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

Jesse Chan, Kaihang Guo

Department of Computational and Applied Math, Rice University

Rice Oil and Gas HPC Conference 2018
March 12-13, 2018

- 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.

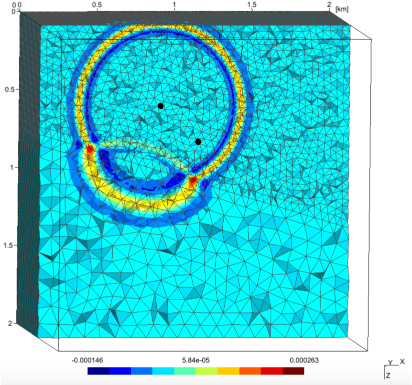
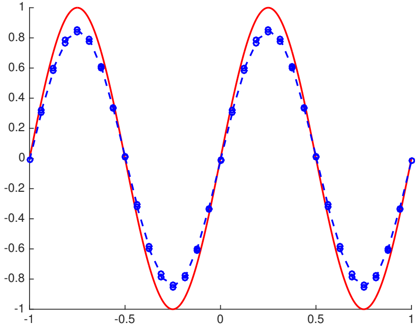


Figure courtesy of Axel Modave.

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.

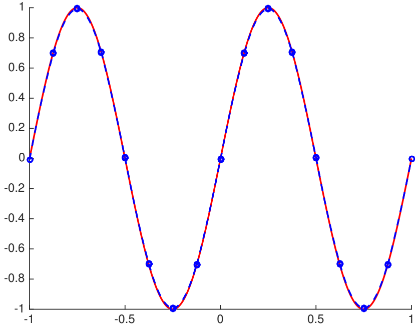


Fine linear approximation.

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.

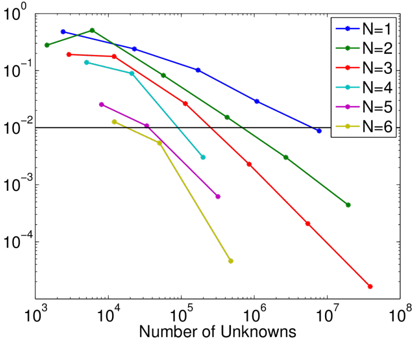


Coarse quadratic approximation.

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.

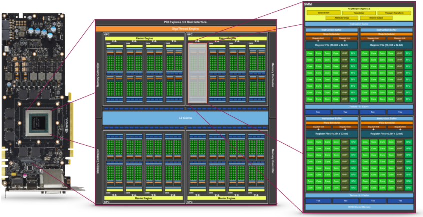


Max errors vs. dofs.

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.

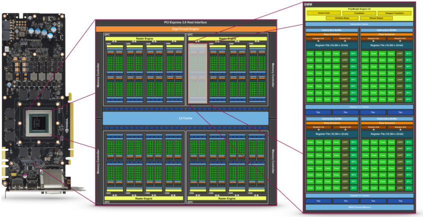


Graphics processing units (GPU).

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.



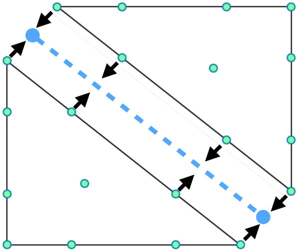
Graphics processing units (GPU).

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).



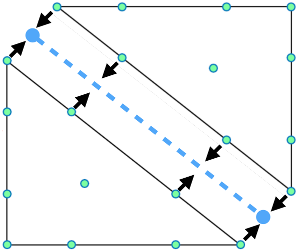
$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \mathbf{D}_x \mathbf{u} + \sum_{\text{faces}} \mathbf{L}_f \left(\text{flux}\right).$$

$$\boldsymbol{M}_{ij} = \int_{D^k} \phi_j(\boldsymbol{x})\phi_i(\boldsymbol{x})$$
$$\mathbf{L}_f = \mathbf{M}^{-1}\mathbf{M}_f.$$

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).



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

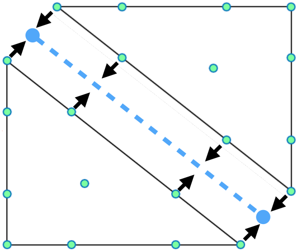
$$\boldsymbol{M}_{ij} = \int_{D^k} \phi_j(\boldsymbol{x}) \phi_i(\boldsymbol{x})$$

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

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).



