A Research Framework for Simplifying the Construction of GFD Models

Frank Giraldo
Department of Applied Mathematics
Naval Postgraduate School, Monterey CA USA
http://faculty.nps.edu/fxgirald

IPAM April 2010

*Funded by ONR-Battlespace Environments, ONR-Computational Mathematics and AFOSR-Computational Mathematics

Motivation for this Work

We are interested in constructing numerical methods for constructing non-hydrostatic mesoscale and global atmospheric models (for NWP applications) as well as constructing coastal ocean models (for tsunami and storm-surge modeling).

To simplify this process, we are developing a framework for constructing research codes. For example:

1. Once a code is running properly it is easier to attach new modules to do new problems.
2. This idea also allows us to reuse all of our data structures for CG/DG (serial and MPI).
3. This idea allows for different sets of interpolation/integration points to be used within the same model (e.g., nodal DG using either Lobatto or Legendre points).
4. The inclusion of new time-integrators is vastly simplified.
5. However, the data structures need to be generalized in order to handle non-conforming grid adaptivity.

Talk Summary

I. Equation Sets
- Hydrostatic Equations
- Nonhydrostatic Equations
- Pseudo-Incompressible Equations

II. Spatial Discretization
- Continuous Galerkin Methods
- Discontinuous Galerkin Methods

III. Suite of Time-Integrators
- Explicit SSP Methods
- Semi-implicit (IMEX) Methods
- Fully-Implicit Methods

IV. Adaptivity
- Conforming Methods
- Non-conforming Methods

I. Equation Sets

Non-hydrostatic Equations
(fully compressible Euler equations)

$$\frac{\partial \rho}{\partial t} + \nabla \bullet \mathbf{U} = 0 \qquad \text{(Mass)}$$

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla \bullet \left(\frac{\mathbf{U} \otimes \mathbf{U}}{\rho} + P\mathbf{I}_3 \right) = -f(\mathbf{k} \times \mathbf{U}) - \rho g \mathbf{k} + S_{\mathbf{U}} \qquad \text{(Momentum)}$$

$$\frac{\partial \Theta}{\partial t} + \nabla \bullet \left(\frac{\Theta \mathbf{U}}{\rho} \right) = S_{\Theta} \qquad \text{(Energy)}$$

$$P = P_A \left(\frac{R\Theta}{P_A} \right)^{\gamma} \qquad \text{(Equation of State)}$$

$$\mathbf{U} = \rho \mathbf{u},$$

$$\Theta = \rho \theta,$$

$$\mathbf{u} = (u, v, w)^T,$$

$$\mathbf{x} = (x, y, z)^T,$$

$$\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y} \frac{\partial}{\partial z} \right)^T$$

$$\theta = \frac{T}{\pi} \quad and \quad \pi = \left(\frac{P}{P_A} \right)^{R/c_p}$$

Unified Equations Framework

(see Dale's, Nigel's, and Rupert's talks)

$$\frac{\partial \rho}{\partial t} + \nabla \bullet \mathbf{U} = 0 \qquad \text{(Mass)}$$

$$\frac{\partial \mathbf{U}_{Horiz}}{\partial t} + \nabla \bullet \left(\frac{\mathbf{U}\,\mathbf{U}_{Horiz}}{\rho} + P\mathbf{I}_2 \right) = -f\left(\mathbf{k}\times\mathbf{U}_{Horiz}\right) + S_{\mathbf{U}}$$

$$\delta_H \left[\frac{\partial W}{\partial t} + \nabla \bullet \left(\frac{\mathbf{U}W}{\rho} \right) \right] + \frac{\partial P}{\partial z} = -\rho g + S_W \qquad \text{(Momentum)}$$

$$\mathbf{U} = \rho \mathbf{u},$$

$$\Theta = \rho\theta,$$

$$\mathbf{u} = (u, v, w)^T,$$

$$\mathbf{x} = (x, y, z)^T,$$

$$\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y} \frac{\partial}{\partial z} \right)^T$$

$$\delta_{PI} \frac{\partial \Theta}{\partial t} + \nabla \bullet \left(\frac{\Theta \mathbf{U}}{\rho} \right) = S_E \qquad \text{(Energy)}$$

$$P = P_A \left(\frac{R\Theta}{P_A} \right)^{\gamma} \qquad \text{(Equation of State)}$$

$$\theta = \frac{T}{\pi} \quad \textit{and} \quad \pi = \left(\frac{P}{P_A} \right)^{R/c_p}$$

II. Spatial Discretization

Galerkin Methods

- Primitive Equations: $\frac{\partial q}{\partial t} + \nabla \cdot \mathbf{F} = S(q)$
- Approximate the solution as: $q_N = \sum_{i=1}^{M_N} \psi_i q_i \qquad \mathbf{F}_N = \mathbf{F}(q_N) \qquad S_N = S(q_N)$
  - **Interpolation O(N)**
- Write Primitive Equations as: $R(q_N) \equiv \frac{\partial q_N}{\partial t} + \nabla \cdot \mathbf{F}_N - S_N = \varepsilon$
- Weak Problem Statement: Find $q_N \in \Sigma(\Omega) \; \forall \; \psi \in \Sigma$

  $\Sigma = \left\{ \psi \in H^1(\Omega) : \psi \in P_N(\Omega_e) \forall \Omega_e \right\}$ (CG)

  $\Sigma = \left\{ \psi \in L^2(\Omega) : \psi \in P_N(\Omega_e) \forall \Omega_e \right\}$ (DG)

  - such that
    - **Integration O(2N)**

    $\int_{\Omega / \Omega_e} \psi R(q_N) d\Omega = 0$

Comparison of CG/DG Methods

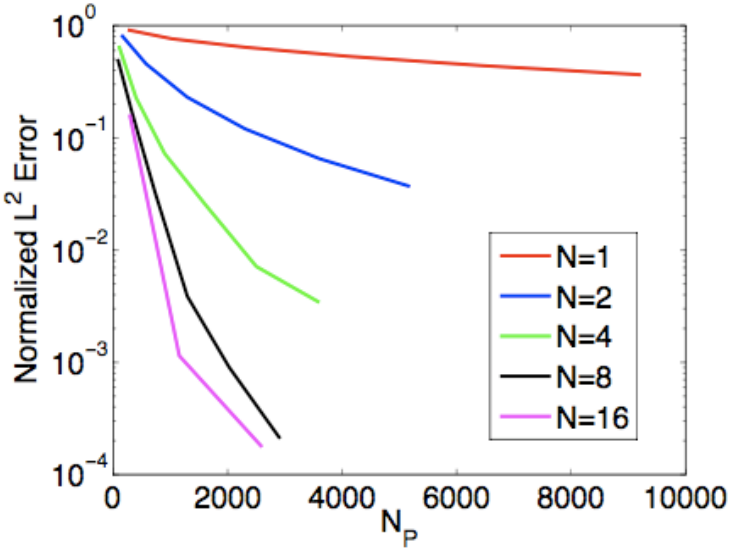
Continuous Galerkin Methods

- High order accurate yet local construction (via DSS)
- Simple to construct efficient semi-implicit time-integrators
- In high-order mode, primarily used with quads and inexact integration (e.g., using Lobatto points avoids non-diagonal mass matrix with slight error since integration is O(2N-1))
- No analog of Lobatto points exist on the triangle so costly to use
- Excellent scalability on MPP

