Copper Mountain March 26-30, 2017

Geometric Multigrid for Scalable DPG Solves in Camellia

Nathan V. Roberts and Jesse Chan
nvrober@sandia.gov, Jesse.Chan@caam.rice.edu
Sandia National Laboratories and Rice University
Outline

1. Motivation/Introduction: DPG, Camellia, HPC
   - Camellia: Design Goals
   - DPG and HPC

2. Our Geometric Multigrid Approach

3. Selected Numerical Results

4. Conclusions
DPG $\neq$ DG
DPG in Brief

DPG approach:

- *Petrov*-Galerkin: test and trial spaces differ
- discontinuous test and trial spaces
- optimal test functions computed on the fly so that
  \[ (v_{e_i}^{\text{opt}}, v)_{V} = b(e_i, v) \quad \forall v \in V \]

- **key choice:** which norm to use on the test space?

DPG features:

- automatic stability even on coarse mesh
- SPD/HPD stiffness matrix \( \Rightarrow \) can use (P)CG
- discontinuous test space \( \Rightarrow \) optimal test solve is local
- Error in \( u_h \) is minimized in the energy norm
  \[ \| u_h \|_E = \sup_{v \in V} \frac{b(u_h, v)}{\|v\|_V} = \| b(u_h, \cdot ) \|_V, \]

- Can measure the error in the energy norm to drive adaptivity.
inf-sup stability

optimal test functions

canonical inf-sup pairing

Can we make
\[ \| \cdot \|_U \approx \| \cdot \|_E \]?

discontinuous test space

computational tractability

min. residual

graph norm on test space

ultraweak formulation

* Note: we approximate the infinite-dimensional test space by taking the polynomial order \( k \) for the trial and “enriching” it somewhat: \( k_{\text{test}} = k_{\text{trial}} + \Delta k \)—in all that follows, \( \Delta k = 1, 2, \) or \( 3 \).
Building the ultraweak formulation

\[ \Delta \phi = f \]

**First-Order System**

\[ \nabla \cdot \psi = f \]
\[ \psi - \nabla \phi = 0 \]

**Integration by Parts**

\[ (\psi \cdot n, v)_{\Gamma_h} - (\psi, \nabla v)_{\Omega_h} = (f, v)_{\Omega_h} \]
\[ (\psi, q)_{\Omega_h} + (\phi, q \cdot n)_{\Gamma_h} - (\phi, \nabla \cdot q)_{\Omega_h} = 0 \]

**Ultraweak (DPG) Variational Formulation**

\[ (\tilde{\psi}_n, v)_{\Gamma_h} - (\psi, \nabla v)_{\Omega_h} \]
\[ + (\psi, q)_{\Omega_h} + (\tilde{\phi}, q_n)_{\Gamma_h} - (\phi, \nabla \cdot q)_{\Omega_h} = (f, v)_{\Omega_h} \]

\[ b((\phi, \psi, \tilde{\phi}, \tilde{\psi}_n), (v, q)) = (f, v)_{\Omega_h} \]
\[ b(u, v) = l(v) \]
DPG Applications to Date

DPG is a general framework, and has been successfully applied to a host of PDE problems, including:

- convection-dominated diffusion
- acoustics/wave propagation
- linear elasticity
- Maxwell’s equations (cloaking problem)
- Burgers’ equations
- Euler equations
- compressible Navier-Stokes
- Stokes
- incompressible Navier-Stokes
- Oldroyd-B Flow

\[ \text{flow past a cylinder, } Re = 40 \]

\(^1\)Bold items have Camellia-based implementations.
**Camellia**¹

**Design Goal:** make DPG research and experimentation as simple as possible, while maintaining computational efficiency and scalability.

Core features:

- rapid specification of new formulations (FEniCS-inspired)
- arbitrary element types (simplices and hypercubes provided)
- $h$- and $p$-adaptivity (with hanging nodes)
- trace and field unknowns (discontinuous and $C^0$)
- scalability via MPI (take advantage of parallelism in optimal test function determination)
- implemented in C++, built atop Trilinos

---

Suitability of DPG for HPC

DPG has several attractive features for HPC:

- **locality**: optimal test functions embarrassingly parallel
- **intensity**: high-order computations take advantage of “free” flops
- **automaticity**: robust adaptivity means less human involvement
Multigrid choices:

- V-cycle
- *multiplicative* smoothing (accelerates convergence at cost of extra residual computation).
- smoother is damped; see our arXiv report for details.
- Prolongation and smoothing details follow...
The basic rule for the prolongation operator $P$ is

- A solution that is exact on the coarse mesh should also be a solution on the fine mesh when prolonged.

