Implementation and Efficiency of Discontinuous Galerkin Spectral Element Methods for Compressible Flows

David A. Kopriva

Department of Mathematics
The Florida State University

DG Spectral Element Methods

Features:

1. Geometric flexibility - finite element method
2. Exponential convergence of the error: $E \sim \frac{e^{-\alpha N}}{K^N}$
3. Exponentially low dissipation errors
4. Exponentially low dispersion errors
5. Robust (esp. compared to strong form spectral methods)

Conventional Wisdom

Conventional wisdom states: DG spectral elements are

1. Too hard to implement
2. Less efficient than other methods, esp. Compact FD

Partly the fault of the spectral community...

We will show that, as usual, conventional wisdom is not necessarily correct.

3.4. Examples of DG methods. A simple and natural choice of numerical fluxes is

$$\widehat{u} = \{u_h\} \text{ on } \Gamma^0, \quad \widehat{u} = 0 \text{ on } \partial\Omega, \quad \text{and} \quad \widehat{\sigma} = \{\sigma_h\} \text{ on } \Gamma.$$

This is the choice proposed by Bassi and Rebay in [10]. With this choice of $\widehat{u}$, we have $\{\widehat{u} - u_h\} = 0$ and $[\![\widehat{u} - u_h]\!] = -[\![u_h]\!]$, so (3.9) gives

$$\sigma_h = \nabla_h u_h + r([\![u_h]\!]). \tag{3.17}$$

Therefore

$$\int_\Gamma \{\widehat{\sigma}\} \cdot [\![v]\!]\, ds = \int_\Gamma \{\nabla_h u\} \cdot [\![v]\!]\, ds - \int_\Omega r([\![u_h]\!]) r([\![v]\!])\, dx, \tag{3.18}$$

where we used the fact that $r([\![u_h]\!]) \in \Sigma_h$ and the definition (3.8) of r in the last step. Substituting in (3.11) we obtain the following primal form for the method of Bassi–Rebay [10]:

$$B_h(u_h, v) = \int_\Omega [\nabla_h u_h \cdot \nabla_h v + r([\![u_h]\!]) r([\![v]\!])]\, dx \\ - \int_\Gamma (\{\nabla_h u_h\} \cdot [\![v]\!] + [\![u_h]\!] \cdot \{\nabla_h v\})\, ds.$$

As a second example, we consider the classic IP method. This was originally proposed as a primal formulation, with

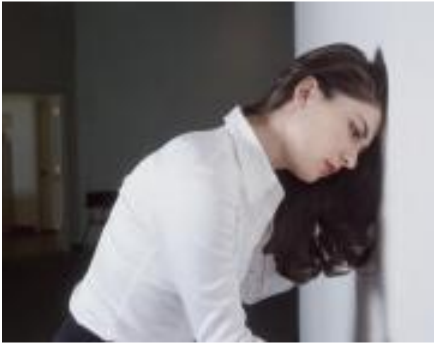
$$B_h(u_h, v) = \int_\Omega \nabla_h u_h \cdot \nabla_h v\, dx - \int_\Gamma ([\![u_h]\!] \cdot \{\nabla_h v\} + \{\nabla_h u_h\} \cdot [\![v]\!])\, ds + \alpha^{\mathrm{j}}(u_h, v), \tag{3.19}$$

where

$$\alpha^{\mathrm{j}}(u_h, v) = \int_\Gamma \mu [\![u_h]\!] \cdot [\![v]\!]\, ds \tag{3.20}$$

DG Spectral Element Approximation

It's Not That Hard!



Compressible Flow Problems

Problems modeled by a system of conservation laws:

$$\vec{q}_t + \nabla \cdot \vec{f} = 0$$

$$\vec{f} = \vec{f^i} + \vec{f^v}$$

where

- Euler Equations

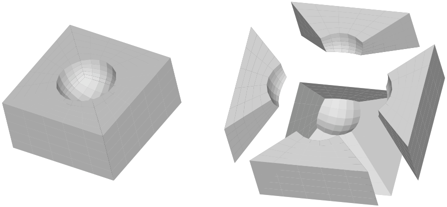
$$\vec{q} = \begin{bmatrix} \rho \\ \rho\vec{u} \\ \rho E \end{bmatrix}, \quad \vec{f^i} = \begin{bmatrix} \rho\vec{u} \\ \rho\vec{u}\otimes\vec{u} + pI \\ \rho u H \end{bmatrix}, \quad \vec{f^v} = 0$$

- Navier-Stokes Equations

$$\vec{f^v} = \begin{bmatrix} 0 \\ -\tau \\ \tau\cdot\vec{u} + k\nabla T \end{bmatrix}$$

