Weight-adjusted Bernstein-Bezier DG methods for wave propagation in heterogeneous media

Jesse Chan, Kaihang Guo

Department of Computational and Applied Mathematics, Rice University

SIAM LA-TX Sectional conference

High order DG methods for wave propagation

- Unstructured (tetrahedral) meshes for geometric flexibility.
- High order: low numerical dissipation and dispersion.
- High order approximations: more accurate per unknown.
- Explicit time stepping: high performance on many-core.

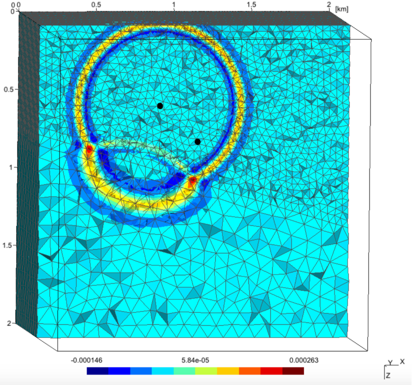
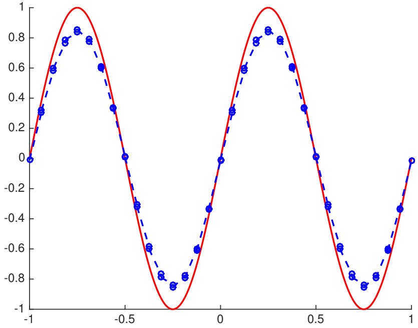


Figure courtesy of Axel Modave.

Goal: accuracy **and** efficiency for heterogeneous media.

High order DG methods for wave propagation

- Unstructured (tetrahedral) meshes for geometric flexibility.
- High order: low numerical dissipation and dispersion.
- High order approximations: more accurate per unknown.
- Explicit time stepping: high performance on many-core.

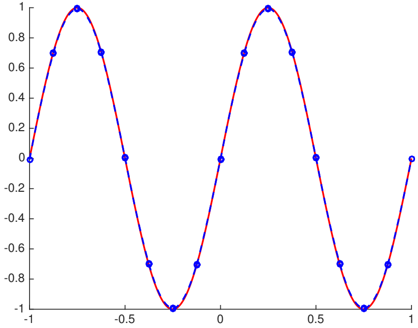


Fine linear approximation.

Goal: accuracy **and** efficiency for heterogeneous media.

High order DG methods for wave propagation

- Unstructured (tetrahedral) meshes for geometric flexibility.
- High order: low numerical dissipation and dispersion.
- High order approximations: more accurate per unknown.
- Explicit time stepping: high performance on many-core.

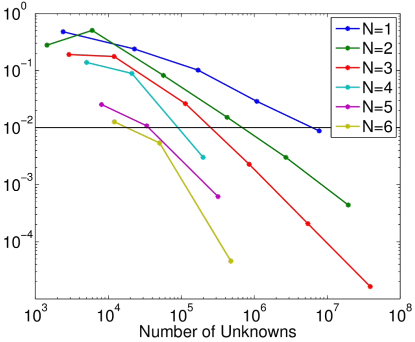


Coarse quadratic approximation.

Goal: accuracy **and** efficiency for heterogeneous media.

High order DG methods for wave propagation

- Unstructured (tetrahedral) meshes for geometric flexibility.
- High order: low numerical dissipation and dispersion.
- High order approximations: more accurate per unknown.
- Explicit time stepping: high performance on many-core.

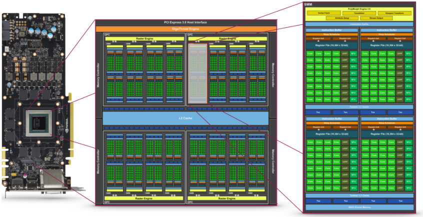


Max errors vs. dofs.

Goal: accuracy **and** efficiency for heterogeneous media.

High order DG methods for wave propagation

- Unstructured (tetrahedral) meshes for geometric flexibility.
- High order: low numerical dissipation and dispersion.
- High order approximations: more accurate per unknown.
- Explicit time stepping: high performance on many-core.

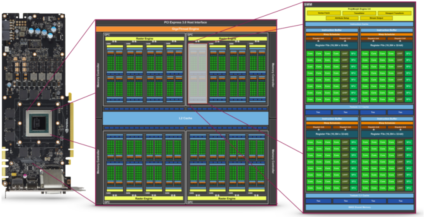


Graphics processing units (GPU).

Goal: accuracy **and** efficiency for heterogeneous media.

High order DG methods for wave propagation

- Unstructured (tetrahedral) meshes for geometric flexibility.
- High order: low numerical dissipation and dispersion.
- High order approximations: more accurate per unknown.
- Explicit time stepping: high performance on many-core.



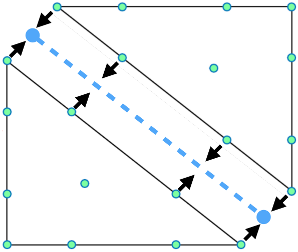
Graphics processing units (GPU).

Goal: accuracy **and** efficiency for heterogeneous media.

Time-domain nodal DG methods

Assume $u(\boldsymbol{x}, t) = \sum \boldsymbol{u}_j \phi_j(\boldsymbol{x})$ on D^k

- Compute numerical flux at face nodes (non-local).
- Compute RHS of (local) ODE.
- Evolve (local) solution using explicit time integration (RK, AB, etc).



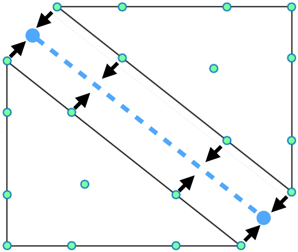
$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \mathbf{D}_x \mathbf{u} + \sum_{\text{faces}} \mathbf{L}_f \left(\text{flux}\right).$$

$$\boldsymbol{M}_{ij} = \int_{D^k} \phi_j(\boldsymbol{x}) \phi_i(\boldsymbol{x})$$

$$\mathbf{L}_f = \mathbf{M}^{-1} \mathbf{M}_f.$$

Assume $u(\boldsymbol{x},t) = \sum \boldsymbol{u}_j \phi_j(\boldsymbol{x})$ on D^k

- Compute numerical flux at face nodes (non-local).
- Compute RHS of (local) ODE.
- Evolve (local) solution using explicit time integration (RK, AB, etc).



