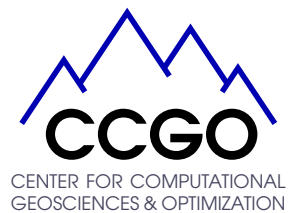
Spectral DG for Wave Propagation and Inversion in Coupled Acoustic-Elastic Media

Georg Stadler



Institute for Computational Engineering and Sciences
The University of Texas at Austin

Joint work with Tan Bui-Thanh, Carsten Burstedde,
Omar Ghattas & Lucas C. Wilcox

Quest Workshop
2010-09-23

Outline

Acoustic and elastic wave equations

Discontinuous Galerkin Discretizations
- Formulation
- Discontinuous spectral element implementation

Numerical examples
- Code verification (long-time, interfaces)
- Code scalability (speedup for CPU and GPU implementation)

Inversion for local wave speeds

Acoustic and elastic wave equations

Governing Equations in velocity-strain form

$$
\begin{aligned}
\frac{\partial \boldsymbol{E}}{\partial t} &= \frac{1}{2}\left(\nabla \boldsymbol{v} + \nabla \boldsymbol{v}^T\right) && \text{in } B \\
\rho \frac{\partial \boldsymbol{v}}{\partial t} &= \nabla \cdot \left(\lambda \operatorname{tr}(\boldsymbol{E})\boldsymbol{I} + 2\mu \boldsymbol{E}\right) + \rho \boldsymbol{f} && \text{in } B \\
\boldsymbol{S}\boldsymbol{n} &= \boldsymbol{t}^{\mathrm{bc}}(t) && \text{on } \partial B \\
\boldsymbol{v} &= \boldsymbol{v}_0(\boldsymbol{x}) && \text{at } t = 0 \\
\boldsymbol{E} &= \boldsymbol{E}_0(\boldsymbol{x}) && \text{at } t = 0
\end{aligned}
$$

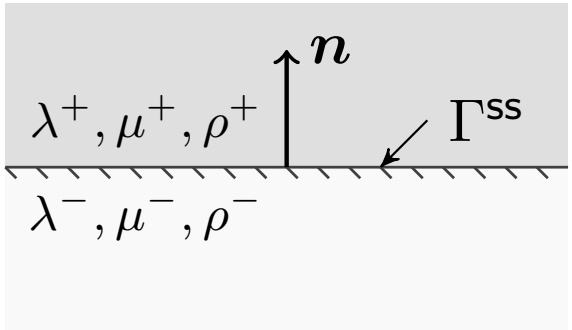
- $\boldsymbol{E}$ — strain tensor
- $\boldsymbol{S}$ — stress tensor
- ρ — mass density
- $\boldsymbol{v}$ — displacement velocity
- f — body force per unit mass
- λ and μ — Lamé parameters
- $\boldsymbol{I}$ — identity tensor
- $\boldsymbol{t}^{\mathrm{bc}}$ — traction bc
- $\boldsymbol{v}_0, \boldsymbol{E}_0$ — initial conditions
- t — time
- $\boldsymbol{x}$ — point in the body
- B — solution body

Acoustic and elastic wave equations

Elastic-Elastic and Elastic-Acoustic Interface conditions

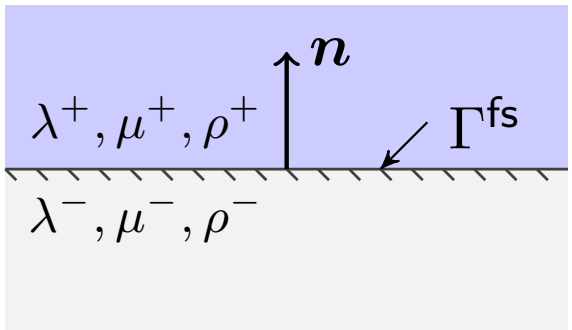
Solid-solid interface Γ^{ss}:

$$\boldsymbol{v}^+ = \boldsymbol{v}^- \qquad \text{on } \Gamma^{ss}$$
$$\boldsymbol{S}^+\boldsymbol{n} = \boldsymbol{S}^-\boldsymbol{n} \qquad \text{on } \Gamma^{ss}$$



Fluid-solid interface Γ^{fs}: (also valid for fluid-fluid)

$$\boldsymbol{n}\cdot\boldsymbol{v}^+ = \boldsymbol{n}\cdot\boldsymbol{v}^- \qquad \text{on } \Gamma^{fs}$$
$$\boldsymbol{S}^+\boldsymbol{n} = \boldsymbol{S}^-\boldsymbol{n} \qquad \text{on } \Gamma^{fs}$$



Outline

Acoustic and elastic wave equations

Discontinuous Galerkin Discretizations
- Formulation
- Discontinuous spectral element implementation

Numerical examples
- Code verification (long-time, interfaces)
- Code scalability (speedup for CPU and GPU implementation)

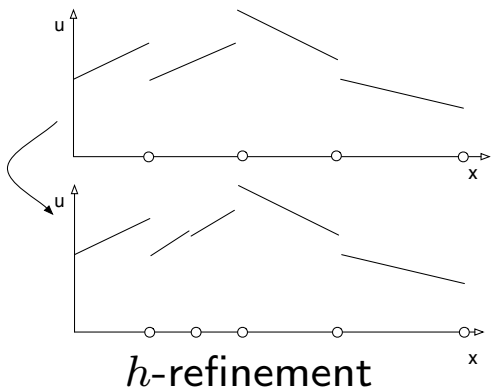
Inversion for local wave speeds

Discontinuous Galerkin

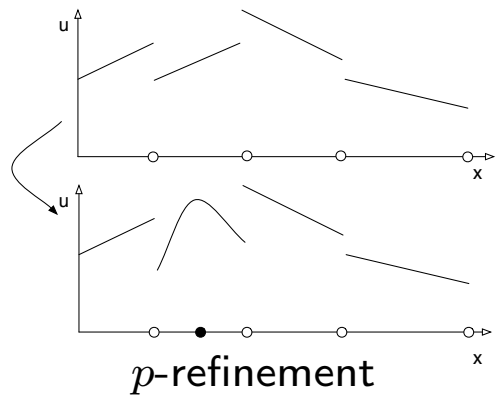
- Solutions are piecewise smooth and likely discontinuous between elements—weak enforcement of continuity between elements and boundary conditions through numerical flux

Discontinuous Galerkin

- Solutions are piecewise smooth and likely discontinuous between elements—weak enforcement of continuity between elements and boundary conditions through numerical flux
- Straight-forward implementation of variable order, variable size elements, and parallelization



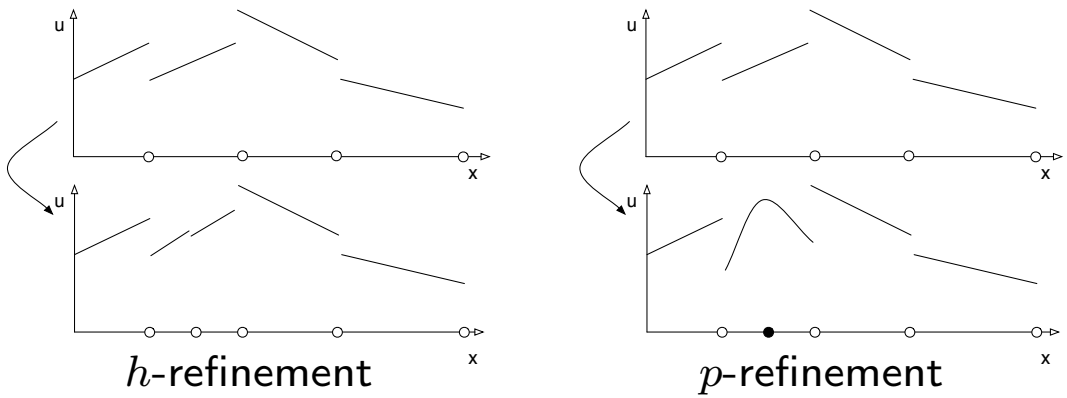
h-refinement



p-refinement

Discontinuous Galerkin

- Solutions are piecewise smooth and likely discontinuous between elements—weak enforcement of continuity between elements and boundary conditions through numerical flux
- Straight-forward implementation of variable order, variable size elements, and parallelization

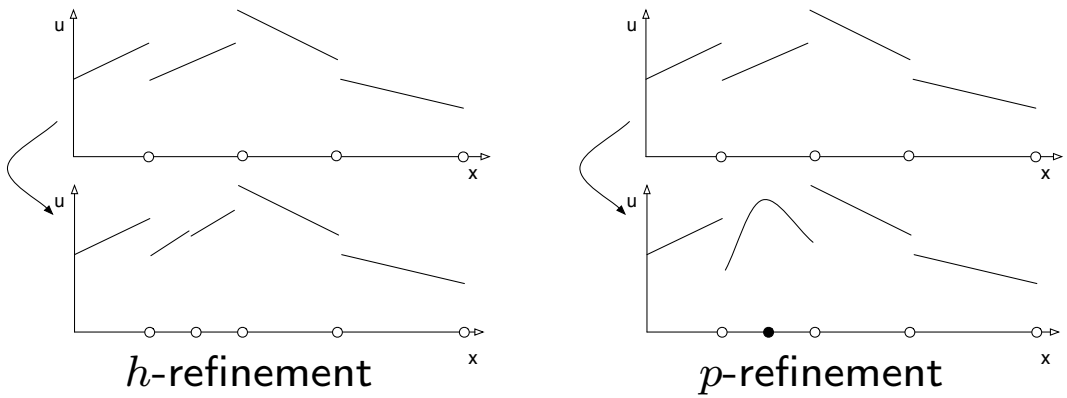


h-refinement p-refinement

- First-order DG formulation allows for straightforward coupling of acoustic and elastic regions through numerical flux

Discontinuous Galerkin

- Solutions are piecewise smooth and likely discontinuous between elements—weak enforcement of continuity between elements and boundary conditions through numerical flux
- Straight-forward implementation of variable order, variable size elements, and parallelization



h-refinement p-refinement

- First-order DG formulation allows for straightforward coupling of acoustic and elastic regions through numerical flux
- Build on scalable parallel 3D-mesh generator, support for parallel adaptive mesh refinement

