Dymore User's Manual

The generalized-α time integration scheme

Contents

1 Introduction

The generalized-α [1] scheme generalizes the classical Hilber-Hughes-Taylor [2] time integrations scheme, which is widely used for structural dynamics problems. For linear problems, these schemes achieve high-frequency numerical dissipation while minimizing unwanted low-frequency dissipation. Both methods have been successfully used for both linear and nonlinear problems, although unconditional stability is proved for linear systems only. For instance, Cardona and Géradin [3] used the Hilber-Hughes-Taylor scheme for integrating the equations of motion in mechanism analysis.

2 The generalized-α scheme

The generalized-α scheme used here is that presented by Arnold and Brüls [4] rather than the original scheme of Chung and Hulbert [1]. It is typically presented for linear structural dynamics problems characterized by the following equations of motion

$$\underline{\underline{M}}\,\underline{\ddot{q}} + \underline{\underline{G}}\,\underline{\dot{q}} + \underline{\underline{K}}\,\underline{q} = \underline{f}(t), \tag{1}$$

where array $\underline{q}$ stores the n generalized coordinates, $\underline{\underline{M}}$, $\underline{\underline{G}}$, and $\underline{\underline{K}}$ are the constant mass, damping, and stiffness matrices of the system, respectively, and $\underline{f}(t)$ the externally applied, time-dependent force array. These equations of motion form a set of linear, second-order, coupled ordinary differential equations.

A typical time step starts and ends at times t_i and t_f, respectively, and $h = t_f - t_i$ is the time step size. Subscripts $(\cdot)_i$ and $(\cdot)_f$ are used to identify quantities evaluated at times t_i and t_f, respectively. The generalized displacement, velocity, and acceleration arrays at time t_i are denoted $\underline{q}_i$, $\underline{\dot{q}}_i$, and $\underline{\ddot{q}}_i$, respectively. Similar notations are defined at the end of the time step using subscript $(\cdot)_f$.

In this formulation, the solution at the end of the time step is written as

$$\underline{q}_f = \underline{q}_i + h\underline{\dot{q}}_i + \left[(\tfrac{1}{2} - \beta)h^2\underline{a}_i + \beta h^2 \underline{a}_f\right], \tag{2a}$$

$$h\underline{\dot{q}}_f = \quad h\underline{\dot{q}}_i + \left[(1-\gamma)h^2\underline{a}_i + \gamma h^2 \underline{a}_f\right], \tag{2b}$$

where β and γ are two parameters that will be selected to achieve desirable stability and accuracy characteristics for the scheme. These equations use algorithmic accelerations, $\underline{a}_f$ and $\underline{a}_i$, which are related to the actual accelerations of the system through the following recurrence relationship,

$$(1-\alpha_m)\underline{a}_f + \alpha_m \underline{a}_i = (1-\alpha_f)\underline{\ddot{q}}_f + \alpha_f \underline{\ddot{q}}_i, \tag{3}$$

where α_m and α_f are two additional parameters that will be selected to achieve desirable stability and accuracy characteristics for the scheme.

The discrete equations of motion, eqs. (1), are satisfied at time t_f,

$$\underline{\underline{M}}\,\underline{\ddot{q}}_f + \underline{\underline{G}}\,\underline{\dot{q}}_f + \underline{\underline{K}}\,\underline{q}_f = \underline{f}(t_f). \tag{4}$$

2.1 Optimal choice of the coefficients

The generalized-α scheme [1] is unconditionally stable, second-order accurate, and present high frequency numerical dissipation when the parameters of the scheme are chosen as follows. The spectral radius of the amplification matrix, ϱ, is a function of the non-dimensional time step, $\bar{h} = h/T$, where T is the natural period of the system. Let ϱ_∞ denote the spectral radius for very large time step sizes, *i.e.*, $\varrho_\infty = \varrho(\bar{h} \to \infty)$. Parameters α_m and α_f are then selected as

$$\alpha_m = \frac{2\varrho_\infty - 1}{\varrho_\infty + 1}, \quad \alpha_f = \frac{\varrho_\infty}{\varrho_\infty + 1}, \tag{5}$$

where $\varrho_\infty \in [0, 1]$.

