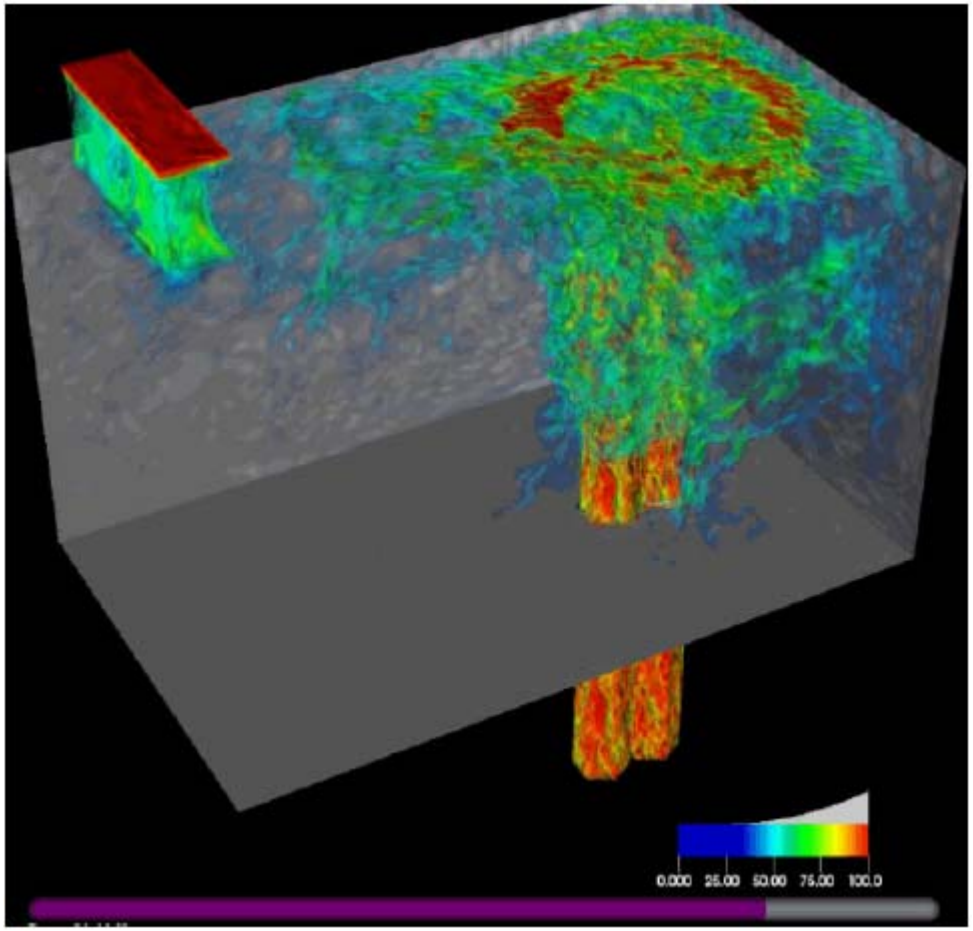
Nek5000 Tutorial

Velocity prediction, ANL MAX experiment.



Paul Fischer
Aleks Obabko
Stefan Kerkemeier
Misun Min
James Lottes
Katie Heisey
Shashi Aithal
Yulia Peet

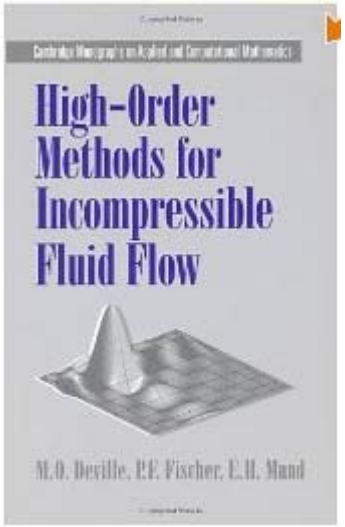
Mathematics and Computer Science Division
Argonne National Laboratory

Outline

- *Nek5000 capabilities*
- *Equations, timestepping, and SEM basics*
- *Parallel / serial issues that you should understand*

Some Resources

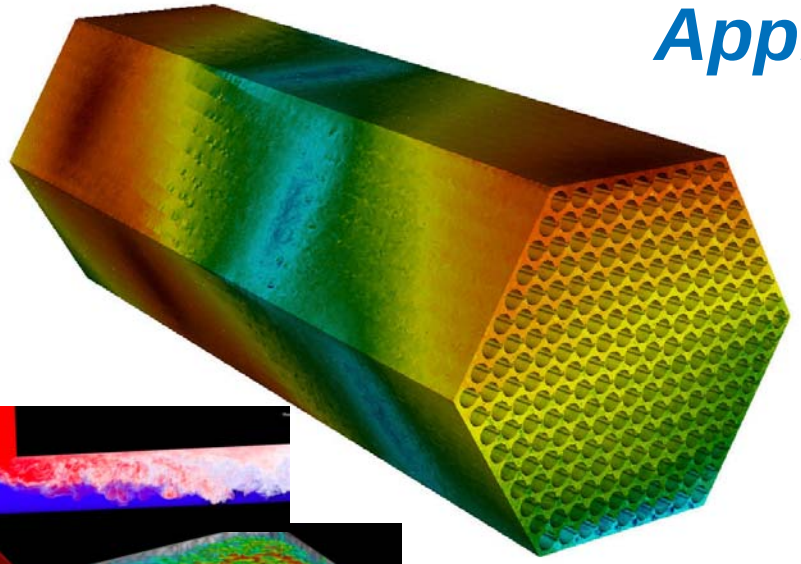
- Code & Algorithm Tips:
  - Nek5000 wiki page (google nek5000)
  - www.mcs.anl.gov/~fischer/Nek5000
- Numerical methods & theory:
  - Deville, Fischer, & Mund, Cambridge Univ. Press, 2002

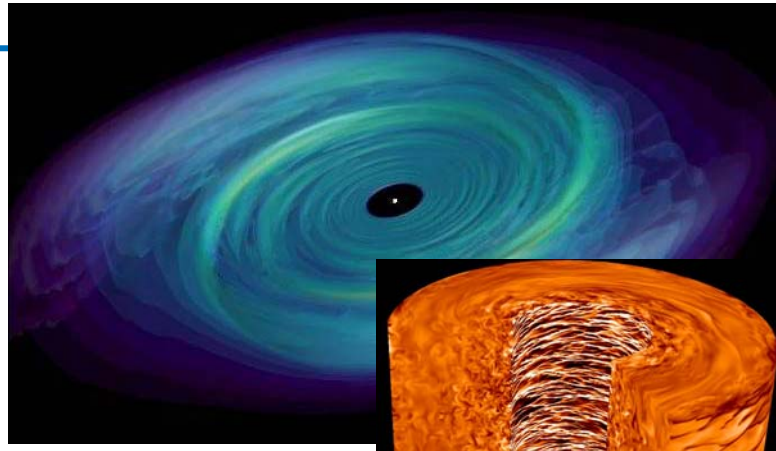


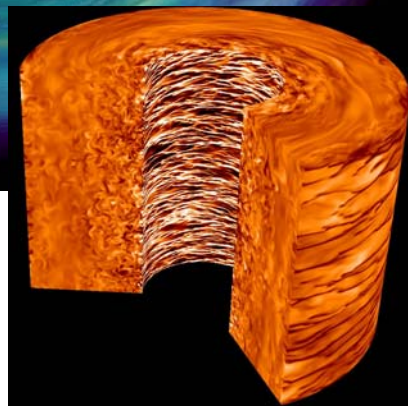
Part I

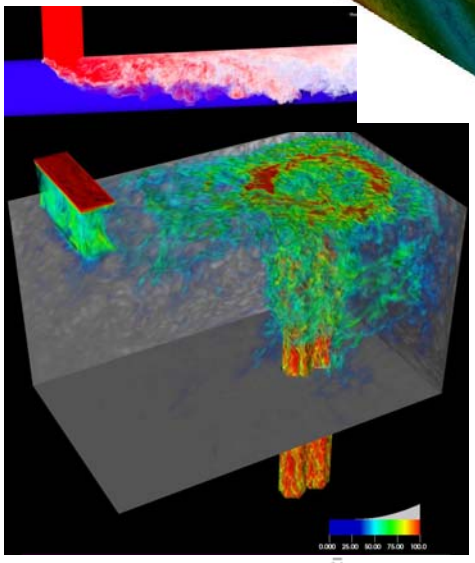
- *Nek5000 capabilities*
  - *Gallery*
  - *Brief history*
  - *Equations solved*
  - *Features overview:*
    - Spectral element discretization
    - Convergence properties (*nek5_svn/examples*)
    - Scalability

Applications



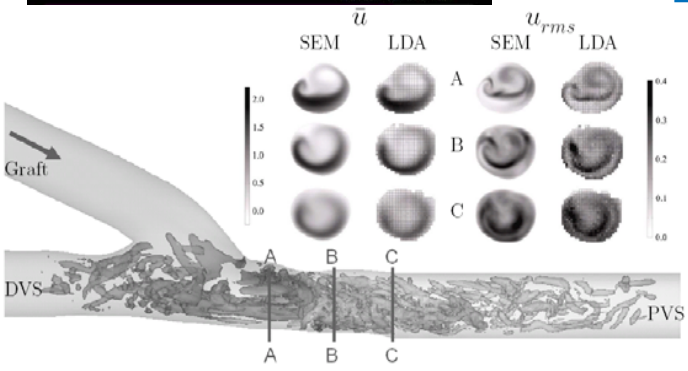


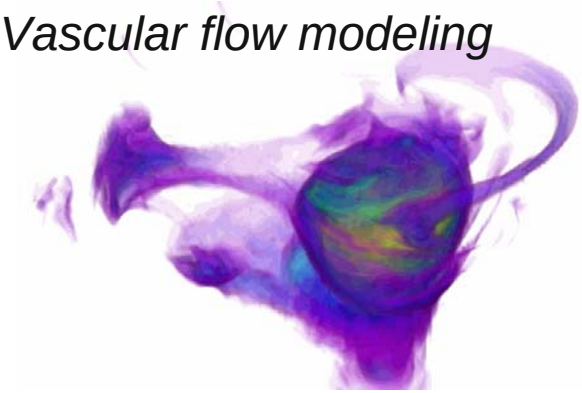


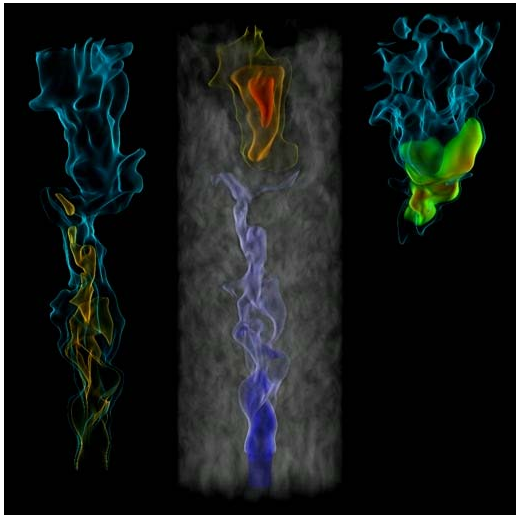


Clockwise from upper left:

- *Reactor thermal-hydraulics*
- *Astrophysics*
- *Combustion*
- *Oceanography*
- *Vascular flow modeling*







Coarse DNS: Channel Flow at Re_b=13,000

Simulations by J. Ohlsson, KTH, Stockholm

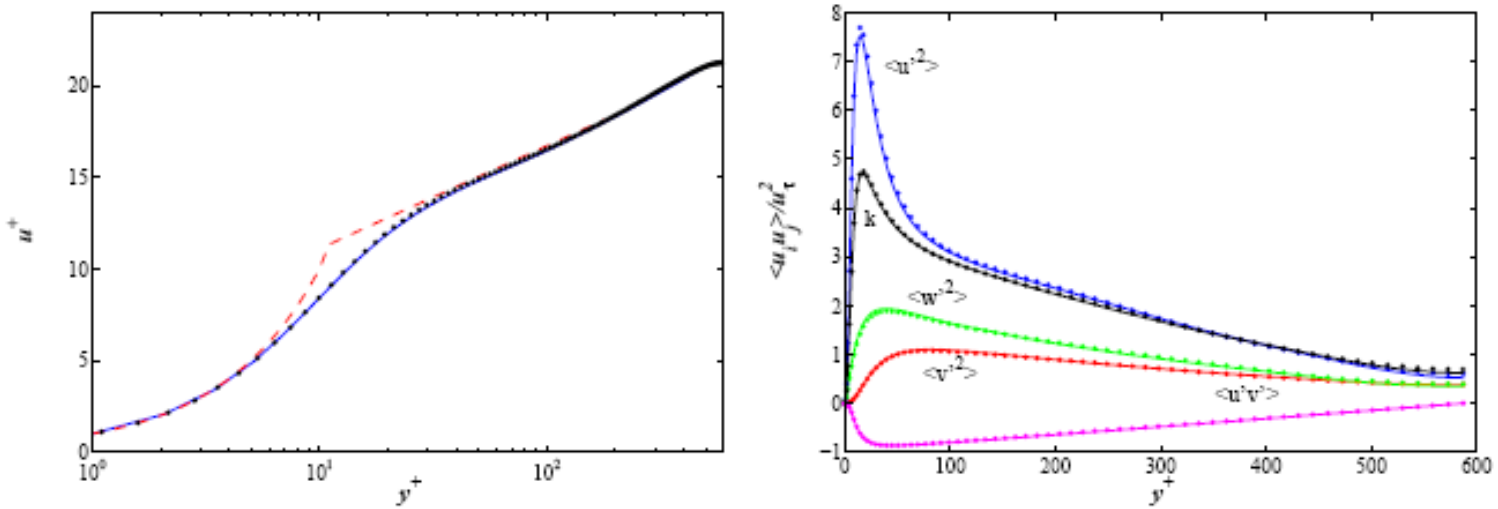
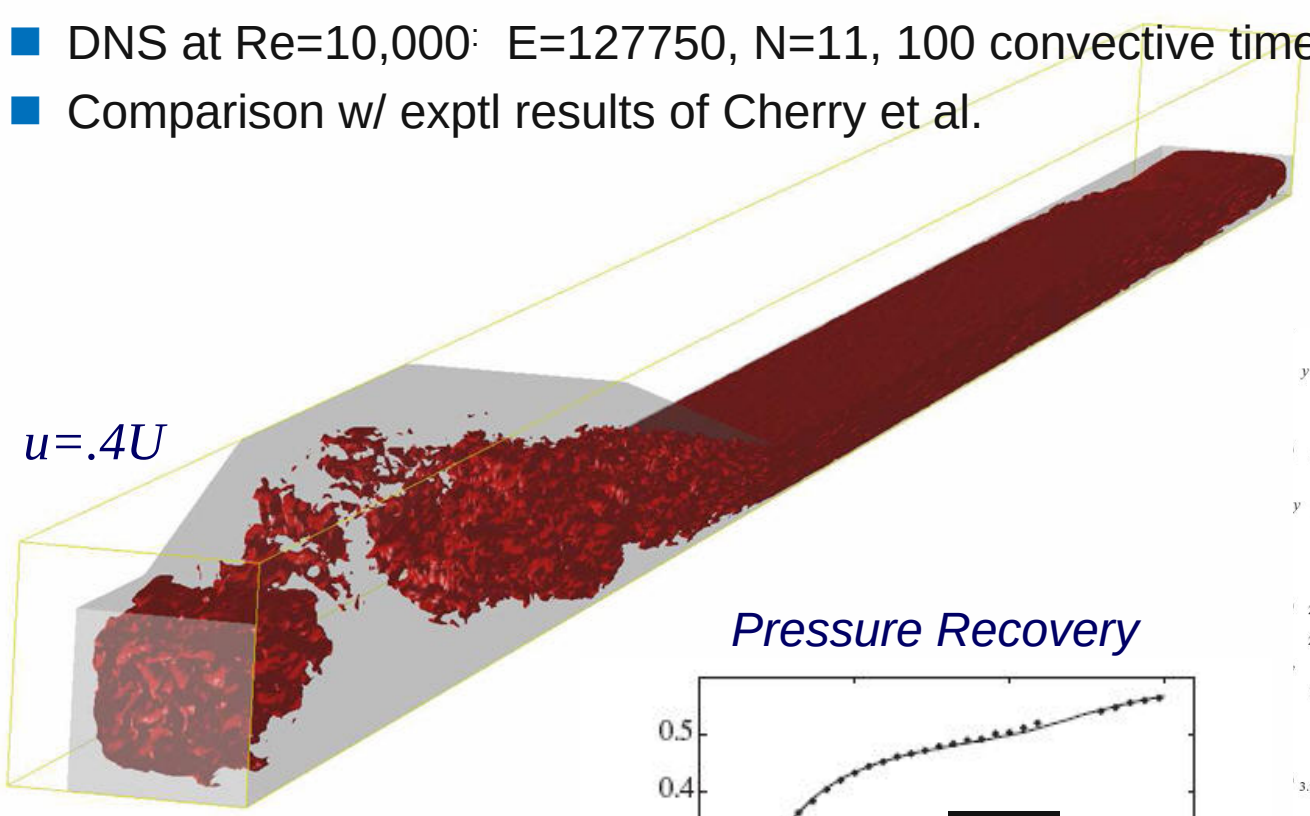


Figure 15: Turbulent channel flow simulations at $Re_\tau = 590$ with polynomial order 15 (a resolution of 288 in the homogenous directions and 192 in the wall-normal direction) showing a) mean velocity profile and b) Reynolds stresses together with the turbulent kinetic energy as defined above. In this case only overintegration was used [Case iii)]. Same time step as for the filtered case. Comparison to DNS results (a resolution of 384 in the homogenous directions and 257 in the wall-normal direction) from Moser et al. [19]. ····· DNS data, ----Log Law, ——Nek5000.

