Galerkin Methods for Incompressible Flow Simulation

Paul Fischer

Mathematics and Computer Science Division

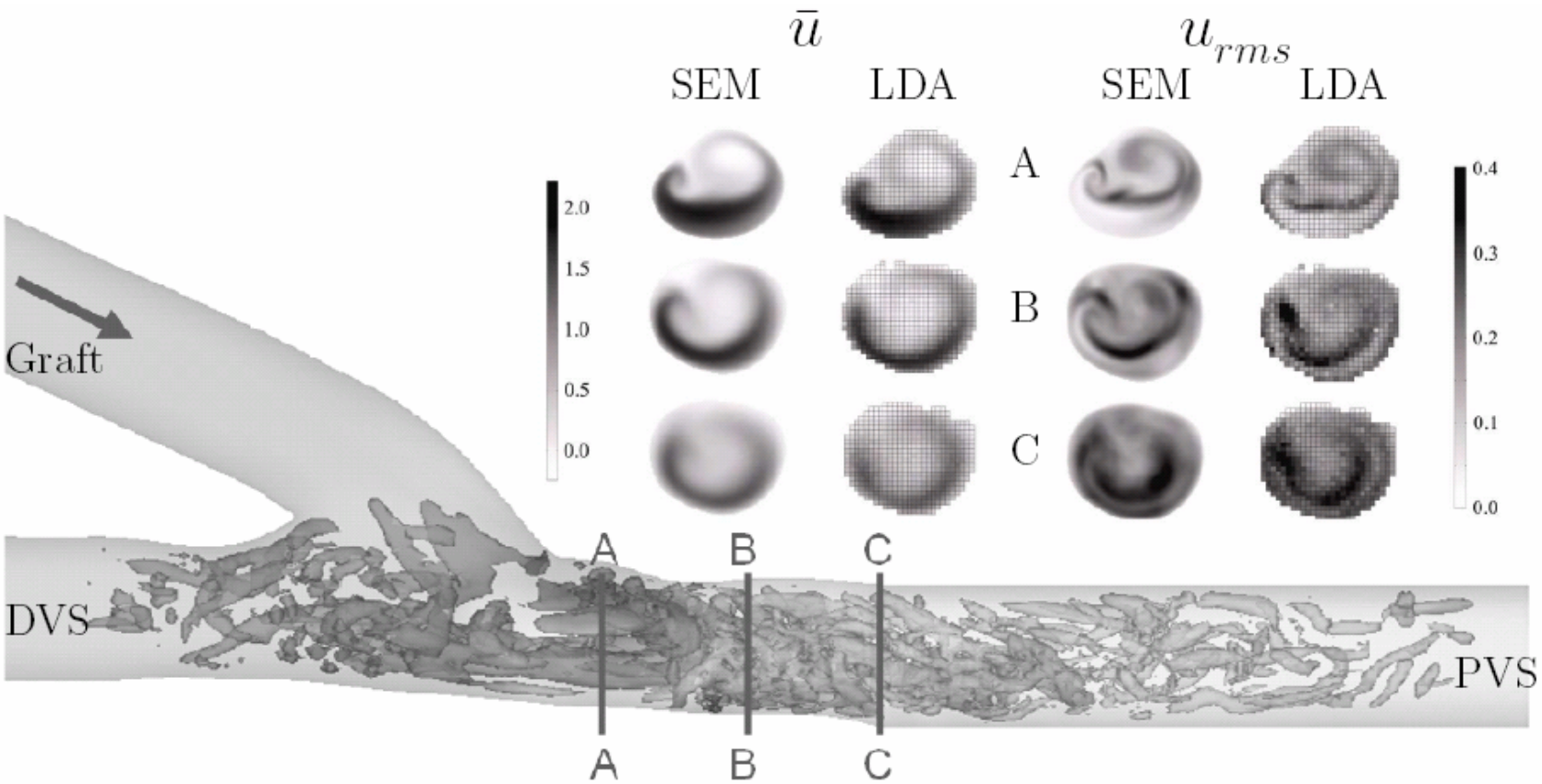
Argonne National Laboratory

Example: Vascular Flow Simulations

Loth, Bassiouny, F (UIC/U Akron / UC / ANL)

Seeking to understand hemodynamics & vascular disease:

- *Combined in vivo, in vitro, numerical studies of AV grafts*
- *Validated code in a complex geometry using LDA measurements*
- *Identified mechanisms for low Re transition to turbulence in AV grafts*



Coherent structures in AV graft at Re=1200; λ_2 criterion of Jeong & Hussein (JFM95)

Outline of Lectures

Lecture 1: Introduction

- Navier Stokes Equations
- Quick overview of some relevant timestepping issues
- Galerkin methods
  - Introduction
  - 1D examples (2D…?)

Lecture 2: Galerkin methods cont'd

- 2D / 3D examples
- Solvers
  - Iterative
  - Fast diagonalization methods

Lecture 3: Putting it all together: Unsteady Navier-Stokes Solver

- Unsteady Stokes / spatial discretization (velocity/pressure spaces)
- System solvers
- Examples: natural convection / driven cavity / Kovasznay flow
- Extensions to 3D

Objectives

- Enable solution of NS in a simple domain
- Develop understanding of
  - costs/benefits of FEM/SEM/spectral methods
  - Differences between low/high order, FEM/FD
- Importance of pressure treatment

Stability of Various Timesteppers

- Derived from model problem $\frac{du}{dt} = \lambda u$
- Stability regions shown in the $\lambda\Delta t$ plane

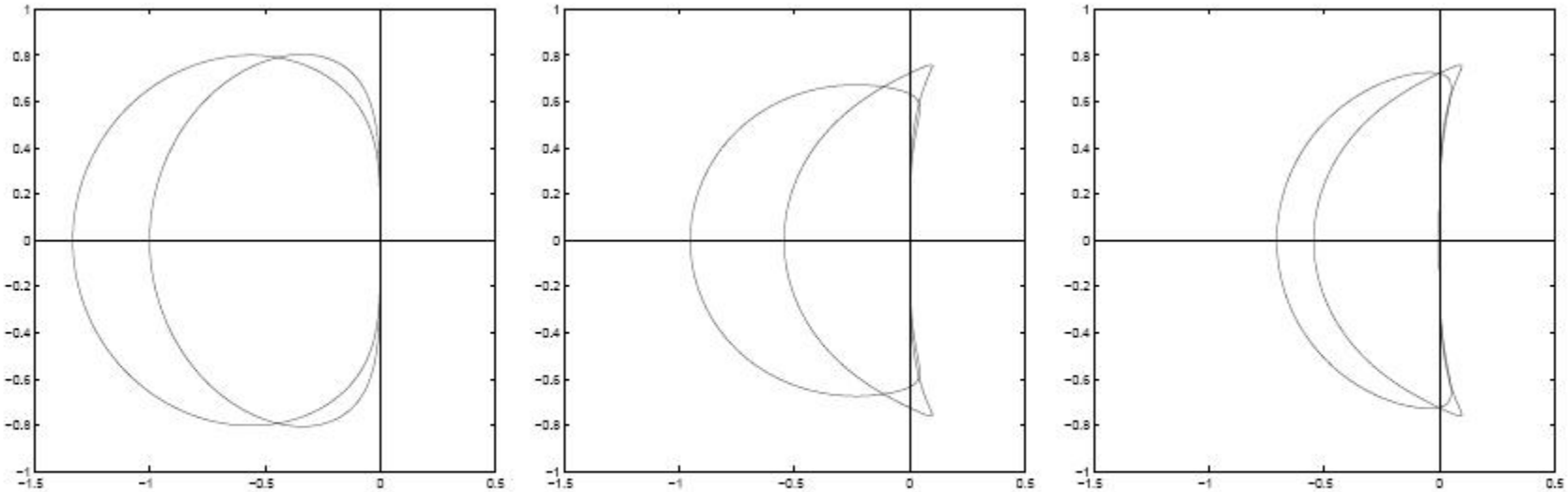


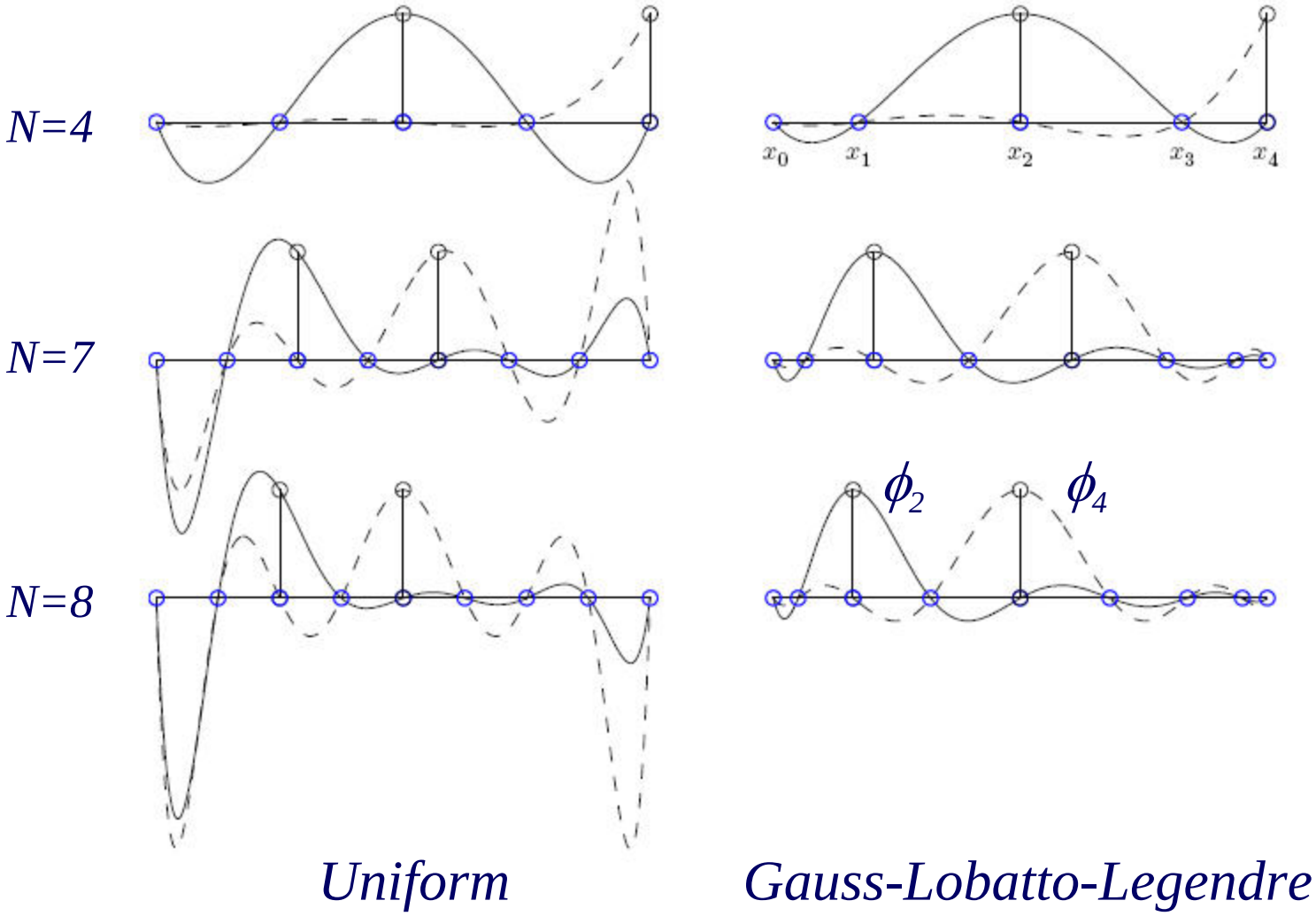
Figure 1: Stability regions for (left) AB2 and BDF2/EXT2, (center) AB3 and BDF3/EXT3, and (right) AB3 and BDF2/EXT2a.