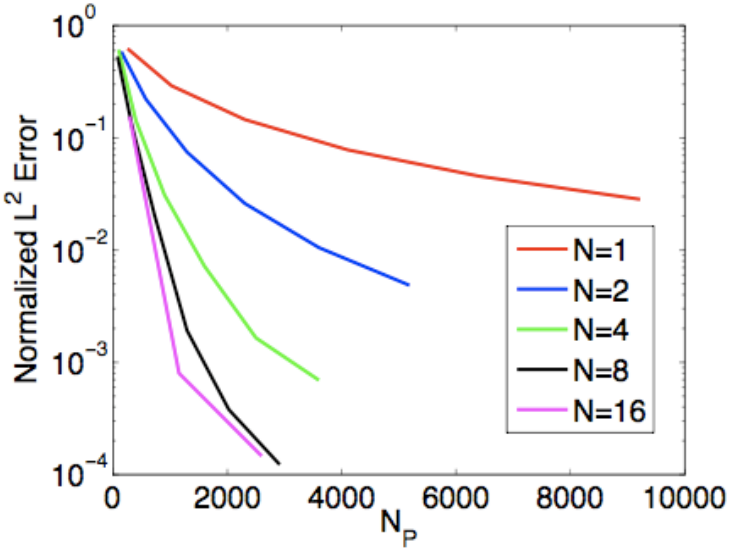
Discontinuous Galerkin Methods

- High order accurate and completely local in nature (no DSS required as in CG)
- High order generalization of the FV (but with compact support)
- Upwinding and BCs implemented naturally (via Riemann solvers)
- Not so easy to construct efficient semi-implicit time-integrators, due to the difficulty in extracting the Schur complement
- Since matrices are all local, using quads or triangles is straightforward and one need not worry as much about exact vs. inexact integration
- On quads, using Legendre points is analogous to using exact integration (for O(2N+1) polynomials) while resulting in diagonal mass matrices
- Excellent scalability on MPP

DG Interpolation/Integration Points
(2D Advection with Q=N integration points)



Lobatto



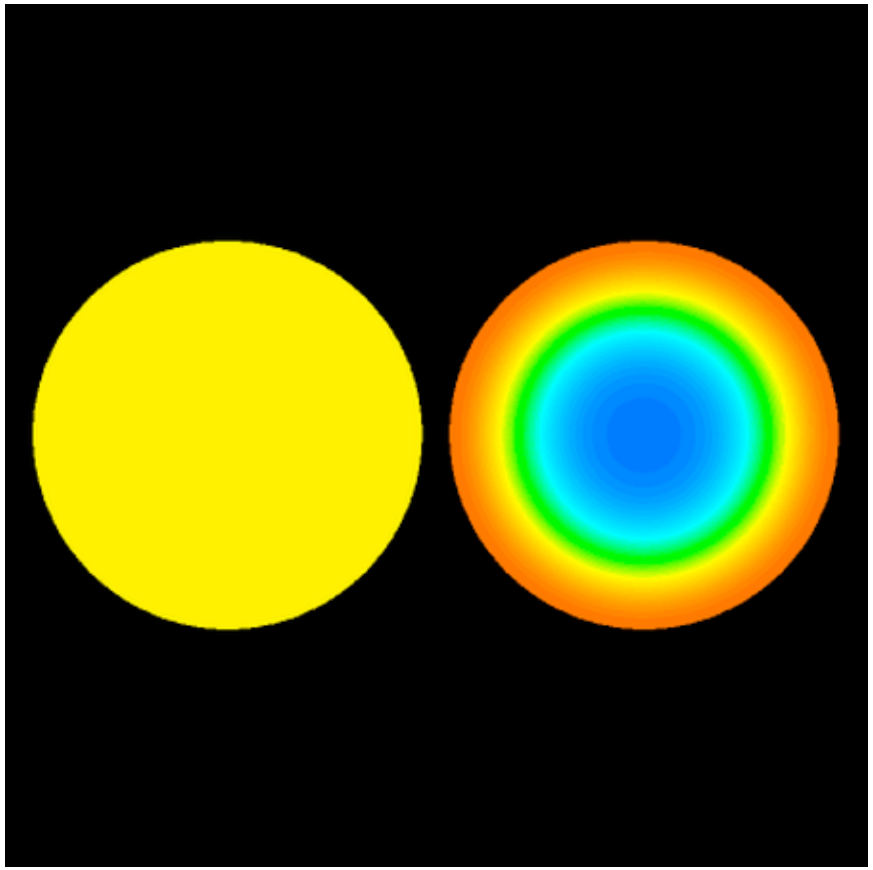
Legendre

Resolution of CG/DG methods is controlled by K (no. elements) and N (poly order) but dofs is controlled as $(KN+1)^d$ for CG and $(K(N+1))^d$ for DG

Scalability of CG Method

(Surface Values for T185 L26 during 0-30 days: Global Hydrostatic Model)