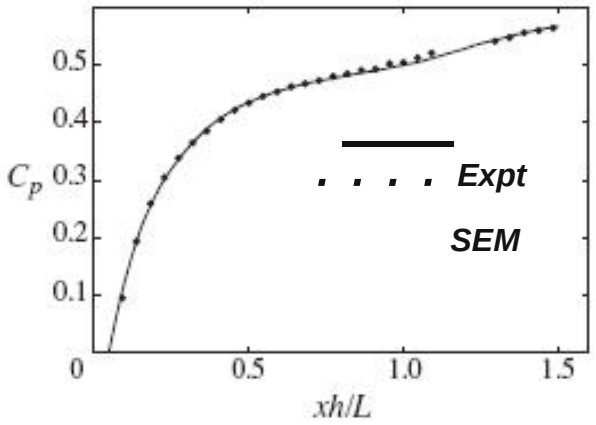
Separation in an Asymmetric Diffuser

Ohlsson, Schlatter, F., and Henningson,, *JFM* (2010)

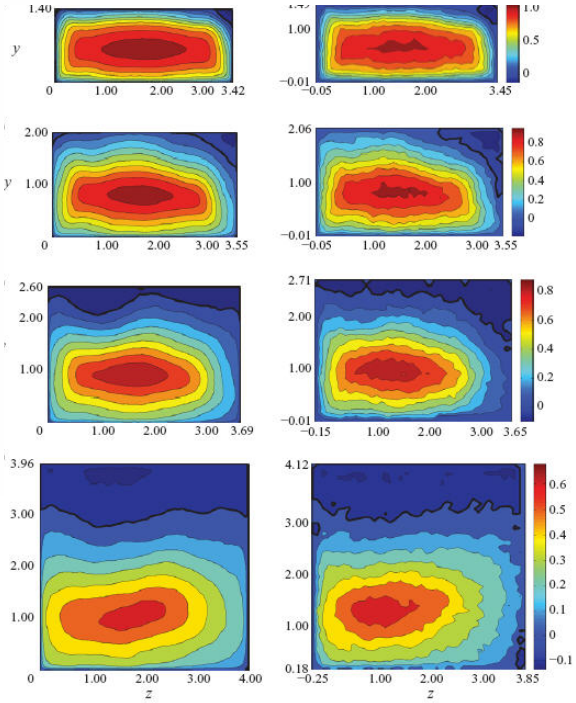
- Flow separation and recovery
- DNS at Re=10,000: E=127750, N=11, 100 convective time units
- Comparison w/ exptl results of Cherry et al.



Pressure Recovery



Axial Velocity

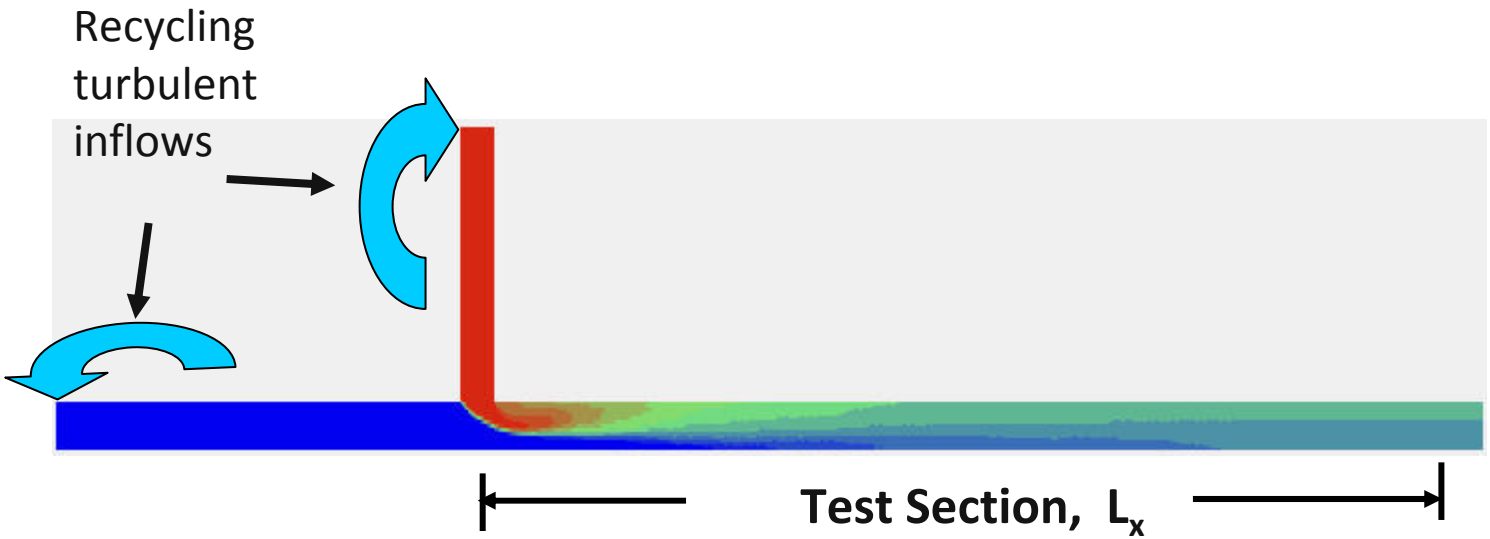


SEM *expt.*

Nek5000 Submission to T-Junction Benchmark

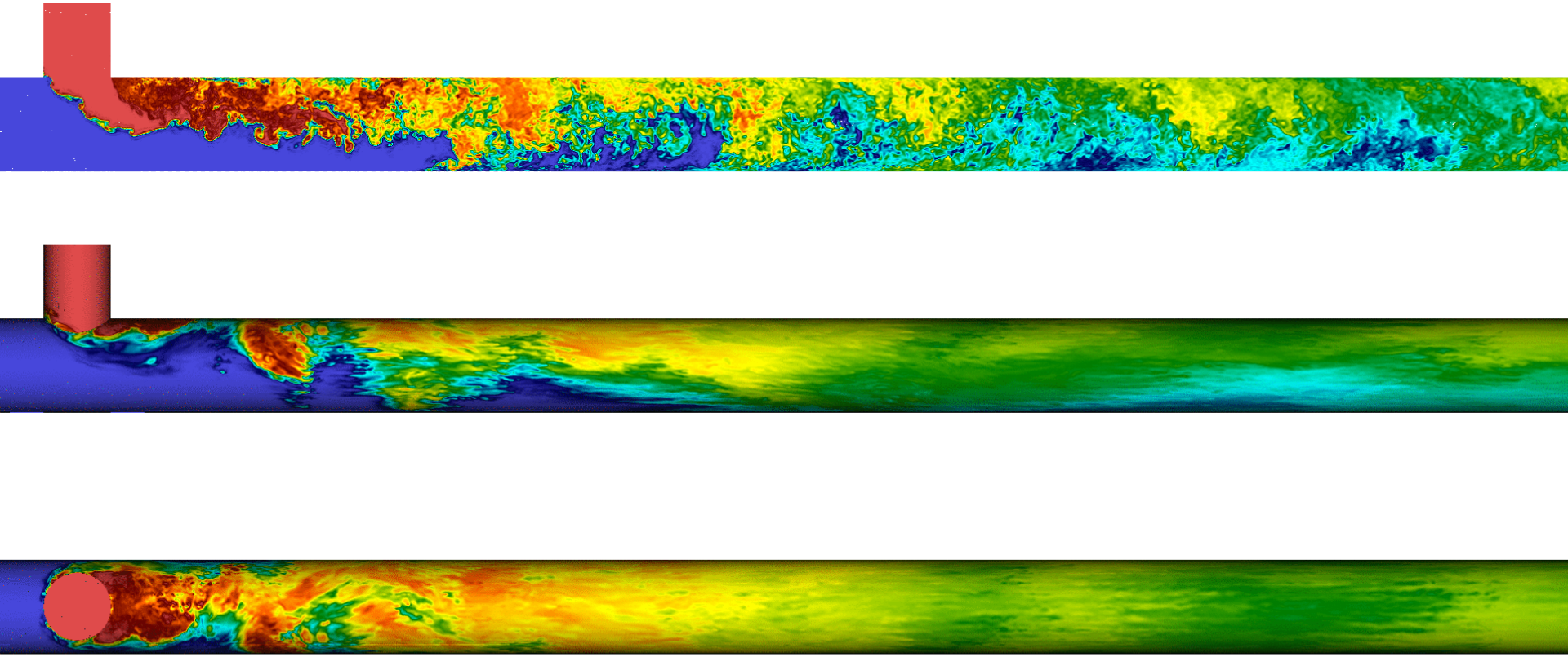
F., Obabko, Tautges, Caceres

- E=62000 spectral elements of order N=7 (n=21 million)
  - Mesh generated by SHARP framework group using CUBIT
  - Loaded into Nek5000 through parallel MOAB interface
  - A ***group*** effort at integrated process, under a time constraint…
- Re_D = 40,000 for inlet pipes (exp.: 90,000)
- Subgrid dissipation modeled with low-pass spectral filter
- L_x ~ 25 D (cost is quadratic in L_x)
- 24 hours on 16384 processors of BG/P (850 MHz) *~ 33x slower than uRANS*



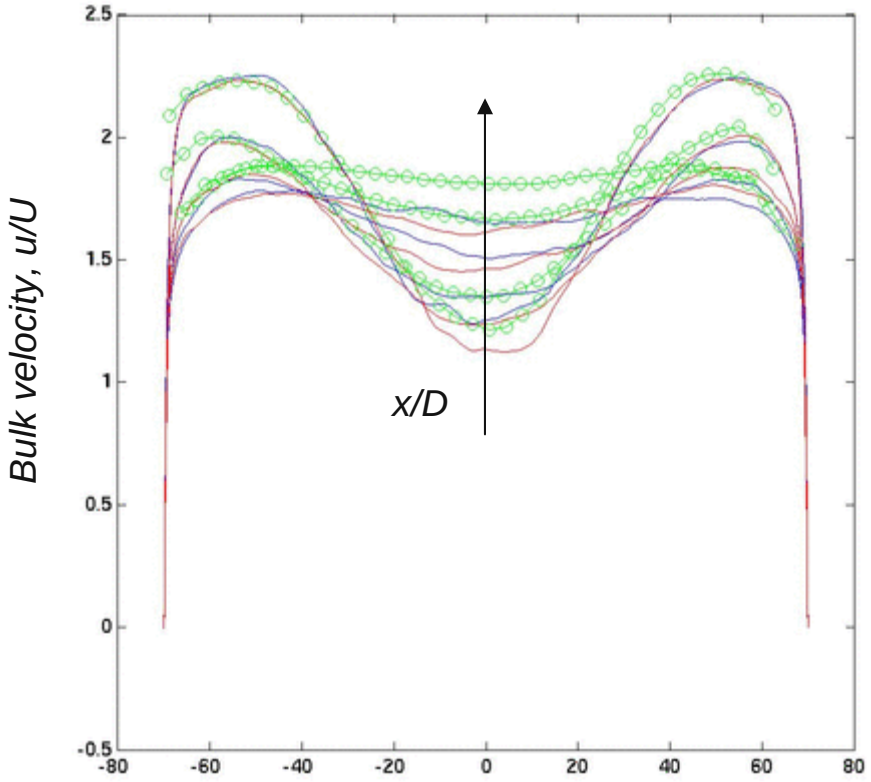
Instantaneous Temperature Distributions:

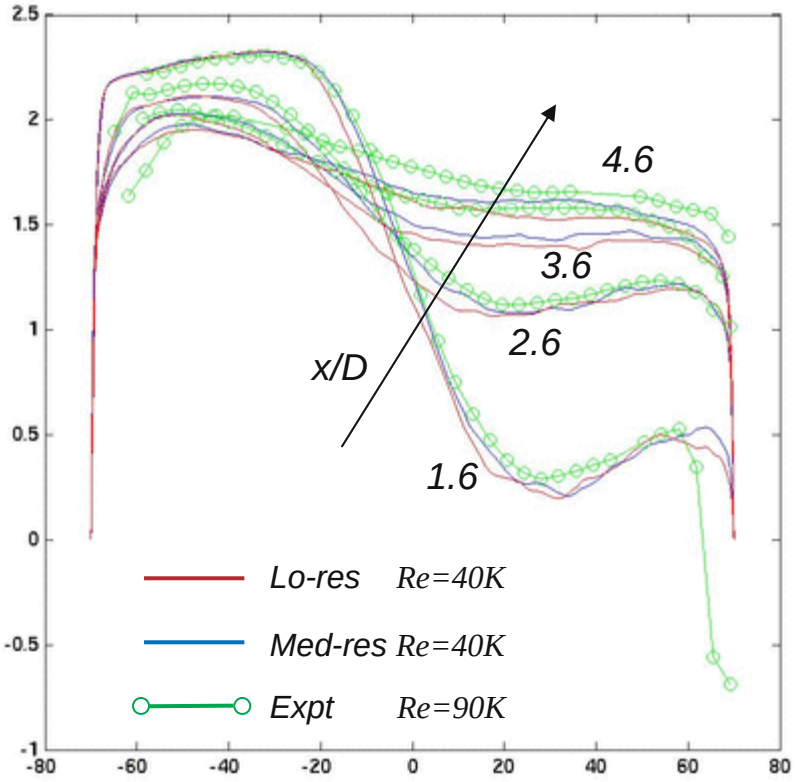
Thermal striping leads to thermal fatigue in structural components