If the spectral radius of the amplification matrix vanishes for $\bar{h} \to \infty$, *asymptotic annihilation* is achieved. Parameters γ and β are then selected as follows

$$\gamma = \frac{1}{2} - \alpha_m + \alpha_f, \quad \beta = \frac{1}{4}(1 - \alpha_m + \alpha_f)^2. \tag{6}$$

Figure 1 shows the spectral radius of the amplification matrix of the generalized-α scheme versus non-dimensional time step size, for six values of the numerical dissipation at infinity, $\varrho_\infty = 0.0$, 0.2, 0.4, 0.6, 0.8 and 1.0. In all six cases, the spectral radius nearly equals unity at low frequency, *i.e.*, for small values of $\bar{h}$. As the frequency increases, the spectral radius decreases monotonically and approaches ϱ_∞ for $\bar{h} \to \infty$. Asymptotic annihilation is achieved for $\varrho_\infty = 0.0$.

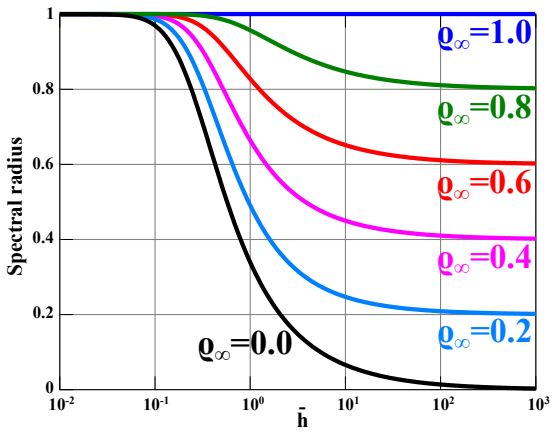

Figure 1: Spectral radius of the amplification matrix for the generalized-α scheme versus $\bar{h} = h/T$. $\varrho_\infty = 0.0$, 0.2, 0.4, 0.6, 0.8 and 1.0.

The accuracy of the generalized-α scheme is characterized by its period elongation and algorithmic damping shown in figs. 2 and 3, respectively. For $\varrho_\infty = 1.0$, the generalized-α scheme is identical to the Newmark algorithm [5]: the scheme presents no algorithmic damping. Of course, as the numerical dissipation at infinity increases, *i.e.*, as ϱ_∞ decreases, the algorithmic damping increases significantly, even at low frequency, as revealed by fig. 3. As ϱ_∞ decreases, the period elongation increases, but to a lesser extend. The scheme, however, remains second-order accurate even when asymptotic annihilation is achieved, *i.e.*, when $\varrho_\infty = 0$. Clearly, numerical dissipation is achieved at the expense of accuracy, although the generalized-α scheme remains second-order accurate for all values of ϱ_∞.

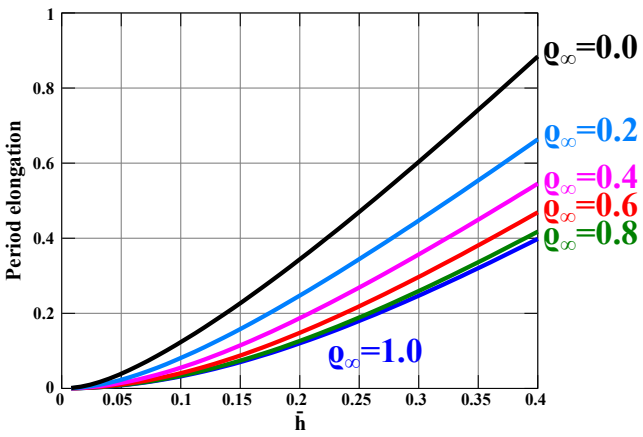

Figure 2: Period elongation of the generalized-α scheme. $\varrho_\infty = 0.0$, 0.2, 0.4, 0.6, 0.8 and 1.0.