Pressure

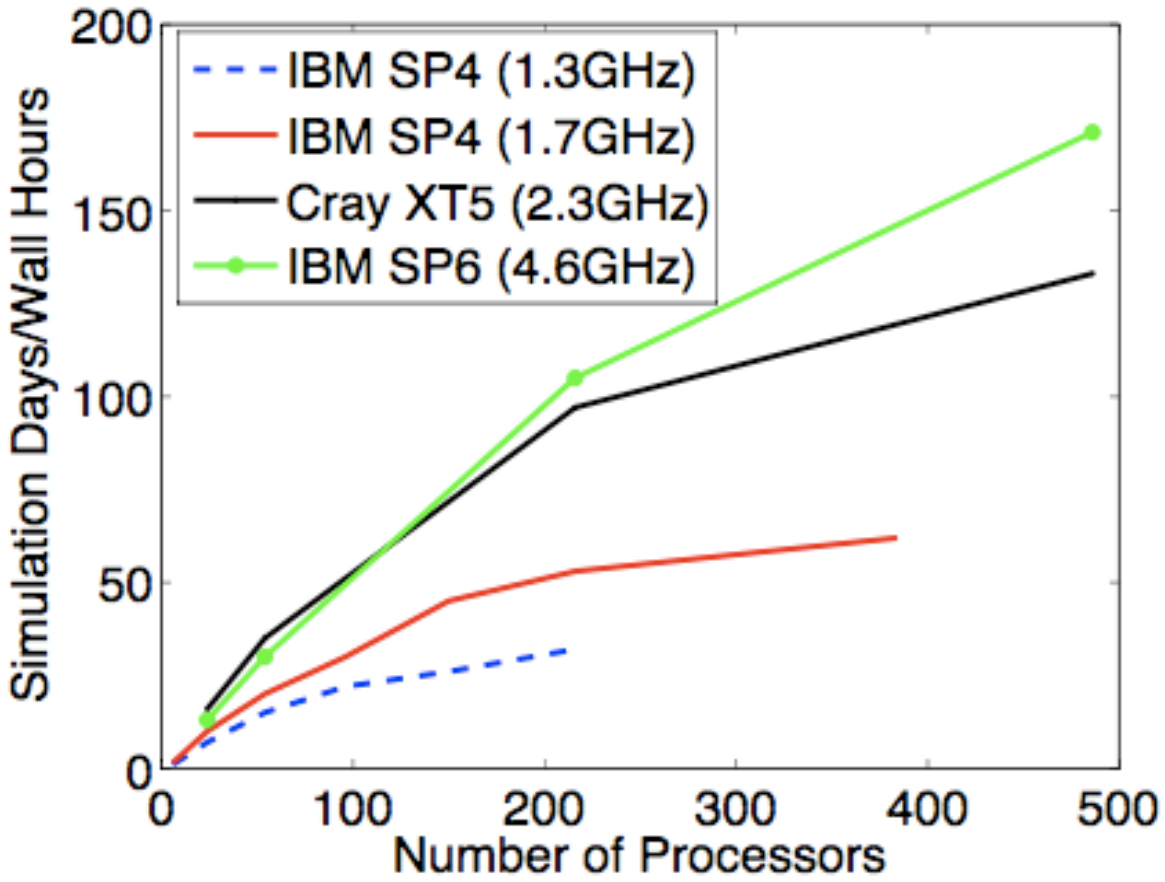


Temperature

30 day simulation of a Baroclinic Instability (Jablonowski-Williamson) for a fully 3D atmospheric model using 8^{th} order polynomials

Scalability of CG Method

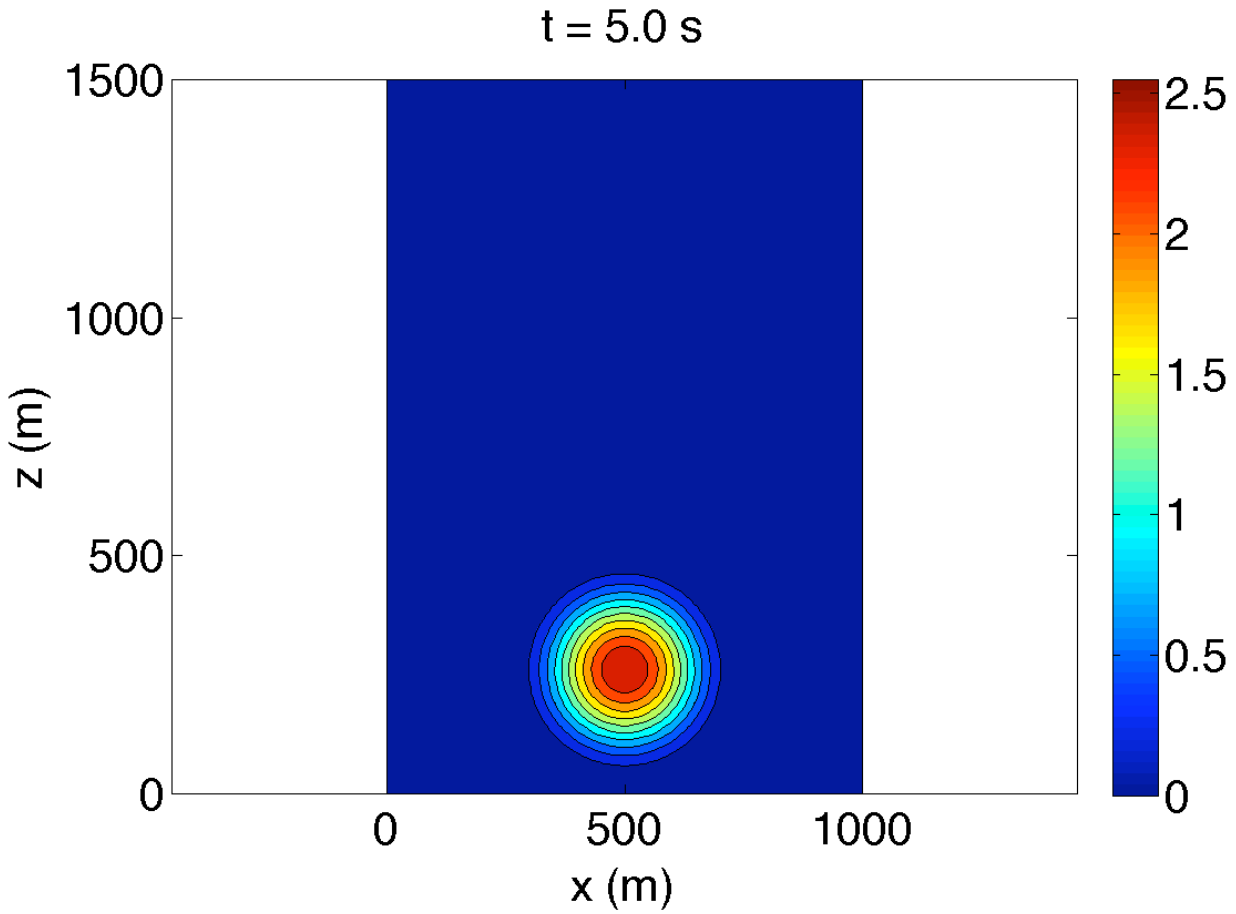
(Global Hydrostatic Model with T239 L30 DT=300 seconds)



Simulations performed at NAVO via PETTT Program

CG Method for 3D Rising Thermal Bubble

(Nonhydrostatic Cartesian Model)



Simulations performed by Jim Kelly

III. Suite of Time-Integrators

1. Explicit Time-Integrators

- Let's rewrite the governing equations as

$$\frac{\partial \mathbf{q}}{\partial t} = S(\mathbf{q})$$

- SSP-RK(2,3,4)