Velocity Comparison Downstream of T-junction

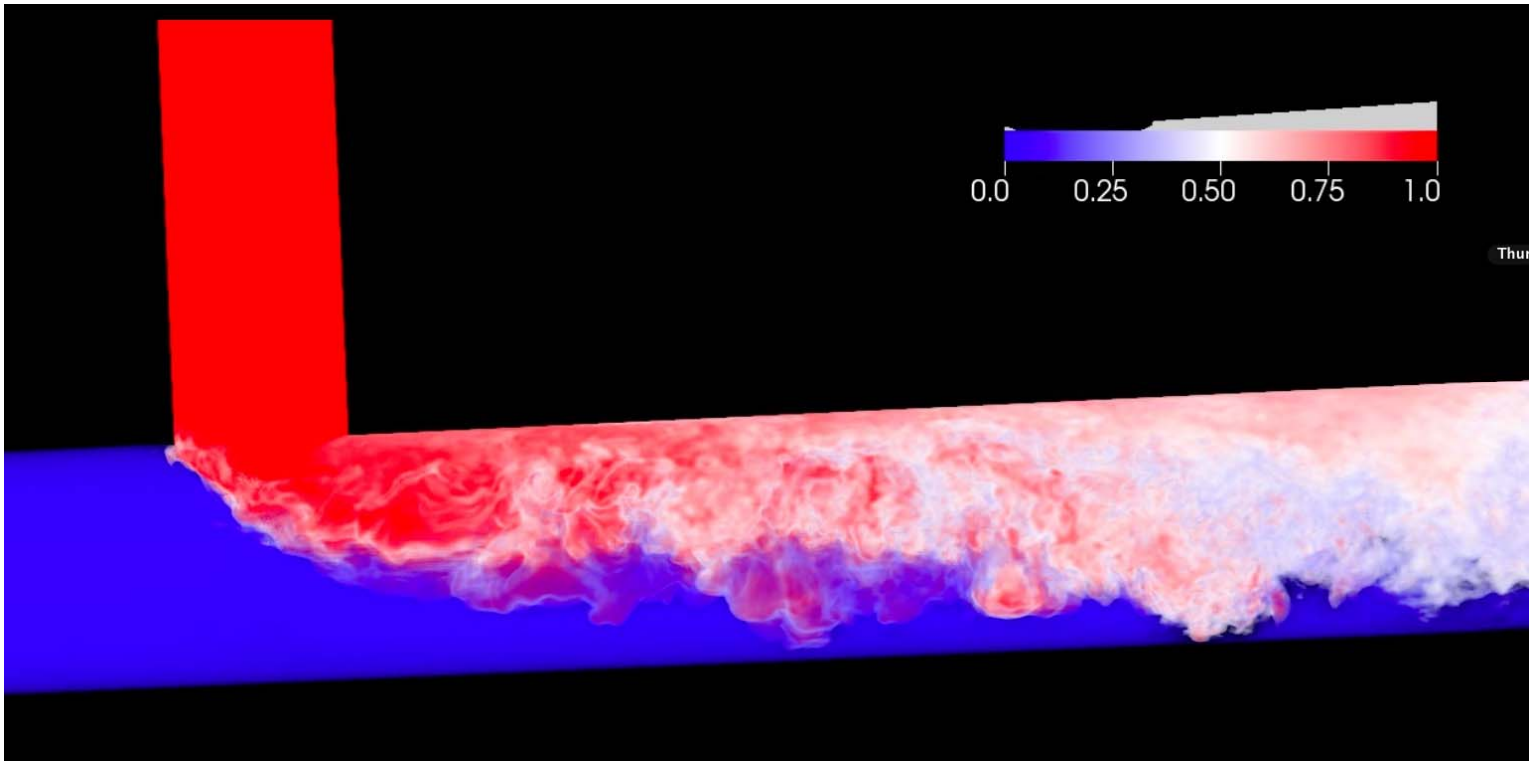
- Medium resolution results are in excellent agreement at x=1.6 & 2.6
- Experiment (Re=90K) exhibits more rapid recovery of profile than simulation (Re=40K)





NEA/OECD 2010 Blind T-Junction Benchmark

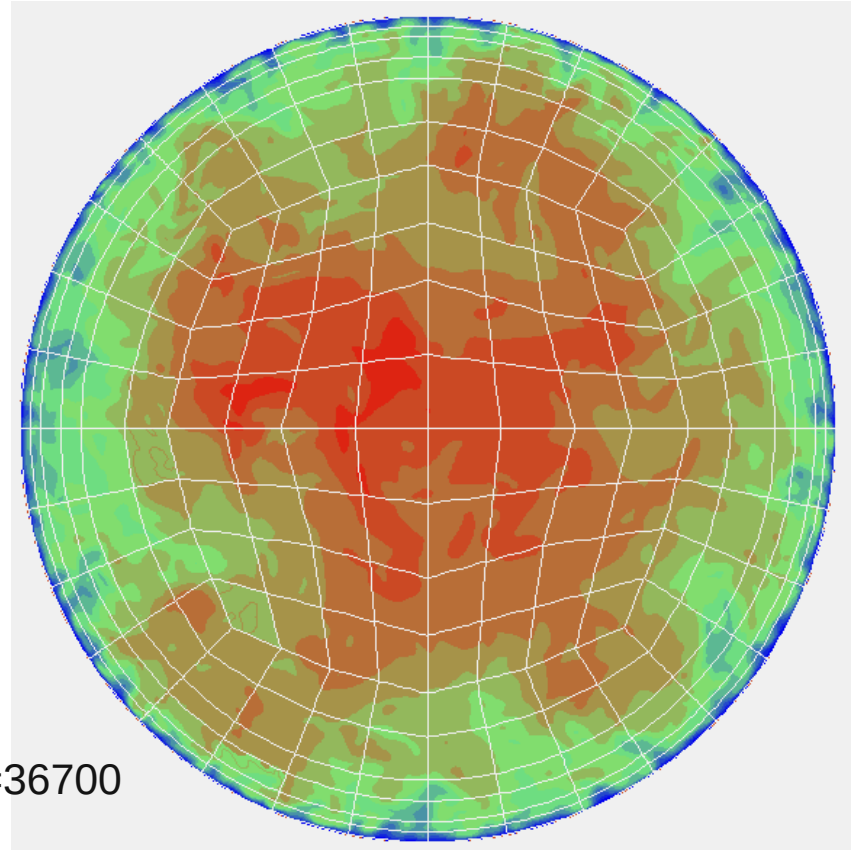
- Experiment with hot/cold inlets at Re ~ 10^5
- Inlet velocity and temperature data provided by Vattenfall.
- Results announced 9/15/10 at CFD4NRS meeting in DC
- Nek5000 submission (21 M points) ranked respectively 1st and 6th in temperature and velocity prediction, of 29 entries w/ 1—71 M points.



Pipe Flow at Re=44,000

Motivated by T-junction benchmark:

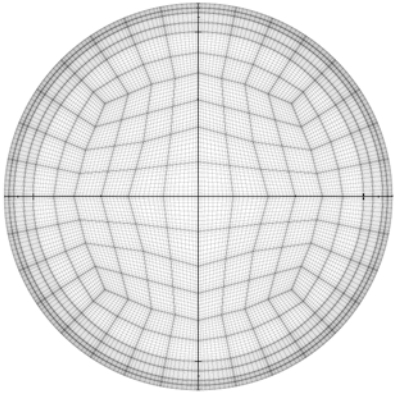
- Target Re=100,000
  - E=12288, N=11, n~16 million
  - Δt=.0003, CFL ~ 0.35
  - P_N-P_N
  - Filter-based dissipation: p101,103=3,.001
  - One of many test runs:
    - *Dyn. Smag.: good wall shear predictions*
      - 1-2 million points
      - No filter

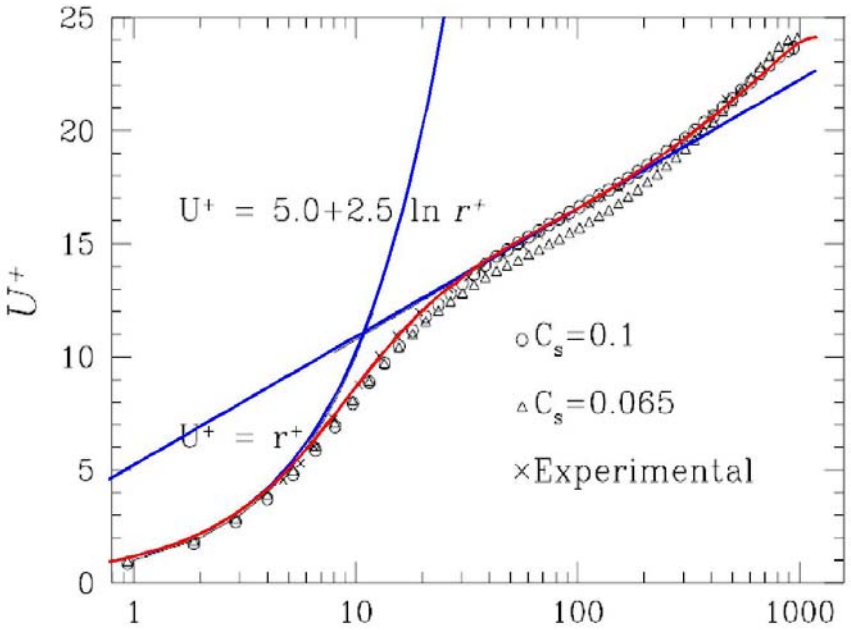


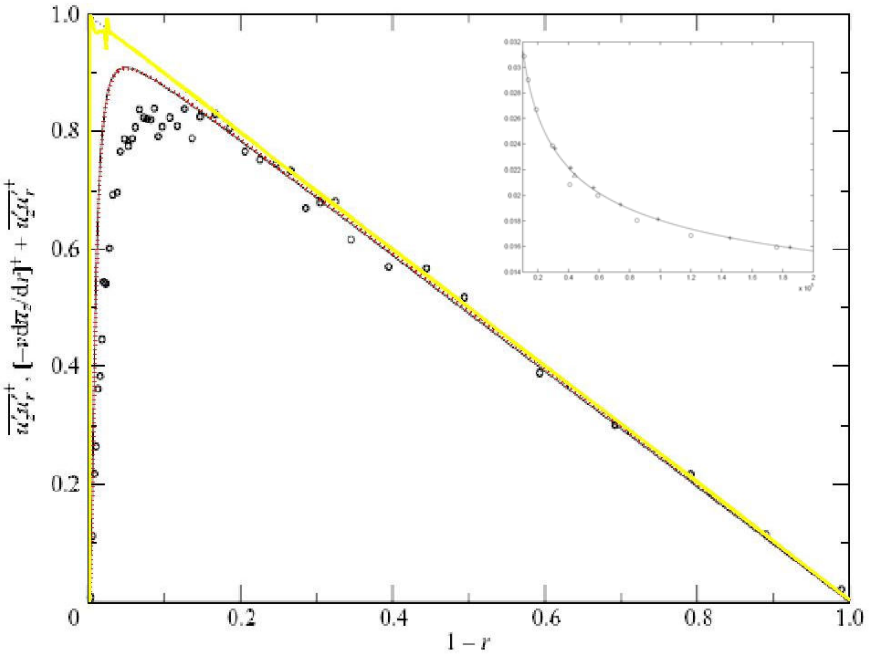
- Comparison with:
  - **Rudman & Hughes (1999)** SEM LES @ Re=36700
    1.2 million points
  - **Wu & Moin (2008)** 2^{nd}-order FD DNS Re=44000
    630 m points

Re=44K Pipe Flow Results

- *16 million point filter-based SEM gives perfect Reynolds stress agreement with 630 million point DNS*
- ***Rigged**…(sort of…): Filter weight chosen to match target WSS.*
  - *Investigating automatic selection criterion*
- *10% variance in friction factor between Princeton & Oregon expts at 44K.*
  - *DNS closest to combined curve fit.*



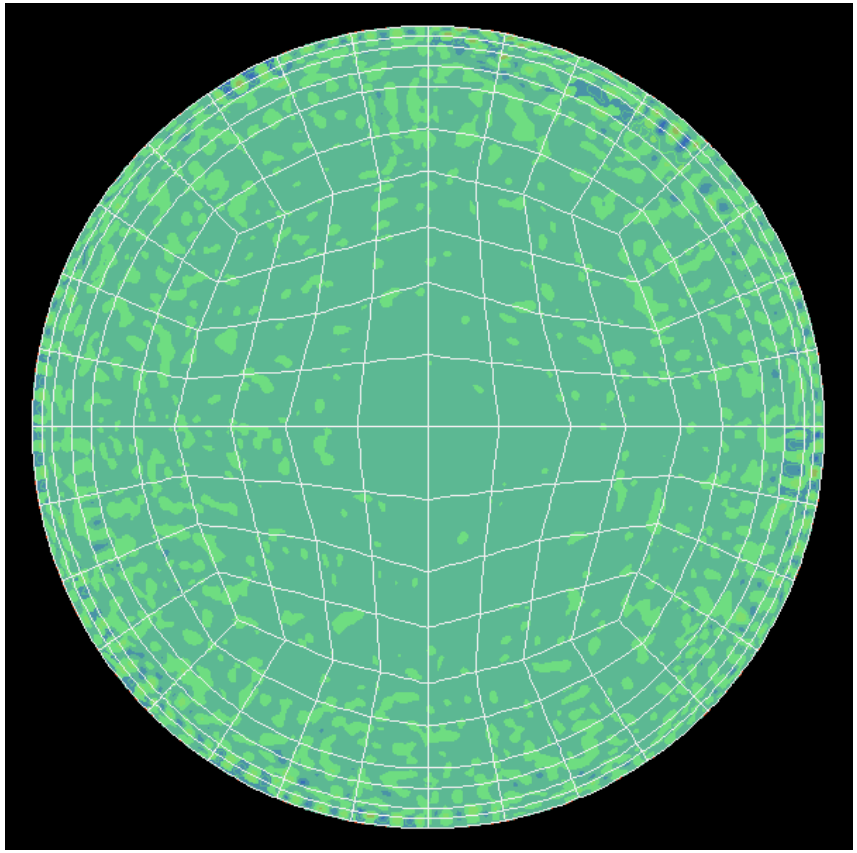




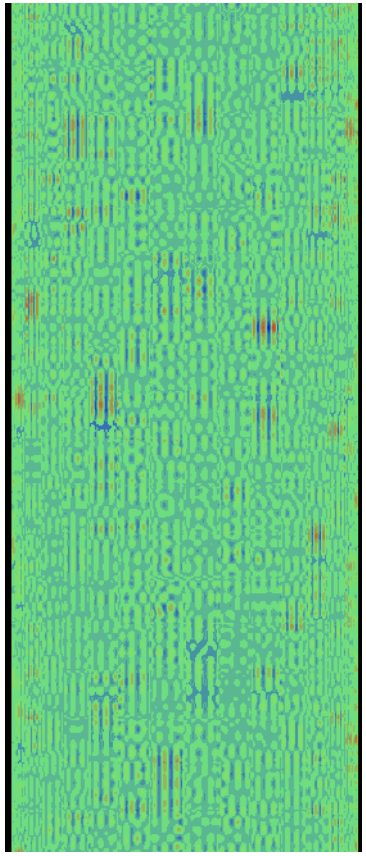
Rudman & Blackburn (1999) – LES; DenToonder & Nieuwstadt (1997) exp (Re=24,600), Wu & Moin (2008) DNS (Re=44,000)