Discontinuous Galerkin discretization

General Form

The DG discretization of the elastic-acoustic wave equations is given by: Find $(\mathbf{E}, \boldsymbol{v}) \in V$ such that for all elements D^e:

$$\int_{\mathrm{D}^e} \frac{\partial \boldsymbol{E}}{\partial t} : \mathbf{C}\boldsymbol{H}\, d\boldsymbol{x} + \int_{\mathrm{D}^e} \rho^- \frac{\partial \boldsymbol{v}}{\partial t} \cdot \boldsymbol{w}\, d\boldsymbol{x} - \int_{\mathrm{D}^e} \frac{1}{2}\left(\nabla \boldsymbol{v} + \nabla \boldsymbol{v}^T\right) : \mathbf{C}\boldsymbol{H}\, d\boldsymbol{x}$$
$$- \int_{\mathrm{D}^e} (\nabla \cdot (\mathbf{C}\boldsymbol{E})) \cdot \boldsymbol{w}\, d\boldsymbol{x} + \int_{\partial \mathrm{D}^e} \mathfrak{F}_{\boldsymbol{v}} : \mathbf{C}\boldsymbol{H} + \mathfrak{F}_{\boldsymbol{E}} \cdot \boldsymbol{w}\, d\boldsymbol{x}$$

for all test functions $(\boldsymbol{H}, \boldsymbol{w})$,where $\mathfrak{F}_{\boldsymbol{v}}$ and $\mathfrak{F}_{\boldsymbol{E}}$ are the numerical fluxes.

Discontinuous Galerkin discretization

General Form

The DG discretization of the elastic-acoustic wave equations is given by: Find $(\mathbf{E}, \boldsymbol{v}) \in V$ such that for all elements D^e:

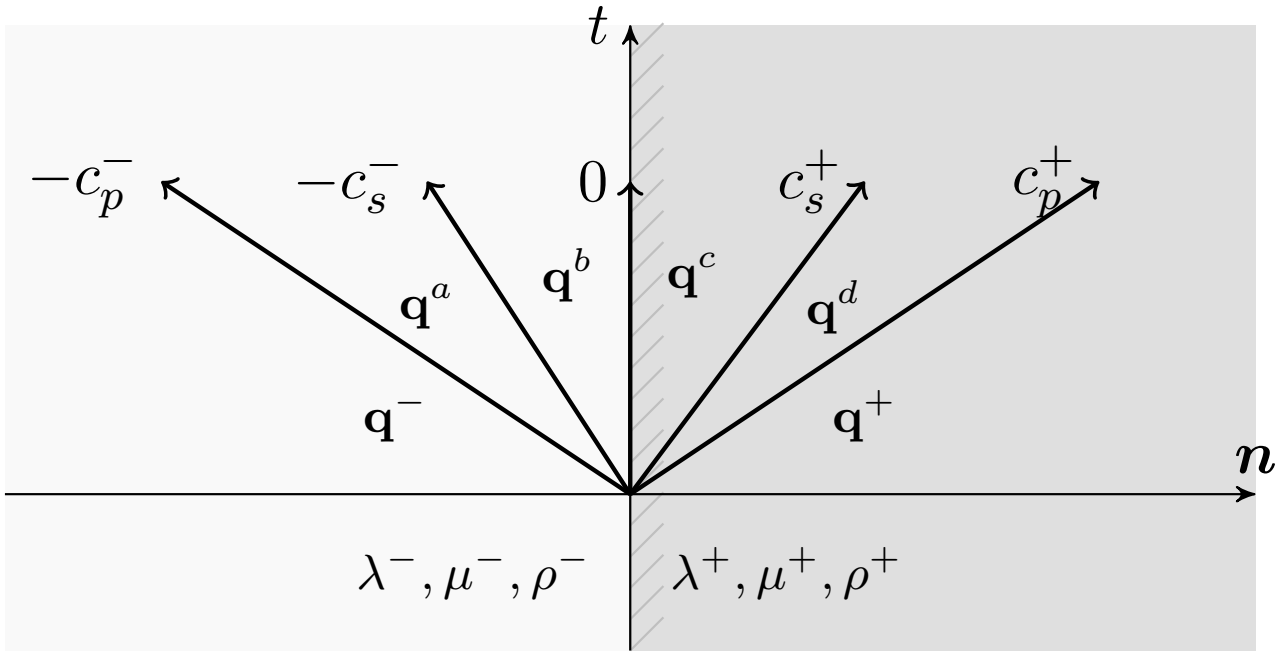
$$\int_{\mathrm{D}^e} \frac{\partial \boldsymbol{E}}{\partial t} : \mathbf{C}\boldsymbol{H}\, d\boldsymbol{x} + \int_{\mathrm{D}^e} \rho^- \frac{\partial \boldsymbol{v}}{\partial t} \cdot \boldsymbol{w}\, d\boldsymbol{x} - \int_{\mathrm{D}^e} \frac{1}{2} \left(\nabla \boldsymbol{v} + \nabla \boldsymbol{v}^T\right) : \mathbf{C}\boldsymbol{H}\, d\boldsymbol{x}$$
$$- \int_{\mathrm{D}^e} (\nabla \cdot (\mathbf{C}\boldsymbol{E})) \cdot \boldsymbol{w}\, d\boldsymbol{x} + \int_{\partial \mathrm{D}^e} \mathfrak{F}_{\boldsymbol{v}} : \mathbf{C}\boldsymbol{H} + \mathfrak{F}_{\boldsymbol{E}} \cdot \boldsymbol{w}\, d\boldsymbol{x}$$

for all test functions $(\boldsymbol{H}, \boldsymbol{w})$,where $\mathfrak{F}_{\boldsymbol{v}}$ and $\mathfrak{F}_{\boldsymbol{E}}$ are the numerical fluxes.

$\Rightarrow$ Compute the numerical flux by solving the Riemann problem with discontinuous material parameters

Numerical flux computation

Elastic-elastic interface



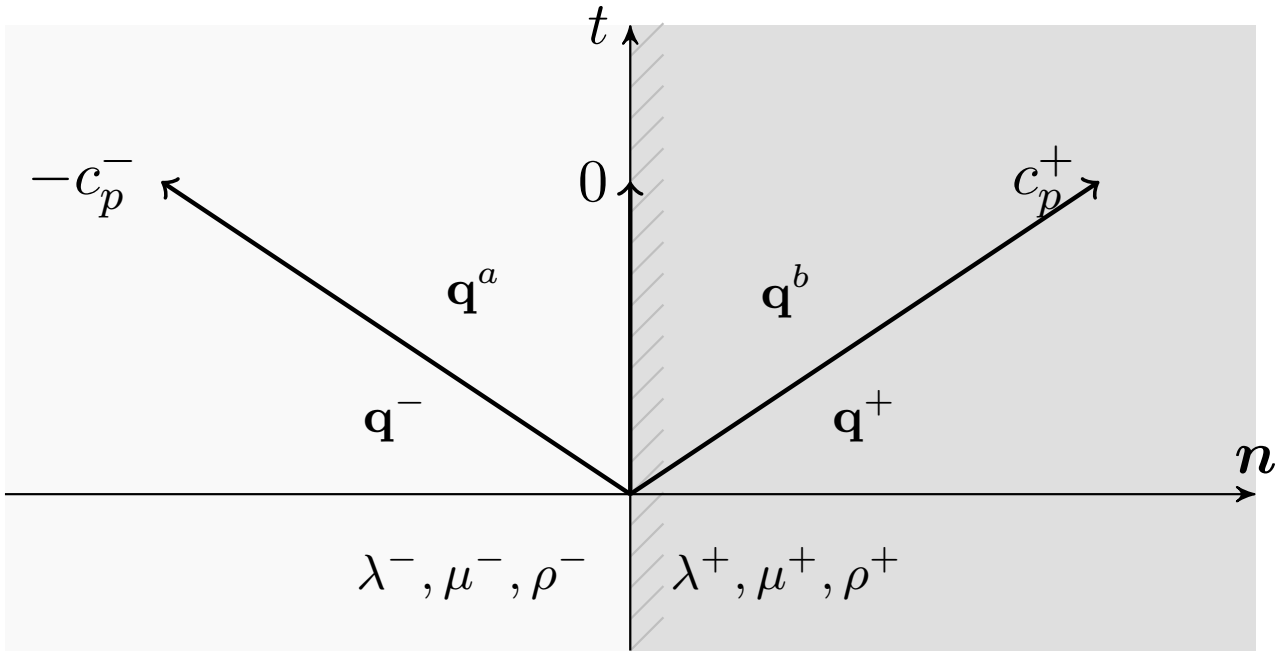
Solve the acoustic-elastic wave equation for a given piecewise constant material with the initial condition

$$\mathbf{q}_0(\boldsymbol{x}) = \begin{cases} \mathbf{q}^- & \text{if } \boldsymbol{n} \cdot \boldsymbol{x} < 0, \\ \mathbf{q}^+ & \text{if } \boldsymbol{n} \cdot \boldsymbol{x} > 0. \end{cases}$$

The upwind numerical flux along $\boldsymbol{n}$ is defined as flux in this Riemann problem across the $\boldsymbol{n} \cdot \boldsymbol{x} = 0$ interface.