  For k=1,…,K

$$\mathbf{q}^{\mathrm{k}} = \sum_{m=0}^{k-1} \left(\alpha_{k,m} \mathbf{q}^{m} + \Delta t \beta_{k,m} S\left(\mathbf{q}^{m}\right) \right), \quad q^{0} = q^{n}, \quad q^{K} = q^{n+1}$$

- SSP-BDF2

  We write as

$$\mathbf{q}^{\mathrm{n+1}} = \alpha_0 \mathbf{q}^{n} + \alpha_1 \mathbf{q}^{n-1} + \gamma \Delta t \left(\beta_0 \, S(\mathbf{q}^{n}) + \beta_1 \, S(\mathbf{q}^{n-1}) \right)$$

  or in general, more compactly, as

$$\mathbf{q}^{\mathrm{n+1}} = \sum_{m=0}^{K-1} \alpha_m \mathbf{q}^{n-m} + \gamma \Delta t \sum_{m=0}^{K-1} \beta_m S(\mathbf{q}^{n-m})$$

2. Semi-Implicit Time-Integrator
(Building an implicit method on top of an explicit one)

- Once again, rewriting the governing equations as

$$\frac{\partial \mathbf{q}}{\partial t} = S(\mathbf{q})$$

- If we knew the linear operator L, then we could write

$$\frac{\partial \mathbf{q}}{\partial t} = \left\{S(\mathbf{q}) - \delta_{SI} L(\mathbf{q})\right\} + \delta_{SI}\left[L(\mathbf{q})\right]$$

- Discretizing by Kth order time-integrator yields

$$\mathbf{q}_{tt} = \hat{\mathbf{q}} + \lambda L\left(\mathbf{q}_{tt}\right)$$

Where

$$\lambda = \delta_{SI}\gamma\Delta t, \qquad \mathbf{q}_{tt} = \sum_{m=-1}^{K-1} \rho_m \mathbf{q}^{n-m}, \qquad \hat{\mathbf{q}} = \rho_{-1}\mathbf{q}^{\text{explicit}} + \sum_{m=0}^{K-1} \rho_m \mathbf{q}^{n-m}$$

and

$$\mathbf{q}^{\text{explicit}} = \sum_{m=0}^{K-1} \alpha_m \mathbf{q}^{n-m} + \gamma\Delta t \sum_{m=0}^{K-1} \beta_m S(\mathbf{q}^{n-m})$$

2. Semi-Implicit Time-Integrators
(Constructing the Schur Complement)

- The implicit problem that we have is: $(I - \lambda L)\mathbf{q}_{tt} = \hat{\mathbf{q}} \longrightarrow A\mathbf{q}_{tt} = b$
- Where the dims are: $A \in R^{dNxdN}$, $\mathbf{q}_{tt} \in R^{dN}$, $b \in R^{dN}$
  - For 2D Euler d=4, 3D d=5, etc.
  - For 2D Euler we solve a $16N^2$ system
  - For 2D SWE d=3, and solve a $9N^2$
- A better approach is to write out the system as follows (for 2D SWE):

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}\begin{pmatrix} \mathbf{u}_{tt} \\ \varphi_{tt} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

- Applying block LU decomposition:

$$\begin{pmatrix} A_{11} & A_{12} \\ 0 & A_{22} - A_{21}A^{-1}{}_{11}A_{12} \end{pmatrix}\begin{pmatrix} \mathbf{u}_{tt} \\ \varphi_{tt} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 - A^{-1}{}_{11}A_{12}b_1 \end{pmatrix}$$

  - Where the Schur Complement is: $\left(A_{22} - A_{21}A^{-1}{}_{11}A_{12}\right)\varphi_{tt} = b_2 - A^{-1}{}_{11}A_{12}b_1 \longrightarrow A\varphi_{tt} = b$
  - And the dimensions are: $A \in R^{NxN}$, $\varphi_{tt} \in R^{N}$, $b \in R^{N}$

2. Semi-Implicit (IMEX) Time-Integrators
(Important Properties of Schur Complement)

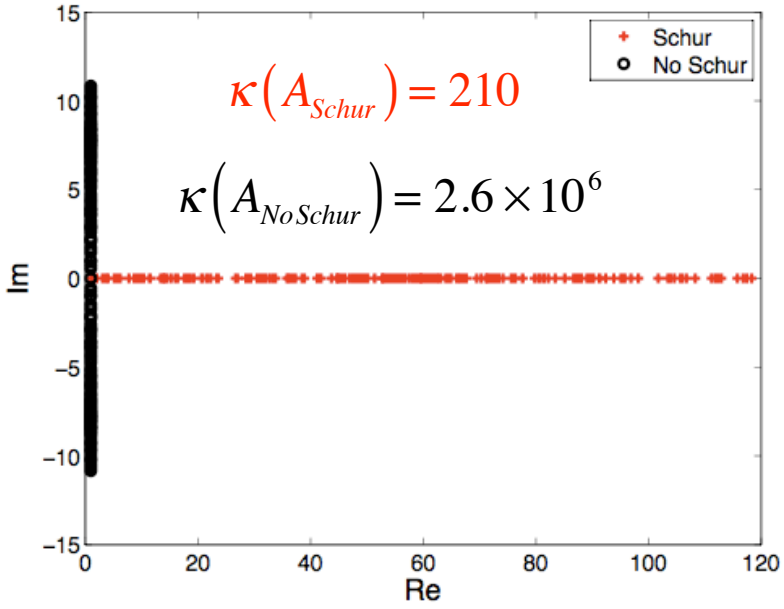
- The original system is:

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} \mathbf{u}_{tt} \\ \varphi_{tt} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

- The Schur Complement system is:

$$\left(A_{22} - A_{21}A^{-1}{}_{11}A_{12}\right)\varphi_{tt} = b_2 - A^{-1}{}_{11}A_{12}b_1$$

- This system is clearly smaller than the original system (NxN instead of 3Nx3N for 2D SWE).
- Equally important is that the Schur System is better conditioned than the original system. This means that fewer iterations are required by an iterative solver to reach convergence.
- **Key Point**: A semi-implicit method should be more efficient than the most efficient explicit methods, but with the Schur Complement the gains are much bigger.