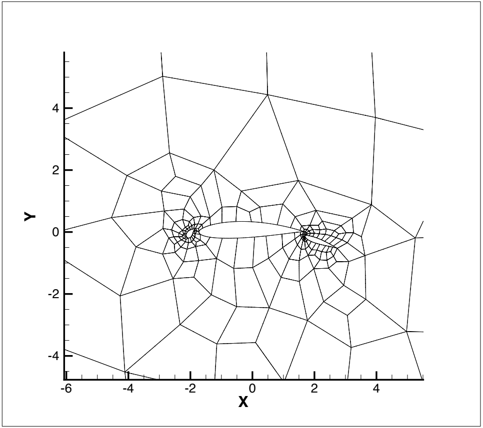
Multi-Element Decomposition

Subdivide domain into multiple elements



Multi-Element Decomposition

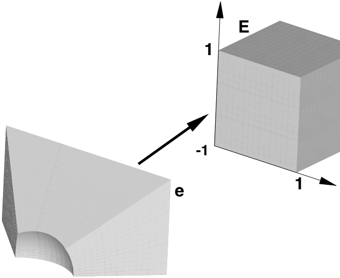
Can be arbitrarily complex



Mapping to Reference Element

Transform:

$$\mathbf{x} = \mathbf{X}(\xi)$$



Equations on Reference Element

Strong form of conservation law:

$$\tilde{Q}_t + \nabla \cdot \tilde{f} = 0$$

where

$$\tilde{Q} = J\mathbf{Q}$$

$$\tilde{f}^i = J\mathbf{a}^i \cdot \mathbf{f} = \sum_{n=1}^{3} J a_n^i \mathbf{f}_n$$

$$J\mathbf{a}^i = J\nabla\xi^i = \mathbf{a}_j \times \mathbf{a}_k = \frac{\partial \mathbf{X}}{\partial \xi^j} \times \frac{\partial \mathbf{X}}{\partial \xi^k} \quad (i,j,k)\ cyclic$$

$$J = \mathbf{a}_i \cdot (\mathbf{a}_j \times \mathbf{a}_k) \quad (i,j,k)\ cyclic$$

The DG Spectral Element Framework

Three characteristics:

1. Approximate

$$\mathbf{q} \approx \mathbf{Q} \in \mathbb{P}^N, \quad \mathbf{f} \approx \mathbf{F} \in \mathbb{P}^M on\ E$$

2. Weak form

$$\int_E \left(\tilde{Q}_t + \nabla \cdot \tilde{F} \right) \phi = 0$$

3. No continuity on $\phi \in \mathbb{P}^N$ between elements

DG Formulation

Integrate by parts

$$\int_E \tilde{Q}_t \phi d\xi + \int_{\partial E} \tilde{F} \cdot \hat{n}_\xi \phi dS - \int_E \tilde{F} \cdot \nabla \phi d\xi = 0$$

Replace boundary fluxes with Riemann solver

$$\int_E \tilde{Q}_t \phi d\xi + \int_{\partial E} \tilde{F}^* \cdot \hat{n}_\xi \phi dS - \int_E \tilde{F} \cdot \nabla \phi d\xi = 0 \quad Form\ I$$

Maybe integrate by parts again

$$\int_E \tilde{Q}_t \phi d\xi + \int_{\partial E} \left(\tilde{F} - \tilde{F}^* \cdot \hat{n}_\xi \right) \phi dS - \int_E \nabla \cdot \tilde{F} \phi d\xi = 0 \quad Form\ II$$

Choices, Choices, Choices

We actually have a *framework* from which to derive methods:

1. Quad/Hex or Tri/Tet elements?
2. Nodal or modal basis?
3. What polynomials?
4. Approximate boundaries with different orders?
5. Approximate solution and fluxes with different orders?
6. Exact integrals or quadrature?
7. Inexact or exact quadrature?
8. Form I or Form II?
9. ???

Too many choices can be overwhelming.

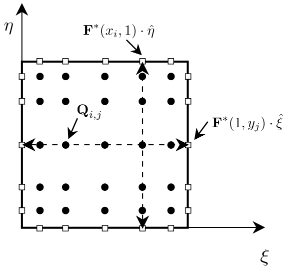
Easy to Implement and Effective Approximation

"Classical" spectral element approximation:

1. Quadrilateral/ Hexahedral elements
   $\Rightarrow$ Efficient tensor product bases
2. Nodal basis
   $\Rightarrow$ Easy for nonlinear/variable coefficient/general complex geometry problems
3. All approximations at same polynomial order
   $\Rightarrow$ Simplifies coding
4. Legendre basis
   $\Rightarrow$ Spectral accuracy
5. Gauss-Type quadrature

Choices Still to Make

1. Gauss or Gauss-Lobatto Quadrature?
   - Gauss exact for polynomials $\mathbb{P}^{2N+1}$
   - Lobatto exact for polynomials $\mathbb{P}^{2N-1}$



2. Integrate by parts once or twice?

Integrate By Parts 1X or 2X?