Numerical flux computation

Acoustic-elastic interface



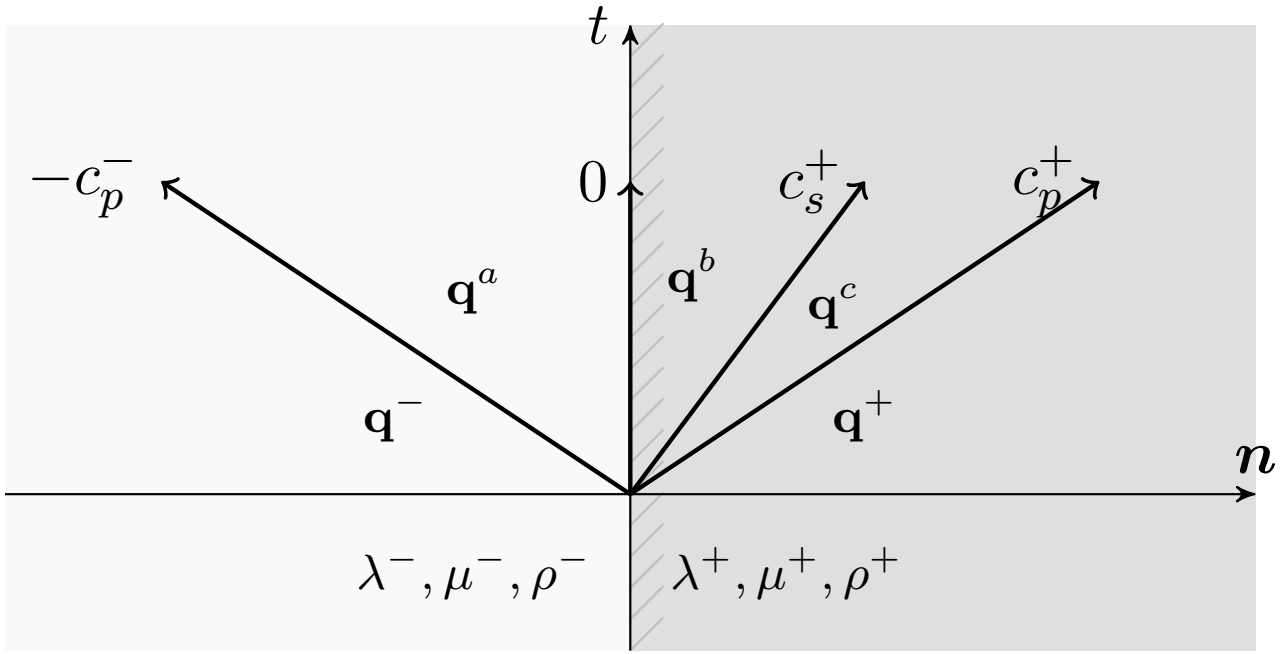
Solve the acoustic-elastic wave equation for a given piecewise constant material with the initial condition

$$\mathbf{q}_0(\boldsymbol{x}) = \begin{cases} \mathbf{q}^- & \text{if } \boldsymbol{n} \cdot \boldsymbol{x} < 0, \\ \mathbf{q}^+ & \text{if } \boldsymbol{n} \cdot \boldsymbol{x} > 0. \end{cases}$$

The upwind numerical flux along $\boldsymbol{n}$ is defined as flux in this Riemann problem across the $\boldsymbol{n} \cdot \boldsymbol{x} = 0$ interface.

Numerical flux computation

Acoustic-acoustic interface



Solve the acoustic-elastic wave equation for a given piecewise constant material with the initial condition

$$\mathbf{q}_0(\boldsymbol{x}) = \begin{cases} \mathbf{q}^- & \text{if } \boldsymbol{n} \cdot \boldsymbol{x} < 0, \\ \mathbf{q}^+ & \text{if } \boldsymbol{n} \cdot \boldsymbol{x} > 0. \end{cases}$$

The upwind numerical flux along $\boldsymbol{n}$ is defined as flux in this Riemann problem across the $\boldsymbol{n} \cdot \boldsymbol{x} = 0$ interface.

Discontinuous Galerkin discretization

Upwind Numerical Flux (unified form)

$$
\begin{aligned}
\mathfrak{F}_{\boldsymbol{v}} &= k_0 \left(\boldsymbol{n} \cdot [\![\mathbf{C}\boldsymbol{E}]\!] + \rho^+ c_p^+ [\![\boldsymbol{v}]\!] \right) \boldsymbol{n} \otimes \boldsymbol{n} \\
&\quad - k_1 \operatorname{sym} \left(\boldsymbol{n} \otimes \left(\boldsymbol{n} \times \left(\boldsymbol{n} \times [\![\mathbf{C}\boldsymbol{E}]\!] \right) \right) \right) \\
&\quad - k_1 \rho^+ c_s^+ \operatorname{sym} \left(\boldsymbol{n} \otimes \left(\boldsymbol{n} \times \left(\boldsymbol{n} \times [\![\boldsymbol{v}]\!] \right) \right) \right), \\
\mathfrak{F}_{\boldsymbol{E}} &= k_0 \left(\boldsymbol{n} \cdot [\![\mathbf{C}\boldsymbol{E}]\!] + \rho^+ c_p^+ [\![\boldsymbol{v}]\!] \right) \rho^- c_p^- \boldsymbol{n} \\
&\quad - k_1 \rho^- c_s^- \boldsymbol{n} \times \left(\boldsymbol{n} \times [\![\mathbf{C}\boldsymbol{E}]\!] \right) \\
&\quad - k_1 \rho^+ c_s^+ \rho^- c_s^- \boldsymbol{n} \times \left(\boldsymbol{n} \times [\![\boldsymbol{v}]\!] \right).
\end{aligned}
$$

Here, $\boldsymbol{n}$ is the unit outward normal and k_0 and k_1 are given by

$$
k_0 = \frac{1}{\rho^- c_p^- + \rho^+ c_p^+}, \quad k_1 = \frac{1}{\rho^- c_s^- + \rho^+ c_s^+}.
$$

Discontinuous Galerkin discretization

Upwind Numerical Flux (unified form)

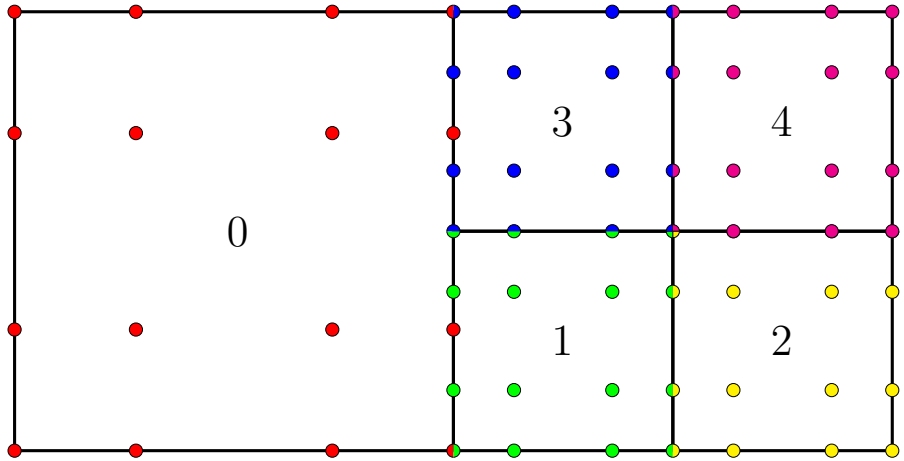
$$
\begin{aligned}
\mathfrak{F}_{\boldsymbol{v}} &= k_0 \left(\boldsymbol{n} \cdot [\![\mathbf{C}\boldsymbol{E}]\!] + \rho^+ c_p^+ [\![\boldsymbol{v}]\!] \right) \boldsymbol{n} \otimes \boldsymbol{n} \\
&\quad - k_1 \operatorname{sym} \left(\boldsymbol{n} \otimes \left(\boldsymbol{n} \times \left(\boldsymbol{n} \times [\![\mathbf{C}\boldsymbol{E}]\!] \right) \right) \right) \\
&\quad - k_1 \rho^+ c_s^+ \operatorname{sym} \left(\boldsymbol{n} \otimes \left(\boldsymbol{n} \times \left(\boldsymbol{n} \times [\![\boldsymbol{v}]\!] \right) \right) \right), \\
\mathfrak{F}_{\boldsymbol{E}} &= k_0 \left(\boldsymbol{n} \cdot [\![\mathbf{C}\boldsymbol{E}]\!] + \rho^+ c_p^+ [\![\boldsymbol{v}]\!] \right) \rho^- c_p^- \boldsymbol{n} \\
&\quad - k_1 \rho^- c_s^- \boldsymbol{n} \times \left(\boldsymbol{n} \times [\![\mathbf{C}\boldsymbol{E}]\!] \right) \\
&\quad - k_1 \rho^+ c_s^+ \rho^- c_s^- \boldsymbol{n} \times \left(\boldsymbol{n} \times [\![\boldsymbol{v}]\!] \right).
\end{aligned}
$$

Here, $\boldsymbol{n}$ is the unit outward normal and k_0 and k_1 are given by

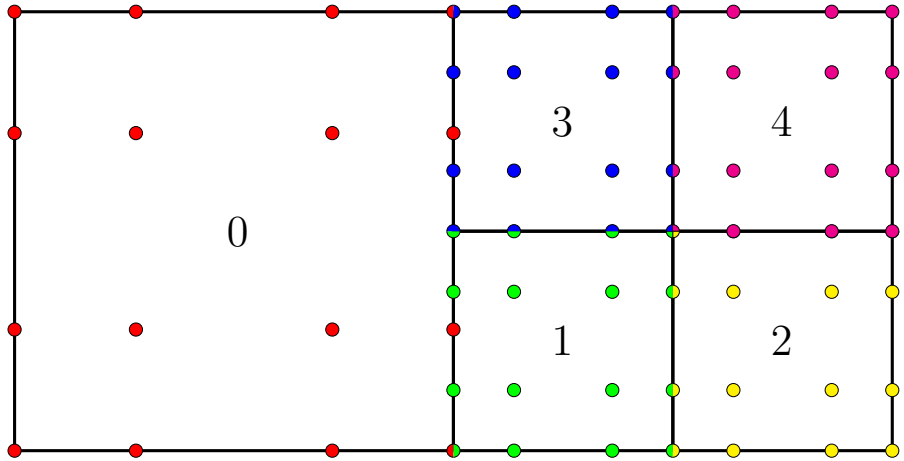
$$
k_0 = \frac{1}{\rho^- c_p^- + \rho^+ c_p^+}, \quad k_1 = \frac{1}{\rho^- c_s^- + \rho^+ c_s^+}.
$$

Note that numerical flux involves local wave speeds from both sides of element interfaces. In the liquid, we simply set $c_s = k_1 = 0$.

