Comparison of Finite Element Methods for Elastodynamics Problem

Reza Abedi¹, Scott Miller²
¹. Mechanical Aerospace and Biomedical Engineering
   University of Tennessee Space Institute
². Applied Research Lab
   Penn State University

12th U.S. National Congress on Computational Mechanics
July 22-25, 2013 Raleigh North Carolina
Comparison Criteria

- Convergence Rates: error vs. element size ($h$)
- Solution Scaling: cost vs. number of elements ($N$)
- Efficiency: cost vs. error
- Memory usage
- Software Engineering aspects: e.g. ease of implementation and extensibility
Numerical method and problem types

- Underlying mathematical model:
  - linear vs nonlinear
  - type of PDE: Elliptic, Parabolic, and Hyperbolic

- Solution features: shocks and singularities

- Specifics of the numerical method:
  - Formulation type (single-field and multi-field formulations)
  - Element type
  - Integration scheme
  - Adaptive Implementations: $h$- and $hp$-adaptive
  - Parallel computing
Sample numerical methods:
A. Time marching schemes

- Displacement and velocity are both primitive fields.

\[ U = \begin{bmatrix} u \\ v \end{bmatrix} \]

- Solve linear momentum balance and kinematic compatibility.

- Consistent mass matrices are used

\[
M = \begin{bmatrix} \hat{u}_i \hat{u}_j & 0 \\ 0 & \hat{v}_i \hat{v}_j \end{bmatrix} \quad K = \begin{bmatrix} 0 & -\hat{u}_i \hat{v}_j \\ 0 & \nabla \hat{v}_i \cdot C[\nabla \hat{v}_j] \end{bmatrix}
\]

\[
M \ddot{U} + K U = 0 \quad \Rightarrow \quad \ddot{U} = M^{-1} K U
\]

- deal.ii is used to implement time marching schemes.
A. Time marching schemes

Standard ODE: \( \dot{u} = f(t, u) \).

- **Forward Euler**: explicit, first order accurate
  \[ \dot{u}_{n+1} = u_n + (\Delta t) f(t_n, u_n) \]

- **Explicit Runge-Kutta (Strong stability Preserving (SSP) variant)**:
  \[
  u_0 = u(t^n) \\
  u_i = k_i(\Delta t)f(t_{i-1}, u_{i-1}) + \sum_{j=1}^{i} c_{ij}u_j \quad \text{for } i = 1 - 5 \\
  u(t^{n+1}) = u_5
  \]

- **Bathe’s method (2007)**
  - Implicit and 2nd order accurate
  - Combines a trapezoidal rule step with a backward difference formula
    \[
    u_{n+1/2} = u_n + \frac{\Delta t}{4} \left( \dot{u}_n + f(t_{n+1/2}, u_{n+1/2}) \right) \\
    u_{n+1} = \frac{1}{3} \left( 4u_{n+1/2} - u_n + (\Delta t)f(t_{n+1}, u_{n+1}) \right)
    \]
B. Spacetime Discontinuous Galerkin (SDG) finite element method

- Discontinuous basis functions
- Element-wise balance property
- Superior performance for resolving discontinuous features without spurious oscillation or excessive dissipation
- Support for nonconforming meshes
- Flexibility in shape functions (polynomial orders)

Figure 1. one-dimensional elastic bar impact problem. (Top: model problem; bottom: exact solution.)

Attention is restricted to the time interval during which the stress wave remains compressive, that is, before the stress wave reaches the free end of the bar. The bar has a length of 4; the density, area and Young's modulus have unit values; the uniform initial speed also has unit value. The bar was uniformly discretized using 200 quadratic rod elements.

Figure 2 shows the stress distribution in the bar for time $t = 2.81$, calculated using the trapezoidal rule algorithm with time step $\Delta t = 0.01$; the dotted line denotes the exact solution. The oscillations in the trapezoidal rule solution induced by the discontinuity are clearly evident. These oscillations are not surprising as it is well-known that the trapezoidal rule algorithm possesses no numerical damping to localize or limit oscillations.

Within the context of structural dynamics, many semidiscrete algorithms have been developed which possess numerical damping but still retain the accuracy characteristics of the trapezoidal rule algorithm for problems with smooth solutions. One of the more successful semidiscrete algorithms is the Hilber-Hughes-Taylor (HHT) method [1-5]. The stress distribution in the bar is shown in Fig. 3 calculated using the HHT-cw algorithm

$p = 2$

$p = 1$
B. **Spacetime Discontinuous Galerkin (SDG) finite element method**

- Direct discretization of spacetime
- Replaces separate temporal integration
- Unified treatment of initial and boundary conditions
- Unstructured spacetime mesh
- Simultaneous mesh gradation in space and time
- Eliminates mesh tangling in moving–grid methods
Causal spacetime mesh and scalable Advancing-Front Solution Strategy
Given a space mesh, Tent Pitcher constructs a spacetime mesh such that every facet on sequence of advancing fronts is spacelike (patch height bounded by causality constraint).

