Time Domain Dynamic Simulations of Locally Nonlinear Large-Scale Systems

Andreas Linderholt[1], Yousheng Chen[1], and Thomas Abrahamsson[2]

1) Department of Mechanical Engineering, Linnaeus University, Växjö, Sweden
2) Department of Applied Mechanics, Chalmers University of Technology, Göteborg, Sweden

The Swedish
National Aeronautical Research Program
NFFP 5

- **NFFP 5 – second call, 2011-2013**

- ***Improved structural dynamics calculation models; compensating for local nonlinearities***

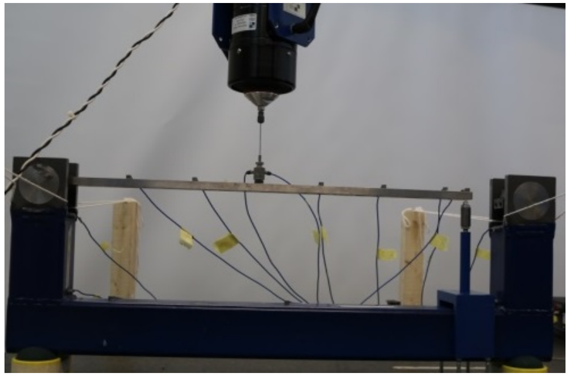
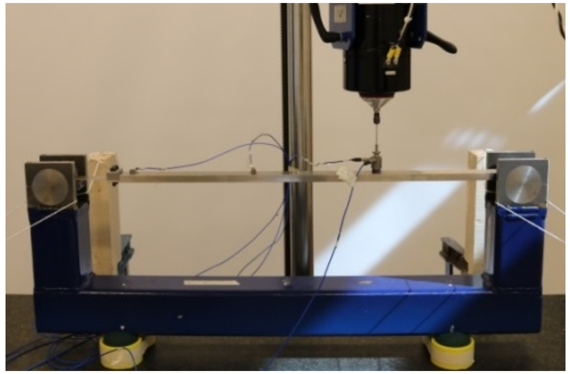
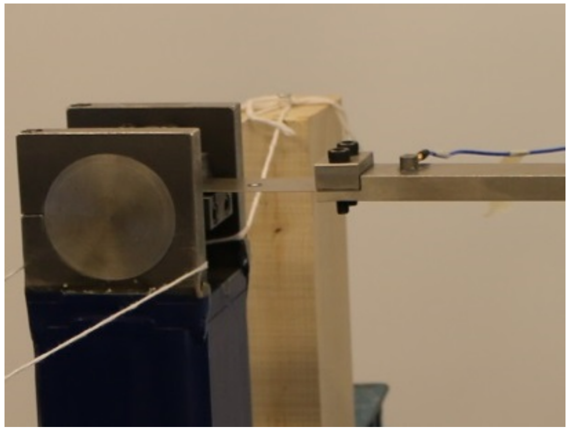
- **Joint venture between Saab, Linnaeus University and Chalmers**

- **Two PhD-students:**

- ***System identification of large-scale linear and nonlinear structural dynamic models*, Vahid Yaghoubi, the doctoral defense was held last May at Chalmers.**

- ***Model Calibration Methods for Mechanical Systems with Local Nonlinearities*, Yousheng Chen, doctoral defense the 25th of October at Linnaeus University in Växjö.**

Including local nonlinearities


Vibrational tests, numerical analyses, correlation, model calibration
Time domain simulations are time consuming