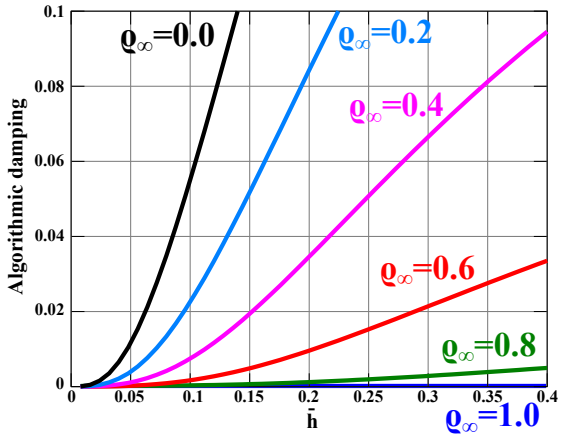

Figure 3: Algorithmic damping of the generalized-α scheme. $\varrho_\infty = 0.0$, 0.2, 0.4, 0.6, 0.8 and 1.0.

3 Practical implementation for nonlinear problems

For nonlinear system, the mass, gyroscopic, and stiffness matrices become functions of the unknowns, and eq. (1) now becomes $\underline{\mathcal{F}}_f = \underline{f}(t_f)$, where $\underline{\mathcal{F}}_f$ is the sum of all forces applied to the system at time t_f. Linearizing these equations leads to

$$\underline{\underline{M}}_f \Delta\underline{\ddot{q}}_f + \underline{\underline{G}}_f \Delta\underline{\dot{q}}_f + \underline{\underline{K}}_f \Delta\underline{q}_f = \underline{f}(t_f) - \underline{\mathcal{F}}_f, \tag{7}$$

where $\underline{\underline{M}}_f$, $\underline{\underline{G}}_f$, and $\underline{\underline{K}}_f$ are the mass, gyroscopic, and stiffness matrices, respectively, expressed in terms of the unknowns at time t_f, and $\Delta\underline{q}_f$, $\Delta\underline{\dot{q}}_f$, and $\Delta\underline{\ddot{q}}_f$, the increments in the generalized coordinates, velocities, and accelerations.

The recurrence relationship, eq. (3), is solved for the incremental algorithmic acceleration to find

$$\Delta\underline{a}_f = c_5\underline{\ddot{q}}_i + c_6\underline{a}_i + c_9\Delta\underline{\ddot{q}}_f, \tag{8}$$

where coefficients c_5, c_6, and c_9 are defined as

$$c_5 = \frac{\alpha_f}{1-\alpha_m}, \quad c_6 = -\frac{\alpha_m}{1-\alpha_m}, \quad c_9 = \frac{1-\alpha_f}{1-\alpha_m}. \tag{9}$$

Introducing eq. (8) into eq. (2a) yields

$$\Delta\underline{q}_f = h\underline{\dot{q}}_i + c_1\underline{\ddot{q}}_i + c_2\underline{a}_i + c_8\Delta\underline{\ddot{q}}_f, \tag{10}$$

where coefficients c_1, c_2, and c_8 are defined as

$$c_1 = \frac{\beta\alpha_f}{1-\alpha_m}h^2, \quad c_2 = (\frac{1}{2} - \frac{\beta}{1-\alpha_m})h^2, \quad c_8 = \frac{\beta(1-\alpha_f)}{1-\alpha_m}h^2. \tag{11}$$

Finally, introducing eq. (8) into eq. (2b) yields

$$\Delta\underline{\dot{q}}_f = \underline{\dot{q}}_i + c_3\underline{\ddot{q}}_i + c_4\underline{a}_i + c_7\Delta\underline{\ddot{q}}_f, \tag{12}$$

where coefficients c_3, c_4, and c_7 are defined as

$$c_3 = \frac{\gamma\alpha_f}{1-\alpha_m}h, \quad c_4 = (1 - \frac{\gamma}{1-\alpha_m})h, \quad c_7 = \frac{\gamma(1-\alpha_f)}{1-\alpha_m}h. \tag{13}$$

Next, eqs. (10) and (12) are introduced in the discrete nonlinear equations of motion, eq. (7), to find

$$\left(\underline{\underline{M}}_f + c_7\underline{\underline{G}}_f + c_8\underline{\underline{K}}_f\right)\Delta\underline{\ddot{q}}_f = \underline{f}(t_f) - \underline{\mathcal{F}}_f - \underline{\underline{K}}_f\left(h\underline{\dot{q}}_i + c_1\underline{\ddot{q}}_i + c_2\underline{a}_i\right) - \underline{\underline{G}}_f\left(\underline{\dot{q}}_i + c_3\underline{\ddot{q}}_i + c_4\underline{a}_i\right).$$