Discontinuous spectral element implementation

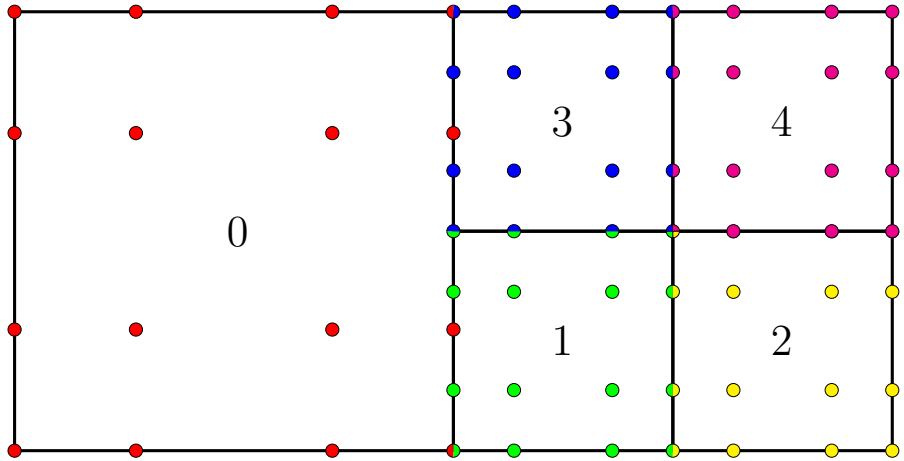


Discontinuous spectral element implementation



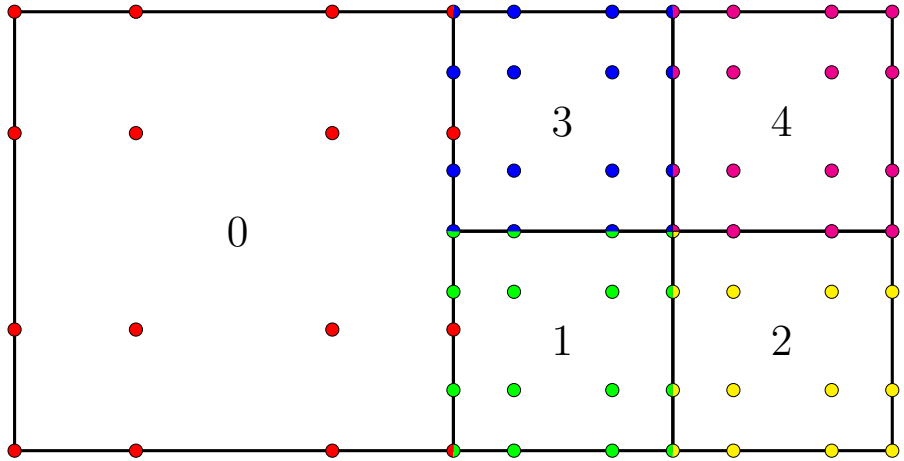
- Allows h-nonconforming hexahedral elements (2:1 balance)

Discontinuous spectral element implementation



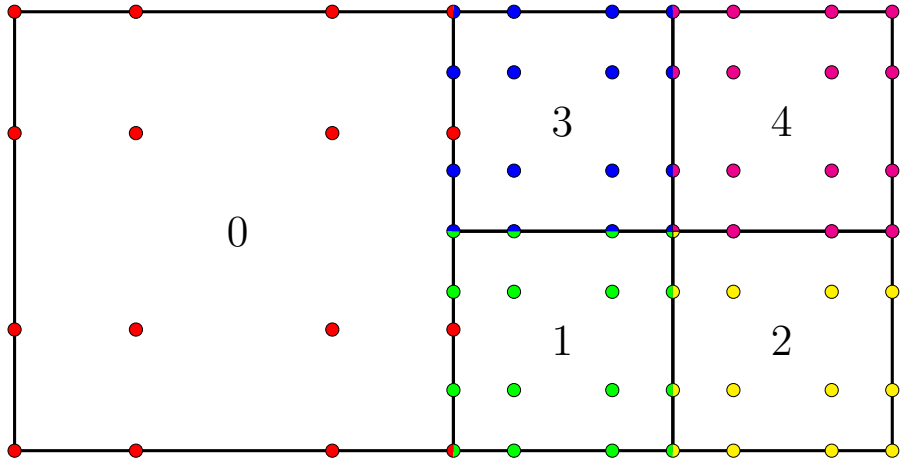
- Allows h-nonconforming hexahedral elements (2:1 balance)
- The element basis is the tensor product of Lagrange polynomials based on the Legendre-Gauss-Lobatto (LGL) nodes (fast implementation).

Discontinuous spectral element implementation



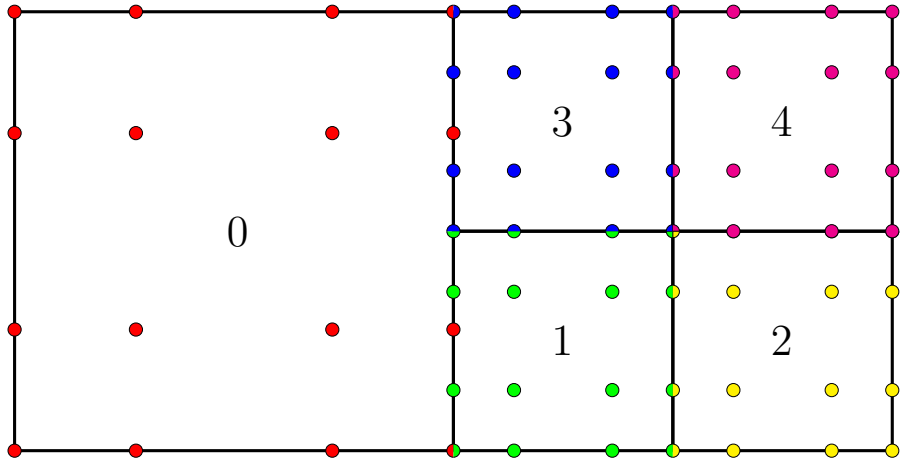
- Allows h-nonconforming hexahedral elements (2:1 balance)
- The element basis is the tensor product of Lagrange polynomials based on the Legendre-Gauss-Lobatto (LGL) nodes (fast implementation).
- LGL quadrature for integration implies diagonal mass matrix

Discontinuous spectral element implementation



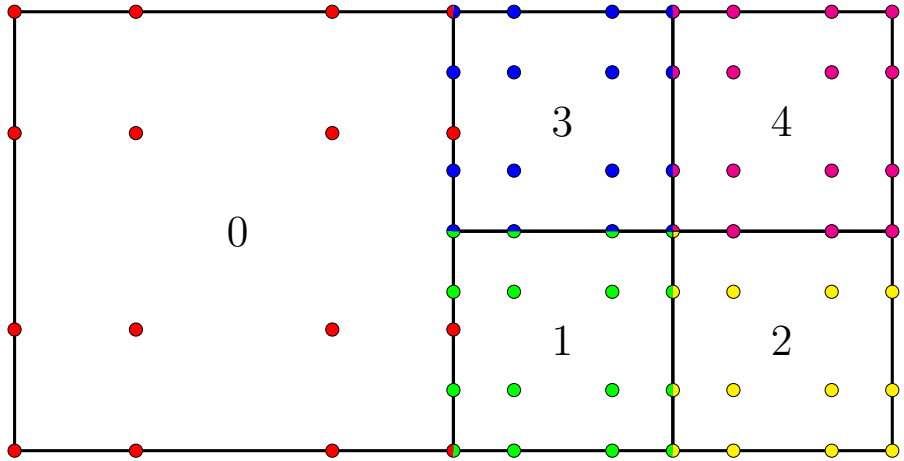
- Allows h-nonconforming hexahedral elements (2:1 balance)
- The element basis is the tensor product of Lagrange polynomials based on the Legendre-Gauss-Lobatto (LGL) nodes (fast implementation).
- LGL quadrature for integration implies diagonal mass matrix
- Allow material parameters to vary on element

Discontinuous spectral element implementation



- Allows h-nonconforming hexahedral elements (2:1 balance)
- The element basis is the tensor product of Lagrange polynomials based on the Legendre-Gauss-Lobatto (LGL) nodes (fast implementation).
- LGL quadrature for integration implies diagonal mass matrix
- Allow material parameters to vary on element
- Careful implementation of flux is required on hanging faces.