Fornberg's Polynomial Interpolant/Derivative Code

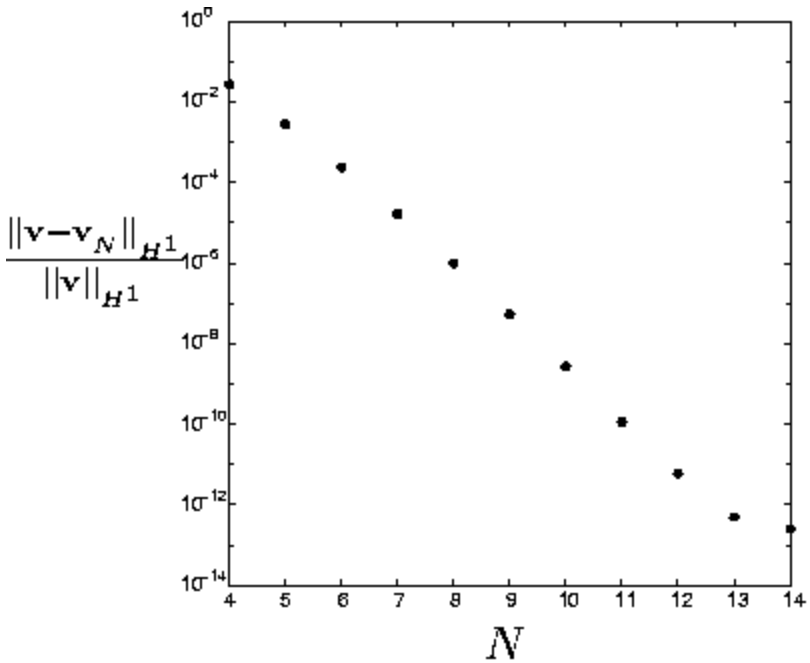
- Code available in f77 and matlab – short, stable, fast

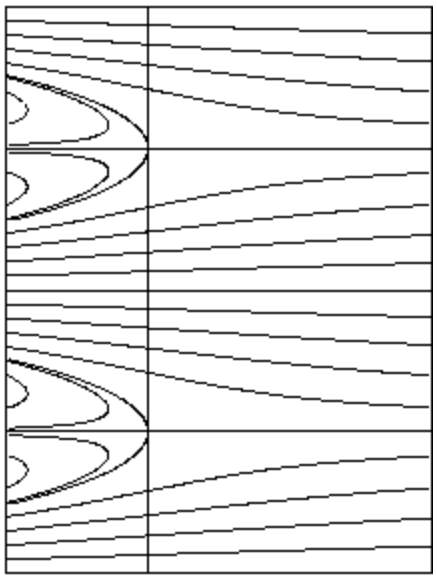
```
c-----------------------------------------------------------------------
      subroutine fd_weights_full(xx,x,n,m,c)
c
c     This routine evaluates the derivative based on all points
c     in the stencils.  It is more memory efficient than "fd_weights"
c
c     This set of routines comes from the appendix of
c     A Practical Guide to Pseudospectral Methods, B. Fornberg
c     Cambridge Univ. Press, 1996.   (pff)
c
c     Input parameters:
c       xx -- point at wich the approximations are to be accurate
c       x  -- array of x-ordinates:   x(0:n)
c       n  -- polynomial degree of interpolant (# of points := n+1)
c       m  -- highest order of derivative to be approxxmated at xi
c
c     Output:
c       c  -- set of coefficients c(0:n,0:m).
c             c(j,k) is to be applied at x(j) when
c             the kth derivative is approxxmated by a
c             stencil extending over x(0),x(1),...x(n).
c
c
      real x(0:n),c(0:n,0:m)
c
      c1       = 1.
      c4       = x(0) - xx
c
```

Lagrange Polynomials: Good and Bad Nodal Point Distributions



Spectral Element Convergence: Exponential with N





Exact Navier-Stokes solution due to Kovazsnay(1948):

$$
\begin{aligned}
v_x &= 1 - e^{\lambda x}\cos 2\pi y \\
v_y &= \frac{\lambda}{2\pi} e^{\lambda x}\sin 2\pi y \\
\lambda &:= \frac{Re}{2} - \sqrt{\frac{Re^2}{4} + 4\pi^2}
\end{aligned}
$$

Lagrangian Bases: Piecewise Linear and Quadratic

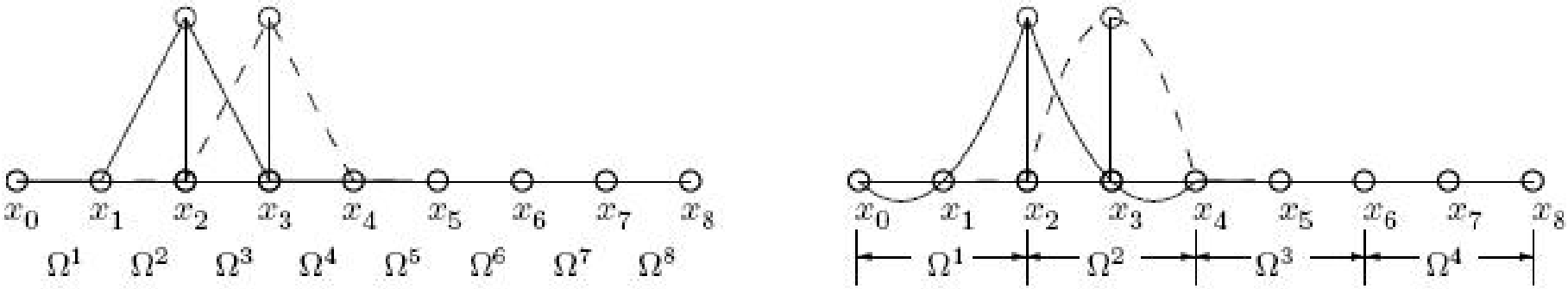


Figure 2: Examples of one-dimensional piecewise linear (left) and piecewise quadratic (right) Lagrangian basis functions, $\phi_2(x)$ and $\phi_3(x)$, with associated element support, Ω^e, $e = 1, \ldots, E$.

- Linear case results in A being tridiagonal (b.w. = 1)
- Q: What is matrix bandwidth for piecewise quadratic case?

