newmark_ss

sharpy.linear.src.lingebm.newmark_ss(*M, C, K, dt, num_damp=0.0001, M_is_SPD=False*) [source]

Produces a discrete-time state-space model of the 2nd order ordinary differential equation (ODE) given by:

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{f(t)}$$

This ODE is discretized based on the Newmark-β integration scheme.

The output state-space model has the form:

$$\begin{aligned}\mathbf{x}_{n+1} &= \mathbf{A_{ss}}\mathbf{x}_n + \mathbf{B_{ss}}\mathbf{f}_n \\ \mathbf{y}_n &= \mathbf{C_{ss}}\mathbf{x}_n + \mathbf{D_{ss}}\mathbf{f}_n\end{aligned}$$

where $\mathbf{y} = \begin{Bmatrix}\mathbf{q}\\ \dot{\mathbf{q}}\end{Bmatrix}$

Note that as the state-space representation only requires the input force $\mathbf{f}$ to be evaluated at time-step n, thus the pass-through matrix $\mathbf{D_{ss}}$ is not zero.

This function retuns a tuple with the discrete state-space matrices $(\mathbf{A_{ss}}, \mathbf{B_{ss}}, \mathbf{C_{ss}}, \mathbf{D_{ss}})$.

Theory

The following steps describe how to apply the Newmark-β scheme to the ODE in order to generate the discrete time-state space-model. It folows the development of [1].

Notation

Bold upper case letters represent matrices, bold lower case letters represent vectors. Non-bold symbols are scalars. Curly brackets indicate (block) vectors and square brackets indicate (block) matrices.

Evaluating the ODE to the time steps t_n and t_{n+1} and isolating the acceleration term:

$$\begin{aligned}\ddot{\mathbf{q}}_n &= -\mathbf{M}^{-1}\mathbf{C}\dot{\mathbf{q}}_n - \mathbf{M}^{-1}\mathbf{K}\mathbf{q}_n + \mathbf{M}^{-1}\mathbf{f}_n \\ \ddot{\mathbf{q}}_{n+1} &= -\mathbf{M}^{-1}\mathbf{C}\dot{\mathbf{q}}_{n+1} - \mathbf{M}^{-1}\mathbf{K}\mathbf{q}_{n+1} + \mathbf{M}^{-1}\mathbf{f}_{n+1}\end{aligned}$$

The update equations of the Newmark-beta scheme are [1]:

$$\begin{aligned}\mathbf{q}_{n+1} &= \mathbf{q}_n + \dot{\mathbf{q}}_n\Delta t + (1/2 - \beta)\ddot{\mathbf{q}}_n\Delta t^2 + \beta\ddot{\mathbf{q}}_{n+1}\Delta t^2 + O(\Delta t^3) \\ \dot{\mathbf{q}}_{n+1} &= \dot{\mathbf{q}}_n + (1 - \gamma)\ddot{\mathbf{q}}_n\Delta t + \gamma\ddot{\mathbf{q}}_{n+1}\Delta t + O(\Delta t^3)\end{aligned}$$

where $\Delta t = t_{n+1} - t_n$.

The stencil is unconditionally stable if the tuning parameters γ and β are chosen as:

$$\begin{aligned}\gamma &= \frac{1}{2} + \alpha \\ \beta &= \frac{(1+\alpha)^2}{4} = \frac{(1/2+\gamma)^2}{4} = \frac{1}{16} + \frac{1}{4}(\gamma + \gamma^2)\end{aligned}$$

where $\alpha > 0$ accounts for small positive algorithmic damping (α is `num_damp` in the code).

Substituting the former relations onto the later ones, rearranging terms, and writing it in state-space form:

$$\mathbf{A_{ss1}}\begin{Bmatrix}\mathbf{q}_{n+1} \\ \dot{\mathbf{q}}_{n+1}\end{Bmatrix} = \mathbf{A_{ss0}}\begin{Bmatrix}\mathbf{q}_n \\ \dot{\mathbf{q}}_n\end{Bmatrix} + \mathbf{B_{ss0}}\mathbf{f}_n + \mathbf{B_{ss1}}\mathbf{f}_{n+1}$$