$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \underbrace{\mathbf{D}_x \mathbf{u}}_{\text{Volume kernel}} + \underbrace{\sum_{\text{faces}} \mathbf{L}_f\,(\text{flux})}_{\text{Surface kernel}}.$$

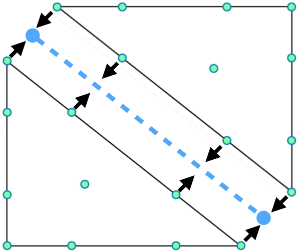
$$\boldsymbol{M}_{ij} = \int_{D^k} \phi_j(\boldsymbol{x})\phi_i(\boldsymbol{x})$$

$$\mathbf{L}_f = \mathbf{M}^{-1}\mathbf{M}_f.$$

Time-domain nodal DG methods

Assume $u(\boldsymbol{x}, t) = \sum \boldsymbol{u}_j \phi_j(\boldsymbol{x})$ on D^k

- Compute numerical flux at face nodes (non-local).
- Compute RHS of (local) ODE.
- Evolve (local) solution using explicit time integration (RK, AB, etc).



$$\underbrace{\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t}}_{\text{Update kernel}} = \underbrace{\mathbf{D}_x \mathbf{u}}_{\text{Volume kernel}} + \underbrace{\sum_{\text{faces}} \mathbf{L}_f \, (\text{flux})}_{\text{Surface kernel}}.$$

$$\boldsymbol{M}_{ij} = \int_{D^k} \phi_j(\boldsymbol{x}) \phi_i(\boldsymbol{x})$$

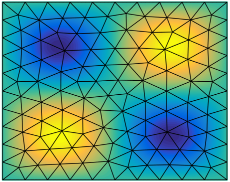
$$\mathbf{L}_f = \mathbf{M}^{-1} \mathbf{M}_f.$$

Outline

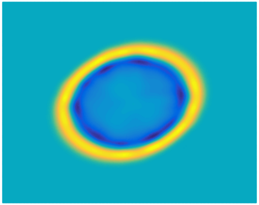
1. Weight-adjusted DG (WADG): high order heterogeneous media
2. Bernstein-Bezier WADG: high order efficiency

Outline

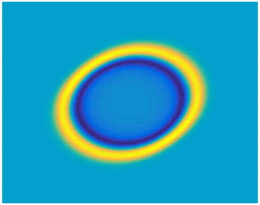
High order approximation of media and geometry



(a) Mesh and exact c^2



(b) Piecewise const. c^2



(c) High order c^2

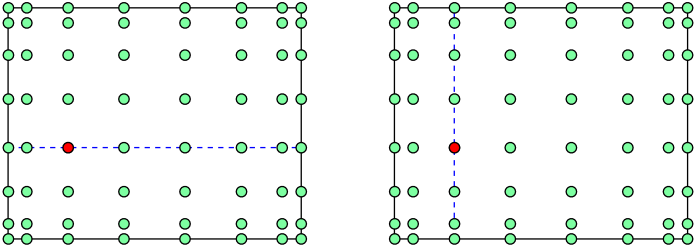
- Piecewise const. c^2: energy stable and efficient, but inaccurate.

$$\frac{1}{c^2(\boldsymbol{x})}\frac{\partial p}{\partial t} + \nabla \cdot \boldsymbol{u} = 0, \qquad \frac{\partial \boldsymbol{u}}{\partial t} + \nabla p = 0.$$

- High order wavespeeds: weighted mass matrices. Stable, but requires pre-computation/storage of inverses or factorizations!

$$\boldsymbol{M}_{1/c^2}\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = \boldsymbol{A}_h\boldsymbol{U}, \qquad \left(\boldsymbol{M}_{1/c^2}\right)_{ij} = \int_{D^k} \frac{1}{c^2(\boldsymbol{x})}\phi_j(\boldsymbol{x})\phi_i(\boldsymbol{x}).$$

Existing approaches: mass lumping



- DG-SEM: collocate at Gauss-Lobatto (or Gauss) points for a diagonal mass matrix. $O(N^4)$ total cost in 3D using Kronecker product.
- Limited to polynomial quads/hexes! Loss of stability or accuracy when extending to simplices (or prisms, pyramids, or non-polynomials).

Chan, Evans (2018). Multi-patch DG-IGA for wave propagation: explicit time-stepping and efficient mass matrix inversion.
Banks, Hagstrom (2016). On Galerkin difference methods.

Weight-adjusted DG: stable, accurate, non-invasive

- Weight-adjusted DG (WADG): energy stable approx. of $\boldsymbol{M}_{1/c^2}$

$$\boldsymbol{M}_{1/c^2}\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} \approx \boldsymbol{M}\left(\boldsymbol{M}_{c^2}\right)^{-1}\boldsymbol{M}\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = \boldsymbol{A}_h\boldsymbol{U}.$$

- New evaluation reuses implementation for constant wavespeed

$$\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = \underbrace{\boldsymbol{M}^{-1}\left(\boldsymbol{M}_{c^2}\right)}_{\text{modified update}} \qquad \underbrace{\boldsymbol{M}^{-1}\boldsymbol{A}_h\boldsymbol{U}}_{\text{constant wavespeed RHS}}$$

- Low storage matrix-free application of $\boldsymbol{M}^{-1}\boldsymbol{M}_{c^2}$ using **quadrature**-based interpolation and L^2 projection matrices $\boldsymbol{V}_q, \boldsymbol{P}_q$.

$$\left(\boldsymbol{M}\right)^{-1}\boldsymbol{M}_{c^2}\text{RHS} = \underbrace{\boldsymbol{M}^{-1}\boldsymbol{V}_q^T W}_{\boldsymbol{P}_q}\operatorname{diag}\left(c^2\right)\boldsymbol{V}_q\left(\text{RHS}\right).$$

Chan, Hewett, Warburton (2016). Weight-adjusted DG methods: wave propagation in heterogeneous media.