To simplify the notation, the following predictor expressions are defined

$$\Delta\underline{q}_p = h\underline{\dot{q}}_i + c_1\underline{\ddot{q}}_i + c_2\underline{a}_i, \tag{14a}$$

$$\underline{\dot{q}}_p = \underline{\dot{q}}_i + c_3\underline{\ddot{q}}_i + c_4\underline{a}_i, \tag{14b}$$

$$\underline{a}_p = c_5\underline{\ddot{q}}_i + c_6\underline{a}_i. \tag{14c}$$

$$\left(\underline{\underline{M}}_f + c_7\underline{\underline{G}}_f + c_8\underline{\underline{K}}_f\right)\Delta\underline{\ddot{q}}_f = \underline{f}(t_f) - \underline{\mathcal{F}}_f - \underline{\underline{K}}_f\Delta\underline{q}_p - \underline{\underline{G}}_f\underline{\dot{q}}_p. \tag{15}$$

4 Practical implementation for constrained problems

For constrained system, the constraint forces must also be considered, together with the Lagrange multipliers used to enforce the constraints. For holonomic constraints written as $\underline{\mathcal{C}}(\underline{q},t)=\underline{0}$, the constraint forces are expressed as

$$\underline{\mathcal{F}}^c = \underline{\underline{B}}^T(\underline{\lambda}+p\underline{\mathcal{C}}), \tag{16}$$

where $\underline{\underline{B}}=\partial\underline{\mathcal{C}}/\partial\underline{q}$ is the constraint matrix, $\underline{\lambda}$ the array of Lagrange multipliers use to enforce the constraints, and p a penalty coefficient. The last term corresponds to the augmented Lagrangian formulation, as proposed by Bayo *et al.* [6, 7]. Linearization of the constraint forces yields

$$\Delta\underline{\mathcal{F}}^c = \left[\underline{\underline{X}}(\underline{\lambda}+p\underline{\mathcal{C}})+p\underline{\underline{B}}^T\underline{\underline{B}}\right]\Delta\underline{q}+\underline{\underline{B}}^T\Delta\underline{\lambda}. \tag{17}$$

The first term of this expression is simply an additional contribution to the stiffness matrix, and the second term is the actual force of constraint.

For constrained systems, the linearized equations of motion of the system, eqs. (7), become

$$\underline{\underline{M}}_f\Delta\ddot{\underline{q}}_f+\underline{\underline{G}}_f\Delta\dot{\underline{q}}_f+\underline{\underline{K}}_f\Delta\underline{q}_f+\underline{\underline{B}}_f^T\Delta\underline{\lambda}_f=\underline{f}(t_f)-\underline{\mathcal{F}}_f, \tag{18}$$

where the stiffness matrix, $\underline{\underline{K}}_f$, now includes an additional contribution, the first term of eq. (17), and the load vector, $\underline{\mathcal{F}}_f$, includes the force of constraint.The constraint must be satisfied together with the equations of motion and after linearization, become

$$\underline{\underline{B}}_f\Delta\underline{q}_f=-\underline{\mathcal{C}}_f. \tag{19}$$

Next, eqs. (10) and (12) are introduced in the discrete nonlinear equations of motion, eqs. (18) and (19), to find

$$\left(\underline{\underline{M}}_f+c_7\underline{\underline{G}}_f+c_8\underline{\underline{K}}_f\right)\Delta\ddot{\underline{q}}_f+\underline{\underline{B}}_f^T\Delta\underline{\lambda}_f=\underline{f}(t_f)-\underline{\mathcal{F}}_f-\underline{\underline{K}}_f\Delta\underline{q}_p-\underline{\underline{G}}_f\dot{\underline{q}}_p, \tag{20a}$$

$$c_8\underline{\underline{B}}_f\Delta\ddot{\underline{q}}_f = -\underline{\mathcal{C}}_f-\underline{\underline{B}}_f\Delta\underline{q}_p. \tag{20b}$$

4.1 Scaling of the constraint equations

A cursory examination of these equations of motion reveals two obvious numerical problems. First, the unknowns of the problem are of different units: accelerations for $\Delta\ddot{\underline{q}}_f$ and forces for the Lagrange multipliers. Second,if the mass and/or damping and/or stiffness of the system become large, one or more of the first three terms of the equations of motion will become large, whereas the constraint equations remain unchanged. In other words, for systems with large mass, damping, or stiffness, the constraint equations become "invisible" to the numerical process.