Discontinuous spectral element implementation



- Allows h-nonconforming hexahedral elements (2:1 balance)
- The element basis is the tensor product of Lagrange polynomials based on the Legendre-Gauss-Lobatto (LGL) nodes (fast implementation).
- LGL quadrature for integration implies diagonal mass matrix
- Allow material parameters to vary on element
- Careful implementation of flux is required on hanging faces.
- Time discretization through method-of-lines (we use RK4 or low-memory 5-stage 4th order)

Outline

Acoustic and elastic wave equations

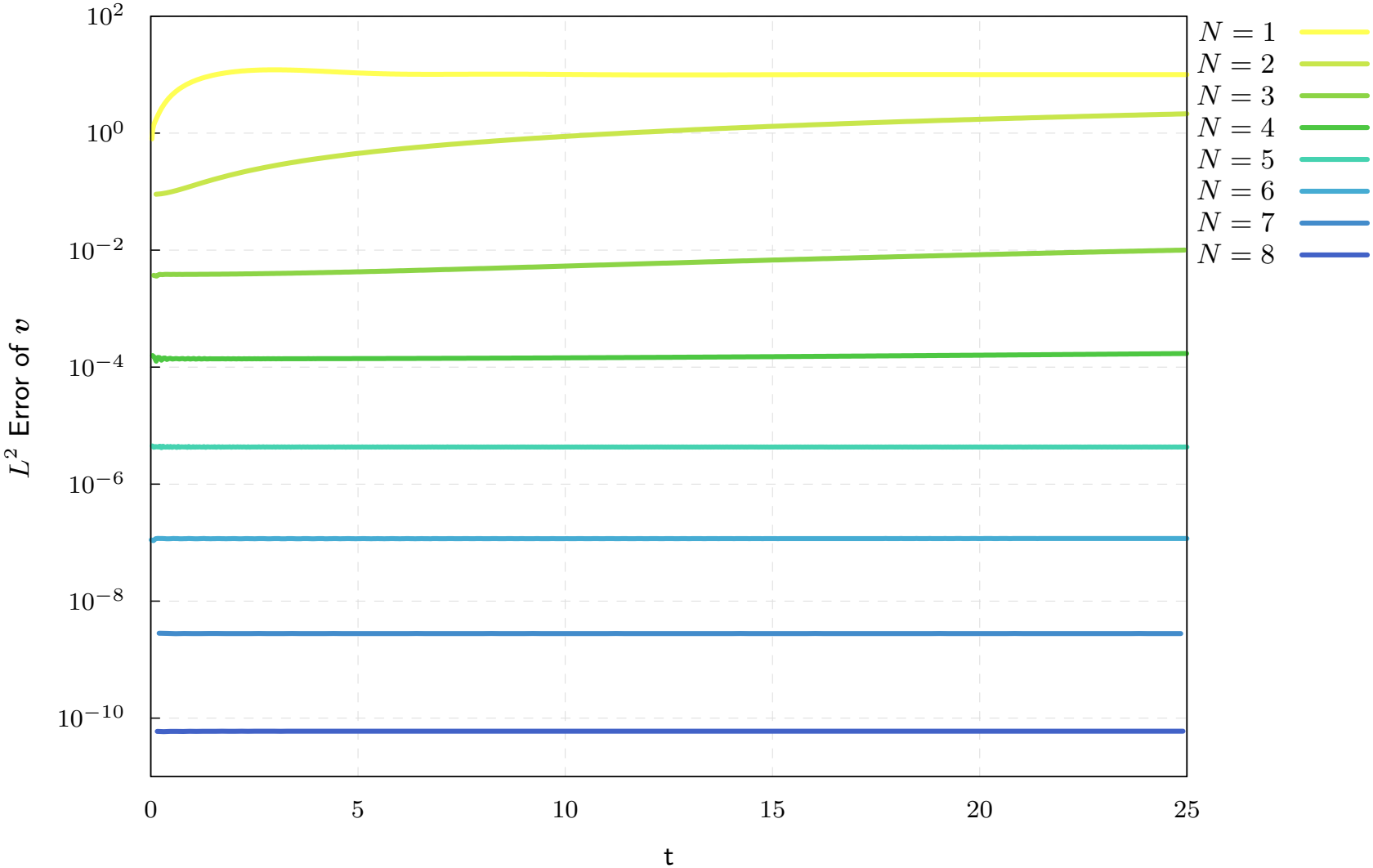
Discontinuous Galerkin Discretizations
- Formulation
- Discontinuous spectral element implementation

Numerical examples
- Code verification (long-time, interfaces)
- Code scalability (speedup for CPU and GPU implementation)

Inversion for local wave speeds

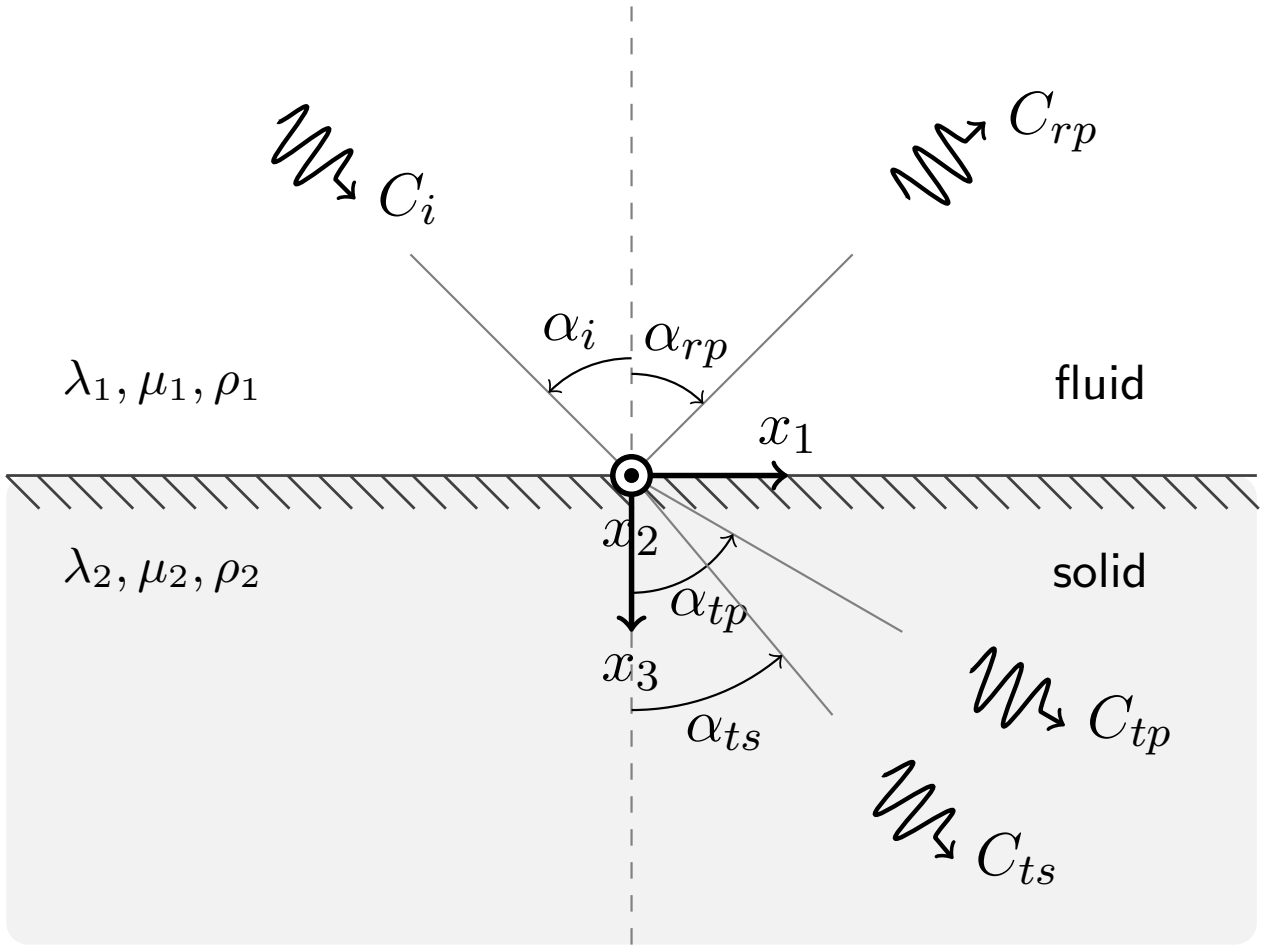
Elastic wave propagation

Plane-wave problem in periodic cube



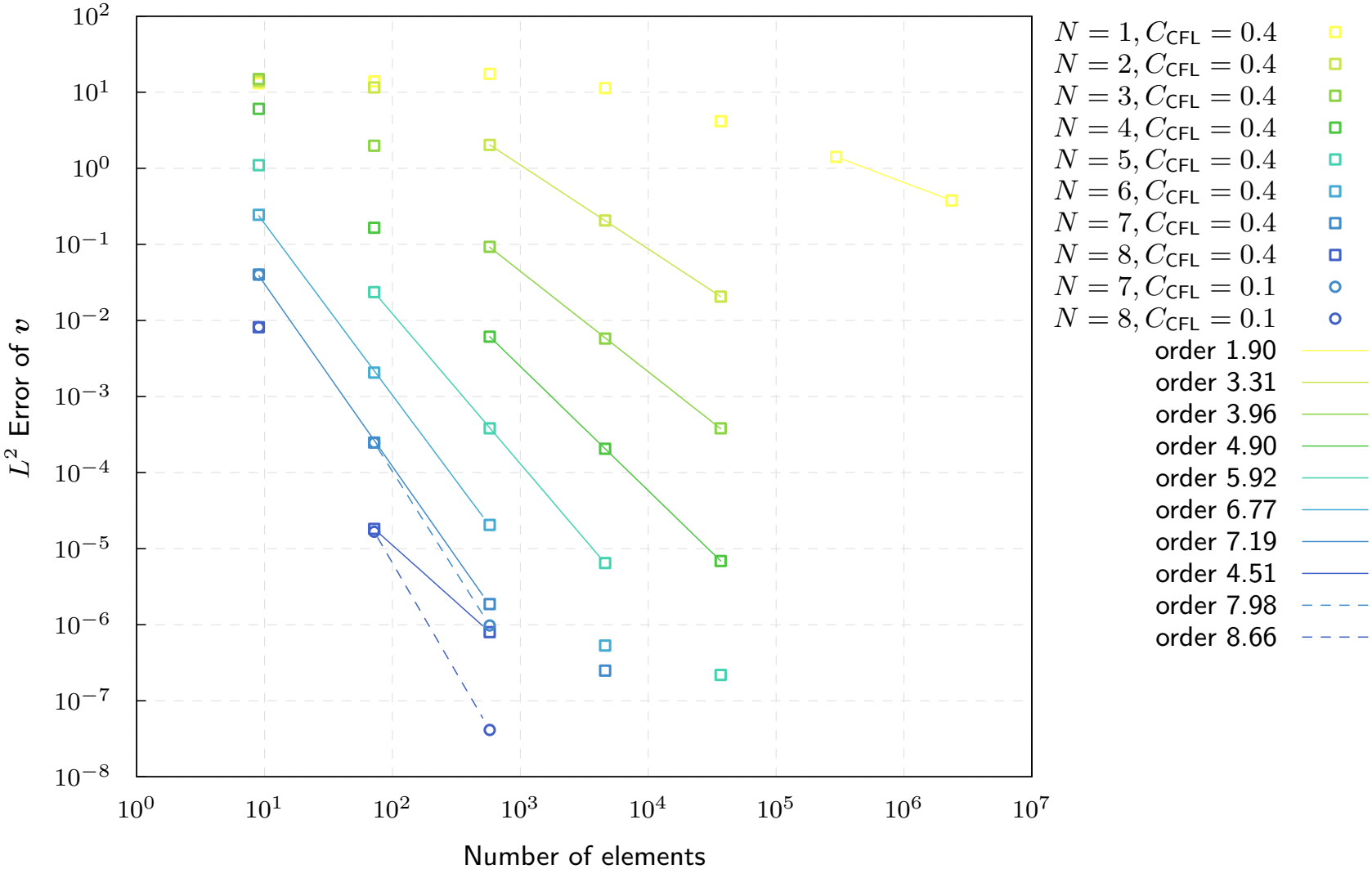
Elastic-acoustic wave propagation

Snell's Law verification (simulation is 3D)