Outline of Presentation;
Simulations of Locally Nonlinear Systems

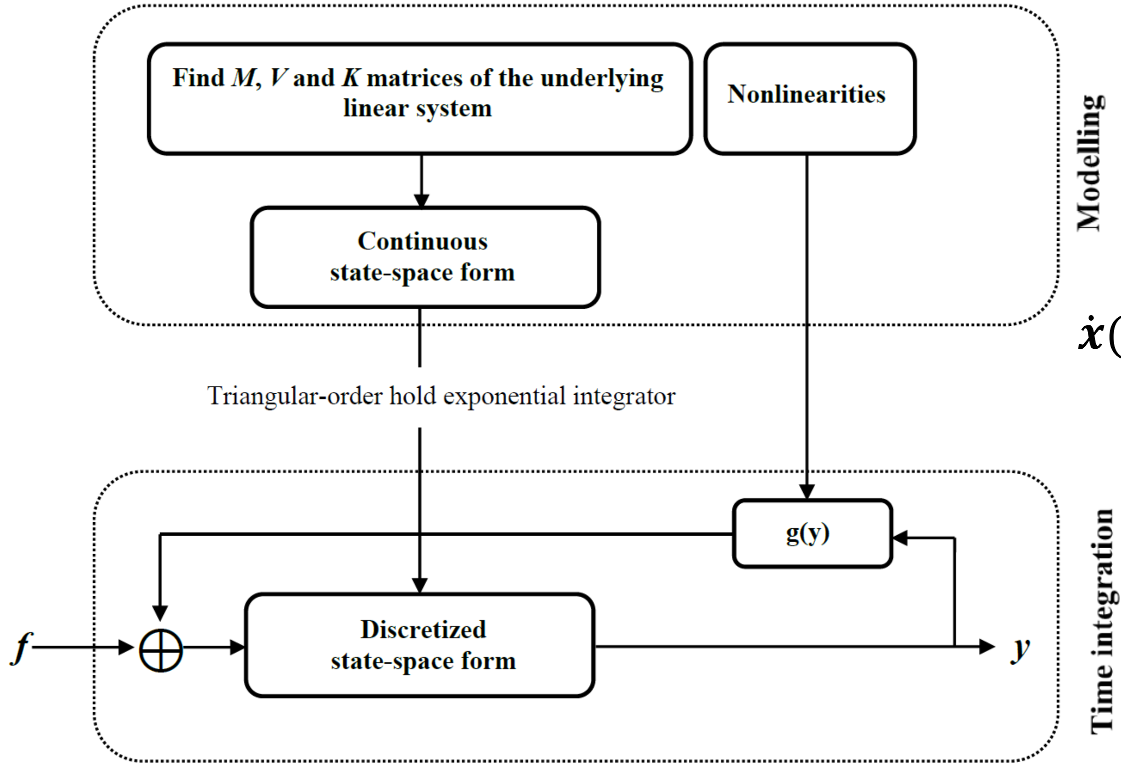
- Structure studied
- Introduction to the state-space based simulation method
- System discretization and time integration of the proposed method
- Case study on the dynamic characteristics of a bouncing ball (an SDOF model and an MDOF model)
- Conclusions

Introduction

- Forced responses of a nonlinear system are essential for the understanding of its nonlinear dynamic characteristics.
- Traditional time domain methods such as Runge-Kutta and Newmark may be inefficient when tackling a large-scale nonlinear system.
- Many research papers deal with development of fast time simulation methods for nonlinear systems. (model reduction)

Overview of the simulation method

$$\mathbf{M}\ddot{\boldsymbol{q}}(t) + \mathbf{V}\dot{\boldsymbol{q}}(t) + \mathbf{K}\boldsymbol{q}(t) = \boldsymbol{f}(t) - \boldsymbol{f}_{\mathrm{NL}}(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t))$$



$$\boldsymbol{x}(t) = [\boldsymbol{q}(t)\ \ \dot{\boldsymbol{q}}(t)]^{\mathrm{T}}$$

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}^{\mathrm{L}}\boldsymbol{f}(t) + \boldsymbol{B}^{\mathrm{NL}}\boldsymbol{f}_{\mathrm{NL}}(t)$$

$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}^{\mathrm{L}}\boldsymbol{f}(t) + \boldsymbol{D}^{\mathrm{NL}}\boldsymbol{f}_{\mathrm{NL}}(t)$$

K-step ahead prediction

The improved method

In this work, we improve the method by integrating the linear and the nonlinear part using the same step size.

Additionally trix to play in order to shorten the simulation time

- combine Matlab routines with C code – the actual time stepping calculations made in C

- Auxiliary matrices making the problem better conditioned

- Matrix diagonalizing – making the state matrix $\boldsymbol{A}$ is the state tri-diagonal

System discretization

- For a general nonlinear mechanical system, the governing equation of motion in a state-space form can be expressed as

$$\begin{cases} \dot{\boldsymbol{x}} = \boldsymbol{A}_{\mathrm{c}}\boldsymbol{x} + \boldsymbol{B}_{\mathrm{c}}\boldsymbol{u}(\boldsymbol{t}) + \boldsymbol{B}_{\mathrm{c}}^{\mathrm{NL}}\boldsymbol{u}^{\mathrm{NL}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ \boldsymbol{y} = \boldsymbol{C}_{\mathrm{c}}\boldsymbol{x} + \boldsymbol{D}_{\mathrm{c}}\boldsymbol{u}(\boldsymbol{t}) + \boldsymbol{D}_{\mathrm{c}}^{\mathrm{NL}}\boldsymbol{u}^{\mathrm{NL}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \end{cases}$$