To resolve these issues, the governing equations, eqs. (20), are recast as

$$\left(\underline{\underline{M}}_f+c_7\underline{\underline{G}}_f+c_8\underline{\underline{K}}_f\right)\Delta\ddot{\underline{q}}_f+r\underline{\underline{B}}_f^T\left(\frac{\Delta\underline{\lambda}_f}{r}\right)=\underline{f}(t_f)-\underline{\mathcal{F}}_f-\underline{\underline{K}}_f\Delta\underline{q}_p-\underline{\underline{G}}_f\dot{\underline{q}}_p, \tag{21a}$$

$$sc_8\underline{\underline{B}}_f\Delta\ddot{\underline{q}}_f = -s\underline{\mathcal{C}}_f-s\underline{\underline{B}}_f\Delta\underline{q}_p. \tag{21b}$$

The Lagrange multipliers were scaled by factor r and the second equation was scaled by factor s. For the two sets of unknowns to share the same units, factor r must have units of mass and hence, it is appropriate to select $r \approx m_r + d_r h + k_r h^2$. In this expression, m_r, d_r, and k_r represent characteristic mass, damping and stiffness coefficients of the system, selected as the average of the diagonal terms of the mass, damping, and stiffness matrices, respectively. To achieve the second goal, factor $c_8 s$ must have units of mass, *i.e.*, $c_8 s \approx m_r + d_r h + k_r h^2$. Because $c_8 = \bar{c}_8 h^2$, where $\bar{c}_8 = \beta(1-\alpha_f)/(1-\alpha_m) \approx 1$, scaling factor s is selected as

$$s = k_r + \frac{d_r}{h} + \frac{m_r}{h^2}. \tag{22}$$

Finally, to preserve the symmetry of the system, scaling factor r is selected as

$$r = c_8 s. \tag{23}$$

Note that this choice gives $r = \bar{c}_8 h^2 s = \bar{c}_8 (m_r + d_r h + k_r h^2)$; as desired, scaling factor r has units of mass.

Introducing the expressions for the scaling factors, eqs. (22) and (23), into the equations of motion, eqs. (21), leads to

$$\begin{bmatrix} \underline{\underline{M}}_f + c_7 \underline{\underline{G}}_f + c_8 \underline{\underline{K}}_f & c_8 s \underline{\underline{B}}_f^T \\ c_8 s \underline{\underline{B}}_f & \underline{\underline{0}} \end{bmatrix} \begin{Bmatrix} \Delta \underline{\ddot{q}}_f \\ \Delta \underline{\bar{\lambda}}_f \end{Bmatrix} = \begin{Bmatrix} \underline{f}(t_f) - \underline{\mathcal{F}}_f - \underline{\underline{K}}_f \Delta \underline{q}_p - \underline{\underline{G}}_f \underline{\dot{q}}_p \\ -s\underline{\mathcal{C}}_f - s\underline{\underline{B}}_f \Delta \underline{q}_p \end{Bmatrix}, \tag{24}$$

where the scaled Lagrange multipliers are defined as

$$\underline{\bar{\lambda}} = \frac{\underline{\lambda}}{c_8 s}. \tag{25}$$

To complete the formulation, a value must be selected for the penalty coefficient introduced in eq. (16). This penalty term gives rise to a stiffness matrix contribution, $p\underline{\underline{B}}^T \underline{\underline{B}}$, shown in eq. (17). To be effective, this term should be of the same order as the other entries of the stiffness matrix and hence, it is reasonable to select $p \approx s$, where scaling factor s is defined by eq. (22). With this choice, it is possible to define a modified Lagrange multiplier

$$\underline{\mu} = \underline{\lambda} + p\underline{\mathcal{C}} = s(c_8 \underline{\bar{\lambda}} + \underline{\mathcal{C}}). \tag{26}$$