Resolution Questions: Pipe Flow, Re > 36,700

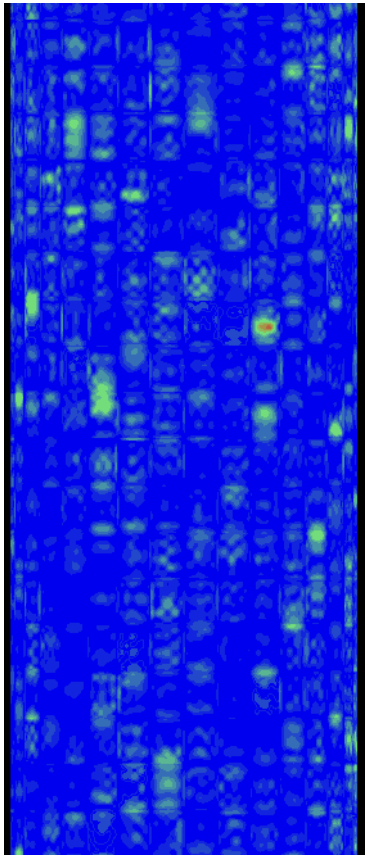
- Plots of w-w* , where w* = low-pass-filtered axial velocity
- Illustrates that azimuthal resolution is weak link in current simulations



$w - w^*$



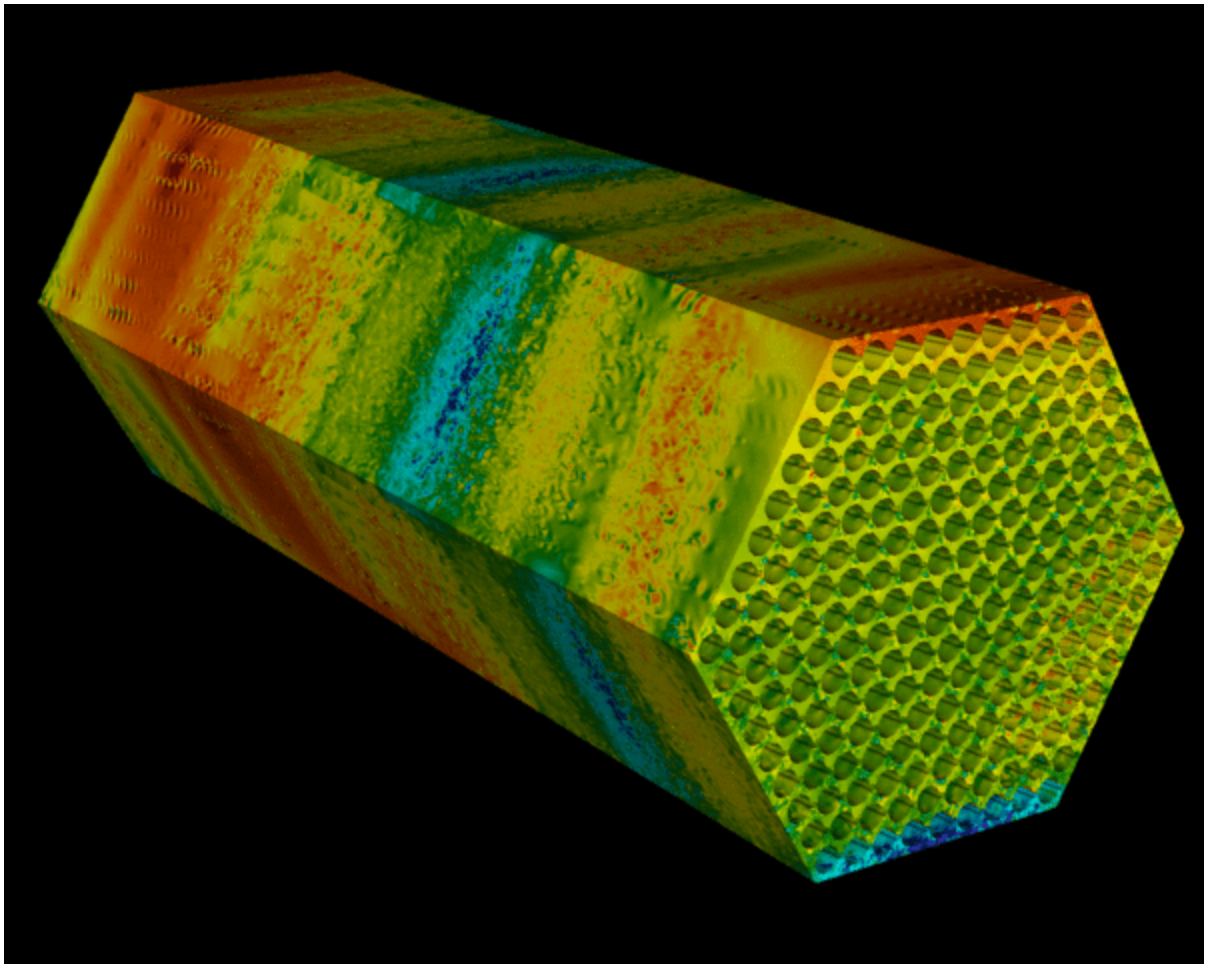
$w - w^*$

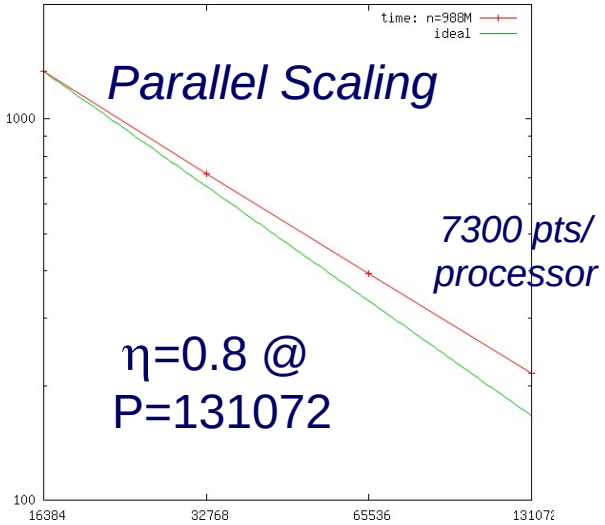


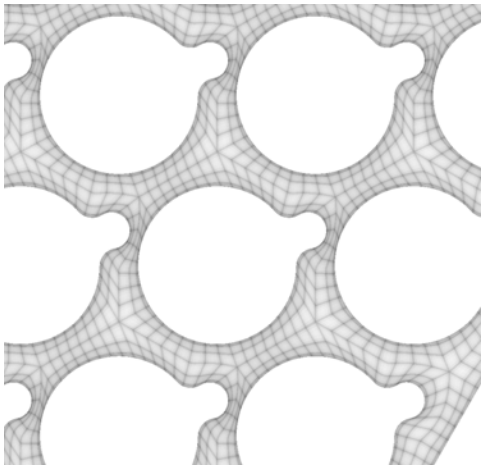
$| u - u^* |$

Parallel Scaling: Subassembly 217 Wire-Wrapped Pins

- 3 million 7^{th}-order spectral elements (n=1.01 billion)
- 16384–131072 processors of IBM BG/P





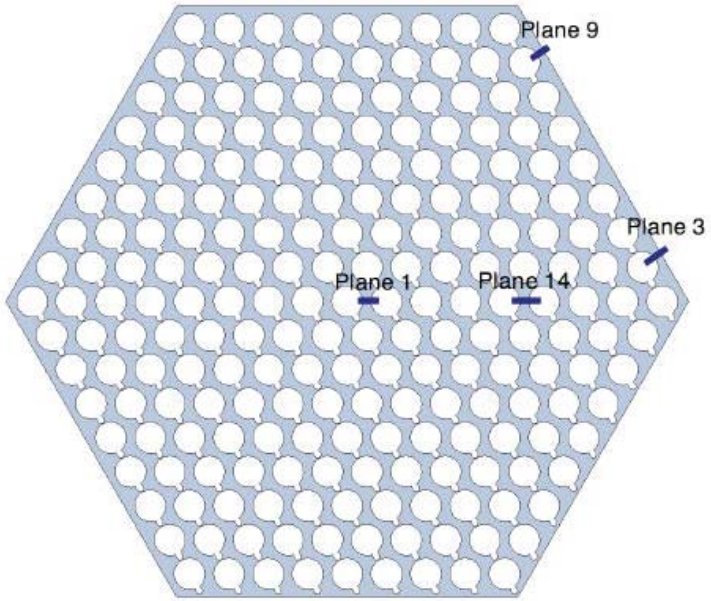


www.mcs.anl.gov/~fischer/sem1b

Nek5000 / Star Cross-Channel Velocity Comparison

HEDL geometry

Re_h = 10,500



W.D. Pointer et al., *Simulations of Turbulent Diffusion in Wire-Wrapped Sodium Fast Reactor Fuel Assemblies, Best Paper Award*, FR09, Kyoto (2009)

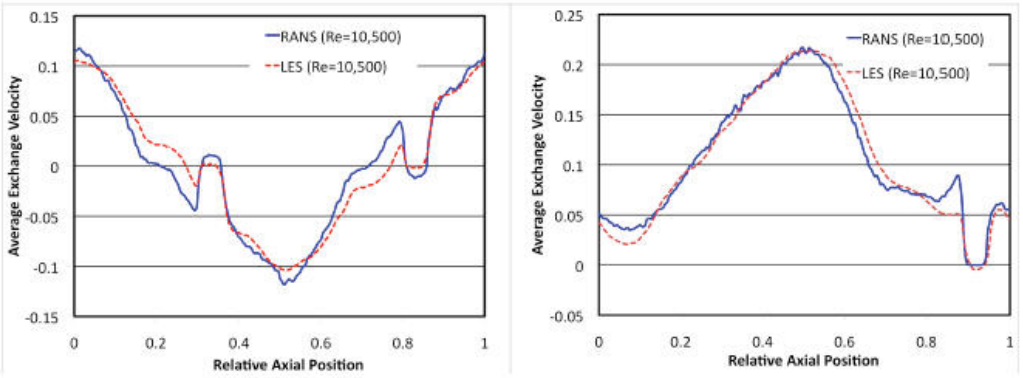


Figure 7. Comparison of predicted normal velocity profile for (a) plane group 1 and (b) plane group 3. RANS data is shown in blue and LES data is shown in red.

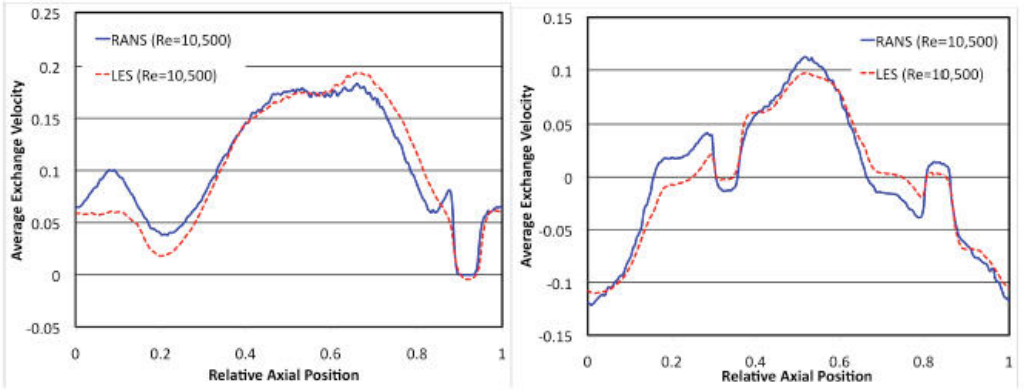
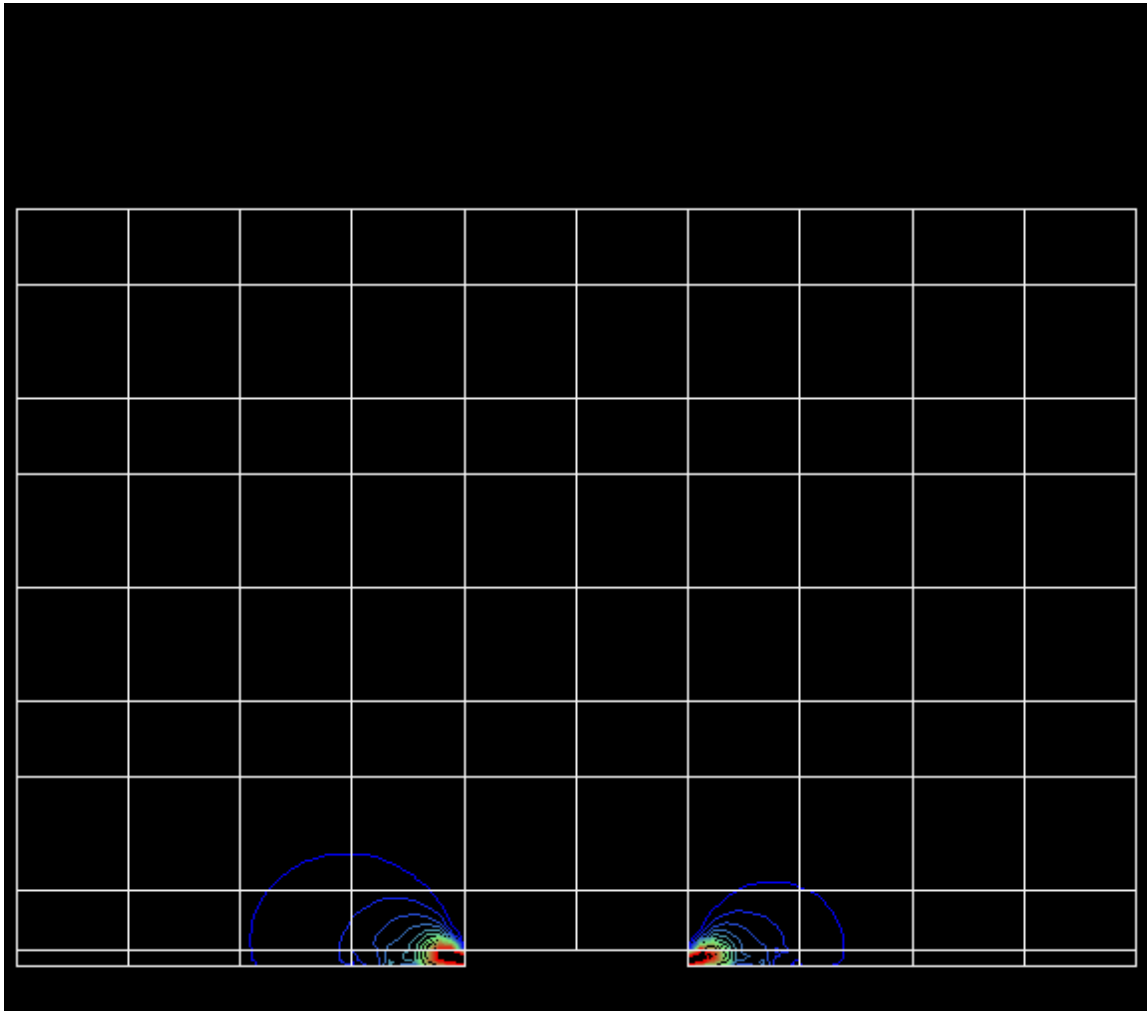


Figure 8. Comparison of predicted normal velocity profile for (a) plane group 9 and (b) plane group 14. RANS data is shown in blue and LES data is shown in red.