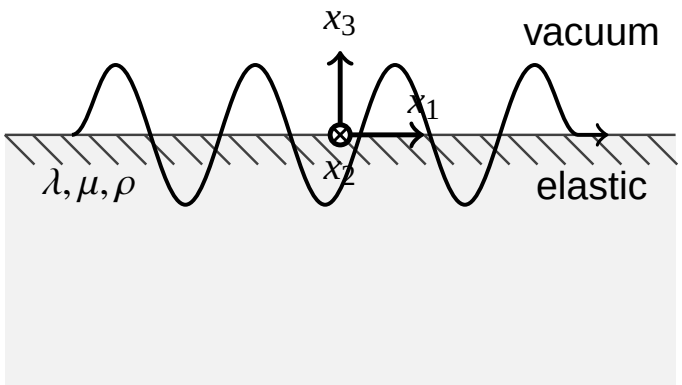
Elastic-acoustic wave propagation

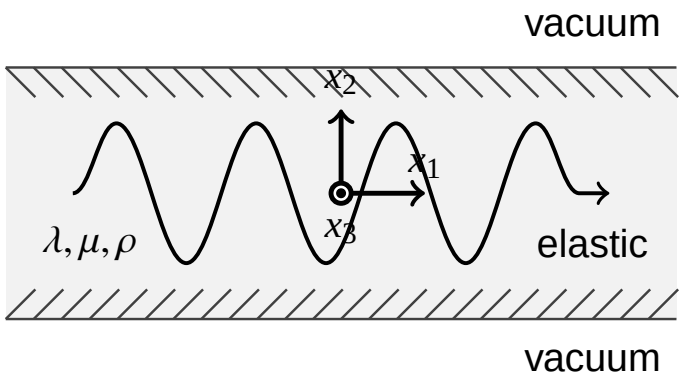
Snell's Law verification

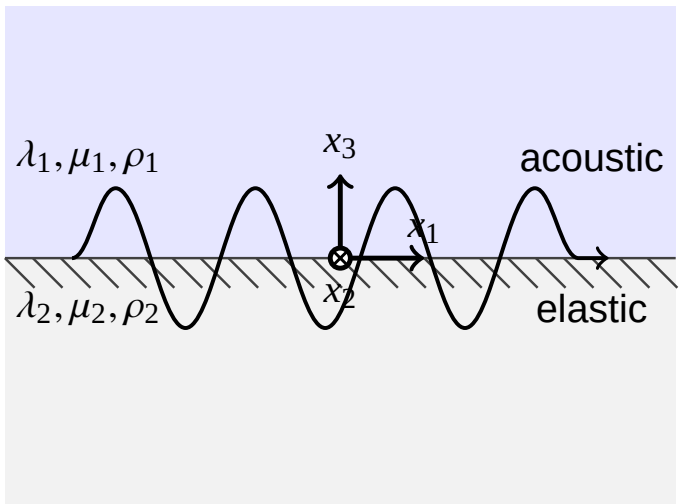


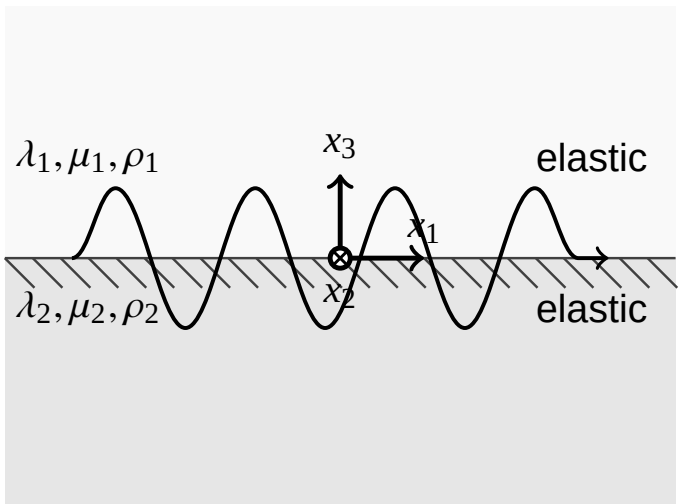
More interface problems

Rayleigh, Lamb, Scholte and Stoneley wave verification



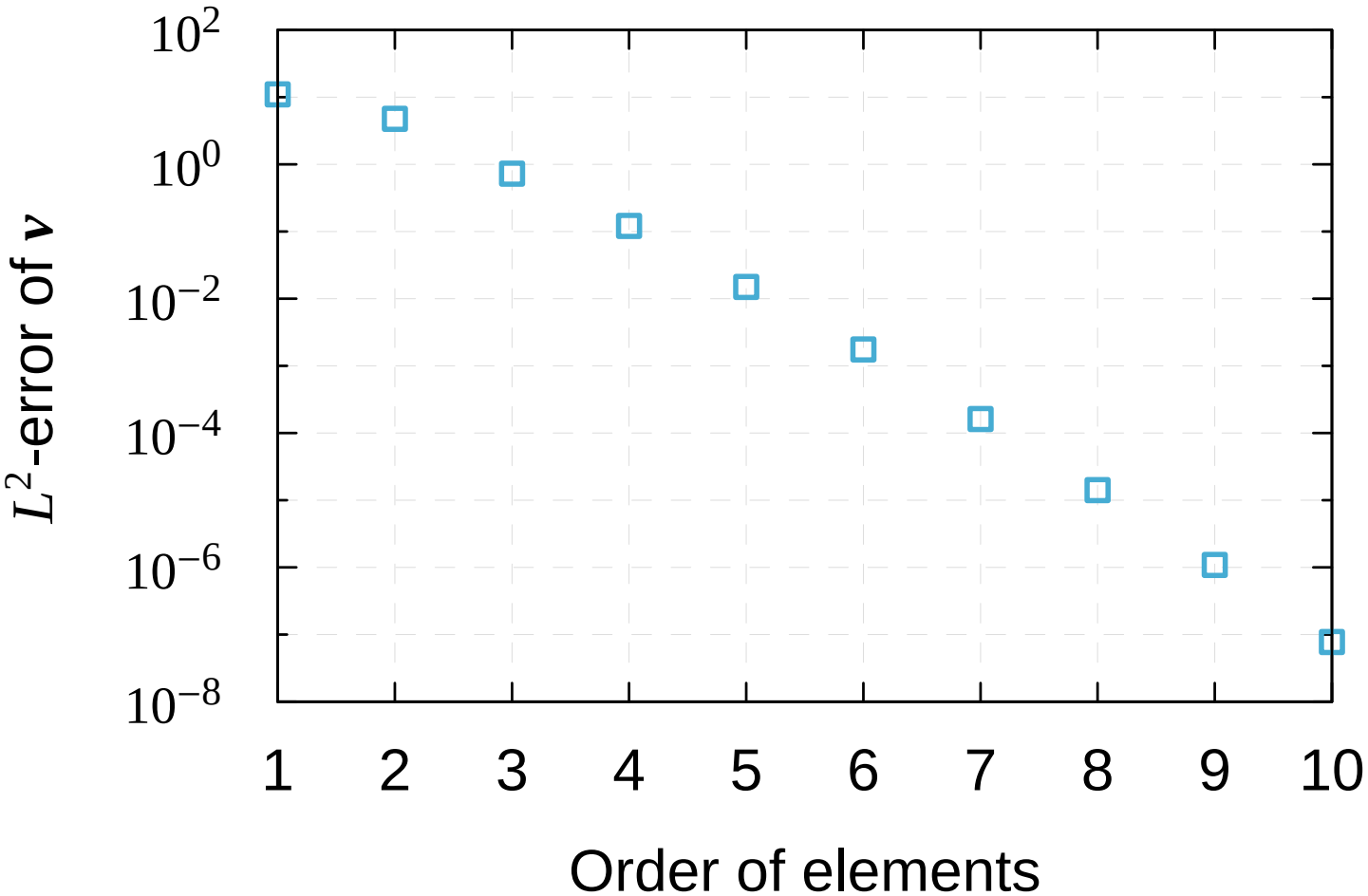






L^2-convergence

Scholte wave example



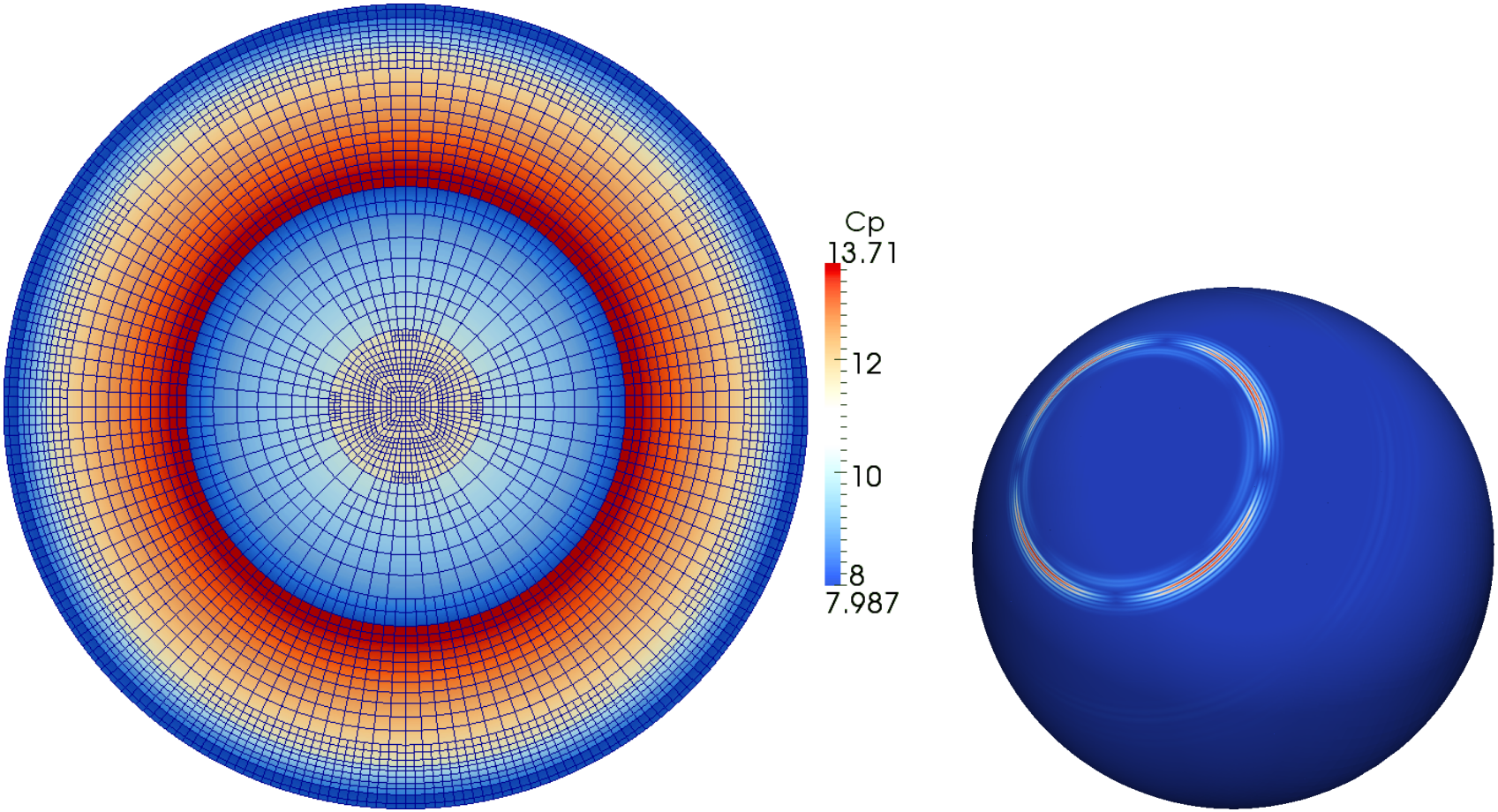
Scalability on Jaguar supercomputer at ORNL

Cray XT-5, 224,256 cores (AMD Istanbul 6 cores/proc, 2 proc/node), 2.3 petaflops

Figure: Image courtesy of the National Center for Computational Sciences, Oak Ridge National Laboratory

Scalability for PREM test problem

Ricker wavelet source with 0.28 Hz (plots are for lower frequency)



Element size tailored to local wave speeds; cubed sphere-based discretization, the mesh resolves interfaces in PREM.

Strong scalability on CPUs

ORNL Jaguar system (Cray XT5 with dual AMD Istanbul 6-core nodes)

# proc cores	meshing time (s)	wave prop per step (s)	par eff wave	Tflops
32,640	6.32	12.76	1.00	25.6
65,280	6.78	6.30	1.01	52.2
130,560	17.76	3.12	1.02	105.5
223,752	47.64	1.89	0.99	175.6

- 6th order elements with at least 10 points per wavelength
- 170 million elements, 53 billion unknowns
- *meshing time* = time for parallel generation of the mesh (adapted to local wave speed) prior to wave propagation solution
- *Wave prop per step* is the runtime (s) per time step of the wave propagation solve
- *Tflops* is double precision teraflops/s based on *PAPI*

Scalability (speedup) on GPUs (with Tim Warburton, Rice)

Speedup on TACC's GPU cluster (Longhorn)

#GPUs	#elem per GPU	Gflops per GPU	time per elem (s)	mesh time (s)
64	27793	79.3485	5.96801e-06	8.82812
128	13896	78.4707	6.05867e-06	4.71147
256	6948	75.4989	6.32710e-06	3.30853
478	3721	70.1133	6.84000e-06	1.85288

- Speedup for run with 1.78M elements of 7-th order
- largest run sustains about 35TF (single precission)
- speedup of 50–100$\times$ (compared to double precission CPU code)
- *Longhorn* = 512 NVIDIA FX 5800 GPUs each with 4GB graphics memory and 512 Intel Nehalem quad core processors connected by QDR InfiniBand interconnect

Outline

Acoustic and elastic wave equations

Discontinuous Galerkin Discretizations
- Formulation
- Discontinuous spectral element implementation

Numerical examples
- Code verification (long-time, interfaces)
- Code scalability (speedup for CPU and GPU implementation)

Inversion for local wave speeds

Inversion for local wave speeds (time domain)

- would like to estimate the Hessian corresponding to

$$\mathcal{J}(c_p, c_s) := \frac{1}{2} \int_0^T \int_B \mathcal{B}(\boldsymbol{x})(\boldsymbol{v} - \boldsymbol{v}_{data})^2 \, \boldsymbol{dx} + R(c_p, c_s)$$

  where the elastic/acoustic wave equation maps (c_p, c_s) into $\boldsymbol{v}$