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

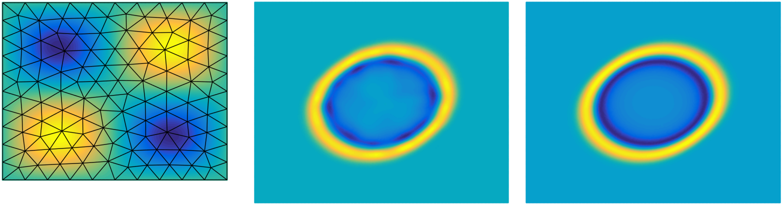
$$\boldsymbol{M}_{ij} = \int_{D^k} \phi_j(\boldsymbol{x}) \phi_i(\boldsymbol{x})$$

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

Outline

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 constant wavespeed c^2: efficient, but spurious reflections.

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

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

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

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

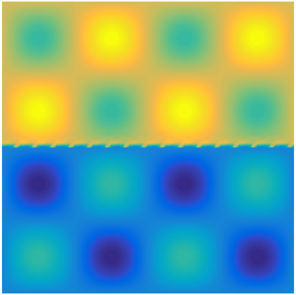
$$\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = \underbrace{\boldsymbol{M}^{-1}\left(\boldsymbol{M}_{c^2}\right)}_{\text{modified update}} \qquad \underbrace{\boldsymbol{M}^{-1}\boldsymbol{A}_h\boldsymbol{U}}_{\text{constant wavespeed RHS}}$$

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

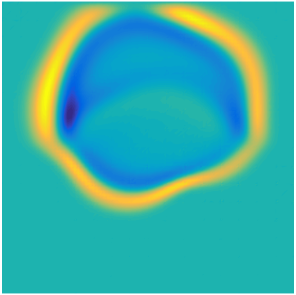
$$\left(\boldsymbol{M}\right)^{-1}\boldsymbol{M}_{c^2}\text{RHS} = \underbrace{\boldsymbol{M}^{-1}\boldsymbol{V}_q^T W}_{\boldsymbol{P}_q}\operatorname{diag}\left(c^2\right)\boldsymbol{V}_q\left(\text{RHS}\right).$$

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

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



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



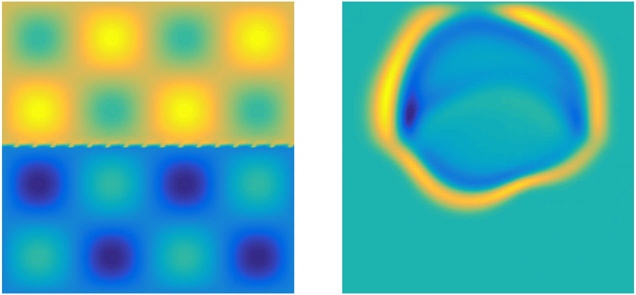
(b) Standard DG

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

- L^2 error is $O(h^{N+1})$; standard DG and WADG difference is $O(h^{N+2})$.
- Can generalize to matrix weights (elastic wave propagation).

Chan 2017. Weight-adjusted DG methods: matrix-valued weights and elastic wave prop. in heterogeneous media (IJNME).

WADG: nearly identical to using 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.

- L^2 error is $O(h^{N+1})$; standard DG and WADG difference is $O(h^{N+2})$.
- Can generalize to matrix weights (elastic wave propagation).

Chan 2017. Weight-adjusted DG methods: matrix-valued weights and elastic wave prop. in heterogeneous media (IJNME).

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

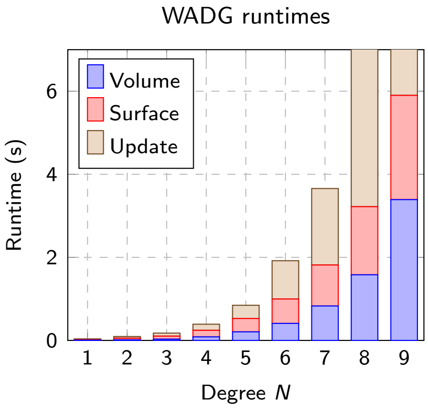
	$N=1$	$N=2$	$N=3$	$N=4$	$N=5$	$N=6$	$N=7$
$\boldsymbol{M}_{1/c^2}^{-1}$	.66	2.79	9.90	29.4	73.9	170.5	329.4
WADG	0.59	1.44	4.30	13.9	43.0	107.8	227.7
Speedup	1.11	1.94	2.30	2.16	1.72	1.58	1.45

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 $\boldsymbol{M}_{1/c^2}^{-1}$!