2D Euler Equations

3. Fully-Implicit Time-Integrators

- Writing the governing equations as

$$\frac{\partial \mathbf{q}}{\partial t} = S(\mathbf{q})$$

- We then discretize this implicitly using implicit methods (eg., BDFK, IRK)

$$\mathbf{q}^{n+1} = \sum_{m=0}^{K-1} \alpha_m \mathbf{q}^{n-m} + \gamma \Delta t S(\mathbf{q}^{n+1})$$

- Where the matrix problem is now
  - Which clearly requires a nonlinear implicit solution (eg., Newton-Krylov methods)

$$\mathbf{q}^{n+1} - \gamma \Delta t S(\mathbf{q}^{n+1}) = \sum_{m=0}^{K-1} \alpha_m \mathbf{q}^{n-m}$$

- We write the problem as the functional:

$$F = \mathbf{q}^{n+1} - \gamma \Delta t S(\mathbf{q}^{n+1}) - \sum_{m=0}^{K-1} \alpha_m \mathbf{q}^{n-m} \equiv 0$$

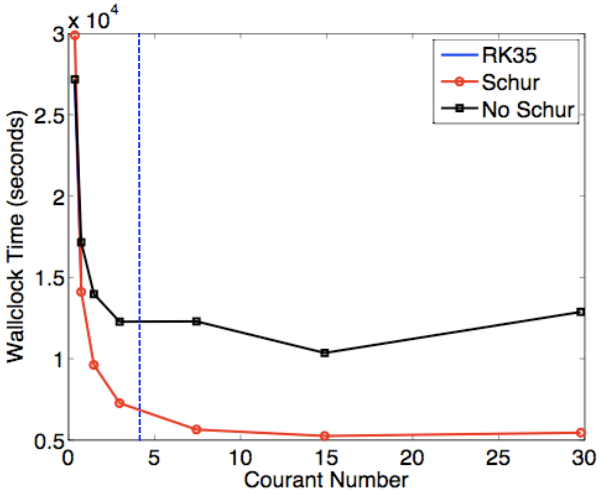
- And then solve the nonlinear problem:

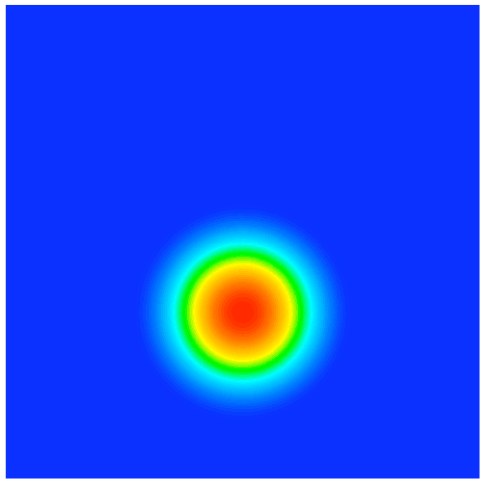
$$F(\mathbf{q})^{(k+1)} = F(\mathbf{q})^{(k)} + \frac{\partial F(\mathbf{q})^{(k)}}{\partial \mathbf{q}} \left(\mathbf{q}^{(k+1)} - \mathbf{q}^{(k)} \right) \equiv 0$$

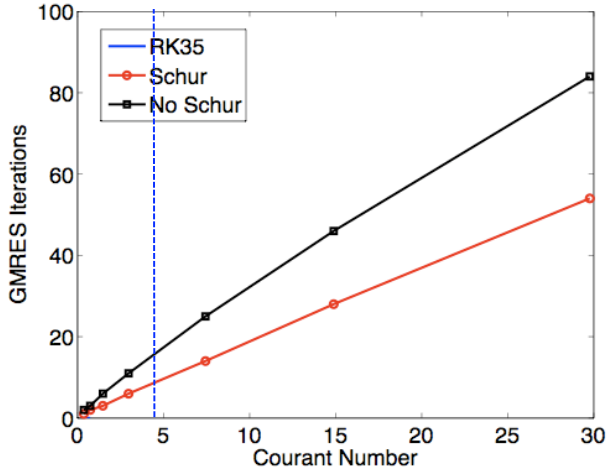
- With the resulting Linear System:

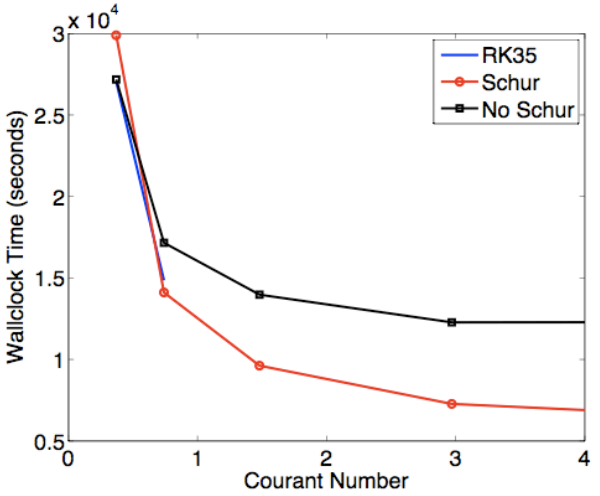
$$\frac{\partial F(\mathbf{q})^{(k)}}{\partial \mathbf{q}} \left(\mathbf{q}^{(k+1)} - \mathbf{q}^{(k)} \right) = -F(\mathbf{q})^{(k)}$$

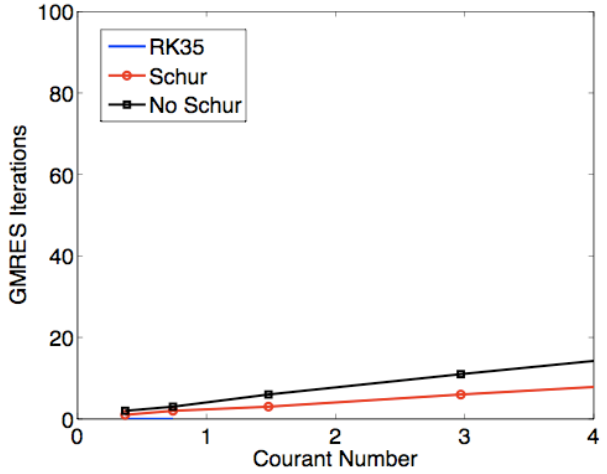
Comparison of SI Time-Integrators
(CG Rising Thermal Bubble)