- Applying the triangular-order hold exponential integrator gives a discrete form

$$\begin{cases} \boldsymbol{x}(k+1) = \boldsymbol{A}_{\mathrm{d}}\boldsymbol{x}(k) + \boldsymbol{B}_{\mathrm{d}}u(k) + \boldsymbol{B}_{\mathrm{d}}^{\mathrm{NL}}\boldsymbol{u}^{\mathrm{NL}}\big(\boldsymbol{q}(k), \dot{\boldsymbol{q}}(k)\big) \\ \mathbf{y}(k+1) = \boldsymbol{C}_{\mathrm{d}}\boldsymbol{x}(k+1) + \boldsymbol{D}_{\mathrm{c}}u(k+1) + \boldsymbol{D}_{\mathrm{d}}^{\mathrm{NL}}\boldsymbol{u}^{\mathrm{NL}}(\boldsymbol{q}(k+1), \dot{\boldsymbol{q}}(k+1)) \end{cases}$$

Time integration

To further shorten the simulation time, the only output states corresponding to the nonlinear forces are used to update the nonlinear forces. The balance equations can be rewritten as

$$\boldsymbol{y}_n(k+1) - \boldsymbol{D}_{\mathrm{n}}^{\mathrm{NL}} \boldsymbol{u}^{\mathrm{NL}}(k+1) = \boldsymbol{R}$$

$\boldsymbol{R}$ is completely determined by the past values together with the currently applied force. This function can be solved using iterations-

no gradients nor Hessians needed – just iterate….

Improvements

- **C code**

Time integration part is written in c code

- **Auxiliary matrix (AM)**

$$\boldsymbol{A}_{\mathrm{c}} = (\boldsymbol{A}_{\mathrm{c}} + \Delta\boldsymbol{A}) - \Delta\boldsymbol{A}$$
$$\mathrm{cond}(\boldsymbol{A}_{\mathrm{c}} + \Delta\boldsymbol{A}) \ll \mathrm{cond}(\boldsymbol{A}_{\mathrm{c}})$$