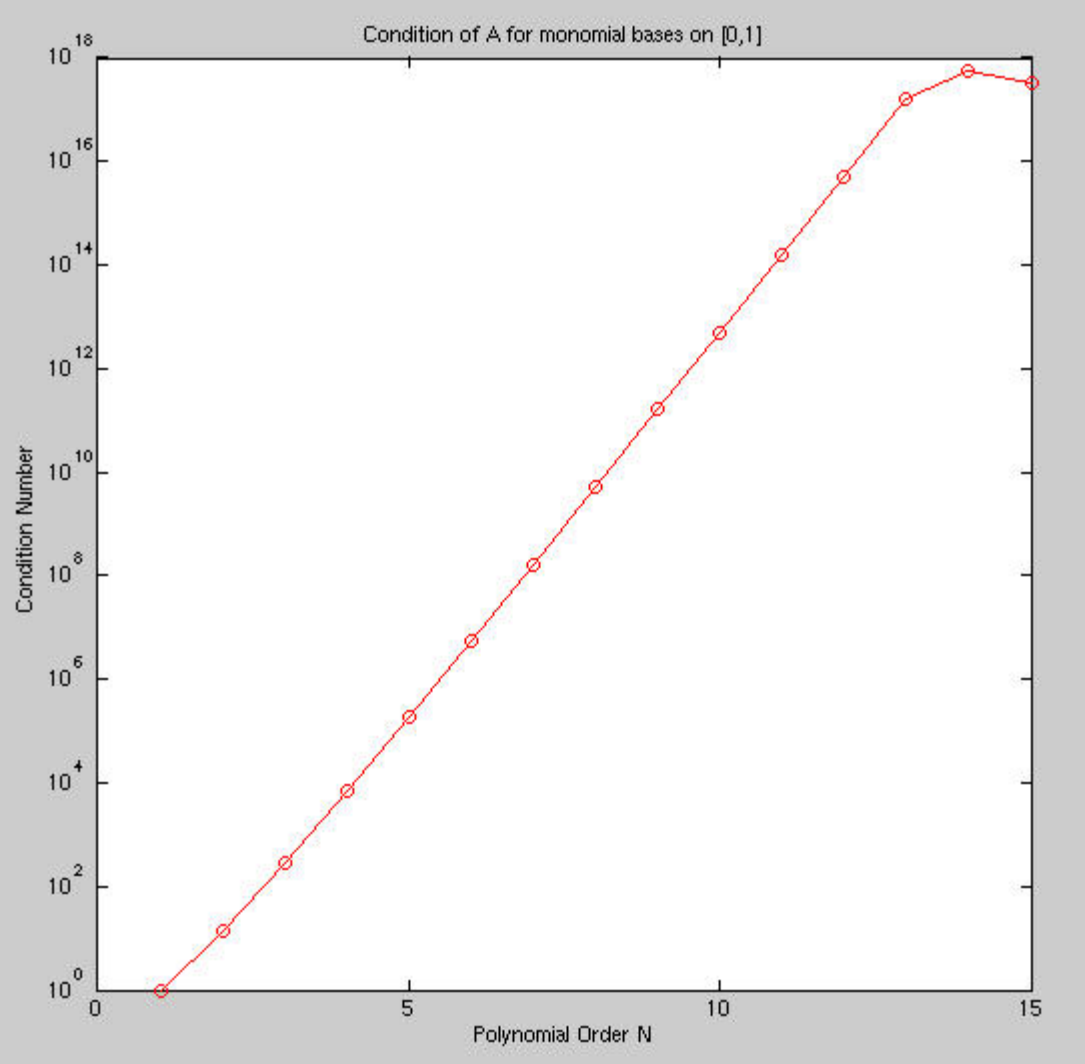
Modal Basis Example

- *Bad basis choice:*

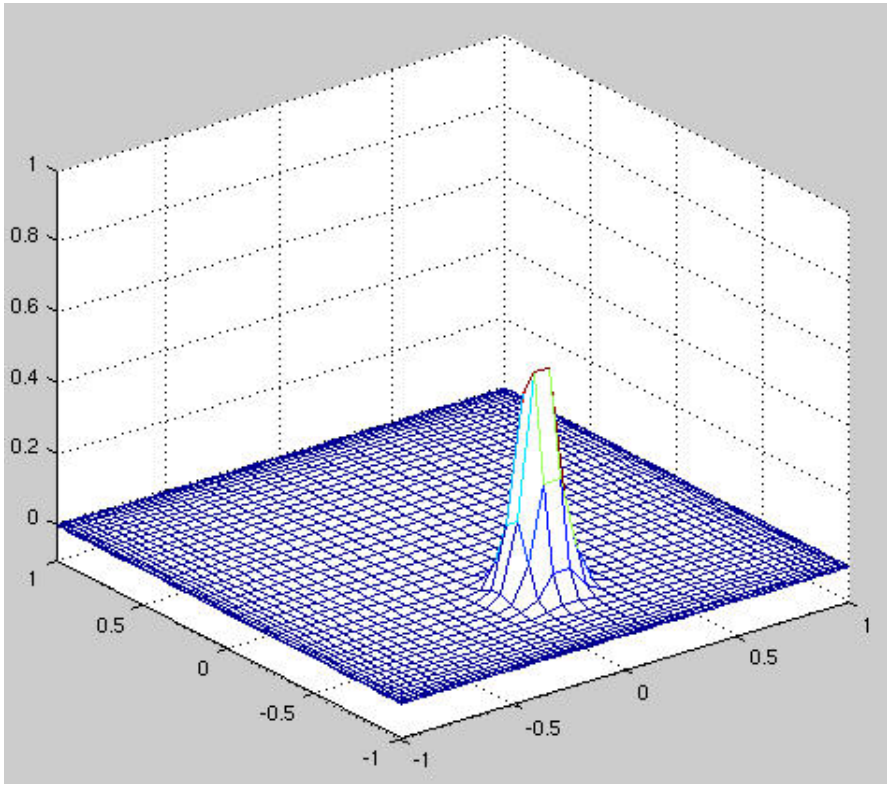
  x^i, $i=1,\ldots,N$

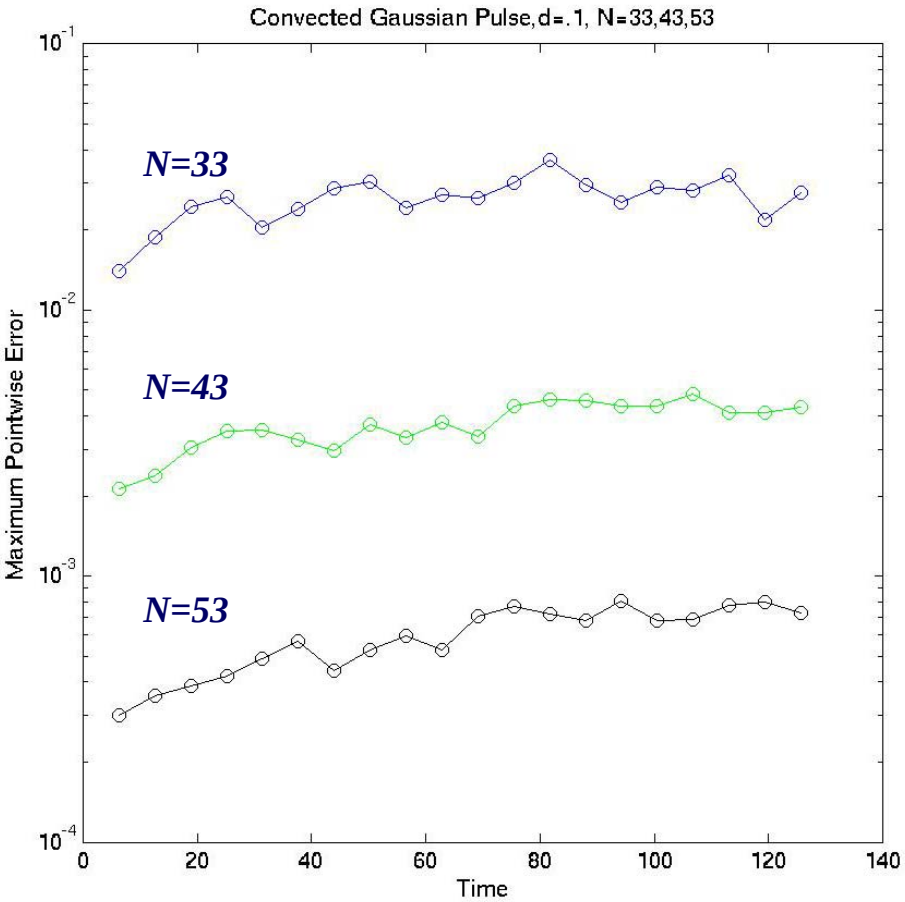
- $A_{ij} = ij / (i+j-1)$

- *Condition number of A grows exponentially with N*



Long-Time Integration: Plane Rotation of Gaussian Pulse





Spectral Element Discretization

- Convection-Diffusion Equation:

$$u_t + \mathbf{c} \cdot \nabla u = \nu \nabla^2 u$$

- Weighted Residual Formulation:

Find $u \in X_0^N \subset H_0^1$ *such that*

$$(v, u_t)_N + (v, \mathbf{c} \cdot \nabla u)_M = \nu (\nabla v, \nabla u)_N \quad \forall v \in X_0^N,$$

- Inner products – integrals replaced by Gaussian quadrature:

  - $(f, g)_M := \sum_{j=0}^{M} \rho_j^M f(\xi_j^M) g(\xi_j^M), \quad$ (1-D, $\Omega = [-1, 1]$)

  - ξ_j^M, ρ_j^M—Mth-order Gauss-Legendre points, weights.

Spectral Element Basis Functions

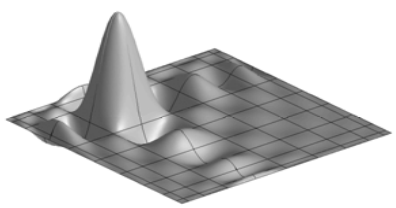
- Nodal basis:

$$u(x,y)|_{\Omega^e} = \sum_{i=0}^{N} \sum_{j=0}^{N} u_{ij}^e \, h_i(r) \, h_j(s)$$

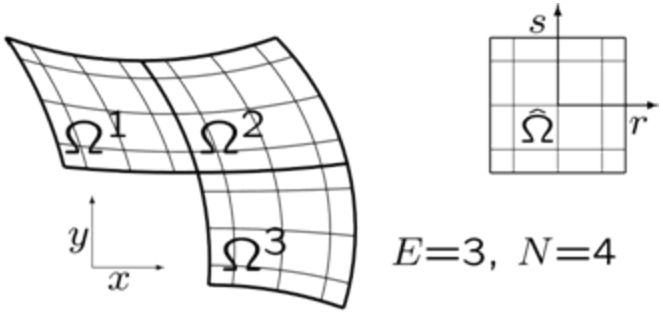
$$h_i(r) \in \mathcal{P}_N(r), \qquad h_i(\xi_j) = \delta_{ij}$$

- ξ_j = Gauss-Lobatto-Legendre quadrature points:

  - stability (*not* uniformly distributed points)
  - allows pointwise quadrature (for *most* operators…)
  - easy to implement BCs and C^0 continuity