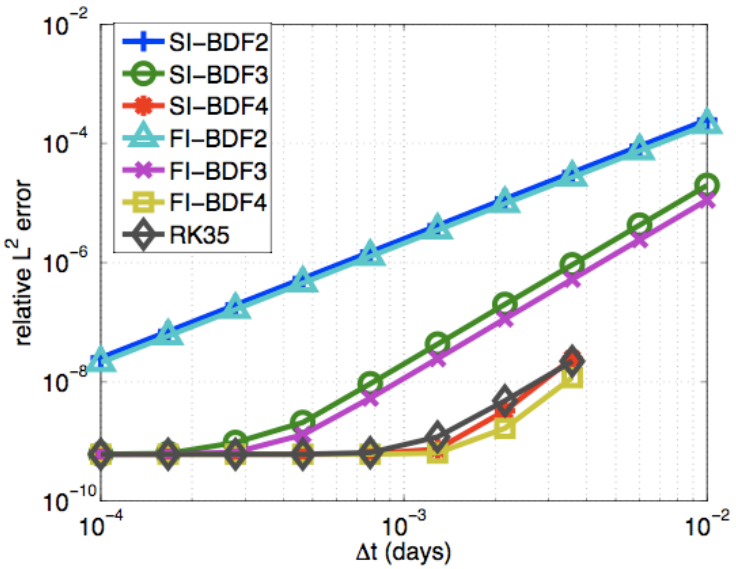


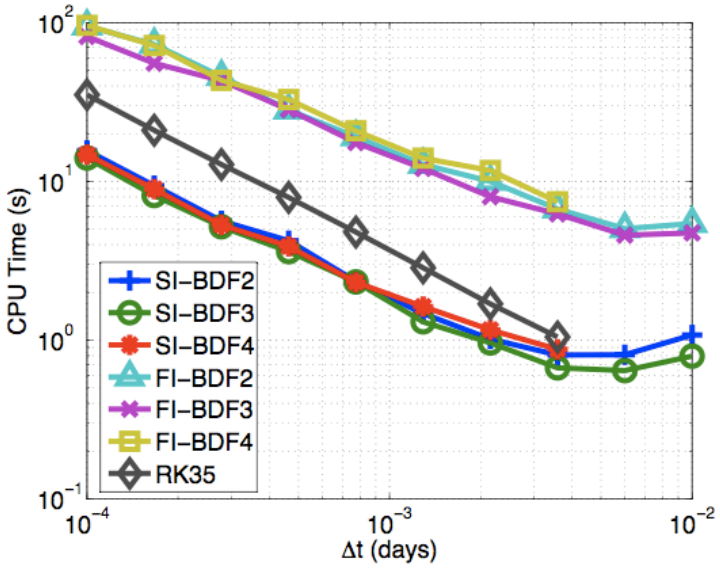


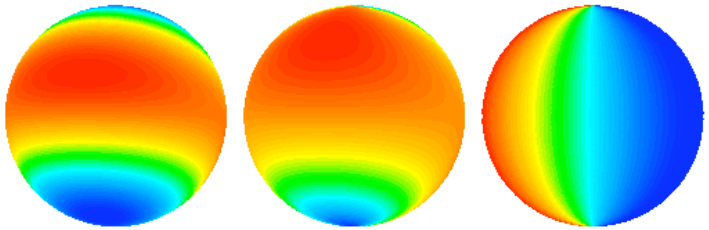




Comparison of FI/SI Time-Integrators
(Läuter Test Case for DG SSWE: No Schur Complement)