A weight-adjusted L^2 inner product

- "Reverse numerical integration": all operations on reference element.
- Let $T_w u = P_N(wu)$, define $T_w^{-1} : P^N \to P^N$ as

$$\left(w T_w^{-1} u, v\right) = (u, v), \qquad \forall v \in P^N.$$

- T_w^{-1} is "inverse" of weighted projection: $T_w T_w^{-1} = T_w^{-1} T_w = P_N$
- Weight-adjusted mass matrix: replace weighted L^2 inner product with "inverse of inverse weighting operator"

$$(wu, v) \implies \left(T_{1/w}^{-1} u, v\right).$$

Koutschan, Lehrenfeld, Schöberl (2011). Computer algebra meets FE: an efficient implementation for Maxwell's equations.

Estimates for WADG

- Generates norm with same equivalence constants

$$w_{\min} \boldsymbol{u}^T \boldsymbol{M} \boldsymbol{u} \leq \boldsymbol{u}^T \boldsymbol{M}_w \boldsymbol{u} \leq w_{\max} \boldsymbol{u}^T \boldsymbol{M} \boldsymbol{u}$$

- Accuracy of weighted "projection" P_w vs. WADG "projection" $\widetilde{P}_w$

$$\left\| u/w - \widetilde{P}_w u \right\|_{L^2} \leq C_w h^{N+1} \|w\|_{W^{N+1,\infty}} \|u\|_{W^{N+1,2}}$$

$$\left\| P_w u - \widetilde{P}_w u \right\|_{L^2} \leq C_{w,N} h^{N+2} \|w\|_{W^{N+1,\infty}} \|u\|_{W^{N+1,2}}$$

- WADG retains high order accuracy for moments: if $v \in P^M$

$$\left| \boldsymbol{v}^T \boldsymbol{M}_w \boldsymbol{u} - \boldsymbol{v}^T \boldsymbol{M} \boldsymbol{M}_{1/w}^{-1} \boldsymbol{M} \boldsymbol{u} \right| \leq C_w h^{2N+2-M} \|w\|_{W^{N+1,\infty}} \|u\|_{W^{N+1,2}} \|v\|_{L^2}$$

Estimates for WADG

- Generates norm with same equivalence constants

$$w_{\min} \boldsymbol{u}^T \boldsymbol{M} \boldsymbol{u} \leq \boldsymbol{u}^T \boldsymbol{M} \boldsymbol{M}_{1/w}^{-1} \boldsymbol{M} \boldsymbol{u} \leq w_{\max} \boldsymbol{u}^T \boldsymbol{M} \boldsymbol{u}$$

- Accuracy of weighted "projection" P_w vs. WADG "projection" $\widetilde{P}_w$

$$\left\| u/w - \widetilde{P}_w u \right\|_{L^2} \leq C_w h^{N+1} \|w\|_{W^{N+1,\infty}} \|u\|_{W^{N+1,2}}$$

$$\left\| P_w u - \widetilde{P}_w u \right\|_{L^2} \leq C_{w,N} h^{N+2} \|w\|_{W^{N+1,\infty}} \|u\|_{W^{N+1,2}}$$

- WADG retains high order accuracy for moments: if $v \in P^M$

$$\left| \boldsymbol{v}^T \boldsymbol{M}_w \boldsymbol{u} - \boldsymbol{v}^T \boldsymbol{M} \boldsymbol{M}_{1/w}^{-1} \boldsymbol{M} \boldsymbol{u} \right| \leq C_w h^{2N+2-M} \|w\|_{W^{N+1,\infty}} \|u\|_{W^{N+1,2}} \|v\|_{L^2}$$

Estimates for WADG

- Generates norm with same equivalence constants

$$w_{\min} \boldsymbol{u}^T \boldsymbol{M}\boldsymbol{u} \leq \boldsymbol{u}^T \boldsymbol{M}\boldsymbol{M}_{1/w}^{-1}\boldsymbol{M}\boldsymbol{u} \leq w_{\max} \boldsymbol{u}^T \boldsymbol{M}\boldsymbol{u}$$

- Accuracy of weighted "projection" P_w vs. WADG "projection" $\widetilde{P}_w$

$$\left\| u/w - \widetilde{P}_w u \right\|_{L^2} \leq C_w h^{N+1} \left\| w \right\|_{W^{N+1,\infty}} \left\| u \right\|_{W^{N+1,2}}$$

$$\left\| P_w u - \widetilde{P}_w u \right\|_{L^2} \leq C_{w,N} h^{N+2} \left\| w \right\|_{W^{N+1,\infty}} \left\| u \right\|_{W^{N+1,2}}$$

- WADG retains high order accuracy for moments: if $v \in P^M$

$$\left| \boldsymbol{v}^T \boldsymbol{M}_w \boldsymbol{u} - \boldsymbol{v}^T \boldsymbol{M}\boldsymbol{M}_{1/w}^{-1}\boldsymbol{M}\boldsymbol{u} \right| \leq C_w h^{2N+2-M} \left\| w \right\|_{W^{N+1,\infty}} \left\| u \right\|_{W^{N+1,2}} \left\| v \right\|_{L^2}$$

Estimates for WADG

- Generates norm with same equivalence constants

$$w_{\min}\boldsymbol{u}^T\boldsymbol{M}\boldsymbol{u} \leq \boldsymbol{u}^T\boldsymbol{M}\boldsymbol{M}_{1/w}^{-1}\boldsymbol{M}\boldsymbol{u} \leq w_{\max}\boldsymbol{u}^T\boldsymbol{M}\boldsymbol{u}$$

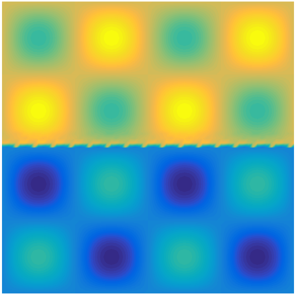
- Accuracy of weighted "projection" P_w vs. WADG "projection" $\widetilde{P}_w$

$$\left\| u/w - \widetilde{P}_w u \right\|_{L^2} \leq C_w h^{N+1} \left\| w \right\|_{W^{N+1,\infty}} \left\| u \right\|_{W^{N+1,2}}$$
$$\left\| P_w u - \widetilde{P}_w u \right\|_{L^2} \leq C_{w,N} h^{N+2} \left\| w \right\|_{W^{N+1,\infty}} \left\| u \right\|_{W^{N+1,2}}$$