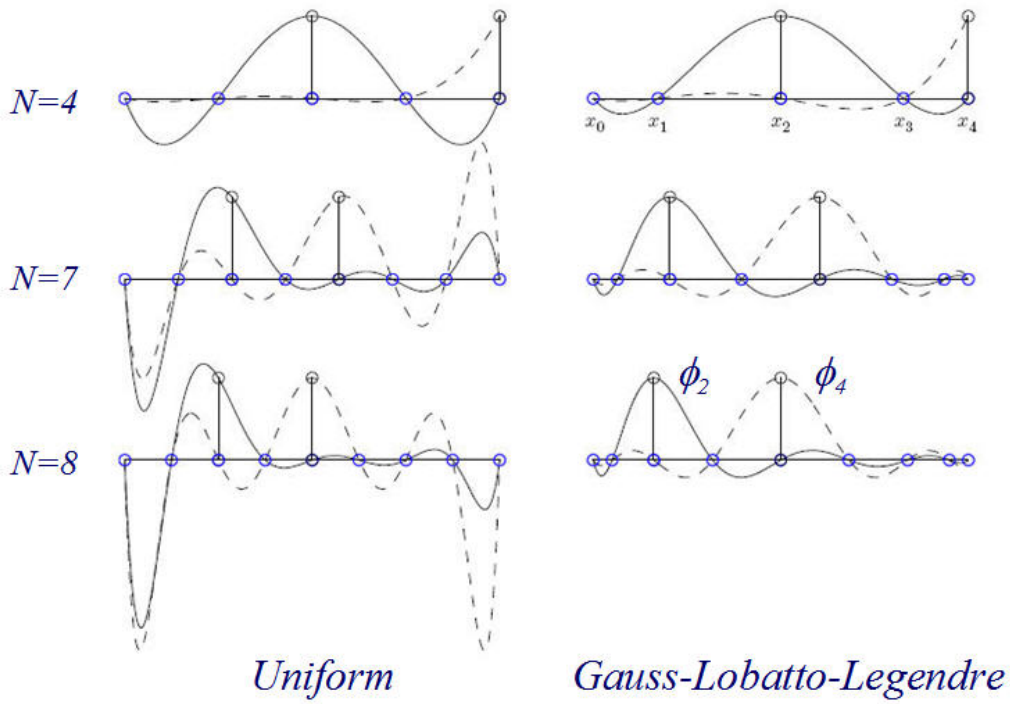


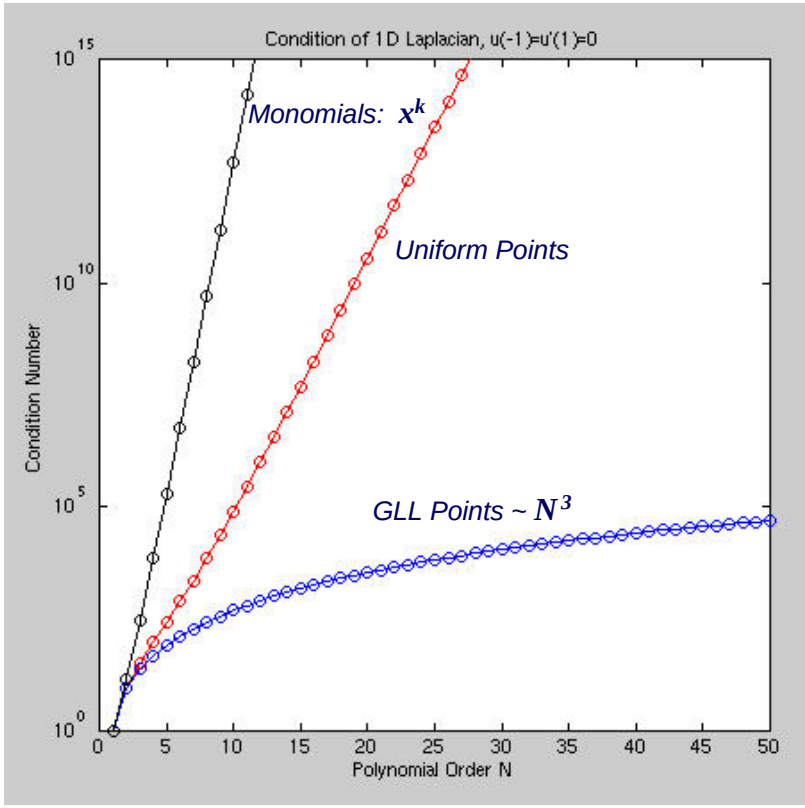
2D basis function, N=10



Stable Lagrange Interpolants

GLL Nodal Basis → good conditioning, minimal round-off



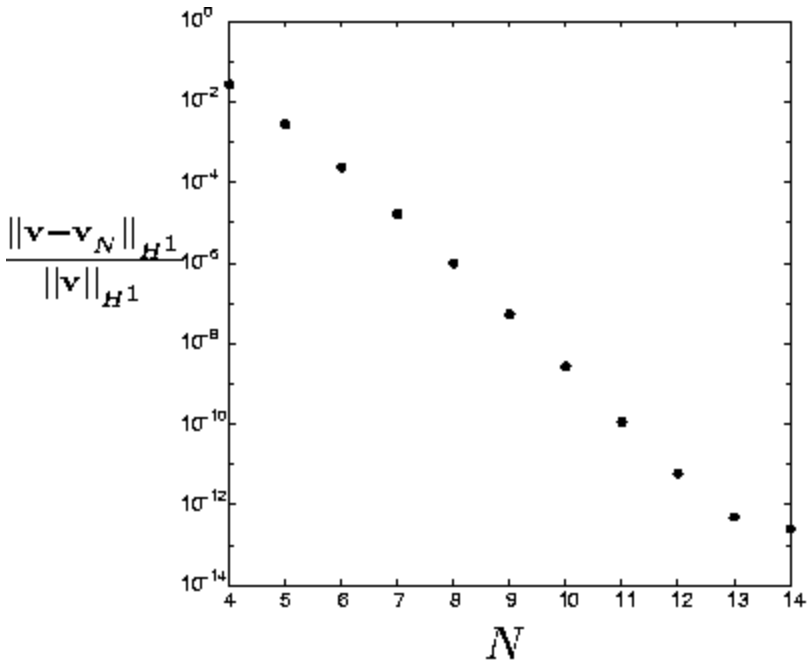


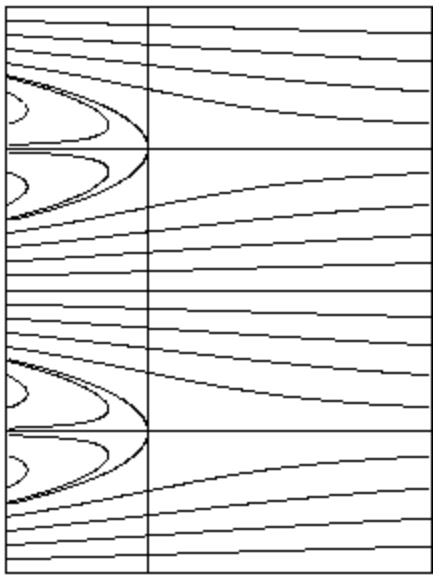
Accuracy

+

Costs

Spectral Element Convergence: Exponential with N





Exact Navier-Stokes solution due to Kovazsnay(1948):

$$
\begin{aligned}
v_x &= 1 - e^{\lambda x}\cos 2\pi y \\
v_y &= \frac{\lambda}{2\pi} e^{\lambda x}\sin 2\pi y \\
\lambda &:= \frac{Re}{2} - \sqrt{\frac{Re^2}{4} + 4\pi^2}
\end{aligned}
$$

Influence of Scaling on Discretization

Large problem sizes enabled by peta- and exascale computers allow propagation of small features (size λ) over distances L >> λ.

- Dispersion errors accumulate linearly with time:

  ~|*correct speed – numerical speed*| * *t* (for each wavenumber)

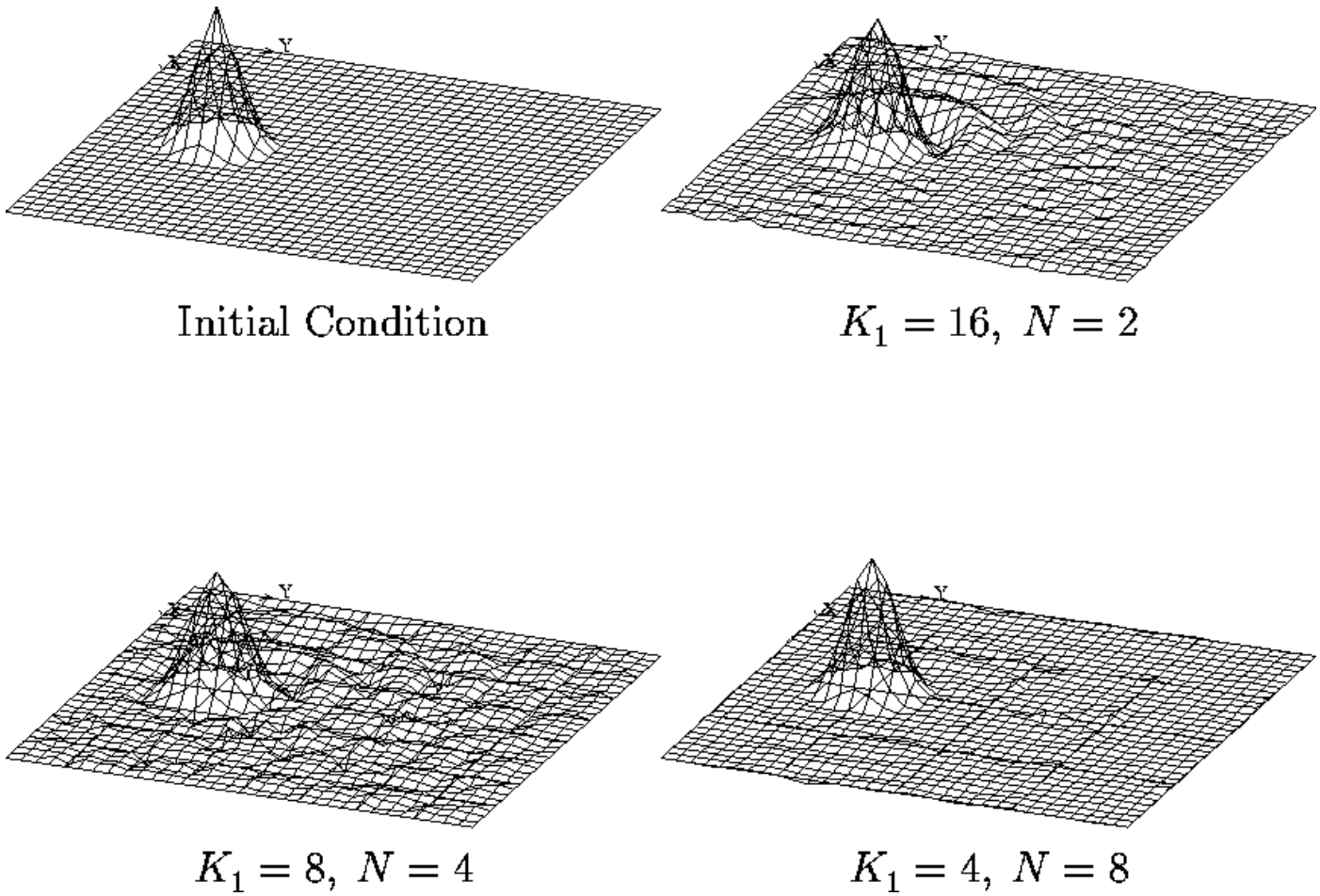
  → $error_{t_final} \sim (L / \lambda) * |$ *numerical dispersion error* |