- $\boldsymbol{v}_{data}$ are waveform observations at receivers, R is a regularization/prior operator, and $\mathcal{B}(\boldsymbol{x})$ is an observation operator that reflects receiver locations and weights
- Lagrangian functional approach
- gradient/Hessian computed via adjoints
- consistent gradient through discretize-then-optimize (DTO) approach
- can show equivalence between DTO and OTD

First-order optimality conditions

- State equations

$$\rho \frac{\partial \boldsymbol{v}}{\partial t} = \boldsymbol{\nabla} \cdot (\mathbf{C}\boldsymbol{E}) + \boldsymbol{f} \qquad \text{in } \Omega \times (0,T)$$

$$\mathbf{C} \frac{\partial \boldsymbol{E}}{\partial t} = \frac{1}{2}\mathbf{C}(\boldsymbol{\nabla v} + \boldsymbol{\nabla v}^T) \qquad \text{in } \Omega \times (0,T)$$

$$\rho \boldsymbol{v} = \mathbf{C}\boldsymbol{E} = \mathbf{0} \qquad \text{in } \Omega \times \{t = 0\}$$

- Adjoint equations ($\boldsymbol{w}$ = adjoint velocity; $\boldsymbol{H}$=adjoint strain)

$$-\rho \frac{\partial \boldsymbol{w}}{\partial t} = \boldsymbol{\nabla} \cdot (\mathbf{C}^* \boldsymbol{H}) - \mathcal{B}(\boldsymbol{v} - \boldsymbol{v}^{\text{obs}}) \qquad \text{in } \Omega \times (0,T)$$

$$\mathbf{C}^* \frac{\partial \boldsymbol{H}}{\partial t} = \frac{1}{2}\mathbf{C}^*(\boldsymbol{\nabla w} + \boldsymbol{\nabla w}^T) \qquad \text{in } \Omega \times (0,T)$$

$$\rho \boldsymbol{w} = \mathbf{C}^* \boldsymbol{H} = \mathbf{0} \qquad \text{in } \Omega \times \{t = T\}$$

- Gradient equations (for general tensor $\mathbf{C}$; no regularization)

$$\boldsymbol{g} = \int_0^T \left[\frac{1}{2}(\boldsymbol{\nabla w} + \boldsymbol{\nabla w}^T) \otimes \boldsymbol{E} + \boldsymbol{H} \otimes \frac{\partial \boldsymbol{E}}{\partial t} - \frac{1}{2}\boldsymbol{H} \otimes (\boldsymbol{\nabla v} + \boldsymbol{\nabla v}^T) \right]$$

Discrete DG adjoint and gradient/sensitivity kernels

- The adjoint DG equation is a wave equation in strain-velocity $(\boldsymbol{H}, \boldsymbol{w})$, in opposite time direction.
- The numerical flux becomes an downwind flux, i.e., an upwind flux in negative time direction
- Gradient/Sensitivity kernel involve DG-specific face terms, for instance the sensitivity w. r. c_p:

$$
\begin{aligned}
\overline{V_p^e} =& 2\rho c_p \left(\nabla \cdot \boldsymbol{w}\right) \operatorname{tr}(\boldsymbol{E}), \\
\overline{S_p^e} =& -k_0^2 \rho^- \boldsymbol{n} \cdot [\![\mathbf{C}\boldsymbol{E}]\!] \boldsymbol{n} \cdot [\![\mathbf{C}\boldsymbol{H}]\!] + k_0^2 \rho^- \left(\rho^+ {c_p}^+\right)^2 [\![\boldsymbol{v}]\!][\![\boldsymbol{w}]\!] \\
& + k_0^2 \rho^- \rho^+ {c_p}^+ \left(\boldsymbol{n} \cdot [\![\mathbf{C}\boldsymbol{E}]\!][\![\boldsymbol{w}]\!] - [\![\boldsymbol{v}]\!] \boldsymbol{n} \cdot [\![\mathbf{C}\boldsymbol{H}]\!]\right) \\
& + 2k_0 \rho^- {c_p}^- \operatorname{tr}(\boldsymbol{E}^-) \left(\boldsymbol{n} \cdot [\![\mathbf{C}\boldsymbol{H}]\!] - \rho^+ {c_p}^+ [\![\boldsymbol{w}]\!]\right).
\end{aligned}
$$

Face terms are needed for exact DG-gradient/sensitivity!