where

$$\begin{aligned}\mathbf{A_{ss1}} &= \begin{bmatrix}\mathbf{I} + \beta\Delta t^2\mathbf{M}^{-1}\mathbf{K} & \beta\Delta t^2\mathbf{M}^{-1}\mathbf{C} \\ \gamma\Delta t\mathbf{M}^{-1}\mathbf{K} & \mathbf{I} + \gamma\Delta t\mathbf{M}^{-1}\mathbf{C}\end{bmatrix} \\ \mathbf{A_{ss0}} &= \begin{bmatrix}\mathbf{I} - \Delta t^2(1/2 - \beta)\mathbf{M}^{-1}\mathbf{K} & \Delta t\mathbf{I} - (1/2 - \beta)\Delta t^2\mathbf{M}^{-1}\mathbf{C} \\ -(1-\gamma)\Delta t\mathbf{M}^{-1}\mathbf{K} & \mathbf{I} - (1-\gamma)\Delta t\mathbf{M}^{-1}\mathbf{C}\end{bmatrix} \\ \mathbf{B_{ss0}} &= \begin{bmatrix}(\Delta t^2(1/2 - \beta)\mathbf{M}^{-1} \\ (1-\gamma)\Delta t\mathbf{M}^{-1}\end{bmatrix} \\ \mathbf{B_{ss1}} &= \begin{bmatrix}(\beta\Delta t^2)\mathbf{M}^{-1} \\ (\gamma\Delta t)\mathbf{M}^{-1}\end{bmatrix}\end{aligned}$$

This is not in standard space-state form because the state update equation depends of the input at t_{n+1}. This term can be eliminated by defining the state

$$\mathbf{x}_n = \begin{Bmatrix}\mathbf{q}_n \\ \dot{\mathbf{q}}_n\end{Bmatrix} - \mathbf{A_{ss1}}^{-1}\mathbf{B_{ss1}}\mathbf{f}_n$$

Then

$$\begin{aligned}\mathbf{x}_{n+1} &= \mathbf{A_{ss1}}^{-1}[\mathbf{A_{ss0}}\mathbf{x}_n + (\mathbf{A_{ss0}}\mathbf{A_{ss1}}^{-1}\mathbf{B_{ss1}} + \mathbf{B_{ss0}})\mathbf{f}_n] \\ \begin{Bmatrix} \dot{\mathbf{q}}_n \\ \ddot{\mathbf{q}}_n \end{Bmatrix} &= \mathbf{x}_n + \mathbf{B_{ss1}}\mathbf{f}_n\end{aligned}$$

See also `sharpy.linear.src.libss.SSconv()` for more details on the elimination of the term multiplying $\mathbf{f}_{n+1}$ in the state equation.

This system is identified with a standard discrete-time state-space model

$$\begin{aligned}\mathbf{x}_{n+1} &= \mathbf{A_{ss}}\mathbf{x}_n + \mathbf{B_{ss}}\mathbf{f}_n \\ \mathbf{y_n} &= \mathbf{C_{ss}}\mathbf{x}_n + \mathbf{D_{ss}}\mathbf{f}_n\end{aligned}$$

where

$$\begin{aligned}\mathbf{A_{ss}} &= \mathbf{A_{ss1}}^{-1}\mathbf{A_{ss0}} \\ \mathbf{B_{ss}} &= \mathbf{A_{ss1}}^{-1}(\mathbf{B_{ss0}} + \mathbf{A_{ss0}}\mathbf{A_{ss1}}^{-1}\mathbf{B_{ss1}}) \\ \mathbf{C_{ss}} &= \mathbf{I} \\ \mathbf{D_{ss}} &= \mathbf{B_{ss1}}\end{aligned}$$

Notation is used in the code

$$\begin{aligned}\mathtt{th1} &= \gamma \\ \mathtt{th2} &= \beta \\ \mathtt{a0} &= (1/2 - \beta)\Delta t^2 \\ \mathtt{b0} &= (1 - \gamma)\Delta t \\ \mathtt{a1} &= \beta\Delta t^2 \\ \mathtt{b1} &= \gamma\Delta t\end{aligned}$$

param M:	Mass matrix $\mathbf{M}$
type M:	np.array
param C:	Damping matrix $\mathbf{C}$
type C:	np.array
param K:	Stiffness matrix $\mathbf{K}$
type K:	np.array
param dt:	Timestep increment
type dt:	float
param num_damp:	Numerical damping. Default `1e-4`
type num_damp:	float
param M_is_SPD:	whether to factorized M using Cholesky (only works for SPD matrices) or LU decomposition. Default: `False`
type M_is_SPD:	bool

returns: with the $(\mathbf{A_{ss}}, \mathbf{B_{ss}}, \mathbf{C_{ss}}, \mathbf{D_{ss}})$ matrices of the discrete-time state-space representation

rtype: tuple

References

[1] - Geradin M., Rixen D. - Mechanical Vibrations: Theory and application to structural dynamics