- For fixed final error ε_f, require: numerical dispersion error ~ $(\lambda / L)\varepsilon_f$, << 1

- High-order methods most efficiently deliver small dispersion errors

  (Kreiss & Oliger 72, Gottlieb et al. 2007)

Excellent transport properties, even for *non-smooth* solutions

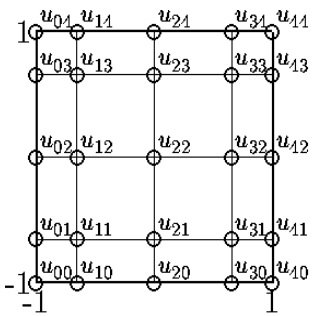


Convection of non-smooth data on a 32x32 grid (K_1 x K_1 spectral elements of order N).

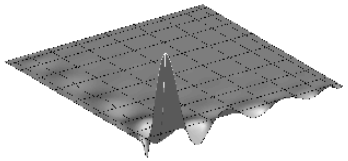
(cf. Gottlieb & Orszag 77)

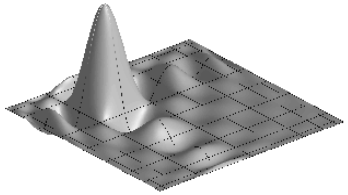
Costs

- **Cost dominated by iterative solver costs, *proportional to***
  - iteration count
  - matrix-vector product + preconditioner cost

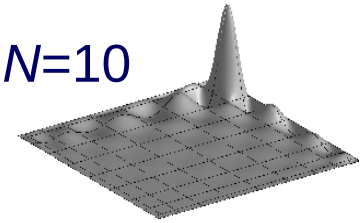


N=4





N=10



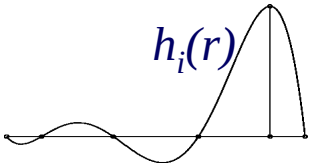
- **Locally-structured tensor-product forms:**
  - minimal indirect addressing
  - fast matrix-free operator evaluation
  - fast local operator inversion via fast diagonalization method (FDM)

    (*Approximate, when element deformed.*)

Tensor-Product Forms: Matrix-Matrix Based Derivative Evaluation

- *Local tensor-product form* (2D),

$$u(r,s) = \sum_{i=0}^{N} \sum_{j=0}^{N} u_{ij} h_i(r) h_j(s), \quad h_i(\xi_p) = \delta_{ip},\ h_i \in \mathbb{P}_N$$



allows derivatives to be evaluated as matrix-matrix products:

$$\left.\frac{\partial u}{\partial r}\right|_{\xi_i,\xi_j} = \sum_{p=0}^{N} u_{pj} \left.\frac{dh_p}{dr}\right|_{\xi_i} = \sum_p \underbrace{\hat{D}_{ip} u_{pj}}_{mxm} =: D_r \underline{u}$$

- *Memory access scales only as O(n)*
- *Work scales as N*O(n), but CPU time is weakly dependent on N* (***WHY?***)

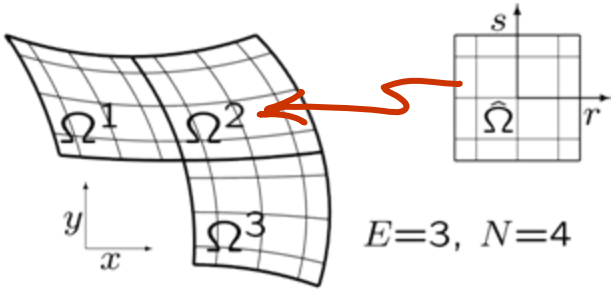
Local "Matrix-Free" Stiffness Matrix in 3D

- For a deformed spectral element, Ω^k,

$$A^k \underline{u}^k = \begin{pmatrix} D_r \\ D_s \\ D_t \end{pmatrix}^T \begin{pmatrix} G_{rr} & G_{rs} & G_{rt} \\ G_{rs} & G_{ss} & G_{st} \\ G_{rt} & G_{st} & G_{tt} \end{pmatrix} \begin{pmatrix} D_r \\ D_s \\ D_t \end{pmatrix} \underline{u}^k$$

$$D_r = (I \otimes I \otimes \hat{D}) \qquad G_{rs} = J \circ B \circ \left(\frac{\partial r}{\partial x}\frac{\partial s}{\partial x} + \frac{\partial r}{\partial y}\frac{\partial s}{\partial y} + \frac{\partial r}{\partial z}\frac{\partial s}{\partial z}\right)$$

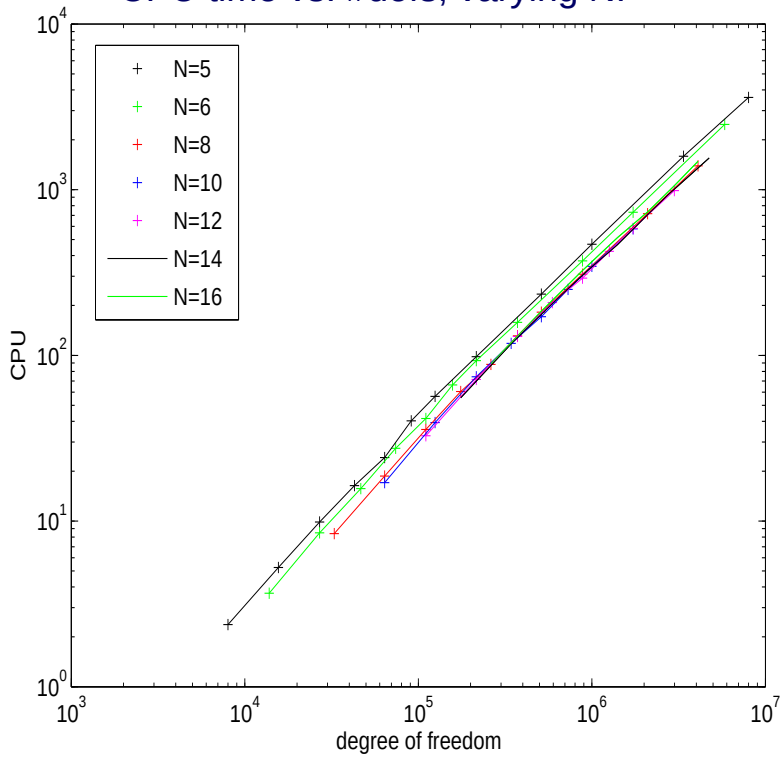
- Operation count is only $O(N^4)$ not $O(N^6)$ [*Orszag '80*]
- Memory access is 7 x number of points (G_{rr}, G_{rs}, etc., are *diagonal*)
- Work is dominated by (fast) matrix-matrix products involving D_r, D_s, etc.



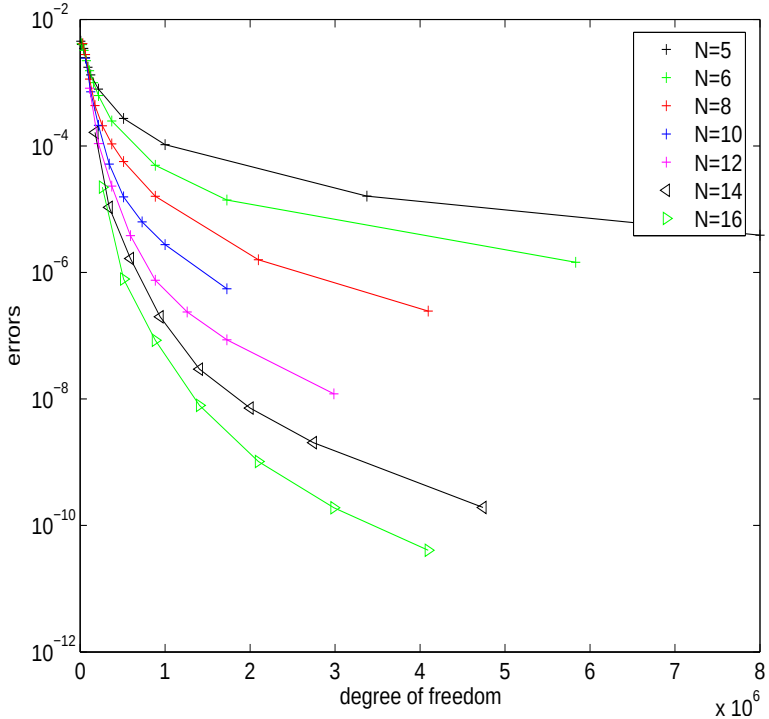
Cost vs. Accuracy: Electromagnetics Example

- For SEM, ***memory*** scales as number of gridpoints, n.
- Work scales as nN, but is in form of (*fast*) matrix-matrix products.