- WADG retains high order accuracy for moments: if $v \in P^M$

$$\left| \boldsymbol{v}^T\boldsymbol{M}_w\boldsymbol{u} - \boldsymbol{v}^T\boldsymbol{M}\boldsymbol{M}_{1/w}^{-1}\boldsymbol{M}\boldsymbol{u} \right| \leq C_w h^{2N+2-M} \left\| w \right\|_{W^{N+1,\infty}} \left\| u \right\|_{W^{N+1,2}} \left\| v \right\|_{L^2}$$

WADG: nearly identical to using M^{-1}_{1/c^2}


(a) $c^2(x, y)$

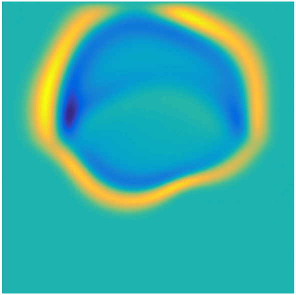
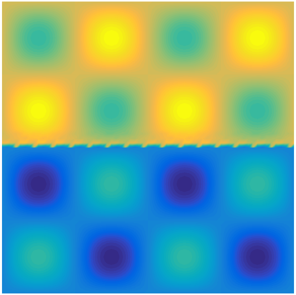

(b) Standard DG

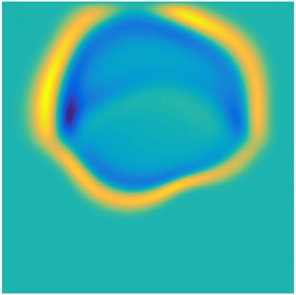
Figure: Standard vs. weight-adjusted DG with spatially varying c^2.

- Observed L^2 error is $O(h^{N+1})$; can prove $O(h^{N+1/2})$ convergence.

WADG: nearly identical to using $\boldsymbol{M}_{1/c^2}^{-1}$



(a) $c^2(x, y)$



(b) Weighted-adjusted DG

Figure: Standard vs. weight-adjusted DG with spatially varying c^2.

- Observed L^2 error is $O(h^{N+1})$; can prove $O(h^{N+1/2})$ convergence.

WADG: more efficient than storing $\boldsymbol{M}_{1/c^2}^{-1}$ on GPUs

	$N=1$	$N=2$	$N=3$	$N=4$	$N=5$	$N=6$	$N=7$
$\boldsymbol{M}_{1/c^2}^{-1}$	.66	2.79	9.90	29.4	73.9	170.5	329.4
WADG	0.59	1.44	4.30	13.9	43.0	107.8	227.7
Speedup	1.11	1.94	2.30	2.16	1.72	1.58	1.45

Time (ns) per element: storing/applying $\boldsymbol{M}_{1/c^2}^{-1}$ vs WADG (deg. $2N$ quadrature).

- Efficiency on GPUs: reduce memory accesses and data movement.
- (Tuned) low storage WADG faster than storing and applying $\boldsymbol{M}_{1/c^2}^{-1}$!

Matrix-valued weights and elastic wave propagation

- Symmetric velocity-stress formulation (entries of $\boldsymbol{A}_i$ either ± 1 or 0)

$$\rho \frac{\partial \boldsymbol{v}}{\partial t} = \sum_{i=1}^{d} \boldsymbol{A}_i^T \frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{x}_i}, \qquad \boldsymbol{C}^{-1} \frac{\partial \boldsymbol{\sigma}}{\partial t} = \sum_{i=1}^{d} \boldsymbol{A}_i \frac{\partial \boldsymbol{v}}{\partial \boldsymbol{x}_i}.$$

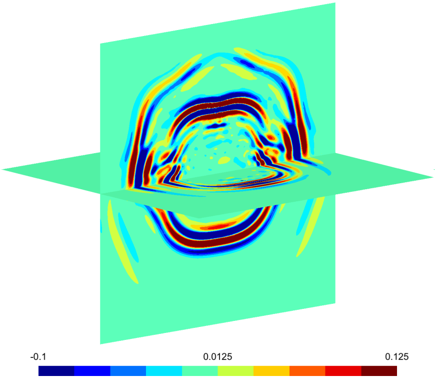
- DG formulation based on penalty fluxes, matrix-weighted mass matrix

$$\boldsymbol{M}_{\boldsymbol{C}^{-1}} = \begin{pmatrix} \boldsymbol{M}_{C_{11}^{-1}} & \dots & \boldsymbol{M}_{C_{1d}^{-1}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{M}_{C_{d1}^{-1}} & \dots & \boldsymbol{M}_{C_{dd}^{-1}} \end{pmatrix}$$

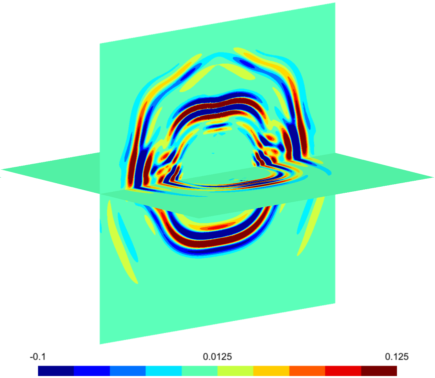
- Weight-adjusted approximation for $\boldsymbol{C}^{-1}$ decouples each component

$$\boldsymbol{M}_{\boldsymbol{C}^{-1}}^{-1} \approx \left(\boldsymbol{I} \otimes \boldsymbol{M}^{-1}\right) \boldsymbol{M}_{\boldsymbol{C}} \left(\boldsymbol{I} \otimes \boldsymbol{M}^{-1}\right).$$