Moving Mesh Examples

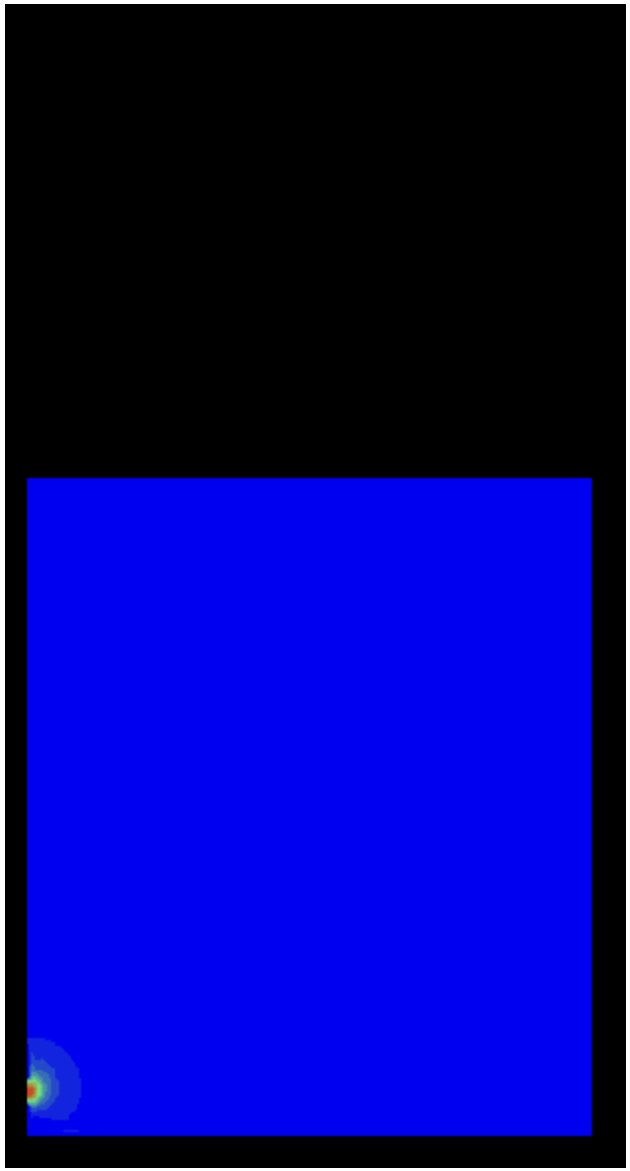
- **peristaltic flow model**
  nek5_svn/examples/peris

- **2D piston, intake stroke:**
  (15 min. to set up and run)



Moving Mesh Examples

- **Free surface case**



Moving Mesh Examples

- **Free surface verification:**
  - *agreement with linear theory to 3 significant digits*
  - *long time solution requires filtering*

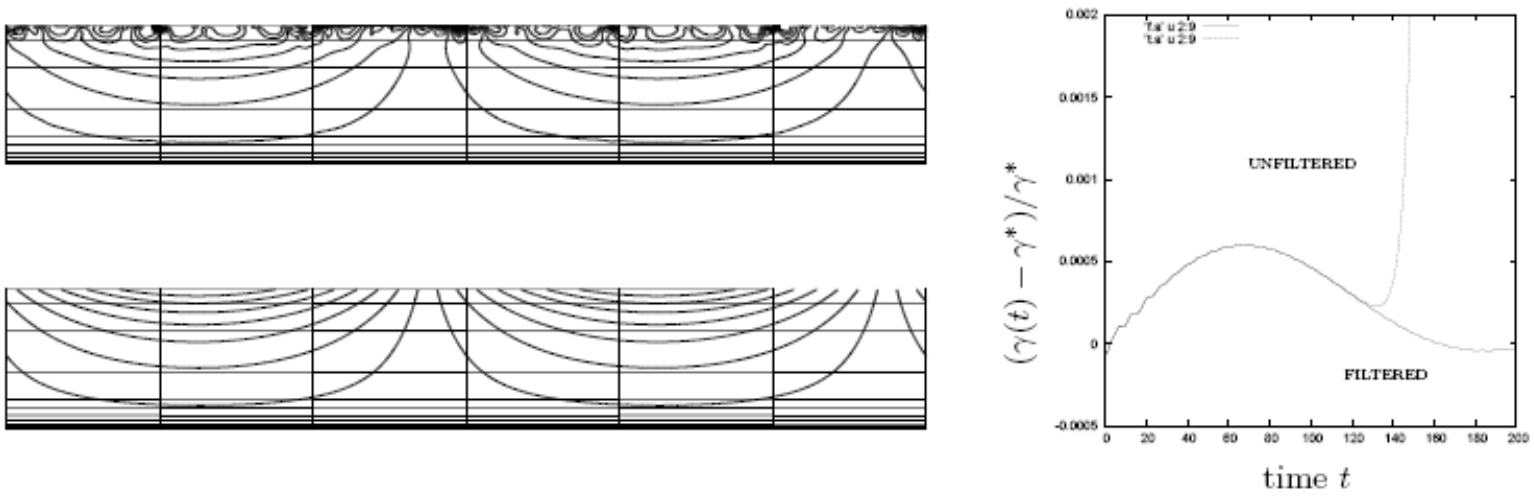


Figure 5: Eigenmodes for free-surface film flow: (left, top) contours of vertical velocity v for unfiltered and (left, bottom) filtered solution at time $t = 179.6$; (right) error in growth rate vs. t.

Nek5000 Brief History

- DNS / LES code for fluid dynamics, heat transfer, MHD, combustion,…
  - 100K lines of code: f77 (70K) & C (30K)
  - Interfaces w/ VisIt & MOAB/Cubit

- Based on high-order spectral element method (Patera ’84, Maday & Patera ’89)
  - Initially Nekton 2.0, first 3D SEM code. (F., Ho, Ronquist, Mavriplis ‘86-’89)

- First commercially-available code for distributed memory computers
  (marketed by Fluent as Nekton into the mid 90s)

- Nek5000 is a highly scalable variant of Nekton
  - Gordon Bell Prize in HPC, 4096 processors (Tufo & F. ’99)
  - 20% of peak on 262,000 processors of BGP (Kerkemeier, Parker & F. ’10)

Motivation for High-Order

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} ~ (L / λ) * | numerical dispersion error |

- For fixed final error $\mathcal{E}_f$, require: numerical dispersion error ~ $(\lambda / L)\mathcal{E}_f$, << 1

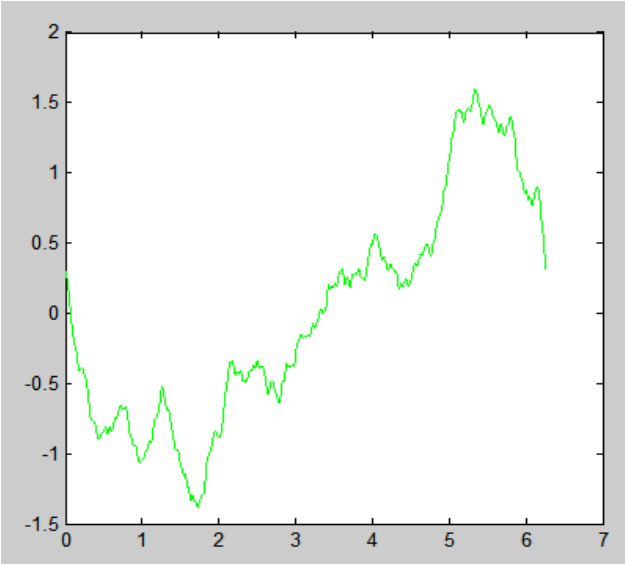
- High-order methods most efficiently deliver small dispersion errors

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

1D Advection Example

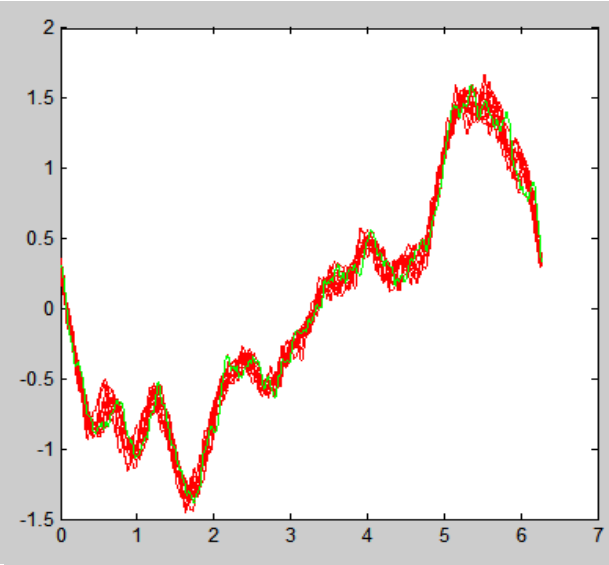
For a complex signal typical of turbulence (-5/3 spectrum), only N >15 case delivers *err* ~ 0.01 after ten revolutions of the signal, out of all cases using 600 points in space.

Exact Solution



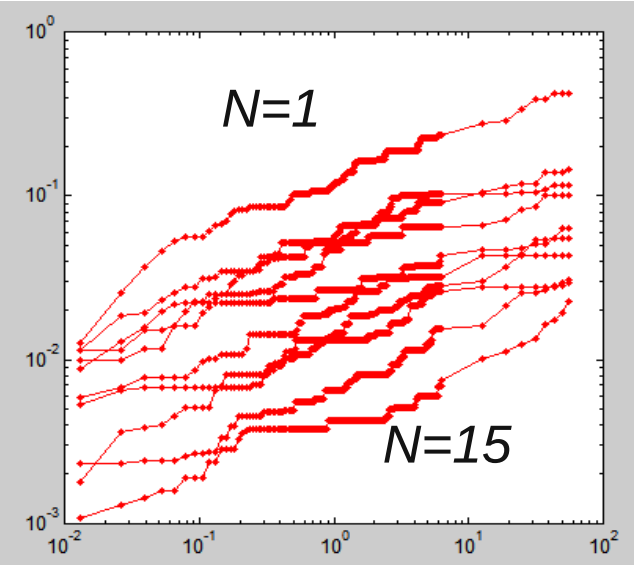
x

N=1 Approximation



x

Error vs time, 600 pts



time

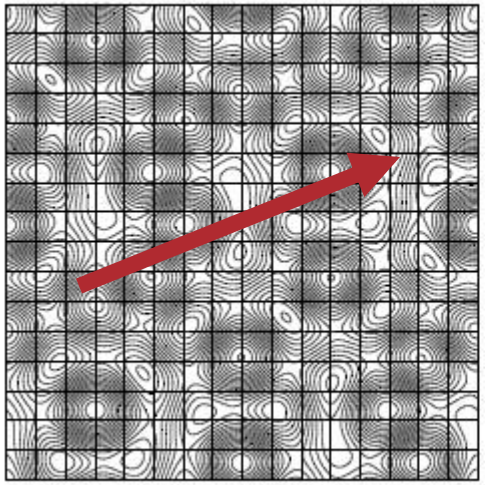
Spectral Element Convergence: Exponential with N

- Exact Navier-Stokes eigenfunctions with $\psi(x,y)$ comprising:

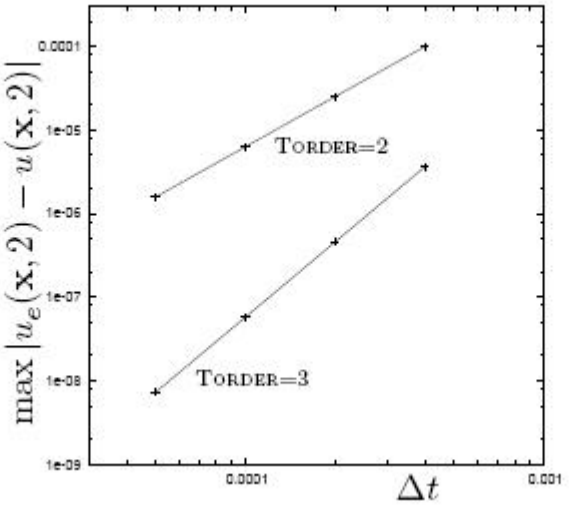
  $\cos(mx)\cos(ny)$ $\cos(mx)\sin(ny)$ $\sin(mx)\cos(ny)$ $\sin(mx)\sin(ny)$

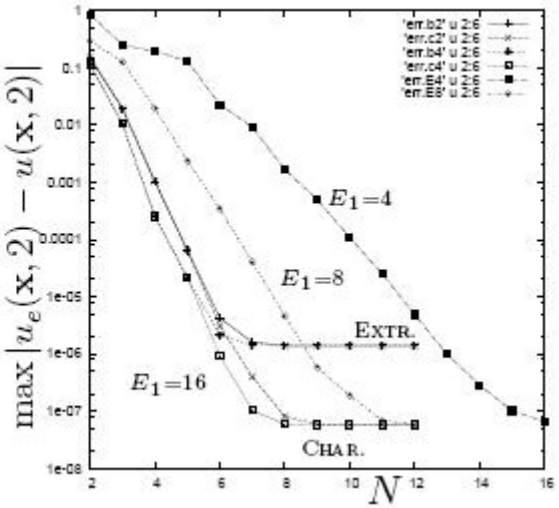
  with $m^2+n^2=\lambda^2$. [O. Walsh '92]

- (E,N)=$(4^2,7)$ yields ϵ=.01 after 25 convective time units, $t\bar{U}\lambda/2\pi$.

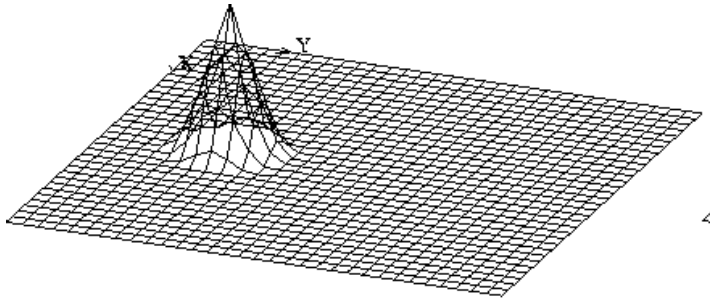


$\bar{U}=(16,5)$

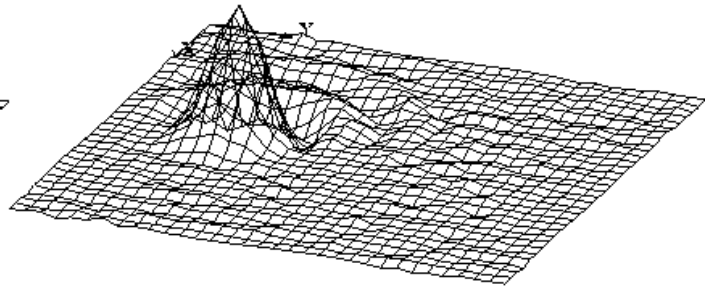




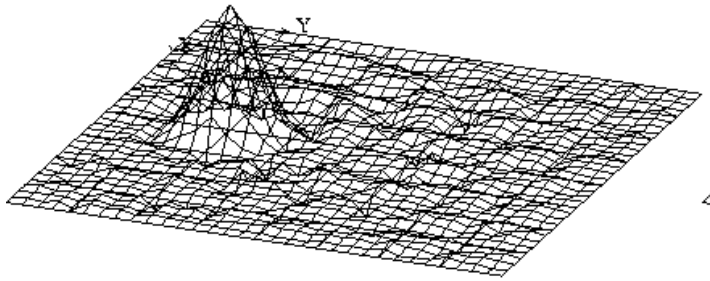
SEM Excellent transport properties, even for non-smooth *solutions*