CPU time vs. #dofs, varying N.



Error vs. #dofs, varying N



Periodic Box; 32 nodes, each with a 2.4 GHz Pentium Xeon

Stability

Stabilizing High-Order Methods

In the absence of eddy viscosity, some type of stabilization is generally required at high Reynolds numbers.

Some options:

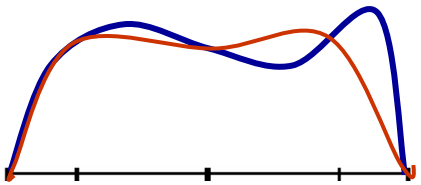
- high-order upwinding (e.g., DG, WENO)
- bubble functions
- spectrally vanishing viscosity
- *filtering*
- *dealiasing*

Filter-Based Stabilization

(Gottlieb et al., Don et al., Vandeven, Boyd, ...)

- *At end of each time step:*
  - Interpolate u onto GLL points for P_{N-1}
  - Interpolate back to GLL points for P_N

$$F_1(u) = I_{N-1}\, u$$



  - Results are smoother with linear combination: *(F. & Mullen 01)*

$$F_\alpha(u) = (1-\alpha)\, u + \alpha\, I_{N-1}\, u \qquad (\alpha \sim 0.05 - 0.2)$$

- *Post-processing — no change to existing solvers*
- *Preserves interelement continuity and spectral accuracy*
- *Equivalent to multiplying by (1-α) the Nth coefficient in the expansion*
  - $u(x) = \Sigma\, u_k\, \phi_k(x) \quad \rightarrow \quad u^*(x) = \Sigma\, \sigma_k u_k\, \phi_k(x), \;\; \sigma_\kappa = 1, \;\; \sigma_N = (1-\alpha)$
  - $\phi_k(x) := L_k(x) - L_{k-2}(x)$ *(Boyd 98)*

Numerical Stability Test: Shear Layer Roll-Up

(Bell et al. JCP 89, Brown & Minion, JCP 95, F. & Mullen, CRAS 2001)

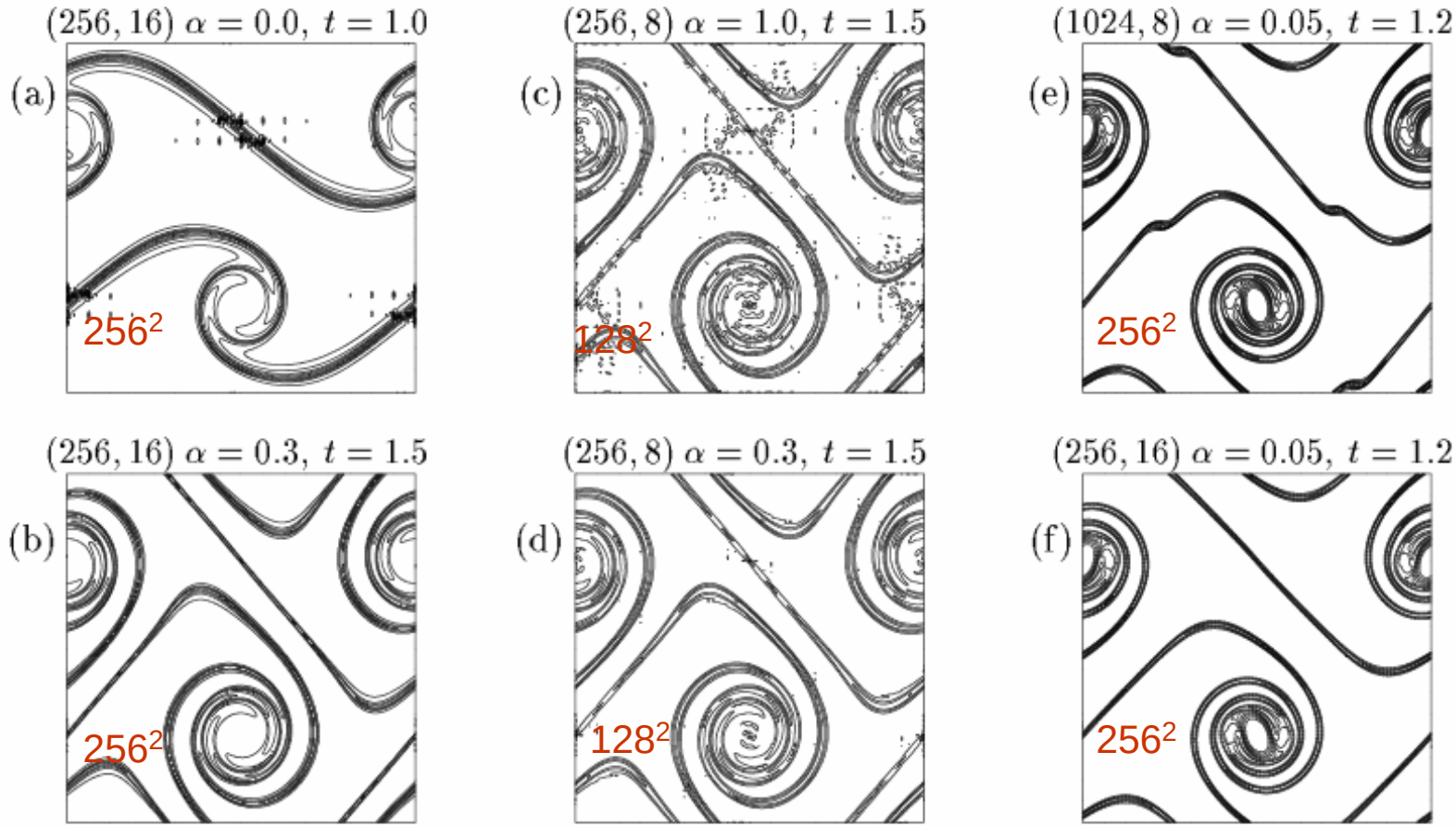


Figure 1: Vorticity for different (K, N) pairings: (a–d) $\rho = 30$, $Re = 10^5$, contours from -70 to 70 by 140/15; (e–f) $\rho = 100$, $Re = 40,000$, contours from -36 to 36 by 72/13. (cf. Fig. 3c in [4]).

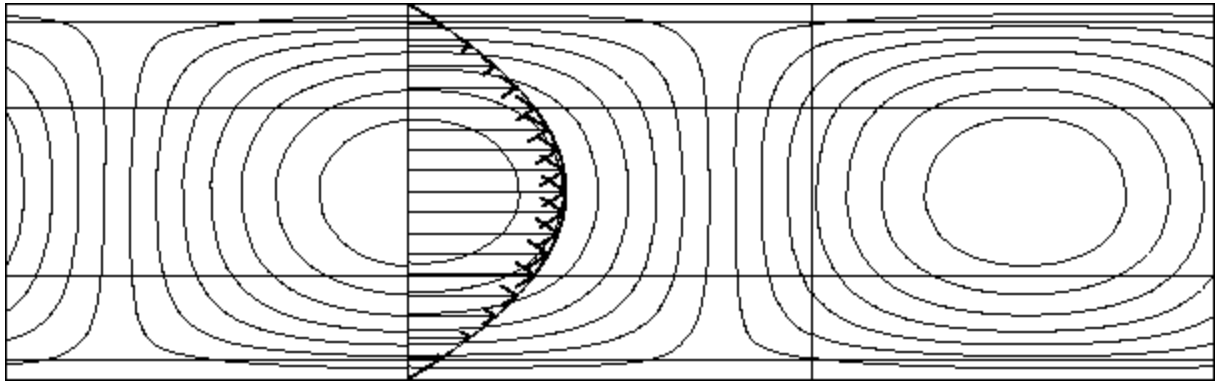
Error in Predicted Growth Rate for Orr-Sommerfeld Problem at Re=7500

(*Malik & Zang 84*)

Spatial and Temporal Convergence

(F. & Mullen, 01)

$\Delta t = 0.003125$			$N = 17$	2nd Order		3rd Order	
N	$\alpha = 0.0$	$\alpha = 0.2$	Δt	$\alpha = 0.0$	$\alpha = 0.2$	$\alpha = 0.0$	$\alpha = 0.2$
7	0.23641	0.27450	0.20000	0.12621	0.12621	171.370	0.02066
9	0.00173	0.11929	0.10000	0.03465	0.03465	0.00267	0.00268
11	0.00455	0.01114	0.05000	0.00910	0.00911	161.134	0.00040
13	0.00004	0.00074	0.02500	0.00238	0.00238	1.04463	0.00012
15	0.00010	0.00017	0.01250	0.00065	0.00066	0.00008	0.00008



Base velocity profile and perturbation streamlines

Filtering permits $Re_{\delta 99}$ > 700 for transitional boundary layer calculations

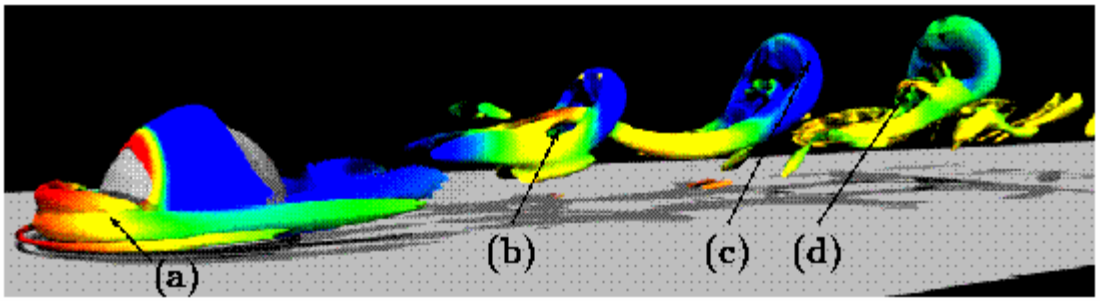
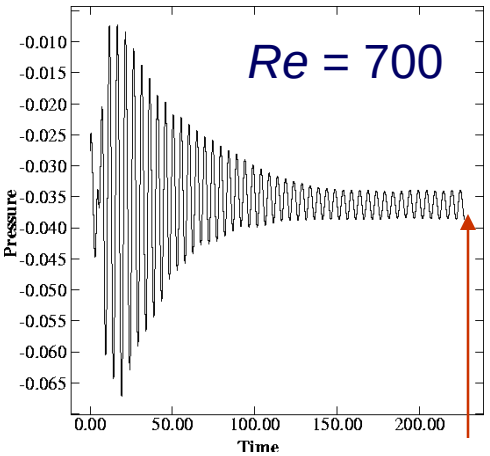