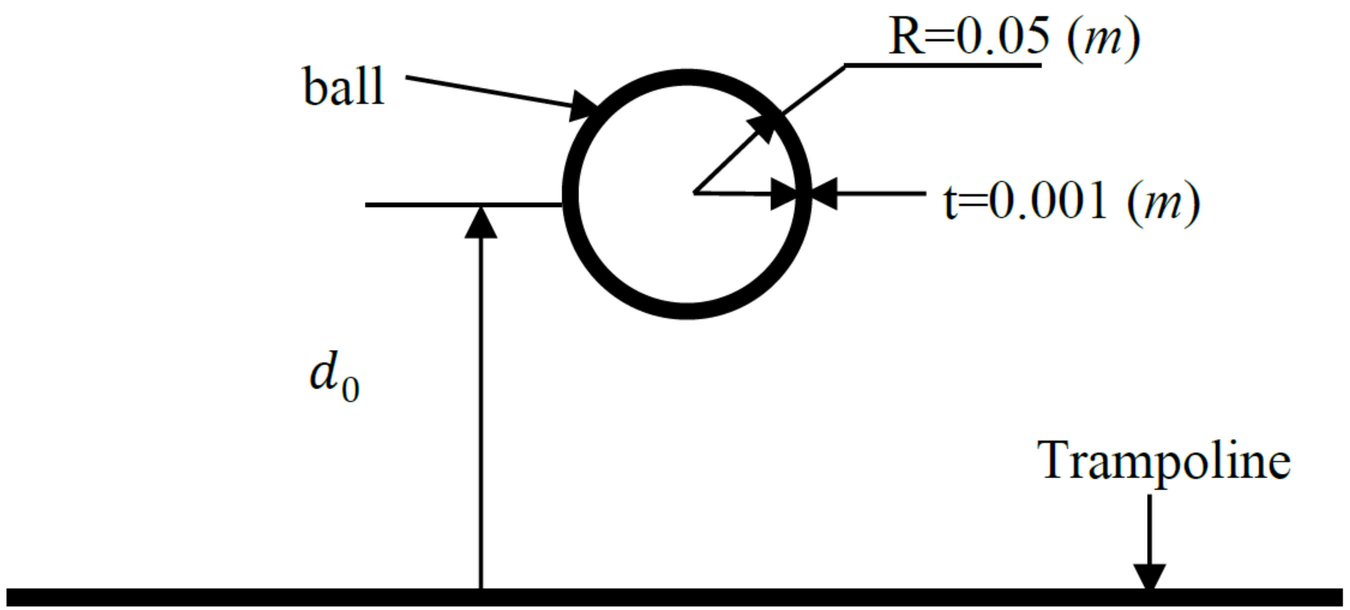
- **Matrix Diagonalization (MD)**

$\widetilde{\boldsymbol{A}}_d = \boldsymbol{T}^{-1}\boldsymbol{A}_{\mathrm{d}}\boldsymbol{T} = diag[\lambda_1, \quad \lambda_2, \quad \cdots, \quad \lambda_N]$ or tri-diagonal

The matrix multiplication $\widetilde{\boldsymbol{A}}_{\mathrm{d}}\boldsymbol{x}(k)$ is replaced by vectors multiplications which reduce the computational cost.

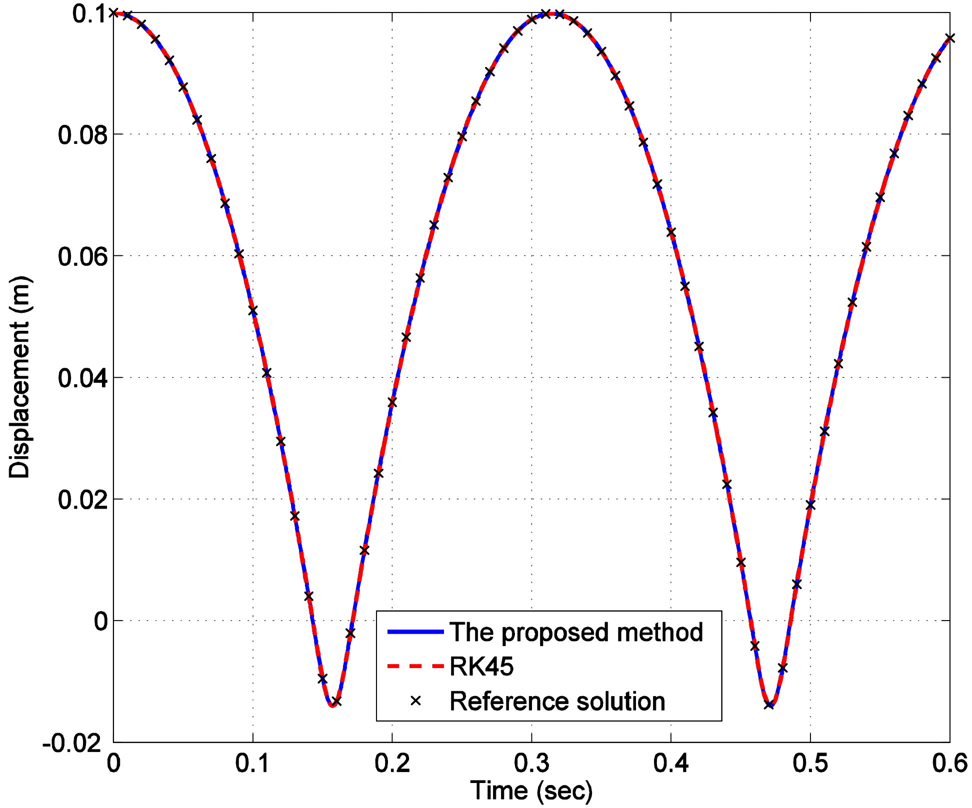
Numerical examples

The dynamic properties of a vertically dropped hollow ball is studied.



The SDOF model

The SDOF model of the bouncing ball and an example of its time response. The reference solution is the fixed time step RK4 using a very small time step.



The simulation results of the SDOF model

Required times to find the time responses of the SDOF system and the deviations of the methods with respect to the ref. solution. The simulations are made for the time interval [0 100] second.