Theorem

(Kopriva and Gassner, 2010) For quadrilateral/hexahedral tensor product discontinuous Galerkin approximations to systems of hyperbolic conservation laws with either Gauss or Gauss-Lobatto quadratures the two forms are **algebraically equivalent** *as long as one uses global polynomial representations for the flux and solutions.*

Gauss Vs Gauss Lobatto

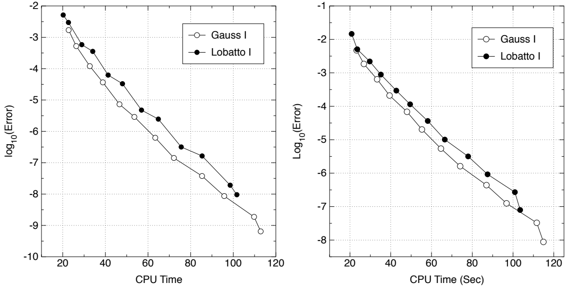
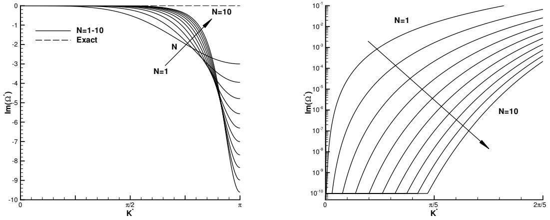


Figure: Maximum error as a function of work for the Gauss and Lobatto approximations. Left: Uniform mesh. Right: Non-Uniform Mesh

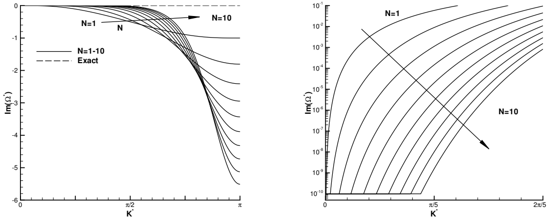
Empirical evidence indicates Gauss is more robust than Gauss-Lobatto. Analysis shows Gauss has higher dissipation in high frequencies, lower dissipation in low frequencies than Gauss-Lobatto.

Gauss Vs Gauss Lobatto



(a) Dissipation relation (b) Logarithm of dissipation error

FIG. 6.1. *Imaginary part of the physical mode for the Gauss DGSEM scheme with* $N = 1$ *up to* $N = 10$*. In the logarithmic plot, the error is cut off at* 10^{-10} *to avoid numerical noise.*



(a) Dissipation relation (b) Logarithm of dissipation error

Final Choice

1. Quadrilateral/ Hexahedral elements
2. Nodal basis, Gauss points
3. All approximations at same polynomial order
4. Legendre-Gauss basis
5. Legendre Gauss quadrature

Implementation

Solution and fluxes by polynomials in (Lagrange) nodal form

$$\mathbf{Q} = \sum_{n=0}^{N} \sum_{m=0}^{N} \mathbf{Q}_{n,m} \ell_n(\xi)\ell_m(\eta)$$

$$\mathbf{F} = \sum_{n=0}^{N} \sum_{m=0}^{N} \left(\mathbf{F}_{n,m}\hat{x} + \mathbf{G}_{n,m}\right) \ell_n(\xi)\ell_m(\eta).$$

Integrate by parts 1x

$$\int_E \frac{\partial \mathbf{Q}}{\partial t}\phi_{i,j} d\xi + \int_{\partial E} \mathbf{F}^* \cdot \hat{n}\phi_{i,j} dS - \int_E \mathbf{F} \cdot \nabla \phi_{i,j} d\xi = 0$$

With $\phi_{i,j} = \ell_i(\xi)\ell_j(\eta)$.

Apply Quadrature to Each Integral

Time derivative integral

$$
\begin{aligned}
&\int_{-1,N}^{1} \frac{d\mathbf{Q}(\xi,\eta)}{dt}\ell_i(\xi)\ell_j(\eta)d\xi d\eta \\
&= \sum_{k=0}^{N}\sum_{l=0}^{N} \frac{d\mathbf{Q}(\xi_k,\eta_l)}{dt}\ell_i(\xi_k)\ell_j(\eta_l)w_k^{(\xi)}w_l^{(\eta)} \\
&= \frac{d\mathbf{Q}_{i,j}}{dt}w_i^{(\xi)}w_j^{(\eta)}
\end{aligned}
$$

Volume and Boundary Integrals

$$\int_{-1,N}^{1} \mathbf{F} \cdot \nabla \phi_{ij} d\xi d\eta = \sum_{k=0}^{N} \mathbf{F}_{k,j} \ell_i'(\xi_k) w_k^{(\xi)} w_j^{(\eta)} + \sum_{k=0}^{N} \mathbf{G}_{i,k} \ell_j'(\xi_k) w_i^{(\xi)} w_k^{(\eta)}$$

$$\int_{\Gamma,N} \phi_{i,j} \mathbf{F}^* \cdot \hat{n} d\Gamma = \mathbf{F}^*(x_i,-1) \cdot (-\hat{\eta})\, \ell_j(-1) w_i^{(\xi)} + \mathbf{F}^*(1,\eta_j) \cdot \hat{\xi} \ell_i(1) w_j^{(\eta)} + \mathbf{F}^*(\xi_i,1) \cdot \hat{\eta} \ell_j(1) w_i^{(\xi)} + \mathbf{F}^*(-1,\eta_j) \cdot \left(-\hat{\xi}\right) \ell_i(-1) w_j^{(\eta)}$$