Matrix-weighted WADG: elastic wave propagation



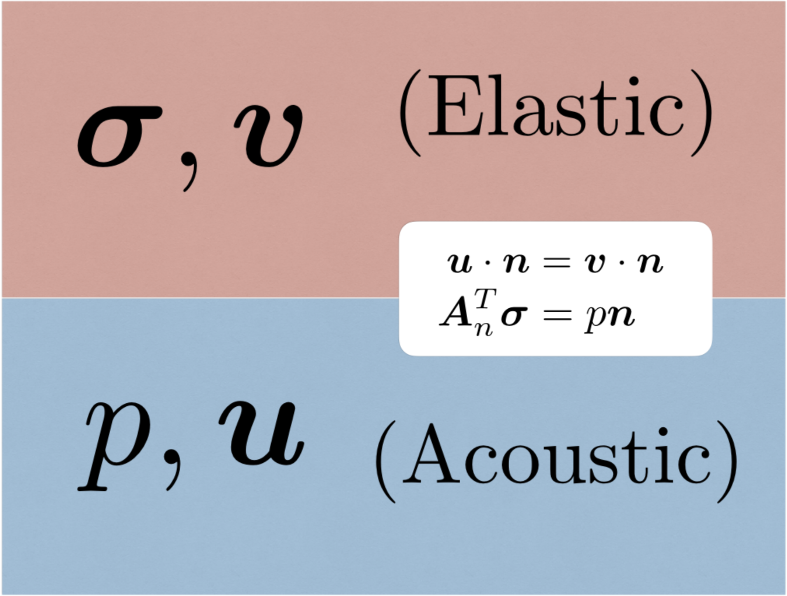
(a) Piecewise constant data



(b) High order data

Figure: $\mathrm{tr}(\boldsymbol{\sigma})$ with $\mu(\boldsymbol{x}) = 1 + H(y) + \frac{1}{2}\cos(3\pi x)\cos(3\pi y)\cos(3\pi z)$, $N = 5$.

Energy stable acoustic-elastic coupling



Energy stable acoustic-elastic coupling

(Elastic)

$$\frac{1}{2}\left\langle p\boldsymbol{n} - \boldsymbol{A}_n^T\boldsymbol{\sigma} - (\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^T)\boldsymbol{A}_n^T\boldsymbol{\sigma}, \boldsymbol{w}\right\rangle + \frac{\tau}{2}\left\langle (\boldsymbol{u} - \boldsymbol{v})\cdot\boldsymbol{n}, \boldsymbol{w}\cdot\boldsymbol{n}\right\rangle$$

$$\frac{1}{2}\left\langle (\boldsymbol{u} - \boldsymbol{v})\cdot\boldsymbol{n}, \boldsymbol{A}_n^T\boldsymbol{q}\right\rangle + \frac{\tau}{2}\left\langle (p\boldsymbol{n} - \boldsymbol{A}_n^T\boldsymbol{\sigma}), \boldsymbol{A}_n^T\boldsymbol{q}\right\rangle$$

$$\boldsymbol{u}\cdot\boldsymbol{n} = \boldsymbol{v}\cdot\boldsymbol{n}$$

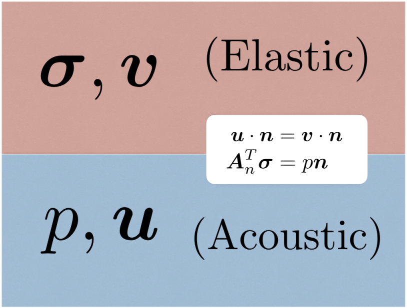
$$\boldsymbol{A}_n^T\boldsymbol{\sigma} = p\boldsymbol{n}$$

$$\frac{1}{2}\left\langle (\boldsymbol{A}_n^T\boldsymbol{\sigma} - p\boldsymbol{n})\cdot\boldsymbol{n}, \boldsymbol{w}\cdot\boldsymbol{n}\right\rangle + \frac{\tau}{2}\left\langle (\boldsymbol{v} - \boldsymbol{u})\cdot\boldsymbol{n}, \boldsymbol{w}\cdot\boldsymbol{n}\right\rangle$$

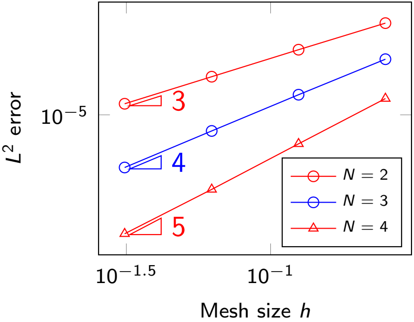
$$\frac{1}{2}\left\langle (\boldsymbol{v} - \boldsymbol{u})\cdot\boldsymbol{n}, q\right\rangle + \frac{\tau}{2}\left\langle (\boldsymbol{A}_n^T\boldsymbol{\sigma} - p\boldsymbol{n})\cdot\boldsymbol{n}, q\right\rangle$$

(Acoustic)

Energy stable acoustic-elastic coupling



(a) Coupling conditions



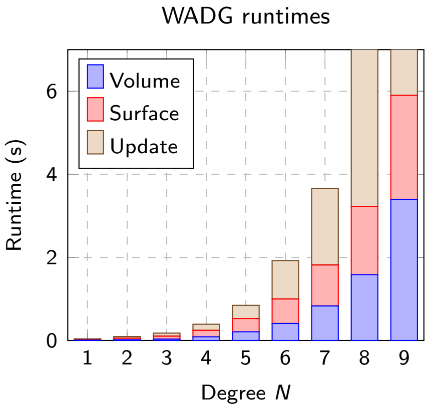
(b) Scholte wave

Straightforward penalty numerical fluxes in terms of interface residuals, energy stable and high order accurate for high order heterogeneous media.

Outline

Computational costs at high orders of approximation

Problem: WADG at high orders becomes **expensive**!

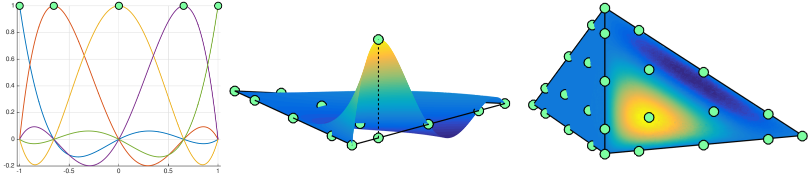


WADG runtimes for 50 timesteps, 98304 elements.

- Large **dense** matrices: $O(N^6)$ work per tet.
- High orders usually use tensor-product elements: $O(N^4)$ vs $O(N^6)$ cost, but less geometric flexibility.
- Idea: choose basis such that matrices are **sparse**.

BBDG: Bernstein-Bezier DG methods

- Nodal DG: $O(N^6)$ cost in 3D vs $O(N^3)$ degrees of freedom.
- Switch to Bernstein basis: sparse and structured matrices.
- Optimal $O(N^3)$ application of differentiation and lifting matrices.