Methods	Proposed simulation method	Proposed simulation method with C code	RK45
simulation time (sec)	6.4	0.2	134

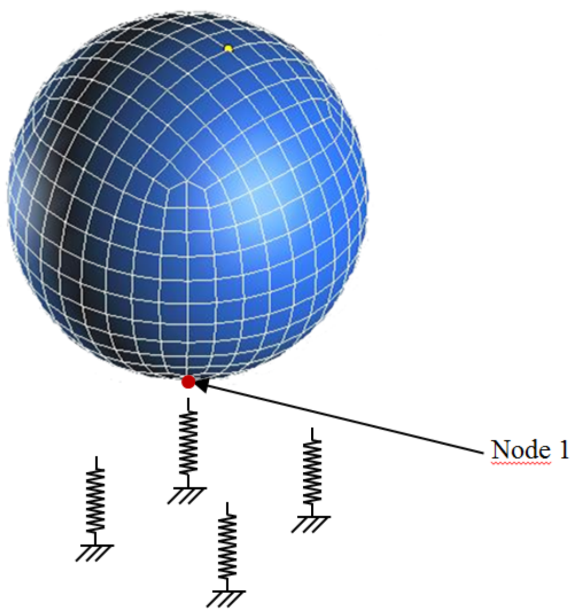
The simulation results of the SDOF model

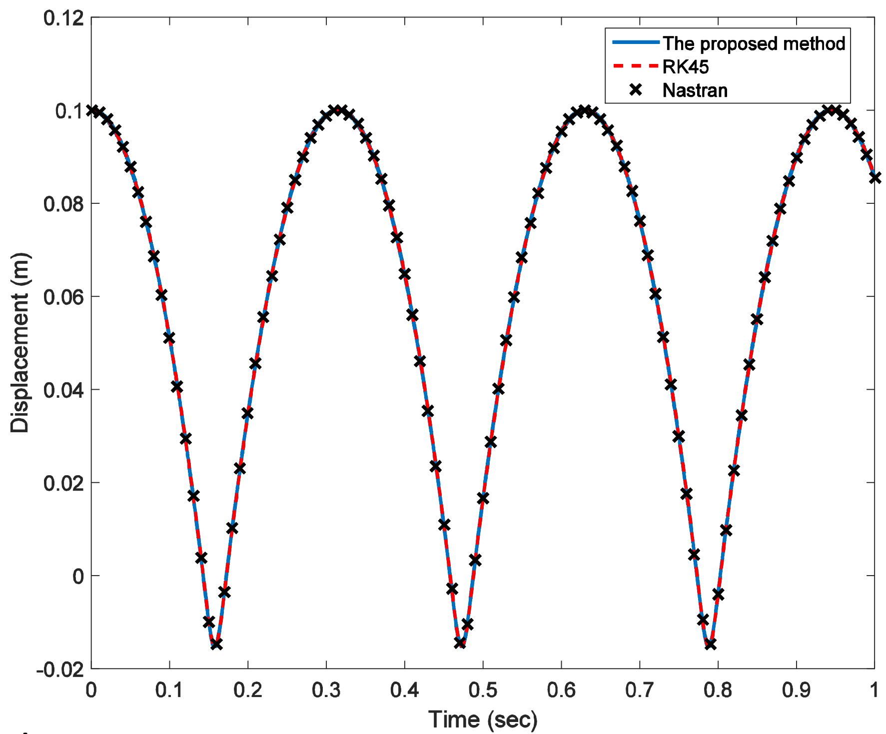
Required times to find the time responses of the SDOF system and the deviations of the methods with respect to the ref. solution. The simulations are made for the time interval [0 100] second.

Methods	Proposed simulation method	Proposed simulation method with C code	RK45
simulation time (sec)	6.4	0.2	134
Deviation	0.0050%	0.0050%	0.0083%

The MDOF model

An MDOF model of the bounce dynamics of a vertically dropped ball and the simulated response using different methods





604 shell elements with 85 discrete independent springs

Simulation results of the MDOF model

Comparison results of the proposed method with and without the Matrix diagonalizing (MD) algorithm.

Time intervals (sec)	1	2	20	100
Required simulation time without MD (sec)	126	218	954	9234
Required simulation time with MD (sec)	153	159	207	747

Simulation results of the MDOF model

The required simulation time for the time interval from zero to 2 second using different methods

Method	Proposed simulation method with AM	Proposed simulation method without AM	RK45	Nastran
Required simulation time (sec)	159	220	67500	6480

For the larger, MDOF, system – the C-code did not decrease the computational time

Concluding remarks

- In this work, an improved state-space based simulation approach is proposed. The improved method makes a one-step ahead prediction which reduce the number of the balancing equations.
- The implementation of C codes, matrix diagonalization and auxiliary matrices can be applied to speed up the simulation additionally.
- The efficiency and accuracy of the proposed method are illustrated using a contact problem which is modelled using an SDOF system and an MDOF system.

Thank you for your attention!