Computational costs at high orders of approximation

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

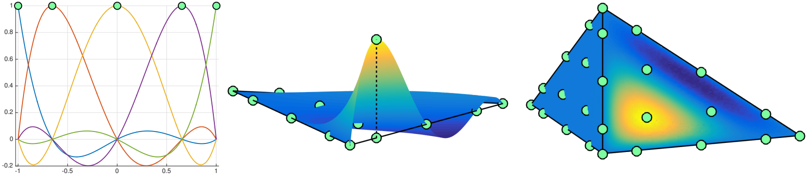


- Large **dense** matrices: $O(N^6)$ work per tet.
- High orders usually use tensor-product elements: $O(N^4)$ vs $O(N^6)$ cost, but less geometric flexibility.
- Idea: choose basis such that matrices are **sparse**.

WADG runtimes for 50 timesteps, 98304 elements.

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.

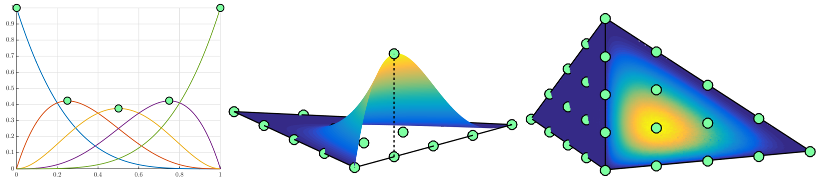


Nodal bases in one, two, and three dimensions.

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.

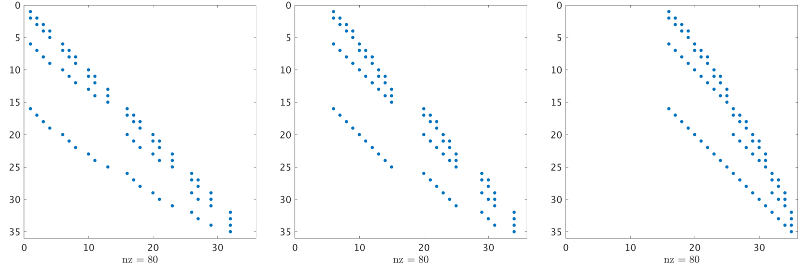


Bernstein bases in one, two, and three dimensions.

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.

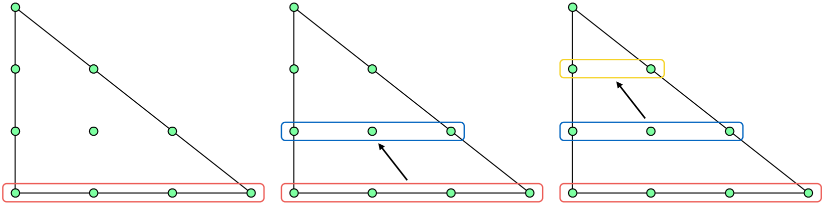


Sparse Bernstein differentiation matrices for the reference tetrahedron.

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

BBDG: Bernstein-Bezier DG methods

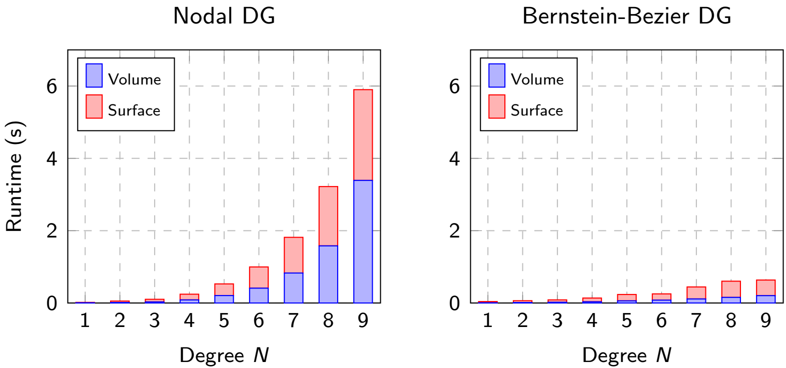
- Nodal DG: $O(N^6)$ cost in 3D vs $O(N^3)$ degrees of freedom.
- Switch to Bernstein basis: sparse and structured matrices.
- Optimal $O(N^3)$ application of differentiation and lifting matrices.