Spatial Discretization

On each element we integrate

$$\frac{d\mathbf{Q}_{i,j}}{dt} + \left\{ \left[\tilde{\mathbf{F}}^*(1,\eta_j)\frac{\ell_i(1)}{w_i^{(\xi)}} - \tilde{\mathbf{F}}^*(-1,\eta_j)\frac{\ell_i(-1)}{w_i^{(\xi)}} \right] + \sum_{k=0}^{N} \tilde{\mathbf{F}}_{k,j}\hat{D}_{ik}^{(\xi)} \right\}$$
$$+ \left\{ \left[\tilde{\mathbf{G}}^*(\xi_i,1)\frac{\ell_j(1)}{w_j^{(\eta)}} - \tilde{\mathbf{G}}^*(\xi_i,-1)\frac{\ell_j(-1)}{w_j^{(\eta)}} \right] + \sum_{k=0}^{N} \tilde{\mathbf{G}}_{i,k}\hat{D}_{jk}^{(\eta)} \right\} = 0$$

Primary Work:

- Computation of fluxes $\tilde{\mathbf{F}}_{k,j}$ and $\tilde{\mathbf{G}}_{i,k}$ from solution
- Computation of Riemann solver $\tilde{\mathbf{F}}^*(\pm 1,\eta_j)$ and $\tilde{\mathbf{G}}^*(\xi_i,\pm 1)$
- Series of dot products (Gauss)
- Series of Matrix-Vector products

DGSEM Time Derivative Algorithm

Gauss-Lobatto Version:

for $j = 0$ **to** M **do**

$\mathbf{F} = xFlux(\mathbf{Q}_j)$

$\mathbf{F}' = MatrixTimesVector(\hat{D}, \mathbf{F})$

$\dot{\mathbf{Q}}_j = -\mathbf{F}'$

$\dot{Q}_{0,j} = \dot{Q}_{0,j} - b_j^L * RiemannSolver(Q_j^{ext}, Q_{0,j}, \hat{n}_j^L)$

$\dot{Q}_{N,j} = \dot{Q}_{N,j} - b_j^R * RiemannSolver(Q_{N,j}, Q_j^{ext}, \hat{n}_j^R)$

end

for $i = 0$ **to** N **do**

$\mathbf{G} = yFlux(\mathbf{Q}_i)$

$\mathbf{G}' = MatrixTimesVector(\hat{D}, \mathbf{G})$

$\dot{\mathbf{Q}}_i = \dot{\mathbf{Q}}_i - \mathbf{G}'$

$\dot{Q}_{i,0} = \dot{Q}_{i,0} - b_i^B * RiemannSolver(Q_i^{ext}, Q_{i,0}, \hat{n}_i^B)$

$\dot{Q}_{i,M} = \dot{Q}_{i,M} - b_i^T * RiemannSolver(Q_{i,M}, Q_i^{ext}, \hat{n}_i^T)$

end

DGSpectral Element Approximation

See... It's Not That Bad!

Viscous Fluxes

Viscous flux depends on gradients:

$$F^v = F^v\left(\vec{u}, \nabla\vec{u}, \nabla T\right)$$

Approximate weakly:

$$\int_E \left(\vec{\Phi} - \nabla\vec{U}\right)\varphi dV = 0 \quad U = \begin{bmatrix} u \\ v \\ T \end{bmatrix}$$

Get gradient approximation that is computed the same as before:

$$\Phi_{i,j} = \frac{1}{J_{i,j}}\left\{D_\xi\left(JU\nabla\xi\right) + D_\eta\left(JU\nabla\eta\right)\right\}$$

$$D_\xi\left(W\right) = \left[\tilde{W}^*(1,\eta_j)\frac{\ell_i(1)}{w_i^{(\xi)}} - \tilde{W}^*(-1,\eta_j)\frac{\ell_i(-1)}{w_i^{(\xi)}}\right] + \sum_{k=0}^{N}\tilde{W}_{k,j}\hat{D}_{ik}^{(\xi)}$$

Yes, But...

"Everyone knows" that spectral element methods are

1. Highly accurate but
2. Expensive to compute relative to finite difference schemes.

Navier Stokes/Acoustics: Discretization Schemes

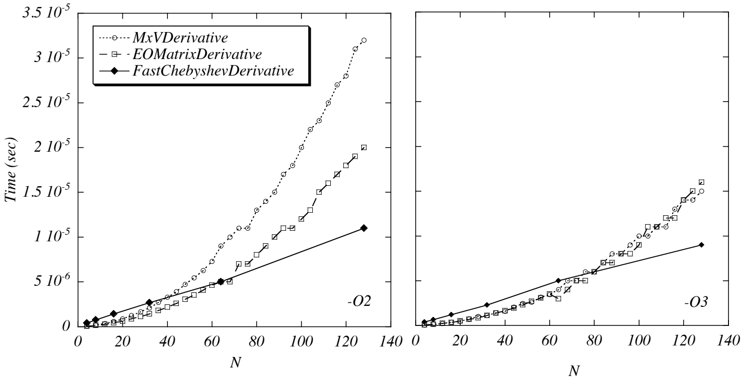
High order schemes are acknowledged as necessary.