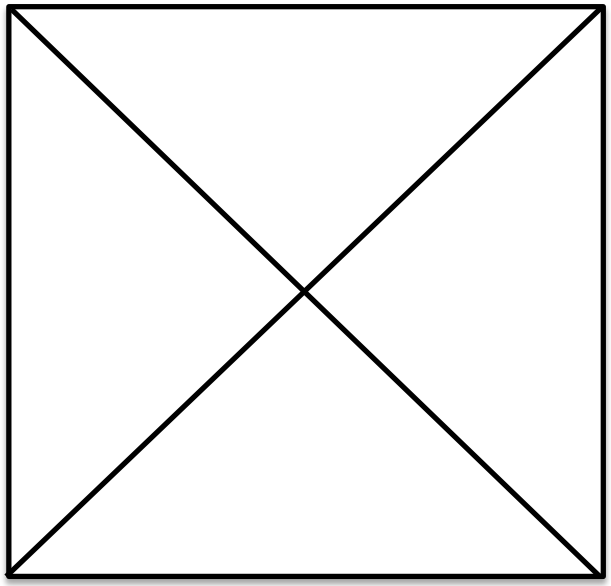


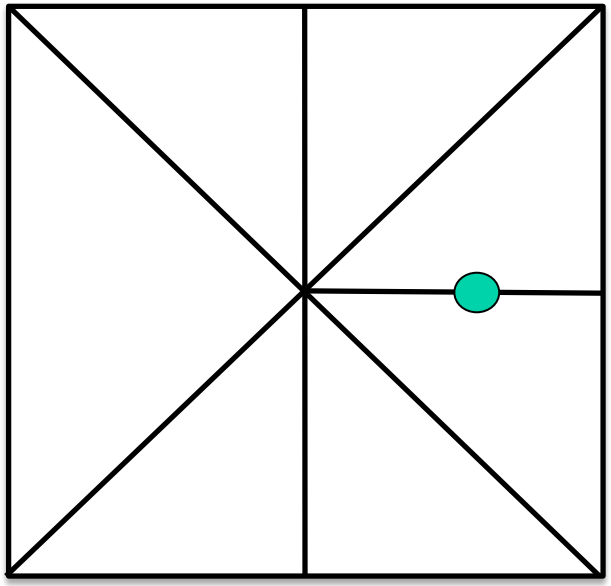




IV. Adaptivity

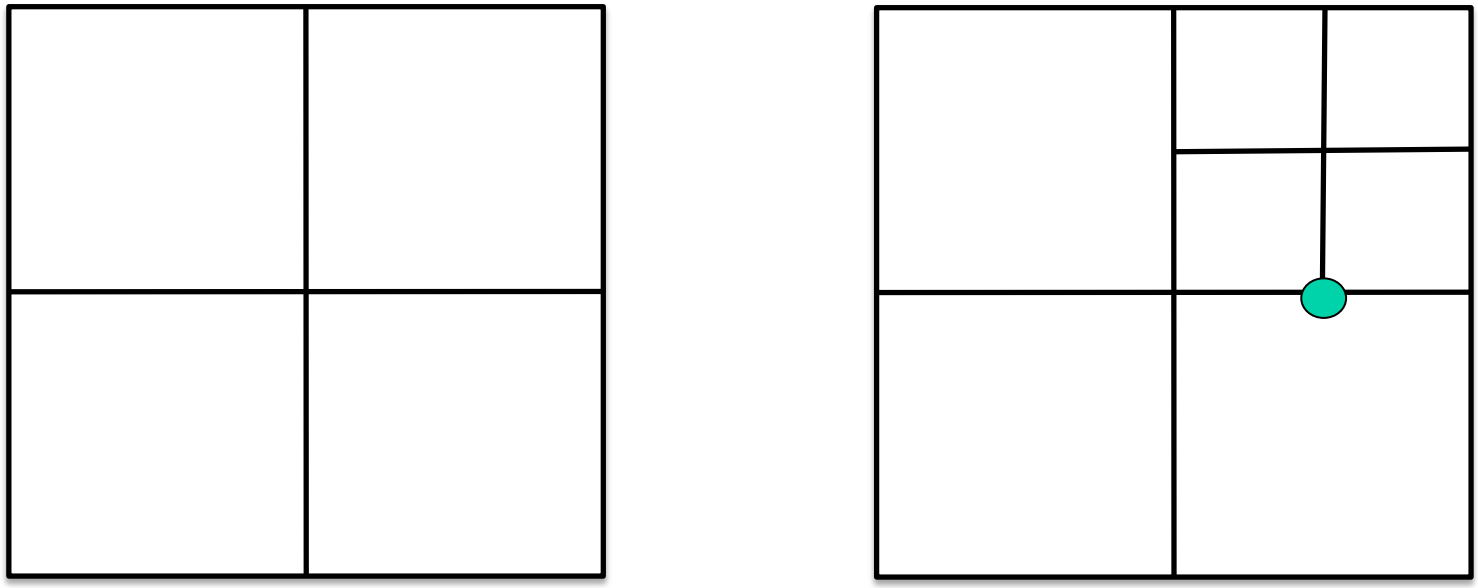
Adaptive Methods





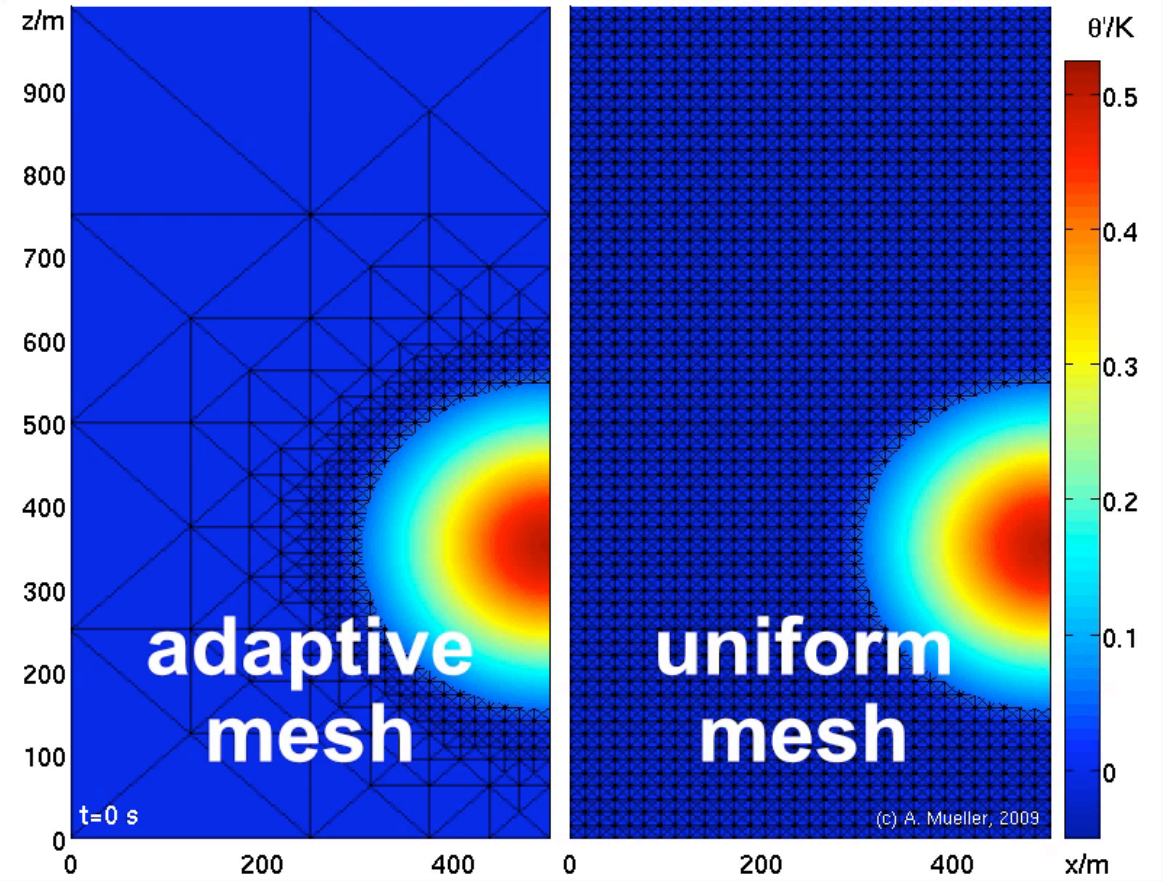
Conforming Grids: A grid point on an interface (edge/side) is owned by only two control volumes (the left and right elements). This kind of adaptive method places all of the work on the grid generator and NO WORK on the solver (what you use to solve your questions: think of your projects 3 and 4, for example). Therefore, one can construct a solver in isolation and interface it with a conforming adaptive method to produce adaptive solutions.