For $p$-multigrid, this is straightforward. What about our traces with $h$-multigrid?
Multigrid: Our Prolongation Operators

The basic rule for the prolongation operator $P$ is

- A solution that is exact on the coarse mesh should also be a solution on the fine mesh when prolongated.

For $p$-multigrid, this is straightforward. What about our traces with $h$-multigrid?

In particular, for the DPG traces under $h$-refinement:

- some traces do not exist in the coarse mesh;
- we define these in terms of the fields ($\hat{u} = \text{tr}(u)$);
- this is further complicated by static condensation: we must reconstruct the fields.
Multigrid: Our Smoothers

For $p$-multigrid smoothers, we use “minimal overlap” additive Schwarz:

![Diagram of a grid with a highlighted square]

Copper Mountain March 26-30, 2017 13
Multigrid: Our Smoothers

For $p$-multigrid smoothers, we use “minimal overlap” additive Schwarz:

For $h$-multigrid smoothers, we use 1-overlap additive Schwarz:
Two-Grid Tests for Stokes

Our coarse grids:

- $p$-multigrid: $k_{\text{coarse}} = k_{\text{fine}}/2$; when $k_{\text{fine}} = 1$, $k_{\text{coarse}} = 0$.
- $h$-multigrid: coarse mesh of same degree as fine, once-coarsened relative to fine mesh.

Our exact solution:

- $u = (e^{x \cdot y} \cos y + \sin y, e^{x \cdot y} \sin y + e^{z \cdot y} \cos y, -e^{z \cdot y} (\cos y - y \sin y))$
- $p = 2e^{x \cdot \sin y} + 2e^{z \cdot \cos y}$
Two-Grid Tests for Stokes

Our coarse grids:
- p-multigrid: \( k_{\text{coarse}} = k_{\text{fine}} / 2 \); when \( k_{\text{fine}} = 1 \), \( k_{\text{coarse}} = 0 \).
- h-multigrid: coarse mesh of same degree as fine, once-coarsened relative to fine mesh.

DPG choices:
- we use static condensation and the graph norm;
- we use \( \Delta k = d = 2 \) or 3 (but little difference for \( \Delta k = 1 \));
- for \( H^1 \) traces, we enrich the corresponding fields (i.e., if Stokes field variables have order 3, then both the velocity traces and the velocity fields will have order 4).
Two-Grid Tests for Stokes

Our coarse grids:

- $p$-multigrid: $k_{coarse} = k_{fine}/2$; when $k_{fine} = 1$, $k_{coarse} = 0$.
- $h$-multigrid: coarse mesh of same degree as fine, once-coarsened relative to fine mesh.

DPG choices:

- we use static condensation and the graph norm;
- we use $\Delta k = d = 2$ or 3 (but little difference for $\Delta k = 1$);
- for $H^1$ traces, we enrich the corresponding fields (i.e., if Stokes field variables have order 3, then both the velocity traces and the velocity fields will have order 4).

Our exact solution:

- $u =$
  $$(-e^x y \cos y + \sin y, e^x y \sin y + e^z y \cos y, -e^z (\cos y - y \sin y))$$
- $p = 2 e^x \sin y + 2 e^z \cos y$
Two-Grid Results: Stokes 2D $p$-Multigrid

- Stokes $p = 1$ (conforming)
- Stokes $p = 2$ (conforming)
- Stokes $p = 4$ (conforming)
Two-Grid Results: Stokes 2D $h$-Multigrid

Stokes 2D $h$-Multigrid

- Stokes $p = 1$ (conforming)
- Stokes $p = 2$ (conforming)
- Stokes $p = 4$ (conforming)
Two-Grid Results: Stokes 3D $p$-Multigrid

3D $p$-Multigrid

- Stokes $p = 1$ (conforming)
- Stokes $p = 2$ (conforming)
Two-Grid Results: Stokes 3D $h$-Multigrid

![Graph showing iteration count vs mesh width for Stokes 3D $h$-Multigrid with two lines representing $p = 1$ (conforming) and $p = 2$ (conforming).]
Approach for More than Two Grids
Lid-Driven Cavity Flow

A classical challenge problem for Stokes flow is lid-driven cavity flow.