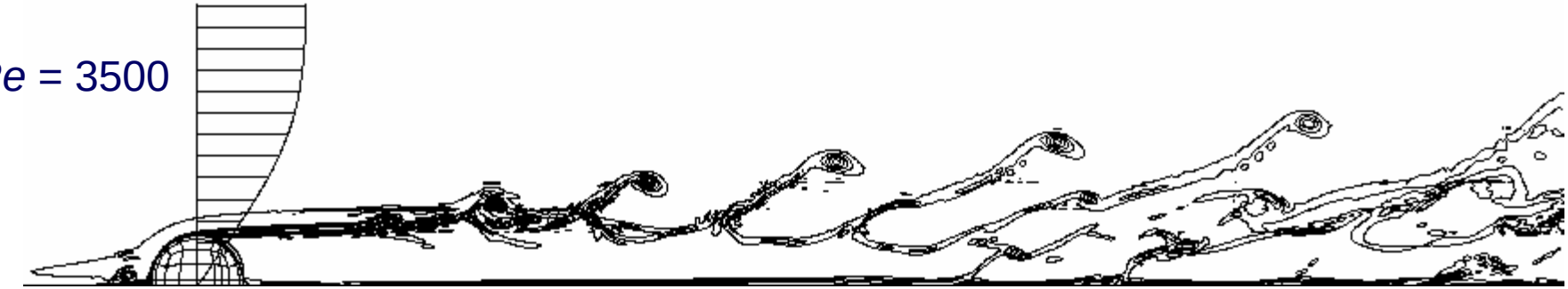
Figure 1: Principal vortex structures identified by $\lambda_2 = -1$ isosurfaces at $Re_\delta = 760$: standing horseshoe vortex (a), interlaced tails (b), hairpin head (c), and bridge (d). Colors indicate pressure. (K=1021, N = 15).



blow up

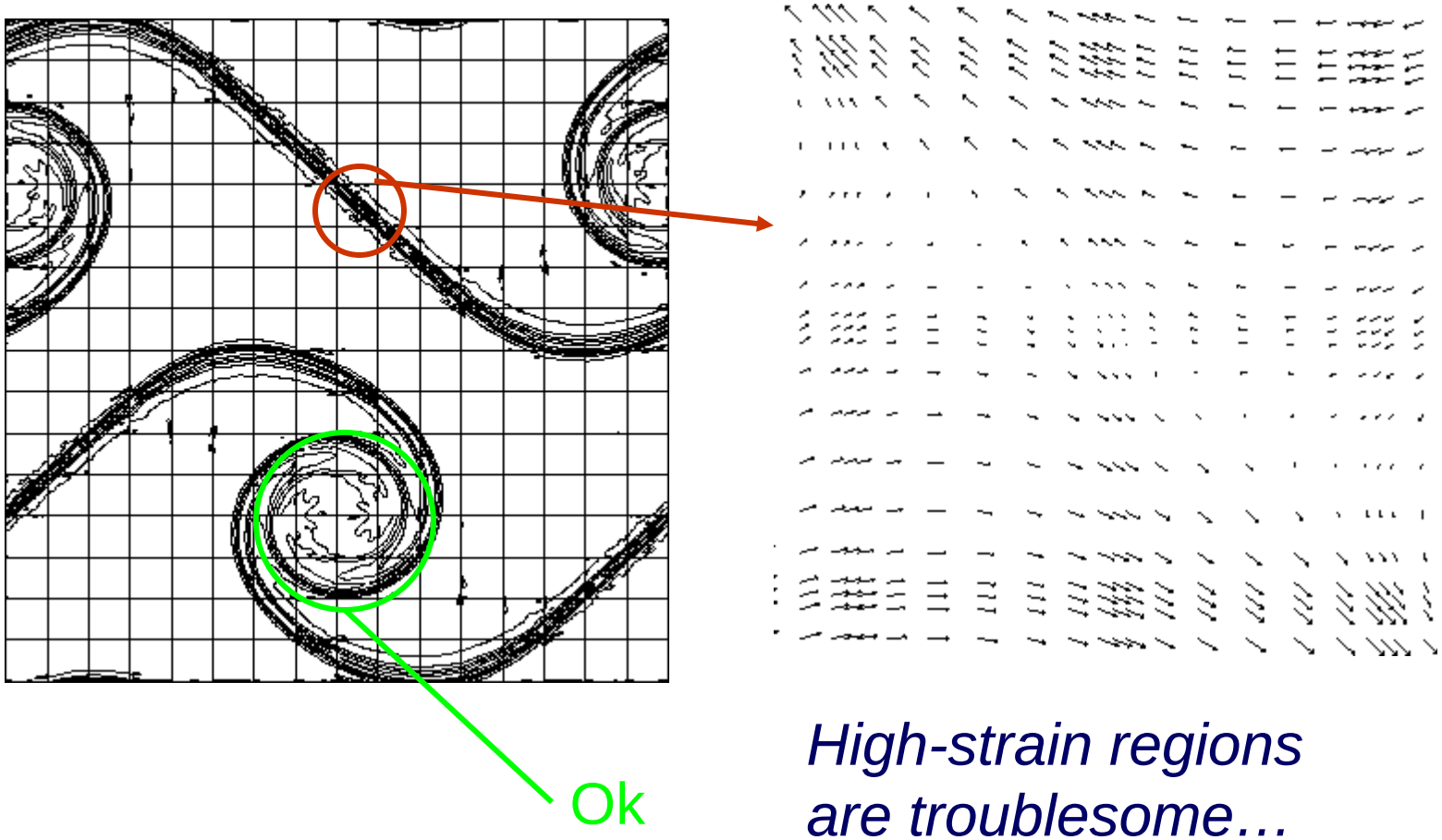
Re = 1000

Re = 3500



Why Does Filtering Work ?
(*Or, Why Do the Unfiltered Equations Fail?*)

Double shear layer example:



High-strain regions are troublesome…

Why Does Filtering Work ?
(*Or, Why Do the Unfiltered Equations Fail?*)

Consider the model problem: $$\frac{\partial u}{\partial t} = -\mathbf{c} \cdot \nabla u$$

Weighted residual formulation: $$B\frac{d\underline{u}}{dt} = -C\underline{u}$$

$$B_{ij} = \int_\Omega \phi_i \phi_j \, dV = \text{symm. pos. def.}$$

$$\begin{aligned} C_{ij} &= \int_\Omega \phi_i \, \mathbf{c} \cdot \nabla \phi_j \, dV \\ &= -\int_\Omega \phi_j \, \mathbf{c} \cdot \nabla \phi_i \, dV - \int_\Omega \phi_j \phi_j \nabla \cdot \mathbf{c} \, dV \\ &= \text{skew symmetric, if } \nabla \cdot \mathbf{c} \equiv 0. \end{aligned}$$

$$B^{-1}C \longrightarrow \text{imaginary eigenvalues}$$

Discrete problem should never blow up.

Why Does Filtering Work ?
(*Or, Why Do the Unfiltered Equations Fail?*)

Weighted residual formulation vs. spectral element method:

$$C_{ij} = (\phi_i, \mathbf{c} \cdot \nabla \phi_j) = -C_{ji}$$

$$\tilde{C}_{ij} = (\phi_i, \mathbf{c} \cdot \nabla \phi_j)_N \neq -\tilde{C}_{ji}$$

This suggests the use of over-integration (dealiasing) to ensure that skew-symmetry is retained

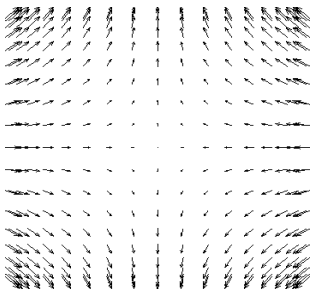
$$C_{ij} = (J\phi_i, (J\mathbf{c}) \cdot J\nabla \phi_j)_M$$

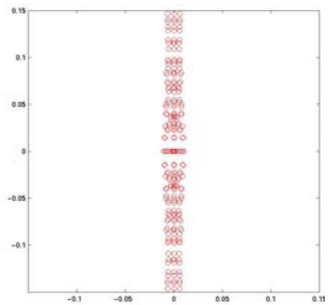
$$J_{pq} := h_q^N(\xi_p^M) \quad \text{interpolation matrix (1D, single element)}$$

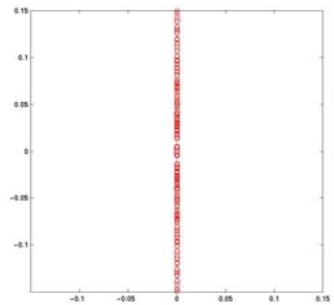
Aliased / Dealiased Eigenvalues: $u_t + \mathbf{c} \cdot \nabla u = 0$

- Velocity fields model first-order terms in expansion of straining and rotating flows.
  - For straining case, $\frac{d}{dt}|u|^2 \sim |\hat{u}_{N\,.}|^2 + |\hat{u}_{.\,N}|^2$
  - Rotational case is skew-symmetric.
  - Filtering attacks the leading-order unstable mode.