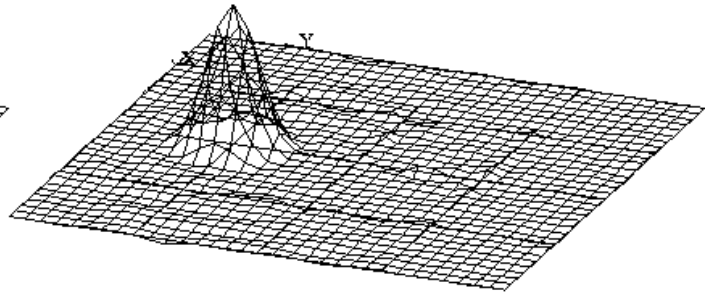
Initial Condition



$K_1 = 16,\ N = 2$



$K_1 = 8,\ N = 4$



$K_1 = 4,\ N = 8$

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

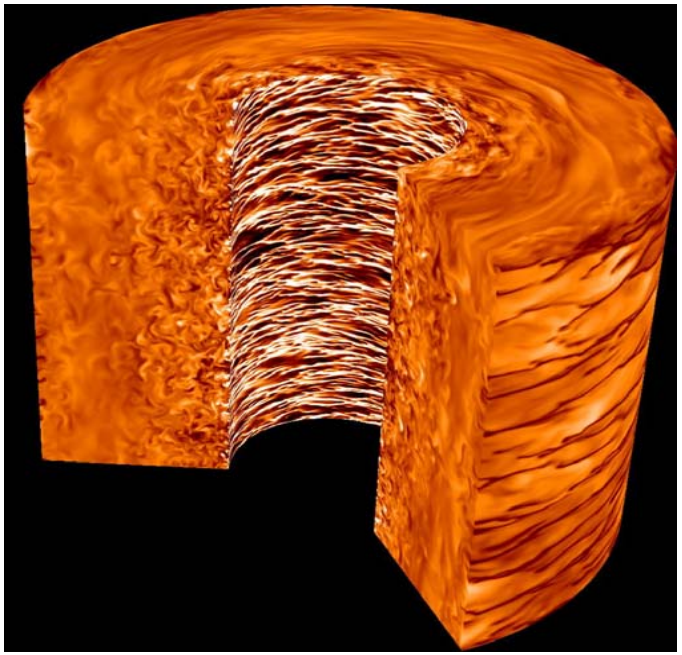
(cf. Gottlieb & Orszag 77)

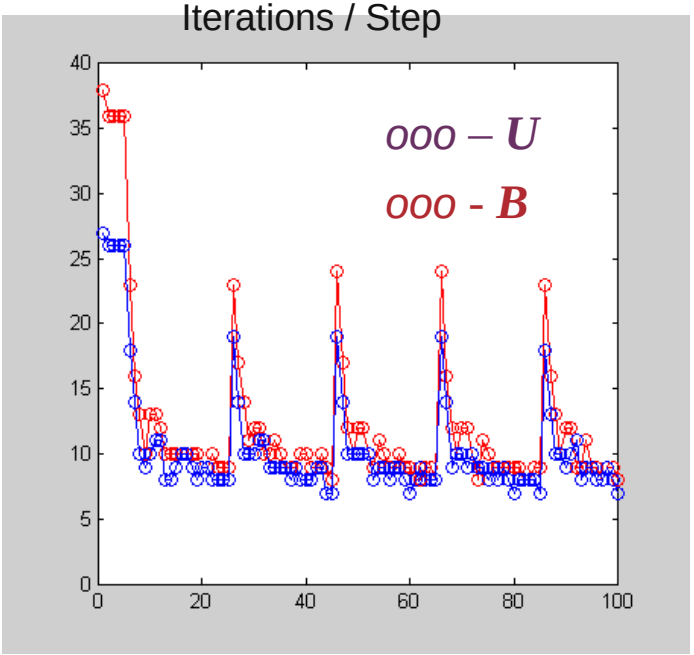
Strengths of Nek5000

- *High-order accuracy at low cost*
  - *Extremely rapid (exponential) convergence in space*
  - *3rd-order accuracy in time*
- *Highly scalable*
  - *Fast scalable multigrid solvers*
  - *Scales to > 290,000 processors with ~10^4 pts/proc on BGP*
- *Extensively tested*
  - *> 10s of platforms over 25 years*
  - *> 150 journal articles & > 60 users worldwide*
  - *> 400 tests after each build to ensure verified source (more tests to be added)*

Solver Performance: Hybrid Schwarz-Multigrid

- *Magneto-rotational instability* *(Obabko, Cattaneo & F.)*
  - *E=140000, N=9 (n = 112 M), P=32768 (BG/L)*
  - *~ 1.2 sec/step*
  - *~ 8 iterations / step for U & B*
  - ***Key is to have a scalable coarse-grid solver***



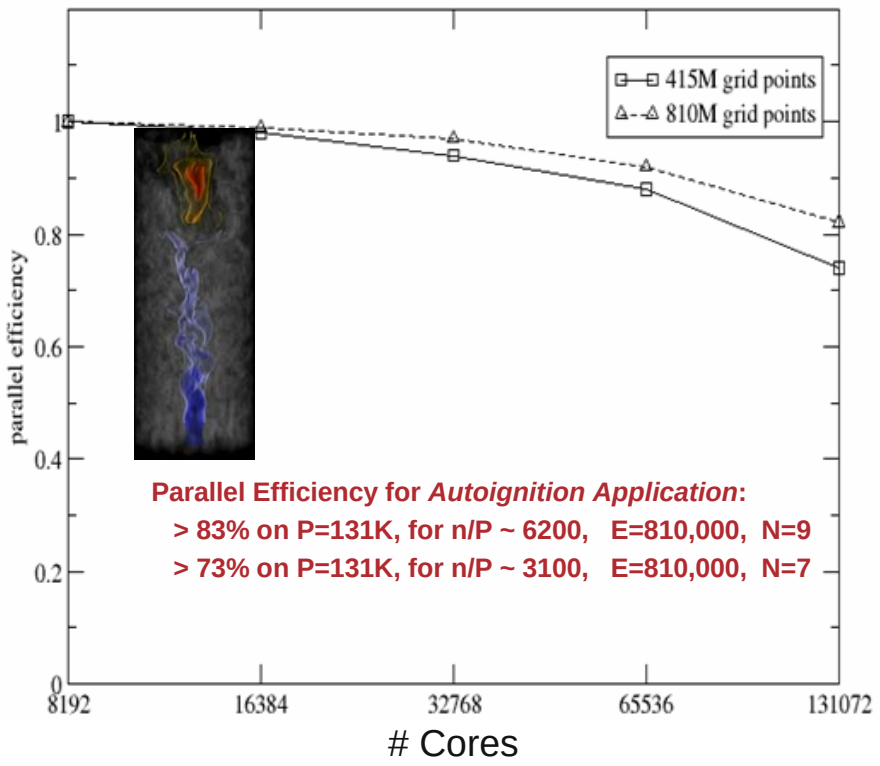


Scaling to P=262144 Cores

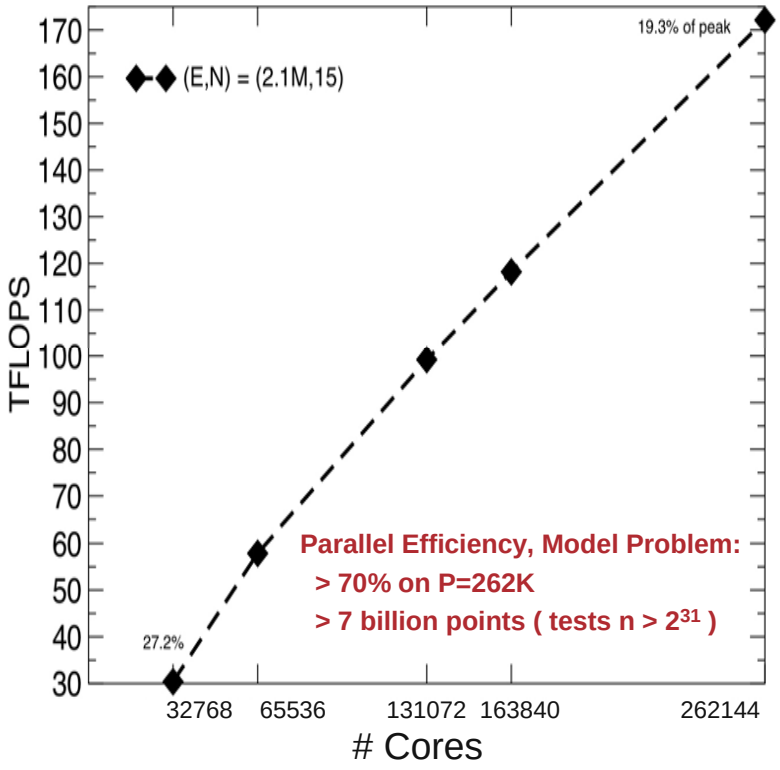
Stefan Kerkemeier
ETHZ / ANL

- Production combustion and reactor simulations on ALCF BG/P demonstrate scaling to P=131072 with n/P ~ 5000-10,000 and η ~ .7
- Test problem with 7 billion points scales to P=262144 on Julich BG/P with η ~ .7
  - tests 64-bit global addressing for ***gs*** communication framework

BG/P Strong Scaling: P=8192 – 131072



P=32768 – 262144

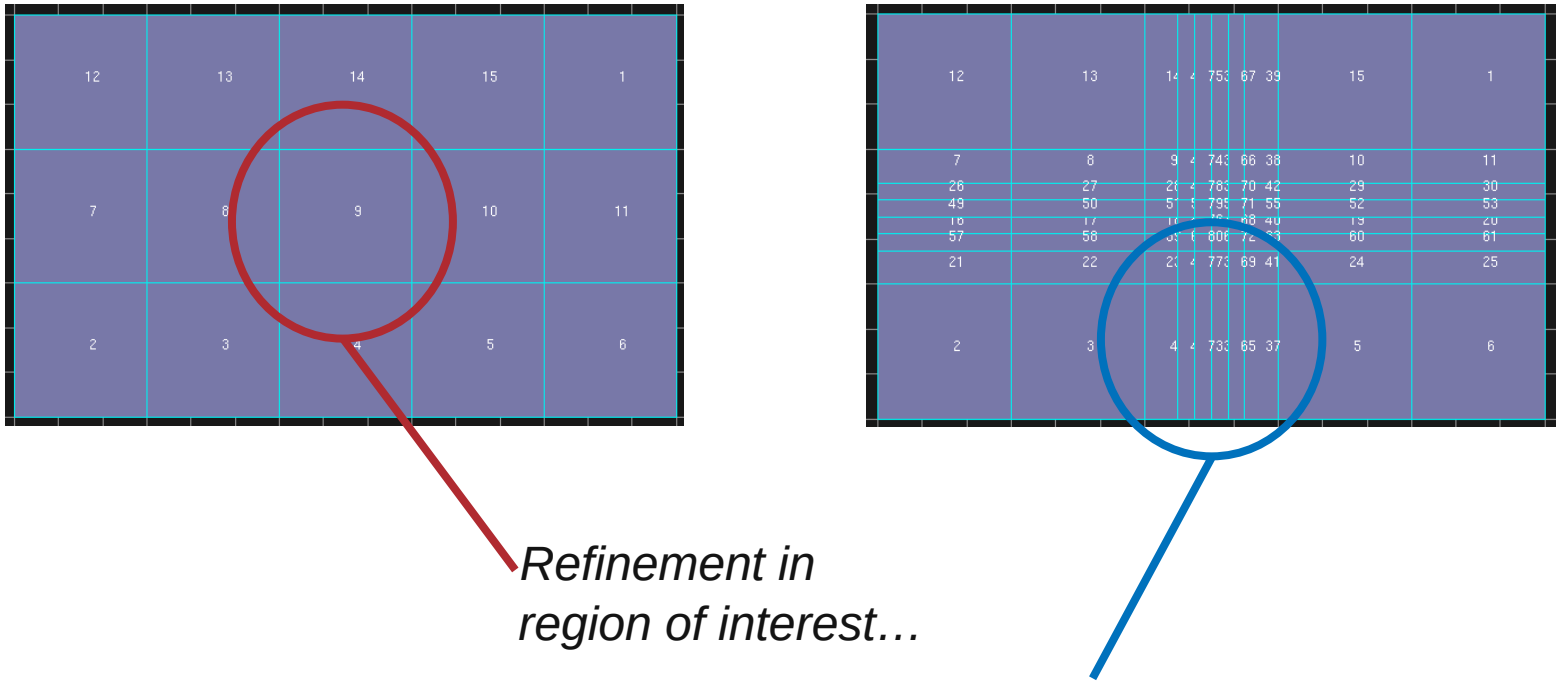


Limitations of Nek5000

- ***No steady-state NS or RANS:***
  - *unsteady RANS under development / test – Aithal*
- ***Lack of monotonicity for under-resolved simulations***
  - *limits, e.g., LES + combustion*
  - *A high priority for 2011-12*
- ***Meshing complex geometries:***
  - *fundamental: meshing always a challenge; hex-based meshes intrinsically anisotropic*
  - *technical: meshing traditionally not supported as part of advanced modeling development*

Mesh Anisotropy

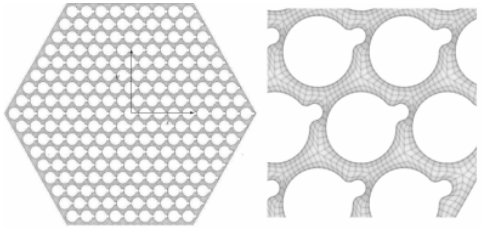
A common refinement scenario (somewhat exaggerated):

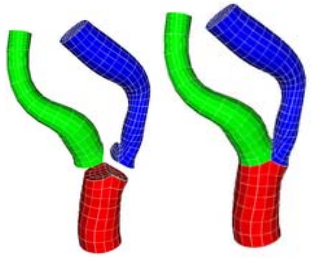


- ***Refinement propagation leads to***
  - unwanted elements in far-field
  - high aspect-ratio cells that are detrimental to iterative solver performance (F. JCP'97)

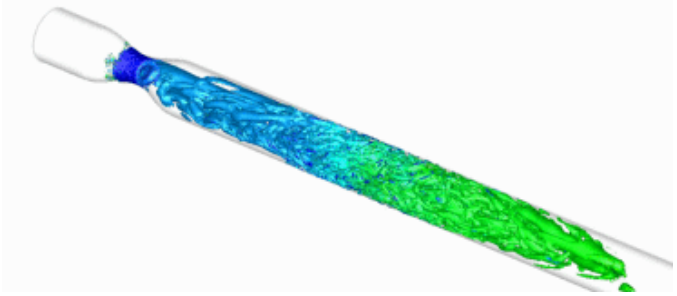
Some Meshing Options

- *genbox: unions of tensor-product boxes*
- *prenek: basically 2D + some 3D or 3D via extrusion (n2to3)*
- *Grow your own: 217 pin mesh via matlab; BioMesh*





- *3rd party: CUBIT + MOAB, TrueGrid, Gambit, Star CD*
- *Morphing:*



Part 2 (a)

Equations, timestepping, and spectral element formulation

…but first, a bit of code structure.

nek5_svn repository

- **Key subdirectories in the repo:**
  - **nek5_svn**
    - ***trunk***
      - **nek** – *makenek* script and source files
      - **tools** – several utilities (prenek, genbox, etc.) and scripts
    - ***examples*** – several case studies
- **Typical steps to run a case:**
  - Create a working directory and copy contents of a similar example case to this directory
  - Modify ***case files*** to suit
  - Copy makenek from nek and type makenek <case>
  - Run job using a script (tools/scripts) and analyze results (postx/VisIt)

nek5_svn repository

- **nek5_svn**
  - |-- 3rd_party
  - |-- branches
  - |-- **examples**
  - | |-- axi
  - | |-- benard
  - | |-- conj_ht
  - | |-- eddy
  - | |-- fs_2
  - | |-- fs_hydro
  - | |-- kovasznay
  - | |-- lowMach_test
  - | |-- moab
  - | |-- peris
  - | |-- pipe
  - | |-- rayleigh
  - | |-- shear4
  - | |-- timing
  - | |-- turbChannel
  - | |-- turbJet
  - | `-- vortex
  - |-- tags
  - |-- tests
  - `-- **trunk**

- **nek5_svn**
  - |-- :
  - |-- :
  - `-- trunk
    - |-- **nek**
    - | | :
    - | |-- source files….
    - | | :
    - `-- **tools**
      - |-- amg_matlab
      - |-- avg
      - |-- genbox
      - |-- genmap
      - |-- makefile
      - |-- maketools
      - |-- n2to3
      - |-- nekmerge
      - |-- postnek
      - |-- prenek
      - |-- reatore2
      - `-- scripts

Base Nek5000 Case Files

- **SIZE** – an f77 include file that determines
  - spatial dimension (ldim =2 or 3)
  - approximation order (lx1,lx2,lx3,lxd) - N := lx1-1
  - upper bound on number of elements per processor: lelt
  - upper bound on total number of elements, lelg

- <case>**.rea** – a file specifying
  - job control parameters (viscosity, dt, Nsteps, integrator, etc.)
  - geometry – element vertex and curvature information
  - boundary condition types
  - restart conditions

- <case>**.usr** – f77 source file specifying
  - initial and boundary conditions
  - variable properties
  - forcing and volumetric heating
  - geometry morphing
  - data analysis options: min/max, runtime average, rms, etc.