1. 5/7 Point stencil optimized finite difference. "DRP"
   - Wide stencils, must filter, ghost points galore, $O(N)$
2. Optimized compact finite difference (e.g. Ashcroft & Zhang, JCP 2003)
   - Tri-diagonal solves, must filter, $O(N)$
3. High order discontinuous Galerkin/Spectral element
   - Matrix-Vector products, $O\left(N^2\right)$

Matrix-Vector Product

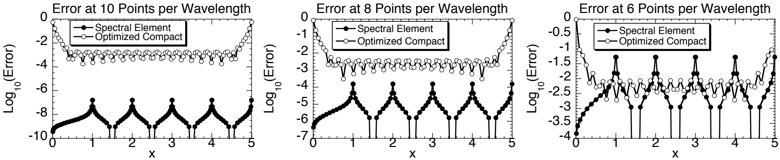
Conventional wisdom: Full $\hat{D}$ makes SEM less efficient than FD.

But matrix-vector products are *fast*:



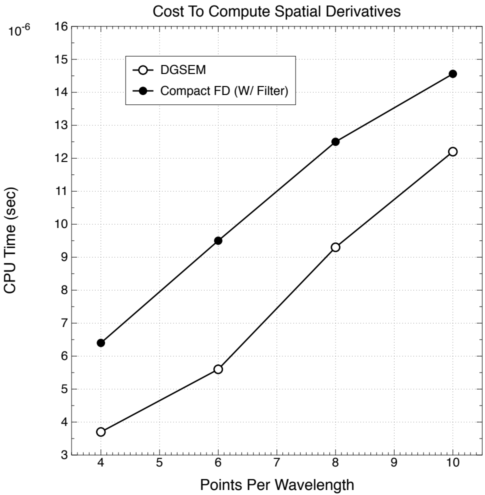
Comparison to Compact Finite Differences

Error to compute derivative of sine waves



Cost Comparison: DGSEM vs Compact FD

Spectral elements are not necessarily more costly than optimized finite differences:



Cost Comparison: DGSEM vs Compact FD

Hard Numbers: CPU cost per point for 3D Navier Stokes.

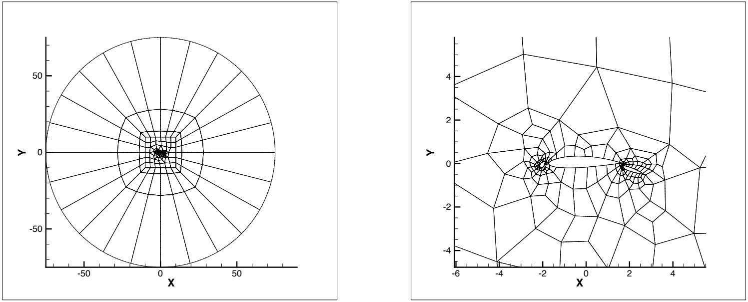
Euler N=6	NS N = 6	NS FD 6
0.26	0.7-0.9	2.5

Table: Cost Per Grid Point ($\mu sec/DOF/EQN/Stage$)

—Finite difference methods are not necessarily cheaper per grid point, either!

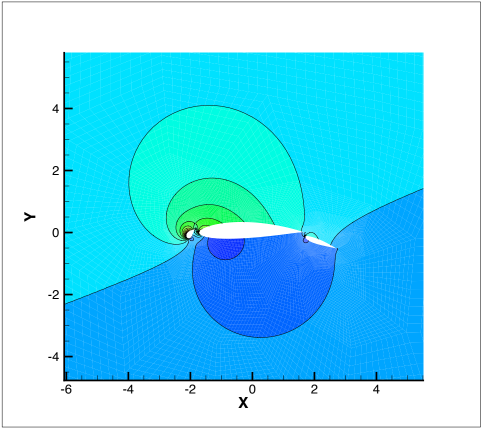
Case Study I

Spectral element solution of flow over 3 Element airfoil



Steady solution

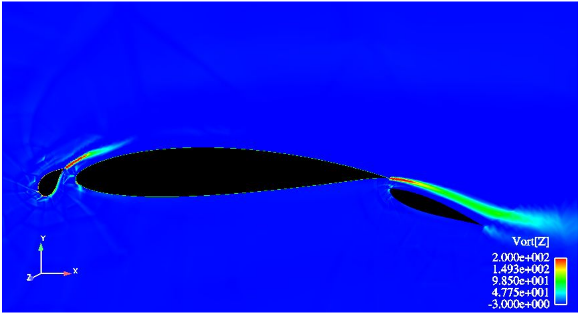
The Good News: 8^{th} order polynomial solution Mach contours...



Look: No vortex streets!

Steady solution

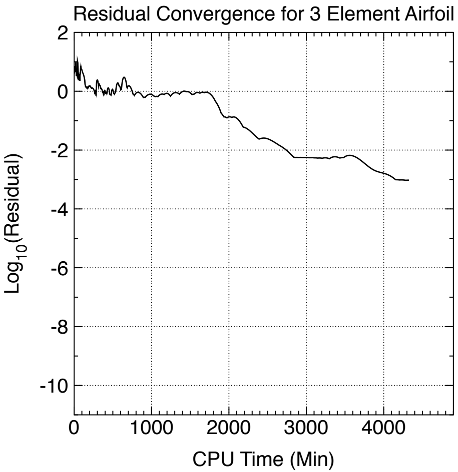
Fluent Solution (Vorticity)...