In summary, the constraint forces, $\underline{\mathcal{F}}_f^c$ and the associated stiffness matrix, $\underline{\underline{K}}_f^c$, are of the following form

$$\underline{\mathcal{F}}_f^c = \begin{Bmatrix} -\underline{\underline{B}}_f^T \underline{\mu} \\ -s\underline{\mathcal{C}}_f \end{Bmatrix}, \quad \underline{\underline{K}}_f^c = \begin{bmatrix} c_8 \left[s\underline{\underline{B}}_f^T \underline{\underline{B}}_f + \underline{\underline{X}}(\underline{\mu}) \right] & c_8 s \underline{\underline{B}}_f^T \\ c_8 s \underline{\underline{B}}_f & \underline{\underline{0}} \end{bmatrix}. \tag{27}$$

4.2 Summary

At each time step, the solution of the nonlinear equations of motion involves the following procedure. The initial conditions are given as $\underline{\dot{q}}_i$, $\underline{\ddot{q}}_i$, and $\underline{a}_i$.

1. The unknowns at time t_f are initialized using the *predictor step*, which assumes that $\underline{\ddot{q}}_f = \underline{0}$, to find $\Delta\underline{q}_f = \Delta\underline{q}_p$, $\underline{\dot{q}}_f = \underline{\dot{q}}_p$, $\underline{\ddot{q}}_f = \underline{0}$, and $\underline{a}_f = \underline{a}_p$, where the predictors $\Delta\underline{q}_p$, $\underline{\dot{q}}_p$, and $\underline{a}_p$ are given by eq. (14).

2. Solve for the incremental accelerations and scaled Lagrange multipliers using eq. (24).

3. Increment the unknowns with the following update formulæ

$$\Delta\underline{q}_f = \Delta\underline{q}_f + c_8\Delta\underline{\ddot{q}}_f, \tag{28a}$$
$$\underline{\dot{q}}_f = \underline{\dot{q}}_f + c_7\Delta\underline{\ddot{q}}_f, \tag{28b}$$
$$\underline{\ddot{q}}_f = \underline{\ddot{q}}_f + \Delta\underline{\ddot{q}}_f, \tag{28c}$$
$$\underline{a}_f = \underline{a}_f + c_9\Delta\underline{\ddot{q}}_f. \tag{28d}$$

4. Iterate over steps 2 and 3 until convergence.

5 Practical implementation in the presence of motion

At each time step, the solution of the nonlinear equations of motion involves the following procedure. The initial conditions are $\hat{p}_i$, the Euler motion parameters representing the motion at the beginning of the time step, $\underline{\mathcal{V}}_i$, the initial velocities, and $\underline{\mathcal{A}}_i = \underline{\dot{\mathcal{V}}}_i$, the initial accelerations. The algorithm uses internal algorithmic acceleration variables $\underline{\mathcal{X}}$, which are initialized at t_0 as $\underline{\mathcal{X}} = \underline{\mathcal{A}}(t_0)$.

When the system configuration is defined by motion quantities, the increment in motion from t_i to t_f is defined by array $\underline{\Delta\mathcal{U}}$, which stacks the incremental motions at each node of the model. At node k, this incremental motion is denoted $\underline{\Delta\mathcal{U}}^{(k)}$. To facilitate the manipulation of motion operations, the Euler motion parameters associated with the incremental motion at each node is constructed as

$$\hat{\underline{p}}^{(k)}_{\text{inc}} = \frac{2\hat{\mathbf{o}} + \underline{\hat{\Delta\mathcal{U}}}^{(k)}}{\|2\hat{\mathbf{o}} + \underline{\hat{\Delta\mathcal{U}}}^{(k)}\|}, \tag{29}$$

where $\underline{\hat{\Delta\mathcal{U}}}^{(k)}$ is the bi-quaternion whose vector parts are those of the incremental motion $\underline{\Delta\mathcal{U}}^{(k)}$ and whose scalar parts vanish.