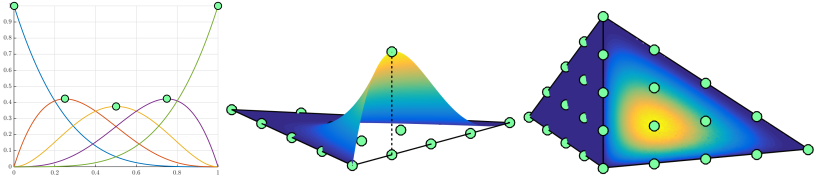


Nodal bases in one, two, and three dimensions.

Chan, Warburton 2015. GPU-accelerated Bernstein-Bezier discontinuous Galerkin methods for wave propagation (SISC).

BBDG: Bernstein-Bezier DG methods

- Nodal DG: $O(N^6)$ cost in 3D vs $O(N^3)$ degrees of freedom.
- Switch to Bernstein basis: sparse and structured matrices.
- Optimal $O(N^3)$ application of differentiation and lifting matrices.

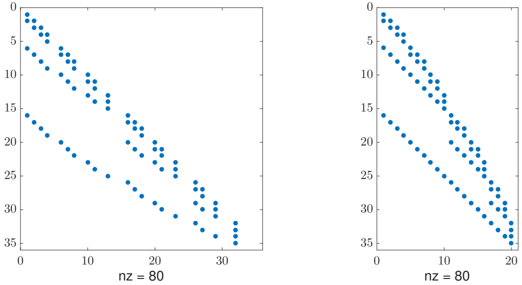


Bernstein bases in one, two, and three dimensions.

Chan, Warburton 2015. GPU-accelerated Bernstein-Bezier discontinuous Galerkin methods for wave propagation (SISC).

BBDG: Bernstein-Bezier DG methods

- Nodal DG: $O(N^6)$ cost in 3D vs $O(N^3)$ degrees of freedom.
- Switch to Bernstein basis: sparse and structured matrices.
- Optimal $O(N^3)$ application of differentiation and lifting matrices.

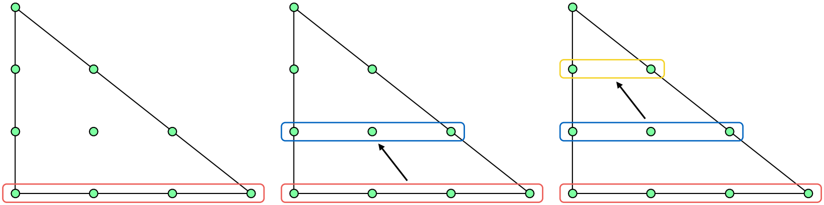


Tetrahedral Bernstein differentiation and degree elevation matrices.

Chan, Warburton 2015. GPU-accelerated Bernstein-Bezier discontinuous Galerkin methods for wave propagation (SISC).

BBDG: Bernstein-Bezier DG methods

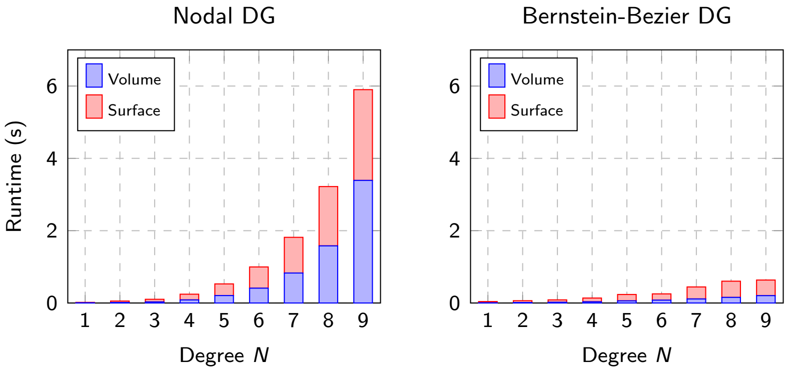
- Nodal DG: $O(N^6)$ cost in 3D vs $O(N^3)$ degrees of freedom.
- Switch to Bernstein basis: sparse and structured matrices.
- Optimal $O(N^3)$ application of differentiation and lifting matrices.



Optimal $O(N^3)$ complexity "slice-by-slice" application of Bernstein lift.

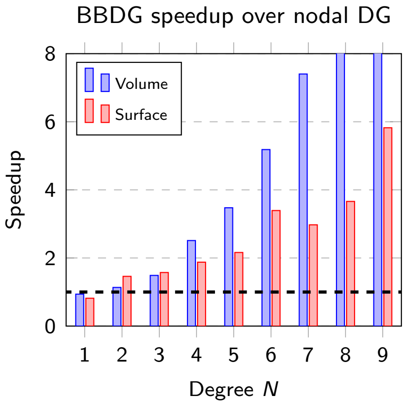
Chan, Warburton 2015. GPU-accelerated Bernstein-Bezier discontinuous Galerkin methods for wave propagation (SISC).

BBDG: efficient volume, surface kernels



$$\underbrace{\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t}}_{\text{Update kernel}} = \underbrace{\mathbf{D}_x \mathbf{u}}_{\text{Volume kernel}} + \underbrace{\sum_{\text{faces}} \mathbf{L}_f \,(\text{flux})}_{\text{Surface kernel}}, \quad \mathbf{L}_f = \mathbf{M}^{-1}\mathbf{M}_f.$$

BBDG: efficient volume, surface kernels



$$\underbrace{\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t}}_{\text{Update kernel}} = \underbrace{\mathbf{D}_x \mathbf{u}}_{\text{Volume kernel}} + \underbrace{\sum_{\text{faces}} \mathbf{L}_f\,(\text{flux})}_{\text{Surface kernel}}, \qquad \mathbf{L}_f = \mathbf{M}^{-1}\mathbf{M}_f.$$

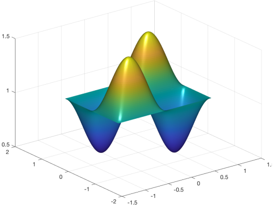
Goal: reduce computational complexity of WADG in 3D

- WADG: stable and accurate, but $O(N^6)$ operations per element.
- BBDG: fast $O(N^3)$ evaluation, but requires piecewise constant media
- Exploit continuous WADG approximation: given $u(\boldsymbol{x})$, compute

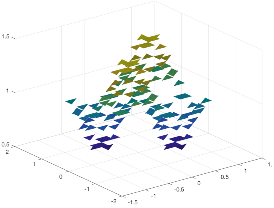
$$P_N\left(u(\boldsymbol{x})w(\boldsymbol{x})\right)$$

Applying $\boldsymbol{M}_w^{-1}$ is always $O(N^6)$ per element, so explicit expression for WADG is a prerequisite for reducing complexity.

