these multivariable functions by table lookup and linear interpolation usually constitutes a sizeable fraction of the total processor time for one integration step.

Eq. (A.1) indicates the necessity of computing both C_{L_n} and $C_{L_{n+1/2}}$. In a full simulation the drag coefficient in Eq. (A.2) would probably be computed as a nonlinear function of Mach number, angle of attack, and perhaps other variables such as flap position and elevator position. Typically, the lift coefficient would involve a nonlinear function of the same variables. Note that the $\dot\alpha$ term in Eq. (30) is synthesized in Eq. (A.3) in terms of Q and F_{wz}/mV_p in accordance with the formula in Eq. (24). The $\dot\alpha$ term actually results from a first-order approximation to the time delay associated with the effect of wing downwash on the horizontal tail. An alternative method for simualing this is to include a pure time delay proportional to $1/Vp$ for the fraction of the $C_{M_\alpha}\alpha$ term due to the downwash effect.

In Eq. (A.8) the dynamic pressure q_n has been used rather than $q_{n+1/2}$, even though the right side of Eq. (A.8) in general is represented at the $n+1/2$ frame. This is because both the density ρ and velocity V_p vary slowly enough that it seems hardly worthwhile to compute the dynamic pressure at both the n and $n+1/2$ frame, although this could be done with a modest additional amount of calculation. Also, in Eqs. (29) and (A.8) we have not included aerodynamic lift terms involving Q and $\dot\alpha$ simply because their effect generally turns out to be negligible.

In the full six-degree-of-freedom flight equations the counterpart of Eq. (26), or its equivalent difference equation (A.11), would be nonlinear state equations for the three Euler angle rates. Or, if quaternions are used, there would be four state equations invloving e_1, e_2, e_3, and e_4, as well as P, Q, and R. The corresponding difference equations would compute $e1_{n+1}$, $e2_{n+1}$, $e3_{n+1}$, and $e4_{n+1}$ from formulas involving $P_{n+1/2}$, $Q_{n+1/2}$, and $R_{n+1/2}$. To convert body axis velocity components to earth-axis velocity components at the $n+1/2$ frame, direction cosines are used. These are computed at the $n+1/2$ frame from quaternions at the $n+1/2$ frame which in turn are computed as the average value of the quaternions at the n and $n+1$ frame. Finally, the earth-axis position coordinates are computed at the $n+1$ frame with modified Euler integration using the earth axis velocity components at the $n+1/2$ frame as the input derivatives. Eqs. (A.11) and (A.13) represent the equivalent to these calculations in our simplified longitudinal case.

References

1. Howe, R.M., "Transfer Function and Characteristic Root Errors for Fixed-Step Integration Algorithms," *Transactions of the Society for Computer Simulation*, 2 (4): 293-320.

2. Benyon, P.K., "A Review of Numerical Methods for Simulation," *Simulation*, 11(5):219-238.

3. Howe, R.M., "The Use of Real-time Predictor-corrector Integration for Flight Simulation," *Proc. of the SCS Simulation Conference Simulators V*, Orlando, 18-21 April, 1988, pp 38-42.

4. Howe, R.M., "Simulation of Linear Systems Using Modified Euler Integration Methods," *Transactions of the Society for Computer Simulation*, 5 (2): 125-152

5. Howe, R.M., "A Performance Comparison of Integration Algorithms in Simulating Flexible Structures," *Proc. of the NASA Workshop on Computational Aspects in the Control of Flexible Systems*, July 12-14, 1988 Williamsburg, VA.

6. Fogarty, L.E., and R.M. Howe, "Computer Mechanization of Six-Degree-of-Freedom Flight Equations," *Simulation*, 11(4): 187-193.

7. Roskam, J., "*Airplane Flight Dynamics and Automatic Flight Controls*," Vol. 1, 1982, pp. 616-624, Roskam Aviation and Engineering Corporation, Route 4, Box 274, Ottawa, Kansas 66067.