Hessian–vector products

- data misfit Hessian–vector products resemble gradient computation; are needed to solve Newton step by CG and to estimate the inverse Hessian for covariance estimate

Hessian–vector products

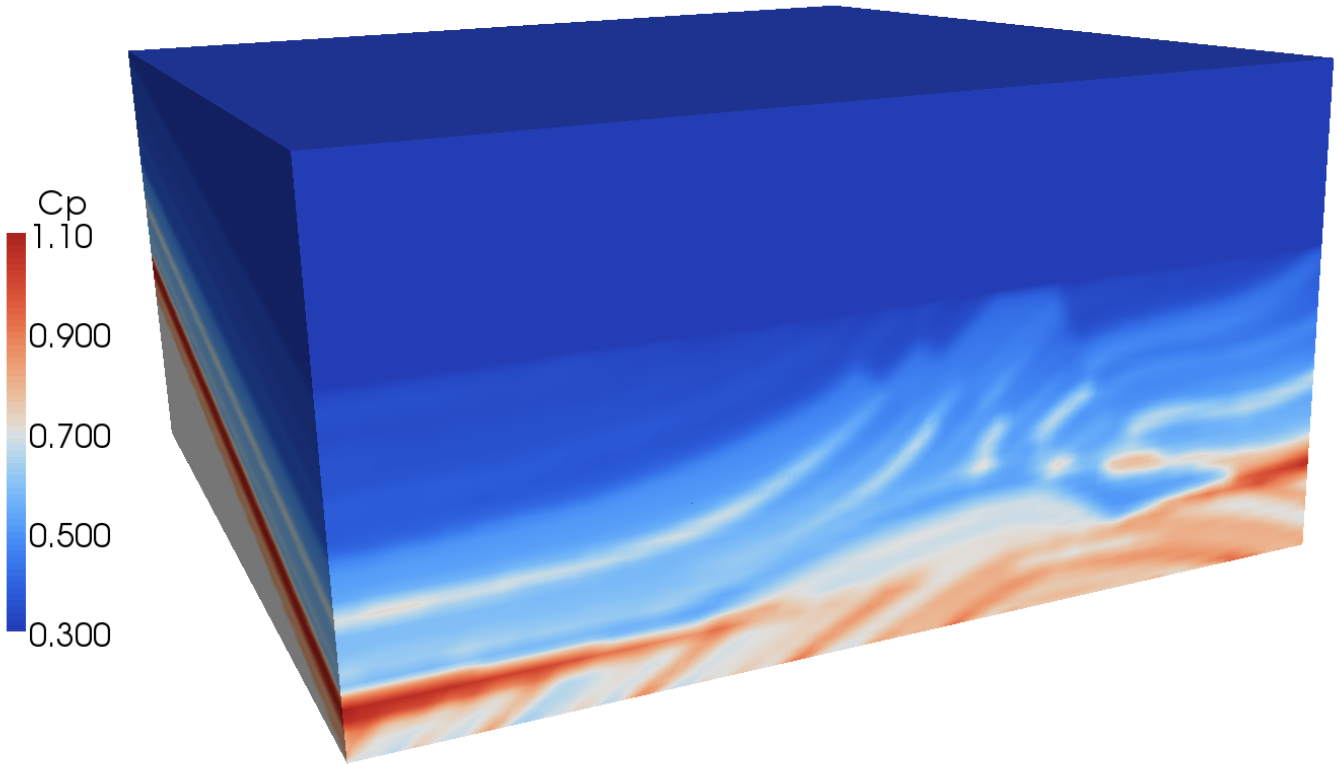
- data misfit Hessian–vector products resemble gradient computation; are needed to solve Newton step by CG and to estimate the inverse Hessian for covariance estimate
- steps to form product of Gauss-Newton Hessian at the "point" (c_p, c_s) in the "direction" $(\hat{c}_p, \hat{c}_s)$:
  1. solve the forward wave equation with given material fields (c_p, c_s) to obtain the forward strain and velocity $(\boldsymbol{E}, \boldsymbol{v})$
  2. linearize the forward wave equation at the point (c_p, c_s) (and corresponding forward strain and velocity) in the direction $(\hat{c}_p, \hat{c}_s)$
  3. solve the linearized forward wave equation to obtain the incremental forward strain and velocity $(\hat{\boldsymbol{E}}, \hat{\boldsymbol{v}})$
  4. solve the adjoint wave equation backwards in time, driven by the incremental forward velocity $\hat{\boldsymbol{v}}$ at observation points, yielding the incremental adjoint strain and velocity $(\hat{\boldsymbol{H}}, \hat{\boldsymbol{w}})$
  5. use the gradient expression to compute the GN Hessian–vector product, but use $(\boldsymbol{E}, \boldsymbol{v})$ with $(\hat{\boldsymbol{H}}, \hat{\boldsymbol{w}})$

Application to inverse wave propagation in Marmousi

Use that inverse Hessian is covariance matrix

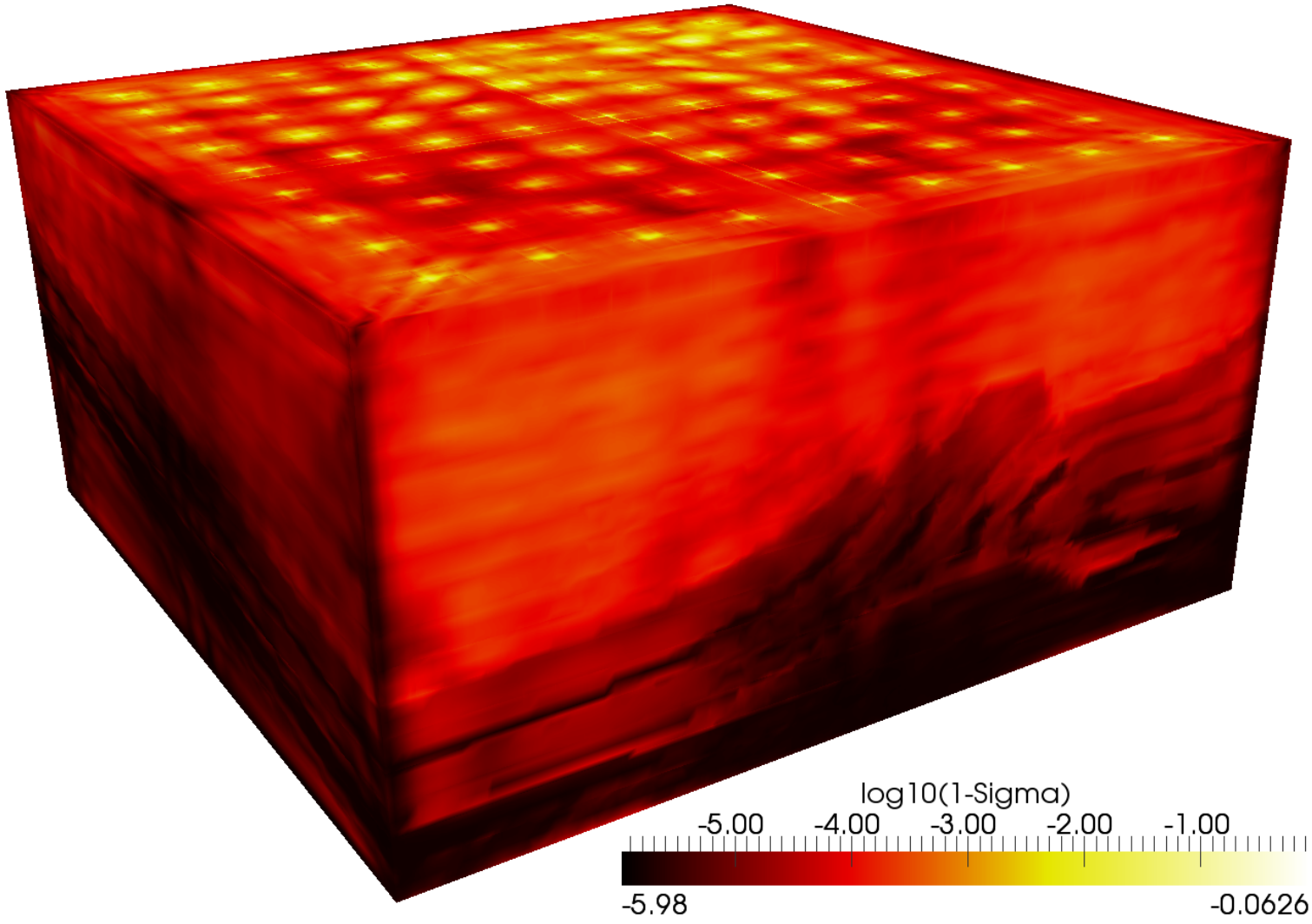
- heterogeneous Marmousi model
- iid data and prior covariances
- wave speeds between 1.5 and 5 km/s
- domain is $10 \times 10 \times 5$ km^3
- Ricker wavelet, central frequency of 8Hz
  $\Rightarrow$ 16–53 wavelengths vertically; 32–106 horizontally
- largest mesh has $64 \times 64 \times 32$ 4th-order elements (16 million DOFs)
- parameter field represented on same mesh as forward/adjoint fields
  $\Rightarrow$ 16 million parameters
- inverse Hessian estimate in 500 Lanczos iterations, 1500 forward/adjoint solves
- low-rank-based variance field approximation computed in 6h on 1600 cores (full Hessian would require $O(10^4)$ hours on 1600 cores)

Target medium, p-wave velocity



- test setting: 5 sources on bottom; 81 receivers on top
- stress-free boundary conditions (should be PML)

Variance field



Summary

- high-order DG for wave propagation through coupled elastic-acoustic media
- near-optimal weak and strong scaling; CPU/GPU implementation
- discretely consistent adjoints and gradients for seismic inversion
- estimate variance using low-rank approximation of Hessian

Partially supported by NSF's PetaApps program (OCI-0749334, OCI-0749045, and OCI-0748898), DOE Office of Science's SciDAC program (DE-FC02-06ER25782), NSF grants CNS-0619838 and DMS-0724746, and AFOSR grant FA9550-09-1-0608.