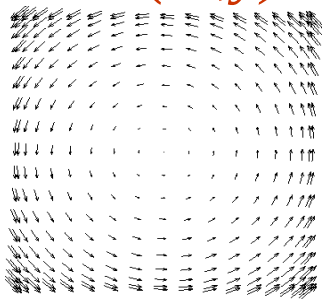
N=19, M=19 *N=19, M=20*

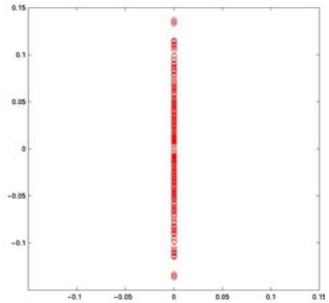


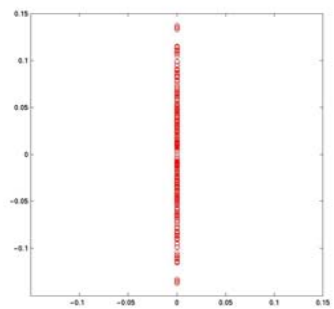




$c = (-x,y)$







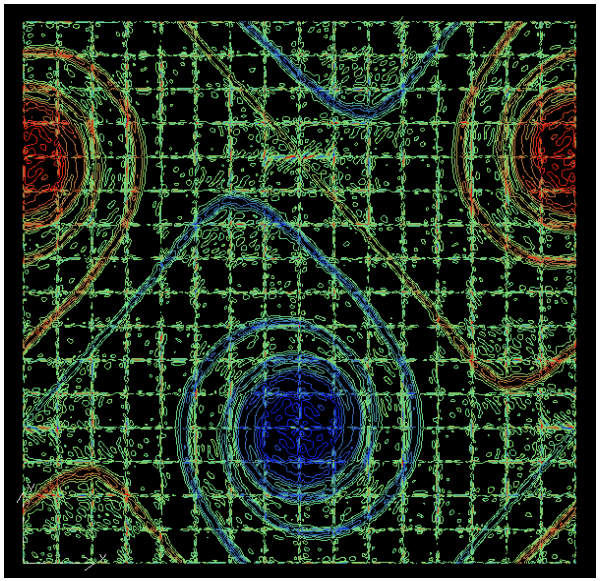
$c = (-y,x)$

Stabilization Summary

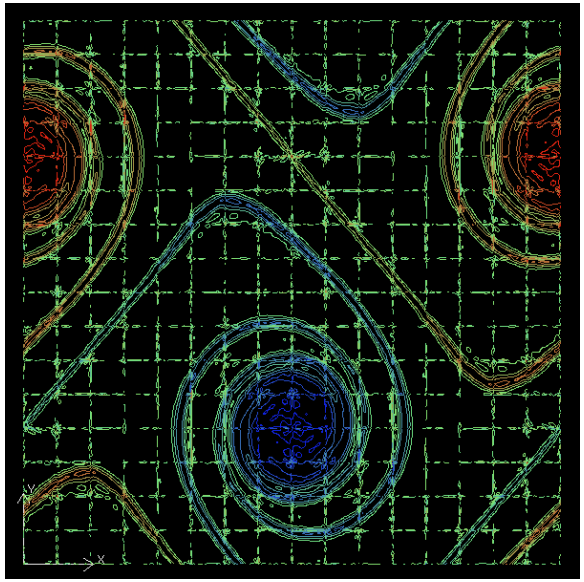
- Filtering acts like well-tuned hyperviscosity
  - Attacks only the fine scale modes (that, numerically speaking, shouldn't have energy anyway…)
  - Can precisely identify which modes in the SE expansion to suppress (unlike differential filters)
  - Does not compromise spectral convergence
- Dealiasing of convection operator recommended for high Reynolds number applications to avoid spurious eigenvalues
  - Can run double shear-layer roll-up problem *forever* with
    - $\nu = 0$,
    - *no filtering*

Dealiased Shear Layer Roll-Up Problem, 128^2

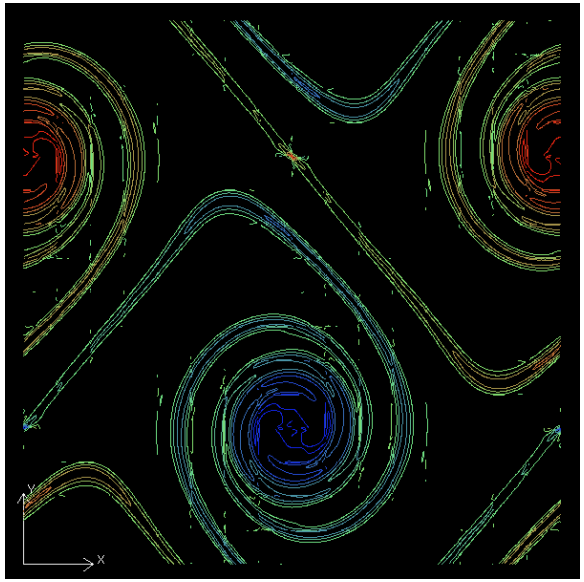
$\nu = 0$, no filter



$\nu = 10^{-5}$, no filter



$\nu = 0$, filter = (.1,.025)



Linear Solvers

Linear Solvers for Incompressible Navier-Stokes

- Navier-Stokes time advancement:

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} = -\nabla P + \frac{1}{Re} \nabla^2 \vec{u}$$

$$-\nabla . \vec{u} = 0$$

- Nonlinear term: *explicit*
  - k th-order backward difference formula / extrapolation
  - characteristics (Pironneau '82, MPR '90)

- Stokes problem: pressure/viscous decoupling:
  - 3 Helmholtz solves for velocity *("easy" w/ Jacobi-precond. CG)*
  - (consistent) Poisson equation for pressure *(computationally dominant)*

P_N - P_{N-2} Spectral Element Method for Navier-Stokes *(MP 89)*

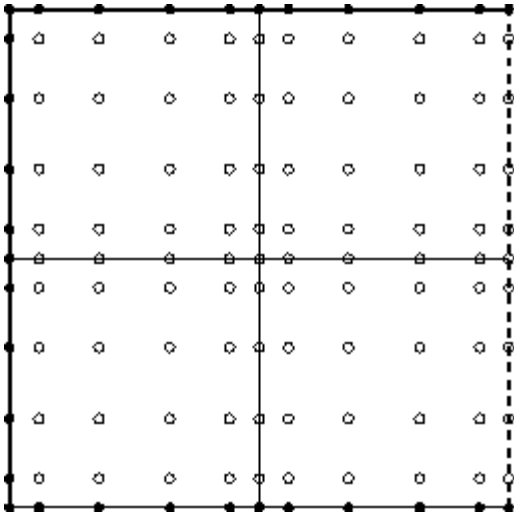
WRT: *Find* $\mathbf{u} \in X^N, p \in Y^N$ *such that:*

$$\frac{1}{Re}(\nabla \mathbf{u}, \nabla \mathbf{v})_{GL} + \frac{1}{\Delta t}(\mathbf{u}, \mathbf{v})_{GL} - (p, \nabla \cdot \mathbf{v})_G = (\mathbf{f}, \mathbf{v})_{GL} \quad \forall\, \mathbf{v} \in X^N \subset H^1$$

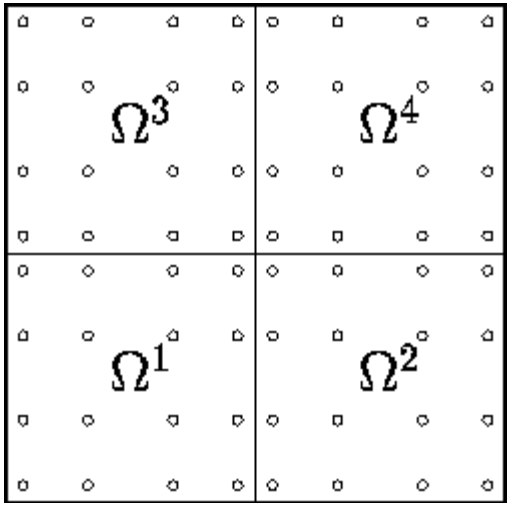
$$- (q, \nabla \cdot \mathbf{u})_G = 0 \qquad \forall\, q \in Y^N \subset L^2$$

Velocity, $\mathbf{u}$ in P_N, continuous

Pressure, p in P_{N-2}, discontinuous



Gauss-Lobatto Legendre points (velocity)



Gauss Legendre points (pressure)

Navier-Stokes Solution Strategy

- Semi-implicit: explicit treatment of nonlinear term.
- Leads to Stokes saddle problem, which is algebraically split

MPR 90, Blair-Perot 93, Couzy 95

$$\begin{bmatrix} \mathbf{H} & -\mathbf{D}^T \\ -\mathbf{D} & 0 \end{bmatrix} \begin{pmatrix} \underline{\mathbf{u}}^n \\ \underline{p}^n - \underline{p}^{n-1} \end{pmatrix} = \begin{pmatrix} \mathbf{B}\underline{\mathbf{f}} + \mathbf{D}^T \underline{p}^{n-1} \\ \underline{f}_p \end{pmatrix}$$

$$\begin{bmatrix} \mathbf{H} & -\frac{\Delta t}{\beta_0}\mathbf{H}\mathbf{B}^{-1}\mathbf{D}^T \\ \mathbf{0} & E \end{bmatrix} \begin{pmatrix} \underline{\mathbf{u}}^n \\ \underline{p}^n - \underline{p}^{n-1} \end{pmatrix} = \begin{pmatrix} \mathbf{B}\underline{\mathbf{f}} + \mathbf{D}^T \underline{p}^{n-1} \\ \underline{g} \end{pmatrix} + \begin{pmatrix} \underline{\mathbf{r}} \\ \underline{0} \end{pmatrix} \quad ,$$

$$E := \frac{\Delta t}{\beta_0}\mathbf{D}\mathbf{B}^{-1}\mathbf{D}^T \quad , \qquad \underline{\mathbf{r}} = O(\Delta t^2)$$

- $\boldsymbol{E}$ - consistent Poisson operator for pressure, SPD
  - Stiffest substep in Navier-Stokes time advancement
  - Most compute-intensive phase
  - Spectrally equivalent to SEM Laplacian, $\boldsymbol{A}$