Snapshot of SIZE

```
      parameter (ldim=2)
      parameter (lx1=14,ly1=lx1,lz1=1,lelt=80,lelv=lelt)
      parameter (lxd=20,lyd=lxd,lzd=1)
      parameter (lelx=1,lely=1,lelz=1)
c
c     NOTE: for IBM BLUE GENE LX1,LXD has to be an even number (double hummer)

      parameter (ldimt= 1)        ! upper limit for passive scalars + T

      parameter (lp =      64)    ! upper limit for number of CPUs
      parameter (lelg = 5000)     ! upper limit for total number of elements

c
c     ******************************************************************
c
      parameter (lzl=3 + 2*(ldim-3))

      parameter (lx2=lx1-0)
      parameter (ly2=ly1-0)
      parameter (lz2=lz1)

      parameter (lx3=lx2)
      parameter (ly3=ly2)
      parameter (lz3=lz2)
```

Snapshots of .rea file

- Parameters section

```
          2 DIMENSIONAL RUN
        118 PARAMETERS FOLLOW
 1.00000      P001: DENSITY
-40.0000      P002: VISCOS
 0.000000E+00 P003:
 0.000000E+00 P004:
 0.000000E+00 P005:
 0.000000E+00 P006:
 1.00000      P007: RHOCP
 1.00000      P008: CONDUCT
 0.000000E+00 P009:
 0.000000E+00 P010: FINTIME
 2000.00      P011: NSTEPS
-0.100000E-02 P012: DT
 0.000000E+00 P013: IOCOMM
 0.000000E+00 P014: IOTIME
 0.000000E+00 P015: IOSTEP
 0.000000E+00 P016: PSSOLVER: 0=default
 1.00000      P017:
 0.500000E-01 P018: GRID < 0 --> # cells
-1.00000      P019: INTYPE
 4.00000      P020: NORDER
 0.100000E-05 P021: DIVERGENCE
 0.100000E-09 P022: HELMHOLTZ
 0.000000E+00 P023: NPSCAL
 0.000000E+00 P024: TOLREL
 0.000000E+00 P025: TOLABS
 2.00000      P026: COURANT/NTAU
 3.00000      P027: TORDER
```

- Geometry and boundary conditions

```
     ELEMENT           5 [    1 ]    GROUP     0
-0.5000000      0.0000000E+00  0.0000000E+00 -0.5000000
 0.5000000      0.5000000       1.000000       1.000000
     ELEMENT           6 [    1 ]    GROUP     0
 0.0000000E+00   1.000000       1.000000       0.0000000E+00
 0.5000000      0.5000000       1.000000       1.000000
     ELEMENT           7 [    1 ]    GROUP     0
-0.5000000      0.0000000E+00  0.0000000E+00 -0.5000000
  1.000000       1.000000       1.500000       1.500000
     ELEMENT           8 [    1 ]    GROUP     0
 0.0000000E+00   1.000000       1.000000       0.0000000E+00
  1.000000       1.000000       1.500000       1.500000
 ***** CURVED SIDE DATA *****
       0 Curved sides follow IEDGE,IEL,CURVE(I),I=1,5, CCURVE
 ***** BOUNDARY CONDITIONS *****
 ***** FLUID   BOUNDARY CONDITIONS *****
P    1  1   7.00000        3.00000       0.000000E+00  0.000000E+00
E    1  2   2.00000        4.00000       0.000000E+00  0.000000E+00
E    1  3   3.00000        1.00000       0.000000E+00  0.000000E+00
v    1  4  0.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00
P    2  1   8.00000        3.00000       0.000000E+00  0.000000E+00
v    2  2  0.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00
E    2  3   4.00000        1.00000       0.000000E+00  0.000000E+00
E    2  4   1.00000        2.00000       0.000000E+00  0.000000E+00
E    3  1   1.00000        3.00000       0.000000E+00  0.000000E+00
E    3  2   4.00000        4.00000       0.000000E+00  0.000000E+00
E    3  3   5.00000        1.00000       0.000000E+00  0.000000E+00
v    3  4  0.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00
E    4  1   2.00000        3.00000       0.000000E+00  0.000000E+00
v    4  2  0.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00
E    4  3   6.00000        1.00000       0.000000E+00  0.000000E+00
```

Snapshot of .usr file

```
c-----------------------------------------------------------------
      subroutine userf  (ix,iy,iz,eg)
      include 'SIZE'
      include 'TOTAL'
      include 'NEKUSE'

      integer e,f,eg
c     e = gllel(eg)

c     Note: this is an acceleration term, NOT a force!
c     Thus, ffx will subsequently be multiplied by rho(x,t)

      ffx = 0.0
      ffy = 0.0
      ffz = 0.0

      return
      end
c-----------------------------------------------------------------
      subroutine userchk
      include 'SIZE'
      include 'TOTAL'
      return
      end
c-----------------------------------------------------------------
      subroutine userbc (ix,iy,iz,iside,ieg)
      include 'SIZE'
      include 'TOTAL'
      include 'NEKUSE'
      ux=0.0
      uy=0.0
      uz=0.0
      temp=0.0
      return
      end
c-----------------------------------------------------------------
      subroutine useric (ix,iy,iz,ieg)
```

Derived Nek5000 Case Files

- <case>**.re2** – binary file specifying
  - geometry – element vertex and curvature information
  - boundary condition types

  This file is not requisite for small problems but important for element counts E > ~10,000

- <case>**.map** – ascii file derived from .rea/.re2 files specifying
  - mesh interconnect topology
  - element-to-processor map

  This file is needed for each run and is generated by running the "genmap" tool (once, for a given .rea file).

- **amg…dat** – binary files derived from .rea/.re2 files specifying
  - algebraic multigrid coarse-grid solver parameters

  These files are needed only for large processor counts (P > 10,000) and element counts (E > 50,000).

Part 2 (b)

Equations, timestepping, and spectral element formulation

Outline

- *Nek5000 capabilities*
- *Equations, timestepping, and SEM basics*
- *Parallel / serial issues that you should understand*

Equation Sets (2D/3D)

- Incompressible Navier-Stokes plus energy equation

$$\begin{aligned} \rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) &= -\nabla p + \nabla\cdot\mu\left(\nabla\mathbf{u} + \nabla^T\mathbf{u}\right) + \mathbf{f} \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}$$

$$\rho C_p\left(\frac{\partial T}{\partial t} + \mathbf{u}\cdot\nabla T\right) = \nabla\cdot k\nabla T + q'''$$

plus additional passive scalars:

$$\rho C_{p_i}\left(\frac{\partial T_i}{\partial t} + \mathbf{u}\cdot\nabla T_i\right) = \nabla\cdot k_i\nabla T_i + q_i''', \quad i = 3, \ldots, n_{flds}$$

- Also supports incompressible MHD, low Mach-number hydro, free-surface, and conjugate heat transfer formulations.

Steady State Equations

- Steady Stokes (plus boundary conditions):

$$\begin{aligned} -\nabla \cdot \mu(\mathbf{x}) \left(\nabla \mathbf{u} + \nabla^T \mathbf{u} \right) + \nabla p &= \mathbf{f}(\mathbf{x}) \\ \nabla \cdot \mathbf{u} &= 0 \end{aligned}$$

- Steady conduction (plus boundary conditions):

$$-\nabla \cdot k(\mathbf{x}) \nabla T + \lambda(\mathbf{x}) T = q'''(\mathbf{x}), \quad \lambda \geq 0$$

Constant Property Equation Set

- Incompressible Navier-Stokes + energy equation

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t}+\mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p+\mu\nabla^2\mathbf{u}+\mathbf{f}$$
$$\nabla\cdot\mathbf{u} = 0$$
$$\rho C_p\left(\frac{\partial T}{\partial t}+\mathbf{u}\cdot\nabla T\right) = k\nabla^2 T+q'''$$

- In Nek parlance, material properties specified in .rea file as:

dimensional	*nondimensional (convective time scale)*
•p1 = ρ	•p1 = 1
•p2 = μ	•p2 = $1/Re$ (or $-Re$)
•p7 = ρC_p	•p7 = 1
•p8 = k	•p8 = $1/Pe$ (or $-Pe$)

  or as variable properties in f77 routine uservp() (.usr file)

- Nek provides a scalable framework to advance these equations with user-defined properties. LES & RANS can be incorporated in this framework. (See /examples.)

Incompressible MHD

$$\frac{\partial \mathbf{u}}{\partial t} - \frac{1}{Re}\nabla^2\mathbf{u} + \nabla p = \mathbf{B}\cdot\nabla\mathbf{B} - \mathbf{u}\cdot\nabla\mathbf{u},$$

$$\nabla\cdot\mathbf{u} = 0$$

$$\frac{\partial \mathbf{B}}{\partial t} - \frac{1}{Rm}\nabla^2\mathbf{B} + \nabla q = \mathbf{B}\cdot\nabla\mathbf{u} - \mathbf{u}\cdot\nabla\mathbf{B},$$

$$\nabla\cdot\mathbf{B} = 0$$

— plus appropriate boundary conditions on $\mathbf{u}$ and $\mathbf{B}$

- Typically, $Re >> Rm >> 1$
- Semi-implicit formulation yields independent Stokes problems for $\mathbf{u}$ and $\mathbf{B}$

Timestepping

Navier-Stokes Time Advancement

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \nu \nabla^2 \mathbf{u}$$
$$\nabla \cdot \mathbf{u} = 0$$

- Nonlinear term: *explicit via BDFk/EXTk or characteristics*

  (Pironneau '82)

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

- Filter velocity (& temperature), if required

(*Order is slightly different for P_N-P_N.)*

Timestepping Design

- ***Implicit:***
  - *symmetric and (generally) linear terms,*
  - *fixed flow rate conditions*

- ***Explicit:***
  - *nonlinear, nonsymmetric terms,*
  - *user-provided rhs terms, including*
    - Boussinesq and Coriolis forcing

- ***Rationale:***
  - *div **u** = 0 constraint is fastest timescale*
  - *Viscous terms: explicit treatment of 2nd-order derivatives → $\Delta t \sim O(\Delta x^2)$*
  - *Convective terms require only $\Delta t \sim O(\Delta x)$*
  - *For high Re, temporal-spatial accuracy dictates $\Delta t \sim O(\Delta x)$*
  - *Linear symmetric is "easy" – nonlinear nonsymmetric is "hard"*

BDF2/EXT2 Example

Consider the convection-diffusion equation,

$$\frac{\partial u}{\partial t} + \mathbf{c} \cdot \nabla u = \nu \nabla^2 u.$$

Discretize in space:

$$B\frac{d\underline{u}}{dt} + C\underline{u} = -\nu A\underline{u}, \qquad (A \text{ is SPD})$$

Evaluate each term at t^n according to convenience:

$$B\frac{d\underline{u}}{dt}\bigg|_{t^n} = B\frac{3\underline{u}^n - 4\underline{u}^{n-1} + \underline{u}^{n-2}}{2\Delta t} + O(\Delta t^2)$$

$$C\underline{u}\Big|_{t^n} = 2C\underline{u}^{n-1} - C\underline{u}^{n-2} + O(\Delta t^2)$$

$$\nu A\underline{u}\Big|_{t^n} = \nu A\underline{u}^n$$

BDFk/EXTk

- BDF3/EXT3 is essentially the same as BDF2/EXT2
  - $O(\Delta t^3)$ accuracy
  - essentially same cost
  - accessed by setting Torder=3 (2 or 1) in .rea file

- For convection-diffusion and Navier-Stokes, the "*EXTk*" part of the timestepper implies a CFL (Courant-Friedrichs-Lewy) constraint

$$\max_{\mathbf{x}\in\Omega} \frac{|\mathbf{u}|\Delta t}{\Delta x} \approx 0.5$$

- For the spectral element method, $\Delta x \sim N^{-2}$, which is restrictive.
  - We therefore often use a characteristics-based timestepper. (IFCHAR = T in the .rea file)

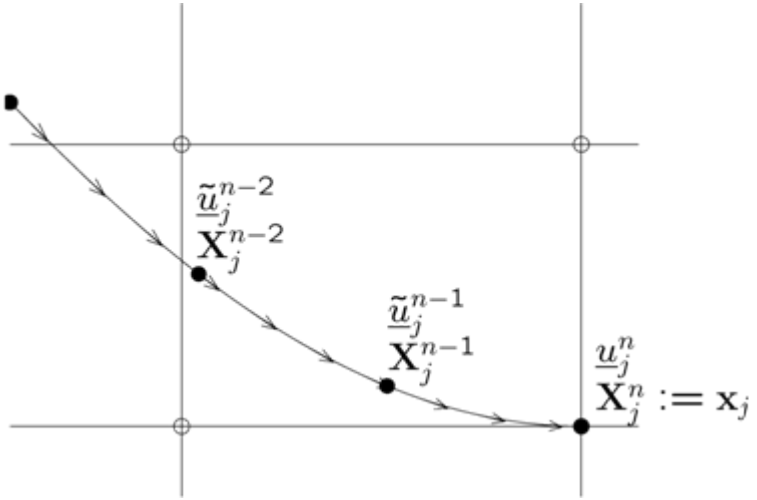
Characteristics Timestepping

- Apply BDFk to material derivative, e.g., for k=2:

$$\frac{Du}{Dt} := \frac{\partial u}{\partial t} + \mathbf{c} \cdot \nabla u$$

$$= \frac{3u^n - 4\tilde{u}^{n-1} + \tilde{u}^{n-2}}{2\Delta t} + O(\Delta t^2)$$

- Amounts to finite-differencing along the characteristic leading into $\boldsymbol{x}_j$



Characteristics Timestepping

- Δt can be >> Δt_{CFL} (e.g., Δt ~ 5-10 x Δt_{CFL})

- Don't need position (e.g., X_j^{n-1}) of characteristic departure point, only the value of $u^{n-1}(x)$ at these points.