Left: schematic of the flow. Right: streamlines. We will start with a 2 × 2, k = 4 mesh, and perform automatic refinements, using our multigrid preconditioner at each refinement step.
Top left to bottom right: sequence of meshes for multigrid operator for refinement 6. From coarsest mesh, refine first in $h$, then jump to fine $k$. 
### Lid-Driven Cavity Flow: Results

<table>
<thead>
<tr>
<th>Ref. #</th>
<th>$\frac{h_{\text{min}}}{h_{\text{min}}}$</th>
<th>Elements</th>
<th>Energy Error</th>
<th>Zero Guess</th>
<th>Prev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1/2</td>
<td>4</td>
<td>7.27e-01</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>1</td>
<td>1/4</td>
<td>10</td>
<td>6.30e-01</td>
<td>22</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>1/8</td>
<td>16</td>
<td>5.82e-01</td>
<td>28</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>1/16</td>
<td>22</td>
<td>2.06e-01</td>
<td>41</td>
<td>20</td>
</tr>
<tr>
<td>4</td>
<td>1/32</td>
<td>28</td>
<td>7.54e-02</td>
<td>46</td>
<td>21</td>
</tr>
<tr>
<td>5</td>
<td>1/64</td>
<td>34</td>
<td>5.90e-02</td>
<td>33</td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td>1/128</td>
<td>70</td>
<td>3.01e-02</td>
<td>55</td>
<td>21</td>
</tr>
<tr>
<td>7</td>
<td>1/256</td>
<td>88</td>
<td>1.55e-02</td>
<td>63</td>
<td>19</td>
</tr>
<tr>
<td>8</td>
<td>1/512</td>
<td>106</td>
<td>8.63e-03</td>
<td>70</td>
<td>15</td>
</tr>
</tbody>
</table>

**Table:** Stokes cavity flow: iteration counts with $k = 4$ to achieve a residual tolerance of $10^{-6}$, starting from a zero initial guess or with the solution from the previous refinement step.
A Scaling Test

For a challenging scaling test, take a $32 \times 32 \times 32$ (32,768-element) quartic, conforming mesh with the same Stokes problem as before. This has $7.6 \times 10^7$ dofs ($1.4 \times 10^7$ trace dofs). Notes:

- Use static condensation
- Coarse mesh has 512 constant elements
- 113 iterations to converge
- Note: mesh initialization involves communication costs that do not scale (takes additional 35 seconds on 32K ranks compared to 4K)
- Use 8 MPI ranks per BG/Q node (2 GB/rank)
A Scaling Test

For a challenging scaling test, take a $32 \times 32 \times 32$ (32,768-element) quartic, conforming mesh with the same Stokes problem as before. This has $7.6 \times 10^7$ dofs ($1.4 \times 10^7$ trace dofs).

Total time to solution achieves 64% of the ideal 8x speedup (83% if mesh initialization costs neglected).
Some timing details

Timing detail for the $32 \times 32 \times 32$ (32,768-element) quartic, conforming Stokes solve (times in seconds).

“GMG Init.” is the time to construct the GMG data structures at each level; includes setting up Solution objects at each level but not constructing prolongation or smoothing operators (these are included in “Solve.”)
Some timing details: Solve

Timing detail for the $32 \times 32 \times 32$ (32,768-element) quartic, conforming Stokes solve (times in seconds).
Conclusions

Summary results:

- Our strategy works well, both in iteration counts and in compute time, to scale the Stokes problems we have considered.
- For Navier-Stokes, this works best for smaller Reynolds numbers (see report for details). Likely need something more specialized for high Reynolds numbers.

Resources:

- Camellia is available under a BSD License at bitbucket.org/nateroberts/Camellia
- Manual available as Argonne Tech Report
Thank you for your attention!

Questions?

For more details:
NVR.

NVR.
Camellia v1.0 manual: Part I.

NVR and Chan, J.