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

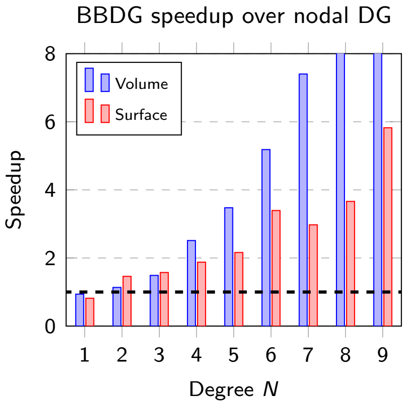
Chan, Warburton 2015. GPU-accelerated Bernstein-Bezier discontinuous Galerkin methods for wave propagation (SISC).

BBDG: efficient volume, surface kernels



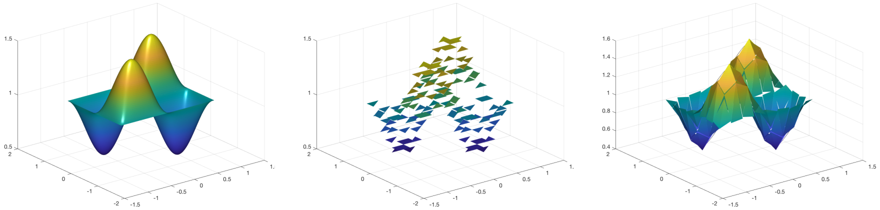
$$\underbrace{\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t}}_{\text{Update kernel}} = \underbrace{\mathbf{D}_x\mathbf{u}}_{\text{Volume kernel}} + \underbrace{\sum_{\text{faces}} \mathbf{L}_f\,(\text{flux})}_{\text{Surface kernel}}, \qquad \mathbf{L}_f = \mathbf{M}^{-1}\mathbf{M}_f.$$

BBDG: efficient volume, surface kernels



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

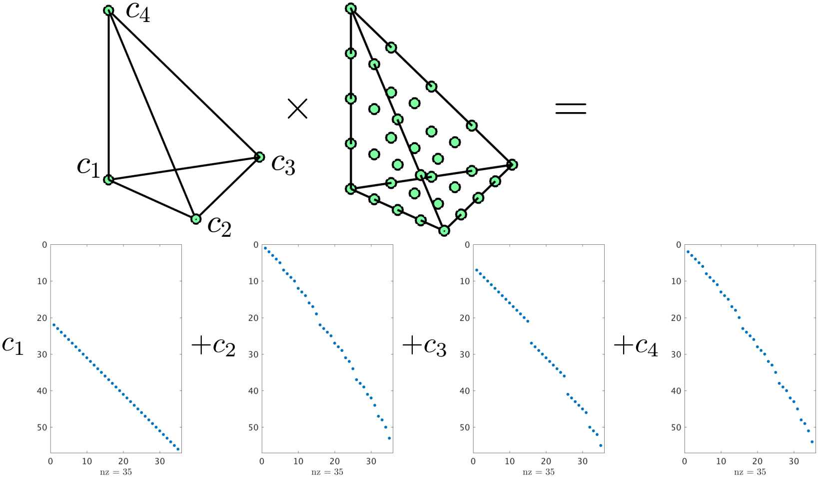
BBWADG: polynomial multiplication and projection



(a) Exact c^2 (b) $M = 0$ approximation (c) $M = 1$ approximation

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

Fast Bernstein polynomial multiplication



Bernstein polynomial multiplication: for fixed M, $O(N^3)$ complexity.

Fast Bernstein polynomial projection

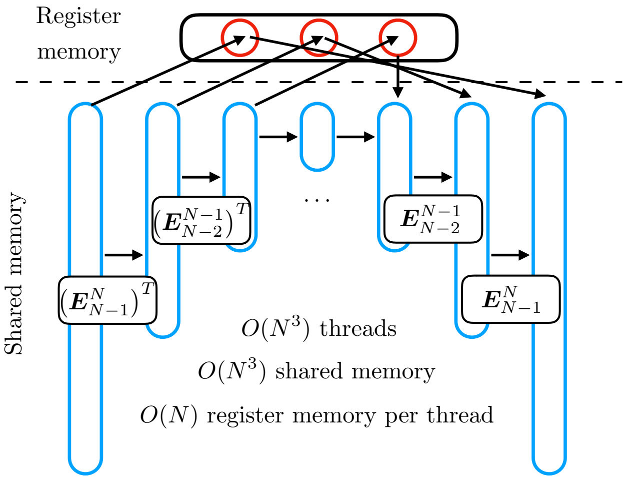
- Given $c^2(\boldsymbol{x})u(\boldsymbol{x})$ as a degree $(N+M)$ polynomial, apply L^2 projection matrix $\boldsymbol{P}_N^{N+M}$ to reduce to degree N.
- Polynomial L^2 projection matrix $\boldsymbol{P}_N^{N+M}$ under Bernstein basis:

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

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

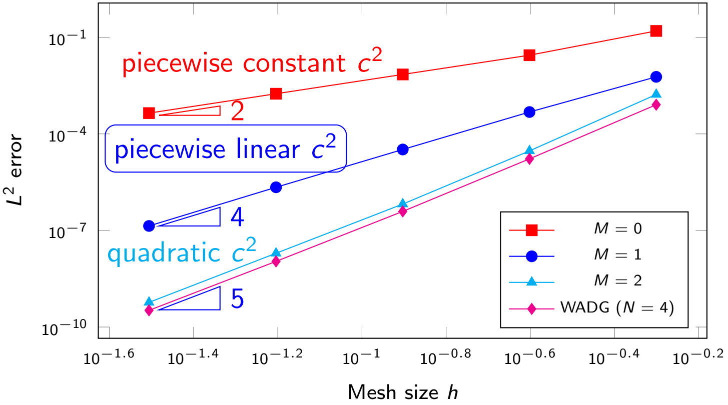
$$\left(c_0 \boldsymbol{I} + \boldsymbol{E}_{N-1}^{N} \left(c_1 \boldsymbol{I} + \boldsymbol{E}_{N-2}^{N-1} \left(c_2 \boldsymbol{I} + \cdots\right) \left(\boldsymbol{E}_{N-2}^{N-1}\right)^T\right) \left(\boldsymbol{E}_{N-1}^{N}\right)^T\right)$$

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



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

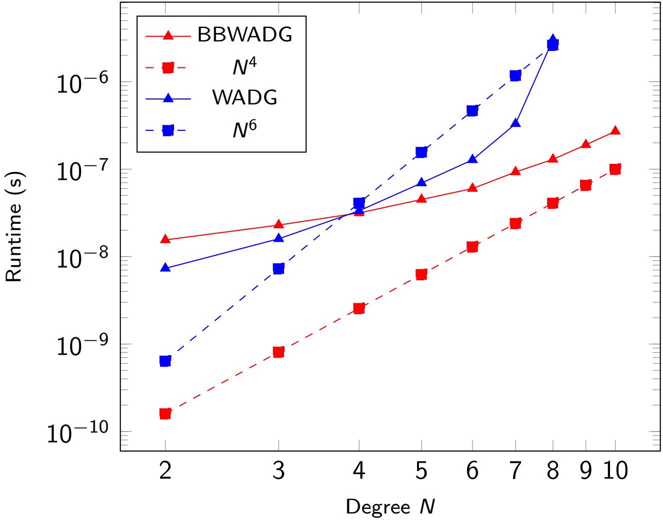
BBWADG: approximating c^2 and accuracy



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 (acoustics)

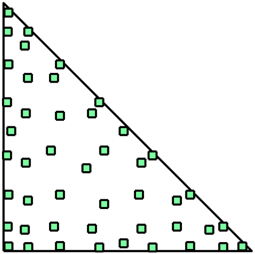
Update kernel for $M = 1$: runtime per element



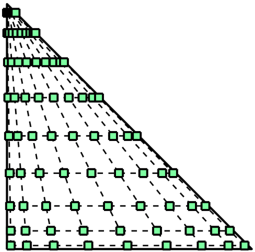
BBWADG: update kernel speedup over WADG (acoustics)

	$N = 3$	$N = 4$	$N = 5$	$N = 6$	$N = 7$	$N = 8$
WADG	1.60e-8	3.34e-8	6.94e-8	1.28e-7	3.31e-7	3.03e-6
BBWADG	2.20e-8	3.30e-8	4.42e-8	6.01e-8	9.46e-8	1.31e-7
Speedup	0.7260	1.0127	1.5706	2.1258	3.4938	23.1591

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: stability and efficiency for heterogeneous 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?

Chan, Hewett, Warburton. 2016. Weight-adjusted DG methods: wave propagation in heterogeneous media (SISC).
Chan 2017. Weight-adjusted DG methods: matrix-valued weights and elastic wave prop. in heterogeneous media (IJNME).
Chan, Warburton 2015. GPU-accelerated Bernstein-Bezier discontinuous Galerkin methods for wave propagation (SISC).