Look: vortex streets!

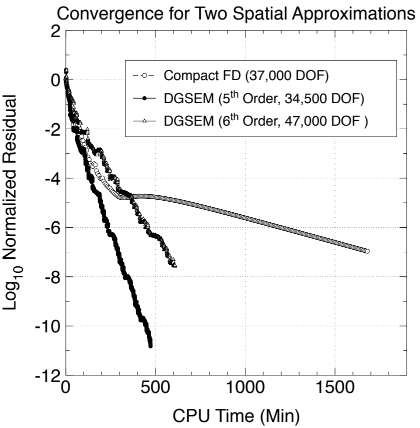
Explicit Convergence to Steady State

The Bad News: 8^{th} order polynomial solution for 3 element airfoil…Yikes



But Aren't FD Methods Better?

Not necessarily... Joukowski Airfoil 2D, Explicit time integration



Time Integration

Recent Implementations:

1. Backward Euler (For Steady State): $\mathbf{Q}^n + \Delta t \mathbf{R}\left(\mathbf{Q}^{n+1}, t_{n+1}\right)$
2. Explicit 1^{st} Stage Singly Diagonal Implicit Runge Kutta (ESDIRK)

$$\mathbf{Q}^{(1)} = \mathbf{Q}^n$$

$$\mathbf{Q}^{(i)} = \mathbf{Q}^n + \Delta t \sum_{j=1}^{i-1} c_{ij}\mathbf{R}\left(\mathbf{Q}^{(j)}, t_j\right) + \gamma \Delta t R\left(\mathbf{Q}^{(i)}, t_i\right)$$

$$\mathbf{Q}^{n+1} = \mathbf{Q}^n + \Delta t \sum_{i=1}^{s} b_i \mathbf{R}\left(\mathbf{Q}^{(i)}, t_i\right)$$

3. Backward Differentiation (BDF)

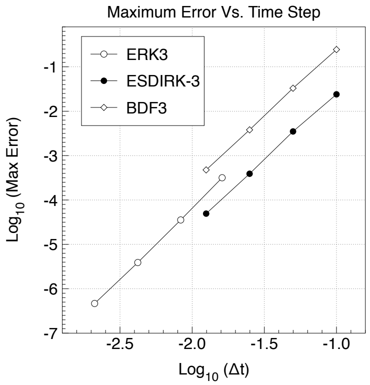
$$\mathbf{Q}^{n+1} = \sum_{j=0}^{s} \alpha_j \mathbf{Q}^{n-j} + \gamma \Delta t \mathbf{R}\left(\mathbf{Q}^{n+1}, t_{n+1}\right)$$

Performance of Time Integration Methods

Linear model hyperbolic system

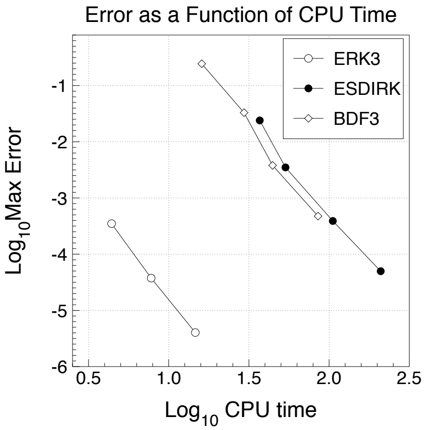
$$Q_t + AQ_x + BQ_y = 0$$

with sinusoidal propagating wave solution.



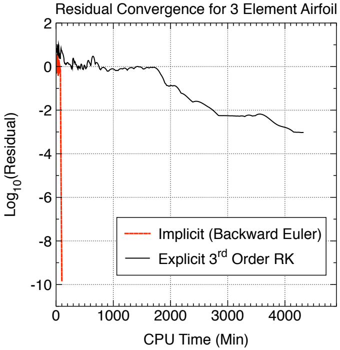
Efficiency of Time Integrators

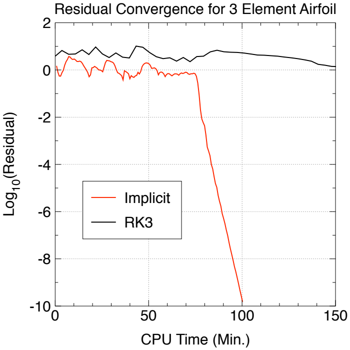
For time accuracy, error looks different in terms of work:



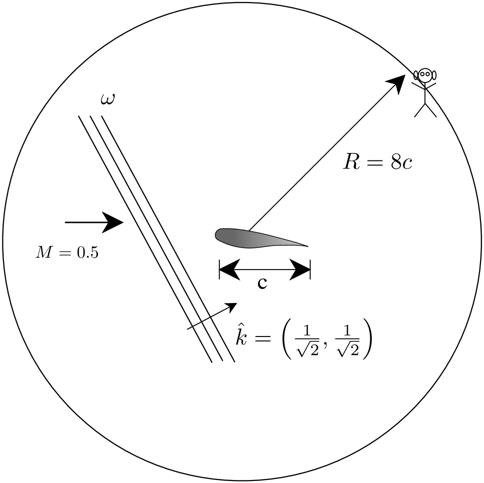
— Affects the choice for time dependent problems...