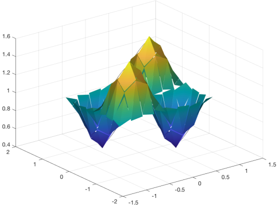
BBWADG: polynomial multiplication and projection



(a) Exact c^2



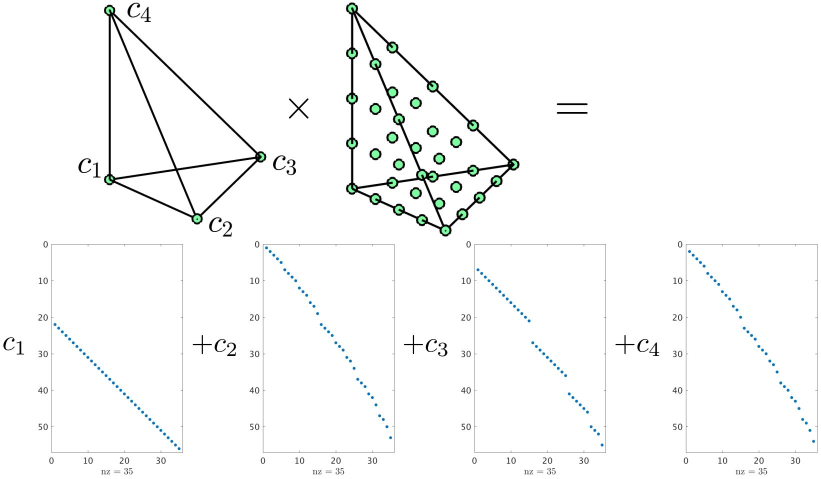
(b) $M = 0$ approximation



(c) $M = 1$ approximation

- $O(N^6)$ update kernel: multiplication by matrices $\boldsymbol{V}_q$ and $\boldsymbol{P}_q$.
- New approach: approx. $c^2(\boldsymbol{x})$ with degree M polynomial, use fast Bernstein algorithms for polynomial multiplication and projection.
- WADG: can reuse fast Bernstein volume and surface kernels.

Fast Bernstein polynomial multiplication



Bernstein polynomial multiplication ($M = 1$ shown), $O(N^3)$ cost for fixed M.

Fast Bernstein polynomial projection

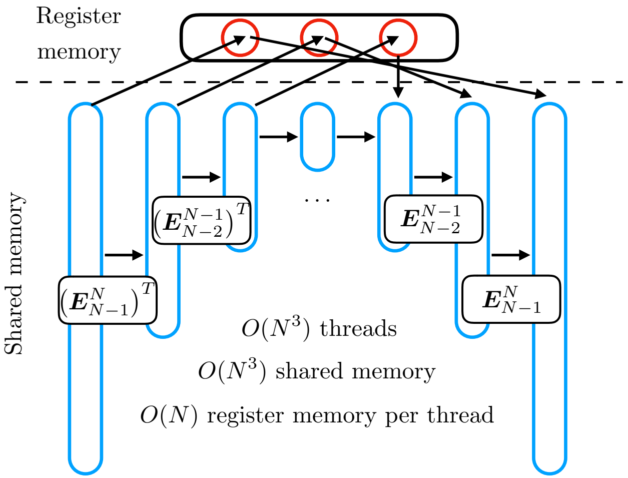
- Given $c^2(\boldsymbol{x})u(\boldsymbol{x})$ as a degree $(N+M)$ polynomial, apply L^2 projection matrix $\boldsymbol{P}_N^{N+M}$ to reduce to degree N.
- Polynomial L^2 projection matrix $\boldsymbol{P}_N^{N+M}$ under Bernstein basis:

$$\boldsymbol{P}_N^{N+M} = \underbrace{\sum_{j=0}^{N} c_j \boldsymbol{E}_{N-j}^{N} \left(\boldsymbol{E}_{N-j}^{N}\right)^T}_{\tilde{\boldsymbol{P}}_N} \left(\boldsymbol{E}_N^{N+M}\right)^T$$

- "Telescoping" form of $\tilde{\boldsymbol{P}}_N$: $O(N^4)$ complexity, more GPU-friendly.

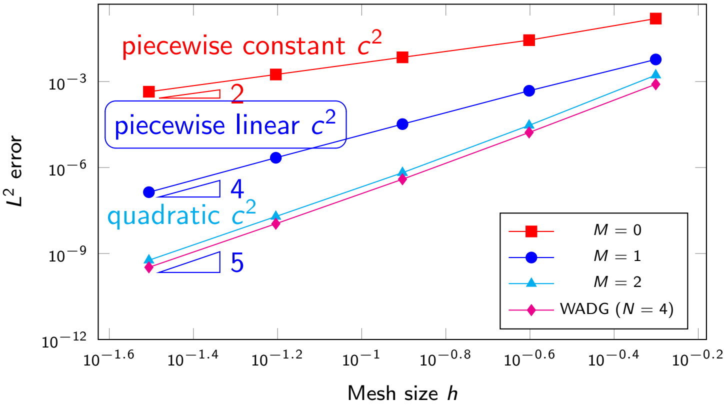
$$\left(c_0\boldsymbol{I} + \boldsymbol{E}_{N-1}^{N}\left(c_1\boldsymbol{I} + \boldsymbol{E}_{N-2}^{N-1}\left(c_2\boldsymbol{I} + \cdots\right)\left(\boldsymbol{E}_{N-2}^{N-1}\right)^T\right)\left(\boldsymbol{E}_{N-1}^{N}\right)^T\right)$$