Similar to CFL condition, except entirely local and not related to stability (required for scalability).
Tent Pitcher:
patch–by–patch meshing & solution

- Meshing and solutions are interleaved
- Patches (‘tents’) of tetrahedra are solve immediately
- patches are created and solved in parallel
FEM comparisons
Convergence rate

\[ e = C h^{ap+b} \quad \Rightarrow \log(e) = \log(C) - (ap + b) \log(1/h) \]

\( a \) and \( b \) depend on both spatial and temporal discretization and integration schemes
Solution Scaling

Various parts of the solution scale differently against the number of elements:

- $n_e$: number of elements in a patch
- $n_p$: number of patches
- $n$: total number of elements = $n_p \langle n_e \rangle$
- $c_p$: computational cost for a patch
- $c$: total cost $\propto n_p c_p$
- $c_p \propto n_e f_{EA} + n_e^\beta f_{LU}$

**SDG Method:**

$c \propto n_p c_p \propto n c_p \quad \Rightarrow \quad c \propto n \quad c_p \propto h^{-(d+1)} c_p$

**Globally coupled time marching schemes:**

$n_p = 1 (n_e = n) \Rightarrow$

$c \propto n \quad f_{EA} + n^\beta \quad f_{LU}$
Solution Scaling

SDG

d=1, 1F, sc, SDG

\[
\begin{array}{c}
\text{log[Time (cpu)]} \\
\text{log[1/h]}
\end{array}
\]

- p = 1
- p = 2
- p = 3
- p = 4
- p = 5
- p = 6

2 elements

4 elements
SDG solutions are between 1 to 2 orders of magnitude more efficient than Bathe’s method.

RK I method (forward Euler) exhibit numerical instabilities at high resolutions.
Lack of regularity: Numerical examples

Benchmark contact problem studied by T. J. R. Hughes et. al. (1976); T. A. Laursen, V. Chawla (1997); A. Czekanski, S. A. Meguid (2001); F. Cirak, M. West (2005), etc.
Lack of regularity
SDG solutions
Lack of Regularity:
Dynamic loading

fixed

$u(t)$

time
Lack of Regularity:
SDG method

- SDG method a natural choice for problems with shocks and other nonsmooth features

- Choice of error measure is very important in problems with discontinuous features

- Riemann fluxes result in small errors in such problems
Variant SDG implementations
Features of adaptive SDG method

- Local adaptivity operations
- flexible h-refinement and p-enrichment
- smaller elements do not pose global time increment constraints
- no projection error

[click to play movie]
Crack-tip wave scattering

[Click to play movie]
Dynamic fracture with damage-delay cohesive model, random defects and nucleation

[Image: Click to play movie]
**hp-Adaptive methods**

Is $p$-enrichment always more efficient for smooth problems?

For larger systems (3F vs 1F), higher order polynomials may not be more efficient!

How about continuous Galerkin methods?
Multi-cell elements

- Eliminates complicated and costly Riemann solutions

- Cost comparison:
  - Assembly cost reduces as there are no interior facets
  - LU cost reduces substantially as there is only one element

- Error comparison:
  - Error increases as element size doubles
Is multi-cell formulation more efficient?

Multi-cell option makes only 3-field formulation more efficient
The computation cost reduction is mostly on LU factorization.

The cost reduction should compensate for the increase in element size ($h$)

3-field formulation ($u, v, E$) becomes more efficient due to more dominant LU part
Parallel version of SDG

- Near-perfect scaling with static load balancing
- Parallelize the meshing process (vs. FE solution)
- Hierarchical and heterogeneous parallel structure (in progress)
Asynchronous load balancing

- load balancing (new diffusion method) Novel, asynchronous continuous load balancing scheme
- Processor cores continually compare work-load; shed or acquire elements to/from neighboring partitions
- keeps all operations local, asynchronous and opportunistic
- Heuristic control of inter-processor boundary quality
- So far, 95% efficiency for strong adaptive refinement
Conclusions

- SDG method can achieve arbitrary high convergence rates in space and time.
- The locally implicit solution scheme yields a linear solution complexity.
- SDG method was substantially more efficient than a few time marching schemes tested.
- Local and asynchronous solution strategy lends itself to adaptive and parallel simulations.
- Future work: Numerical and Mathematical analysis of the SDG method and various other Finite Element Methods.