Implicit Solution of 3 Element Airfoil





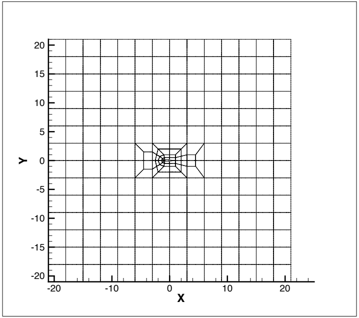
NASA/CP-2000-209790
Time Dependent - Time accuracy required

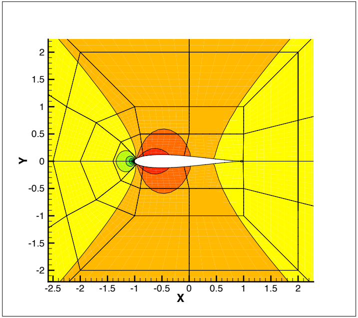


Solution Approach

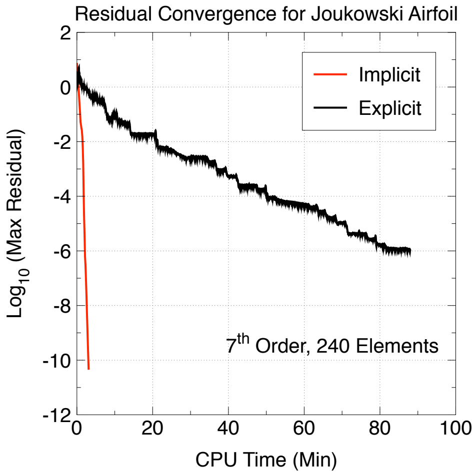
Problem solved in two stages:

1. Solution of steady-state
2. Addition of vorticity wave solved to time periodic state

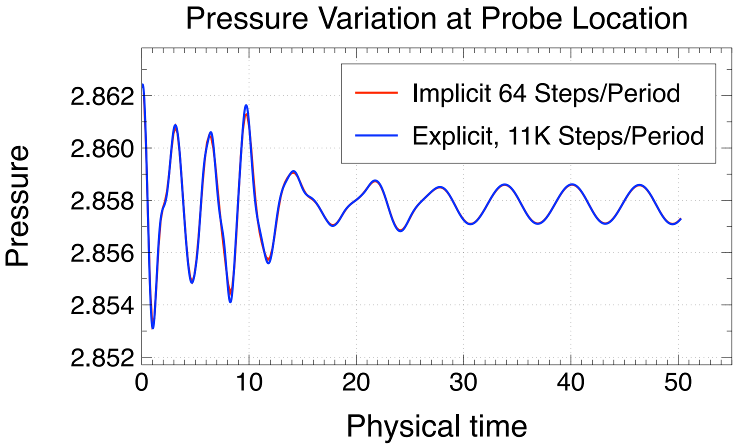




Steady State Cost

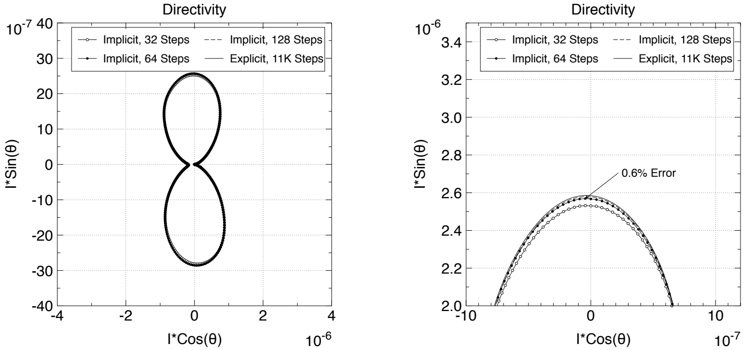


Convergence to Time Periodic State



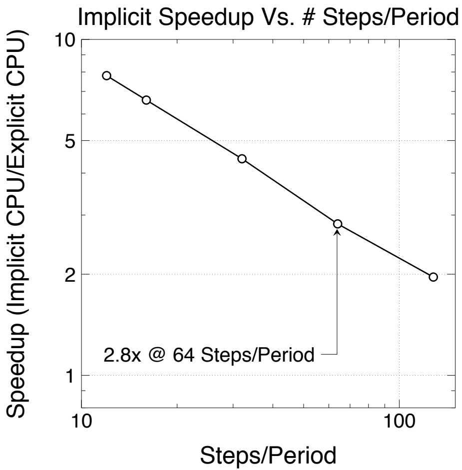
Solution: Directivity

At least 64 steps per period needed for accurate solution:



Efficiency of Implict Vs. Explicit Time Integration

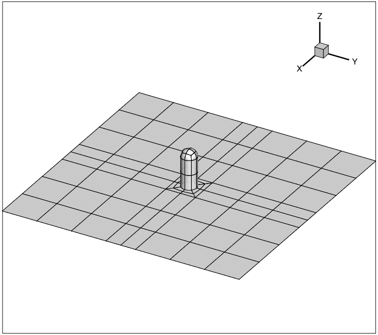
For desired accuracy, speedup of only 2-3 times over explicit:

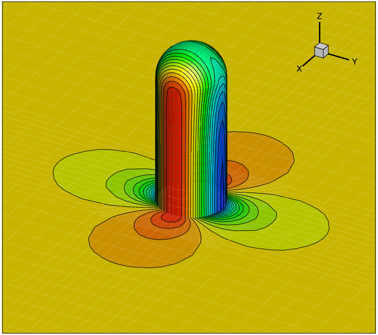


Adding Parallelism

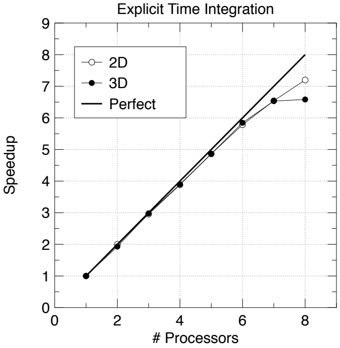
Computation of time derivatives is highly parallelizable.

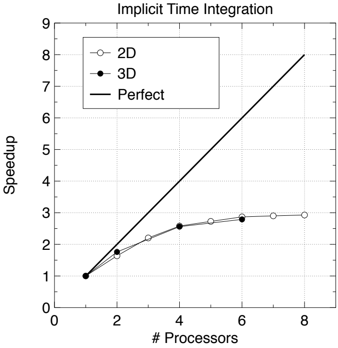
Case Study III:



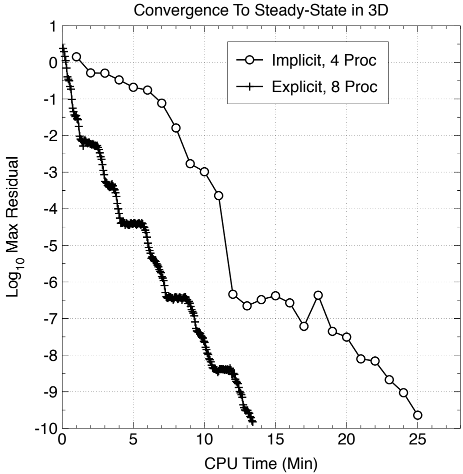


Speedup: Euler





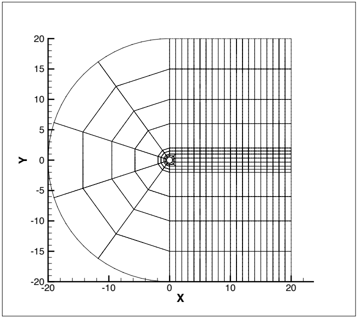
Parallel Convergence

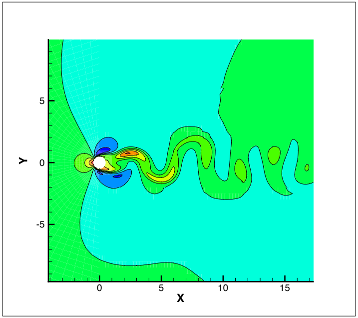


Adding Parallelism

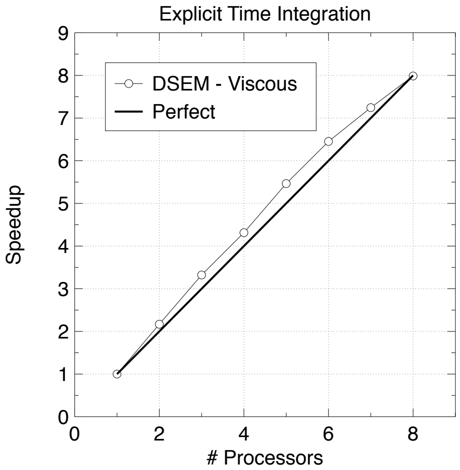
Computation of time derivatives is highly parallelizable.

Case Study IV:





Speedup: Navier Stokes



Conclusions

- DGSEM's Don't have to be difficult to implement!
  - Matrix-Vector Multiplies + Dot products. (No Tri-diagonal solves)
  - Boundary conditions easy - Riemann solver. 125^{th} order as easy as Roe's scheme. (No ghost points)
- For Efficient Steady-State Computations...
  - Significant speedups of implicit over explicit, at least in 2D
  - Preconditioning is critical
  - Storage of preconditioner limits order to about $N = 4$ in 3D
- For Efficient Time Dependent (Wave Propagation) Computations...
  - Advantage of implicit over explicit reduced. (2x-3x)
  - Parallelism of explicit approximation may negate advantages gained by implicit

But at least we beat Compact FD!