Sketch of GPU algorithm for $\tilde{P}_N$



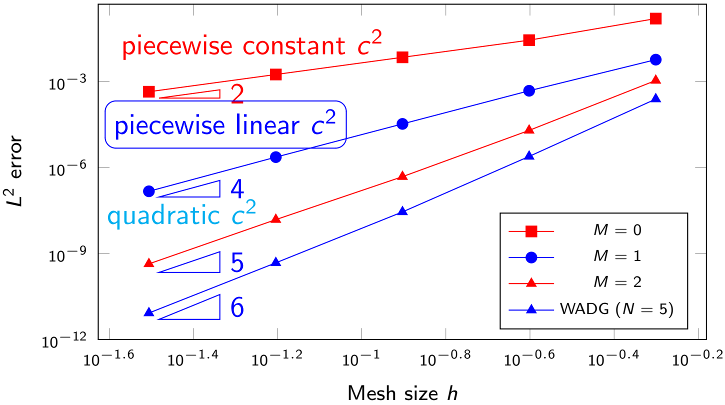
$$\left(c_0 \boldsymbol{I} + \boldsymbol{E}^N_{N-1}\left(c_1 \boldsymbol{I} + \boldsymbol{E}^{N-1}_{N-2}\left(c_2 \boldsymbol{I} + \cdots\right)\left(\boldsymbol{E}^{N-1}_{N-2}\right)^T\right)\left(\boldsymbol{E}^{N}_{N-1}\right)^T\right)$$

BBWADG: effect of approximating c^2 on accuracy



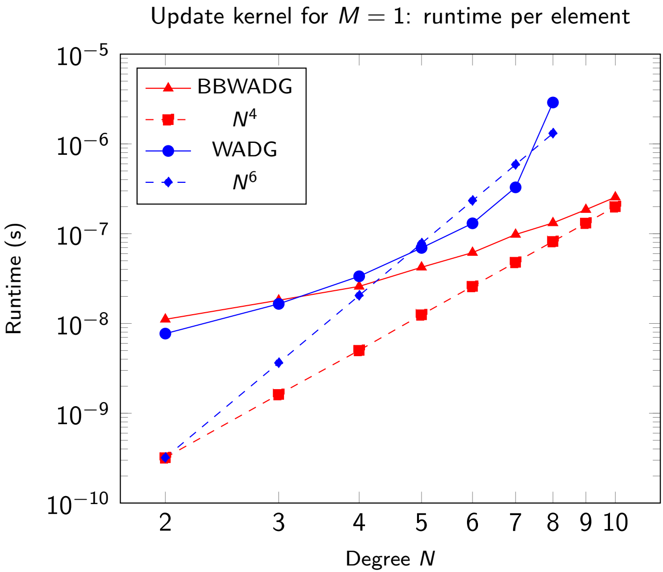
Approximating smooth $c^2(\boldsymbol{x})$ using L^2 projection:
$O(h^2)$ for $M = 0$, $O(h^4)$ for $M = 1$, $O(h^{M+3})$ for $0 < M \leq N - 2$.

BBWADG: effect of approximating c^2 on accuracy

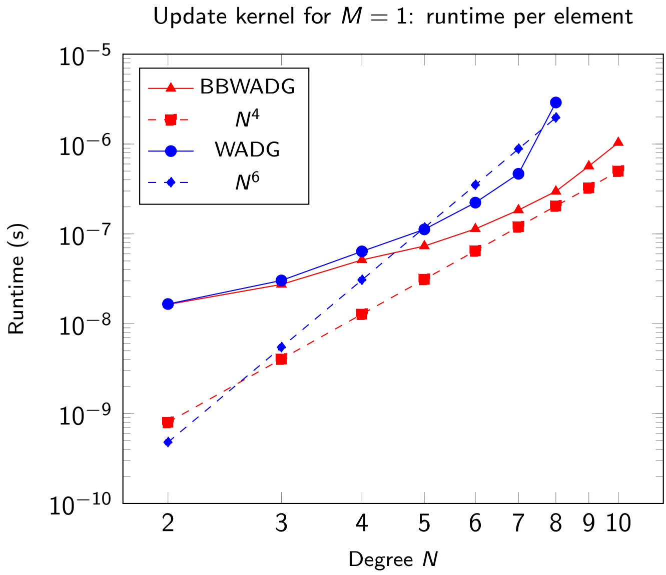


Approximating smooth $c^2(\boldsymbol{x})$ using L^2 projection:
$O(h^2)$ for $M = 0$, $O(h^4)$ for $M = 1$, $O(h^{M+3})$ for $0 < M \leq N - 2$.

BBWADG: computational runtime (3D acoustics)



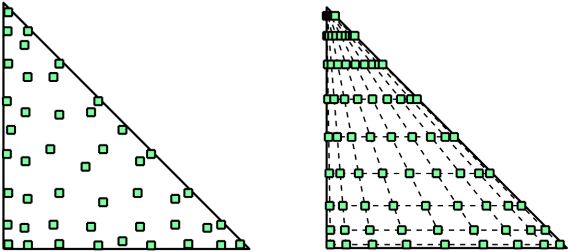
BBWADG: computational runtime (3D elasticity)



BBWADG: update kernel speedup (3D acoustics)

	$N=3$	$N=4$	$N=5$	$N=6$	$N=7$	$N=8$
WADG	1.65e-8	3.35e-8	6.94e-8	1.31e-7	3.28e-7	2.89e-6
BBWADG	1.81e-8	2.59e-8	4.22e-8	6.16e-8	9.79e-8	1.32e-7
Speedup	0.9116	1.2934	1.6445	2.1266	3.3504	21.8939

For $N \geq 8$, quadrature (and WADG) becomes much more expensive.



(a) $N=7$ quadrature (b) $N=8$ quadrature

Summary and acknowledgements

- Weight-adjusted DG: provable stability, high order accuracy, and efficiency in heterogeneous acoustic and elastic media.
- BBWADG: improved complexity for approximate wavespeeds.
- This work is supported by the National Science Foundation under DMS-1712639 and DMS-1719818.

Thank you! Questions?

Guo, Chan (2018). Bernstein-Bézier weight-adjusted DG methods for wave propagation in heterogeneous media.
Chan (2018). Weight-adjusted DG methods: matrix-valued weights and elastic wave prop. in heterogeneous media.
Chan, Hewett, Warburton (2017). Weight-adjusted DG methods: wave propagation in heterogeneous media.
Chan, Warburton (2017). GPU-accelerated Bernstein-Bezier discontinuous Galerkin methods for wave propagation.