1. *Predictor step*:

$$\underline{\Delta\mathcal{U}} = h\underline{\mathcal{V}}_i + c_1\underline{\mathcal{A}}_i + c_2\underline{\mathcal{X}}, \tag{30a}$$
$$\underline{\mathcal{V}}_f = \underline{\mathcal{V}}_i + c_3\underline{\mathcal{A}}_i + c_4\underline{\mathcal{X}}, \tag{30b}$$
$$\underline{\mathcal{X}} = c_5\underline{\mathcal{A}}_i + c_6\underline{\mathcal{X}} \tag{30c}$$
$$\underline{\mathcal{A}}_f = \underline{0} \tag{30d}$$

Using eq. (29), the Euler motion parameters for node k at time t_f become

$$\hat{p}^{(k)}_f = \underline{\underline{\mathcal{A}}}(\hat{p}^{(k)}_i)\hat{p}^{(k)}_{\text{inc}}. \tag{31}$$

2. Solve for the incremental accelerations

$$\left[\underline{\underline{M}}_f + c_7\underline{\underline{G}}_f + c_8\underline{\underline{K}}_f\right]\Delta\underline{\mathcal{A}}_f = \underline{f}(t_f) - \underline{\mathcal{F}}_f. \tag{32}$$

3. *Corrector step*:

$$\underline{\Delta\mathcal{U}} = \underline{\Delta\mathcal{U}} + c_8\Delta\underline{\mathcal{A}}_f, \tag{33a}$$
$$\underline{\mathcal{V}}_f = \underline{\mathcal{V}}_f + c_7\Delta\underline{\mathcal{A}}_f, \tag{33b}$$
$$\underline{\mathcal{A}}_f = \underline{\mathcal{A}}_f + \Delta\underline{\mathcal{A}}_f \tag{33c}$$
$$\underline{\mathcal{X}} = \underline{\mathcal{X}} + c_9\Delta\underline{\mathcal{A}}_f \tag{33d}$$

   The Euler motion parameters at time t_f are evaluated using eq. (31).

4. Iterate over steps 2 and 3 until convergence.

5.1 Summary

The overall computation implementation of the generalized-α scheme is given in table 1.

Table 1: Overall computation implementation of the generalized-α scheme.

Loop over time steps {

  Loop over rejections {

    Compute coefficients using eqs. (9), (11), *and* (13).

    Perform the predictor step eqs. (30).

    Loop over iterations {

      Compute matrices and applied loads.

      Solve for incremental accelerations using eqs. (32).

      Increment configuration using eq. (31)

    }

    if solution is acceptable: end loop over rejections;

    else: adjust time step size or contact conditions, start next rejection.

  }

  Update configuration.

}

References

[1] J. Chung and G.M. Hulbert. A time integration algorithm for structural dynamics with improved numerical dissipation: The generalized-α method. *Journal of Applied Mechanics*, 60:371–375, 1993.

[2] H.M. Hilber, T.J.R. Hughes, and R.L. Taylor. Improved numerical dissipation for time integration algorithms in structural dynamics. *Earthquake Engineering and Structural Dynamics*, 5:283–292, 1977.

[3] A. Cardona and M. Géradin. Time integration of the equations of motion in mechanism analysis. *Computers & Structures*, 33(3):801–820, 1989.

[4] M. Arnold and O. Brüls. Convergence of the generalized-α scheme for constrained mechanical systems. *Multibody System Dynamics*, 18(2):185–202, 2007.

[5] N.M. Newmark. A method of computation for structural dynamics. *Journal of the Engineering Mechanics Division*, 85:67–94, 1959.

[6] E. Bayo, J. García de Jalón, and M.A. Serna. A modified Lagrangian formulation for the dynamic analysis of constrained mechanical systems. *Computer Methods in Applied Mechanics and Engineering*, 71:183–195, November 1988.

[7] E. Bayo, J. García de Jalón, A. Avello, and J. Cuadrado. An efficient computational method for real time multibody dynamic simulation in fully Cartesian coordinates. *Computer Methods in Applied Mechanics and Engineering*, 92:377–395, 1991.