Adaptive Methods



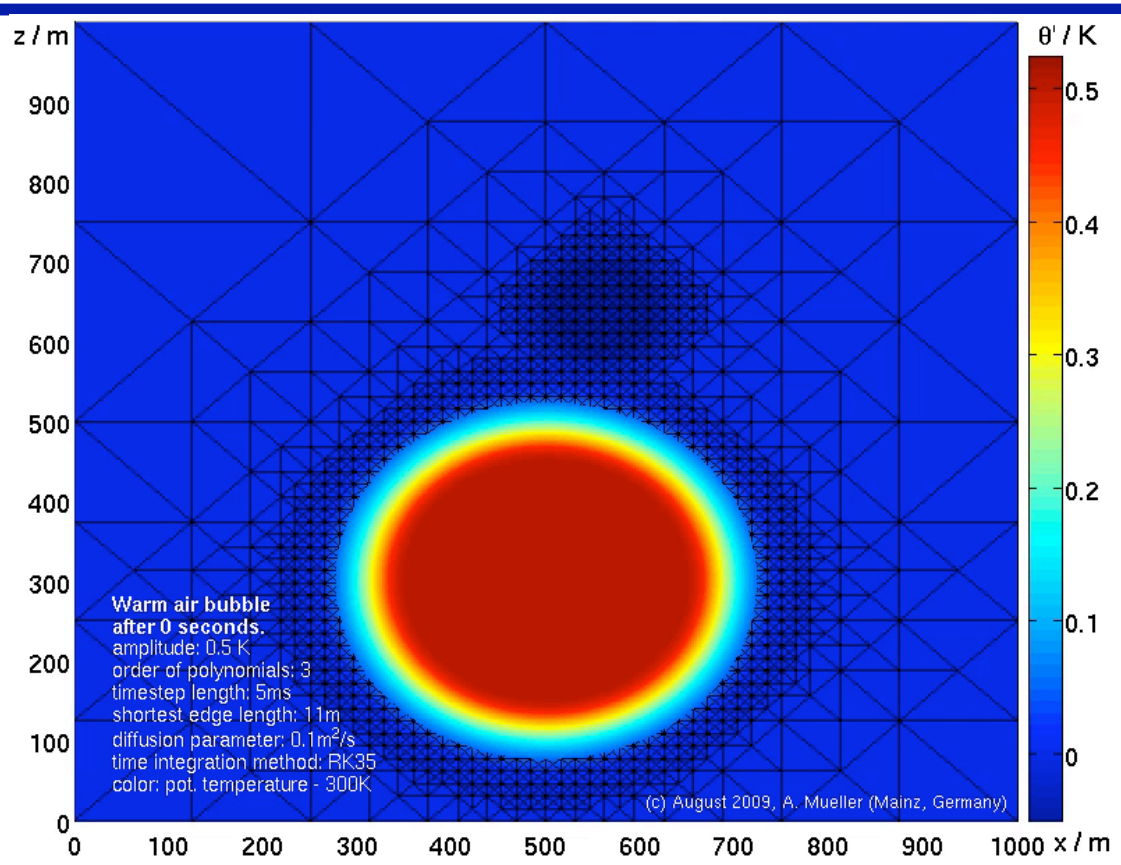
Non-Conforming Grids: A grid point on an interface can be owned by more than two control volumes. This kind of adaptive method places much of the burden on the solver as it now needs to be able to handle non-conforming grids but greatly simplifies the adaptivity process. This kind of grid can produce very efficient adaptive methods and is the idea used in the adaptive mesh refinement (AMR) methods of, e.g., Marsh Berger, Phil Colella, etc.

DG Method with Conforming Adaptivity



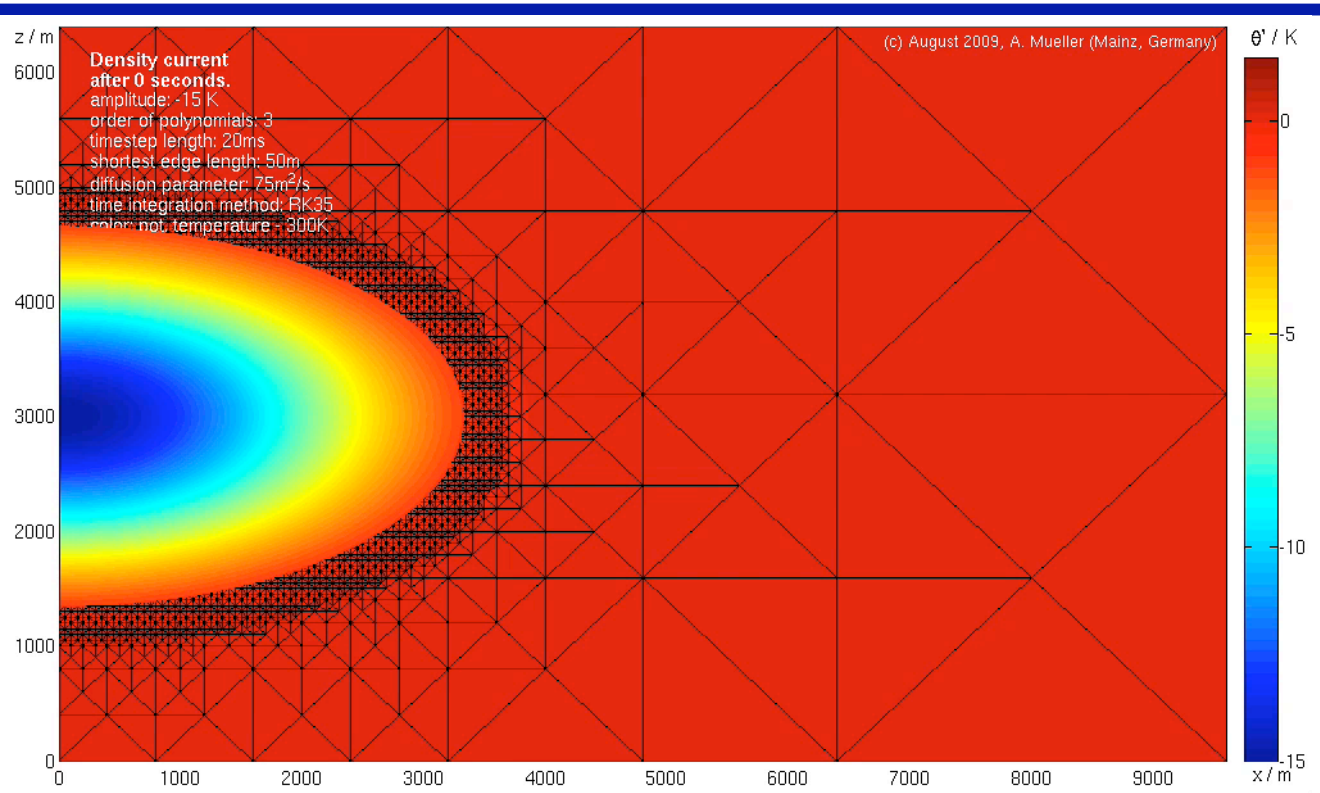
Simulations performed by Andreas Mueller in collaboration with Jörn Behrens

DG Method with Conforming Adaptivity



Simulations performed by Andreas Mueller in collaboration with Jörn Behrens

DG Method with Conforming Adaptivity



Simulations performed by Andreas Mueller in collaboration with Jörn Behrens

Closing Remarks

- This modeling framework allows us to produce codes relatively quickly and offers a simple means to validate new codes (once one component is working).
- Switching interpolation/integration points for DG is very simple in this approach and requires very few changes to any existing code (e.g., from Lobatto to Legendre).
- For the time-integrators, it makes it easy to test which approach is best for which type of problem.
- Also we anticipate that new time-integrators can be incorporated with minimal effort. We hope to show this to be the case (e.g., SI-RK methods, extrapolation methods, exponential Rosenbrock and Rupert's blended method).
- Did not address this here, but the success of SI/FI methods are heavily reliant on good preconditioners. We believe we have a good strategy for the Schur form (manuscript in preparation) but not for the No Schur form. This is good news for SI but not yet for FI.
- Efficiency is somewhat affected by this approach but makes the codes very easy to adapt to new needs (e.g., the CG methods carry along data structures that are only needed by the DG methods)
- Rewriting our codes to make them fully modular took some effort initially, but then facilitates the modification of the code and is simplifying the MPI implementation (in progress).