These values satisfy the pure hyperbolic problem:

$$\frac{\partial \tilde{u}}{\partial s} + \mathbf{c} \cdot \nabla \tilde{u} = 0, \qquad s \in [t^{n-1}, t^n]$$

$$\tilde{u}(\mathbf{x}, t^{n-1}) := u^{n-1}(\mathbf{x}),$$

which is solved via explicit timestepping with Δs ~ Δt_{CFL}

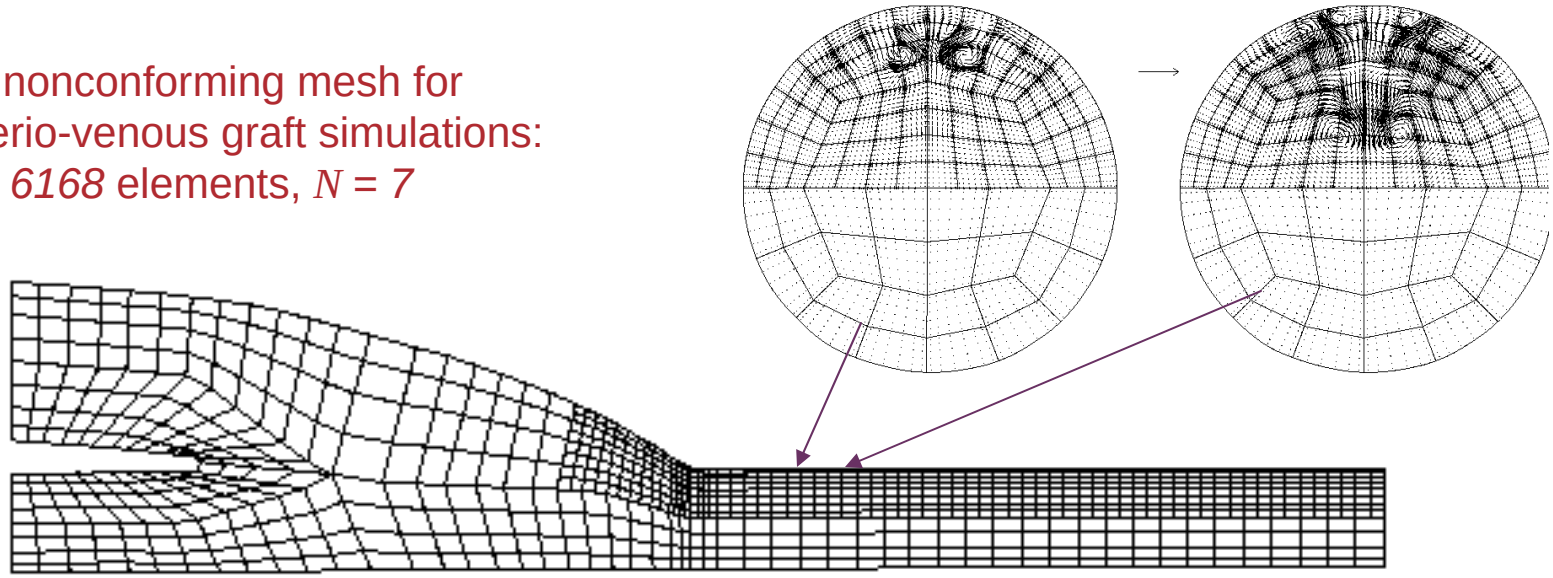
Spatial Discretization

Spectral Element Method

(Patera 84, Maday & Patera 89)

- Variational method, similar to FEM, using *GL* quadrature.
- Domain partitioned into E high-order quadrilateral (or hexahedral) elements (decomposition may be nonconforming - *localized refinement*)
- Trial and test functions represented as N th-order tensor-product polynomials within each element. (N ~ 4 -- 15, typ.)
- EN^3 gridpoints in 3D, EN^2 gridpoints in 2D.
- Converges *exponentially fast* with N for smooth solutions.

3D nonconforming mesh for arterio-venous graft simulations: $E = 6168$ elements, $N = 7$



Spectral Element Method: Poisson Example

- The SEM is a weighted residual method.
- Consider Poission eqn:

$$-\nabla^2 u = f,\ u|_{\partial}\,\Omega = 0$$

- Postulate a representation of the solution, e.g.,

$$u(\mathbf{x}) = \sum_{j=1}^{n} u_j \phi_j(\mathbf{x}),$$

and insist that the residual $r(\mathbf{x}) := f + \nabla^2 u$ be orthogonal to all functions in the approximation space:

$$\int_{\Omega} v(f + \nabla^2 u) d\mathbf{x} = 0, \quad \text{for } v = \phi_i,\ i = 1, \ldots, n$$

Spectral Element Method: Poisson Example

Integrate 2nd-order term by parts:

$$-\int_\Omega v\nabla^2 u d\mathbf{x} = \int_\Omega \nabla v \cdot \nabla u d\mathbf{x} - \int_{\partial\Omega} v\nabla u \cdot \hat{\mathbf{n}} dA$$

- Surface integral vanishes because of boundary conditions.

- Rearranging and inserting basis functions yields:

$$\sum_{j=1}^{n} a_{ij}u_j = \sum_j b_{ij}f_j \iff A\underline{u} = B\underline{f}$$

$$\begin{aligned} a_{ij} &:= \int_\Omega \nabla\phi_i \cdot \nabla\phi_j d\mathbf{x} && \text{stiffness matrix} \\ b_{ij} &:= \int_\Omega \phi_i\phi_j d\mathbf{x} && \text{mass matrix} \\ f(\mathbf{x}) &:= \sum_{j=1}^{n} f_j\phi_j(\mathbf{x}) \end{aligned}$$

SEM Function Representation

- Key point is that there is a continuous representation of all variables:

$$u(\mathbf{x}) = \sum_{j=1}^{n} u_j \phi_j(\mathbf{x})$$

- Since $\phi_j(\mathbf{x})$ is known a priori, we know how to differentiate and integrate.

- Moreover, choose ϕ_js to be computationally convenient

1D Basis Functions

$$u(x) := \sum_{i=0}^{N} u_i h_i(x), \quad h_i(x) \in \mathbb{P}_N$$

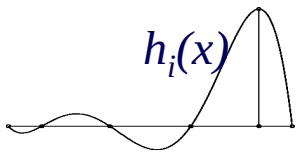
- Stable high-order basis for N[th]-order polynomial approximation space:
  - poor choices:

    $h_i(x) = x^i$

    $h_i(x) =$ Lagragian interpolant on uniform points, $x_i = i \cdot \Delta x$
  - good choices:

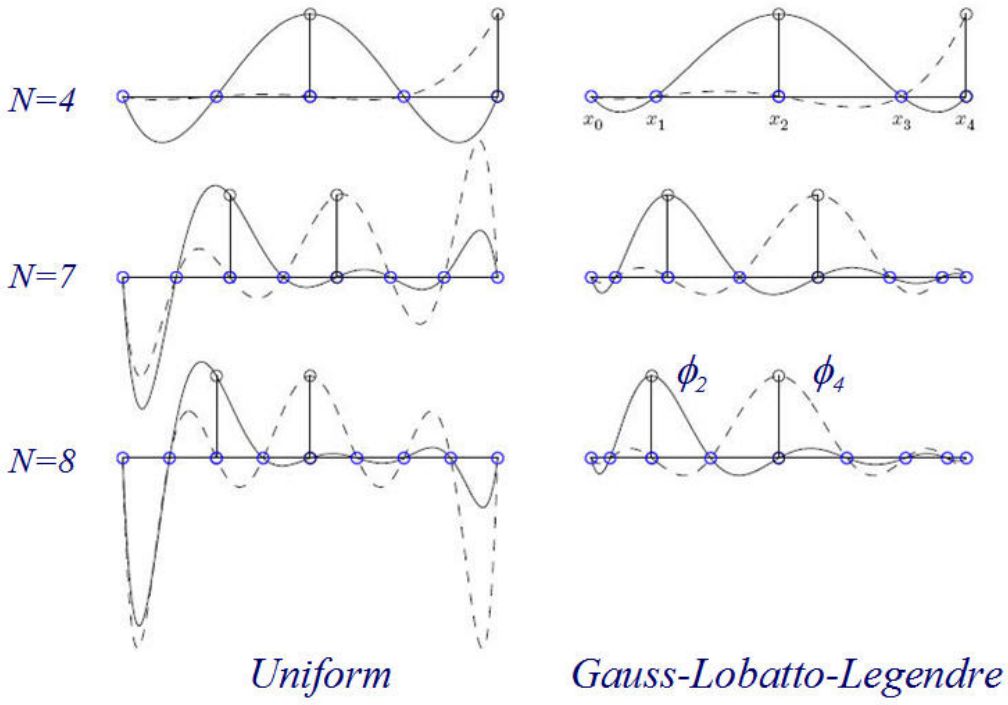
    $h_i(x) = L_i(x)$ (any orthogonal polynomial)

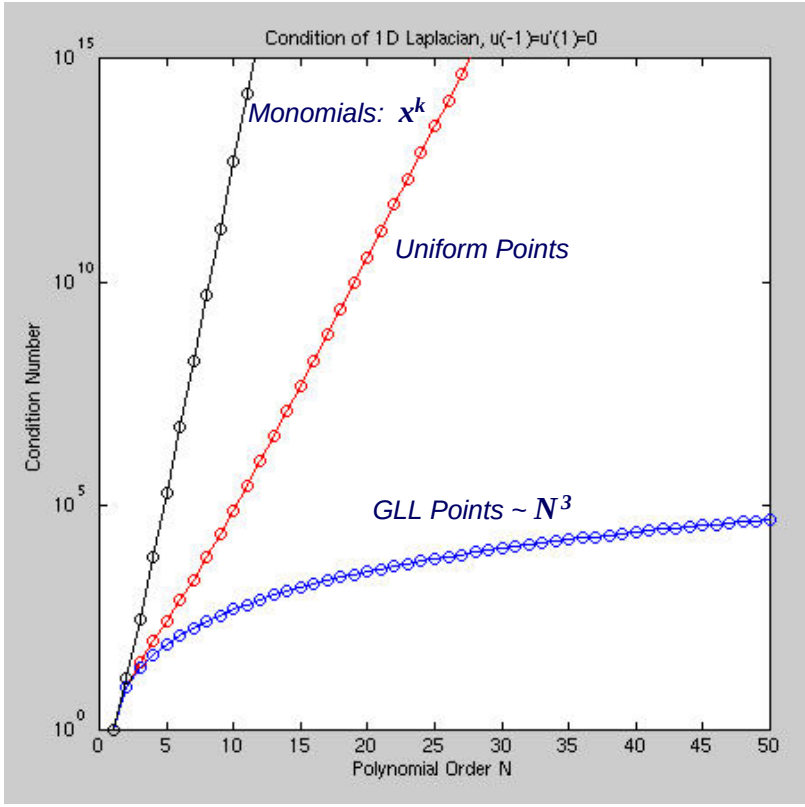
    $h_i(x) =$ Lagragian interpolant on Gauss points



Condition Number of 1D Stiffness Matrix

GLL Nodal Basis → good conditioning, minimal round-off error





Integration (projection) in 1D

- Replace integrals with Gauss-Lobatto-Legendre quadrature:

$$\int_\Omega \nabla v \cdot \nabla u \, dV = \int_\Omega v f \, dV$$

with

$$(\nabla v, \nabla u)_N = (v, f)_N$$

where

$$(f, g)_N := \sum_{k=0}^{N} \rho_k f(\xi_k) g(\xi_k),$$

$\{\xi_k\}$ = Gauss-Lobbato-Legendre quadrature points

$\{\rho_k\}$ = Gauss-Lobbato-Legendre quadrature weights

- Yields a diagonal mass matrix; preserves spectral accuracy.
  (However, beware stability issues….)

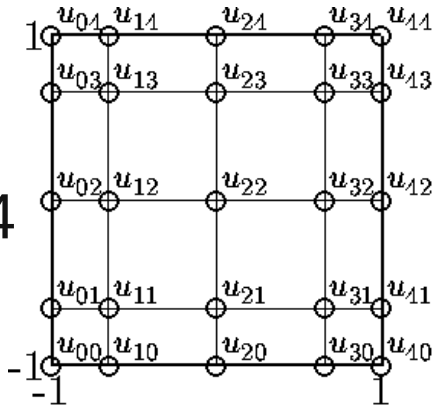
Extension to 2D

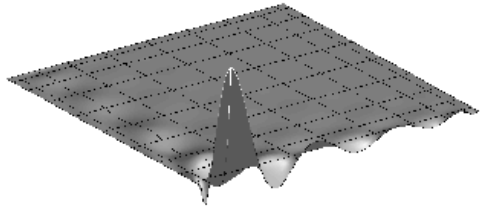
Nodal bases on the Gauss-Lobatto-Legendre points:

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

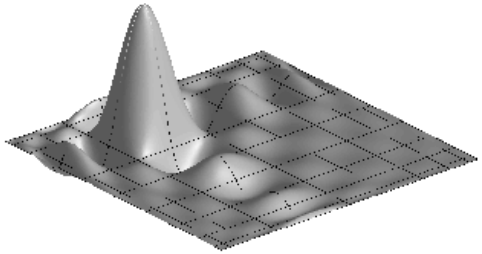
basis coefficients

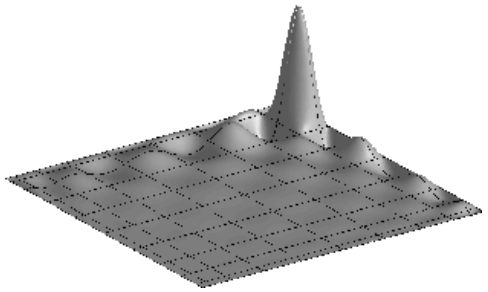
N=4





N=10

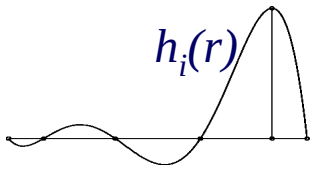




Matrix-Matrix Based Derivative Evaluation

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

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

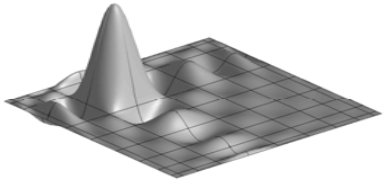


allows derivatives to be evaluated as matrix-matrix products:

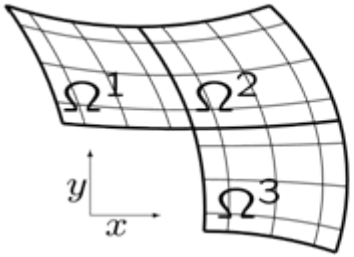
$$\left.\frac{\partial u}{\partial r}\right|_{\xi_i,\xi_j} = \sum_{p=0}^{N} u_{pj} \left.\frac{dh_p}{dr}\right|_{\xi_i} = \sum_{p} \underbrace{\hat{D}_{ip} u_{pj}}_{mxm} =: D_r \underline{u}$$

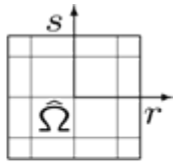
- *In 3D, ~90% of all ops in mxm – high optimization important.*

Mapped Geometries



2D basis function, N=10





E=3, N=4

Geometry takes same form as solution,

$$\mathbf{x}(r,s) = \sum_{i=0}^{N} \sum_{j=0}^{N} \mathbf{x}_{ij} h_i(r) h_j(s),$$

with $\mathbf{x} := (x, y)$, from which

$$\left.\frac{\partial x}{\partial r}\right|_{\xi_i,\xi_j} = \sum_p D_{ip} x_{pj}$$

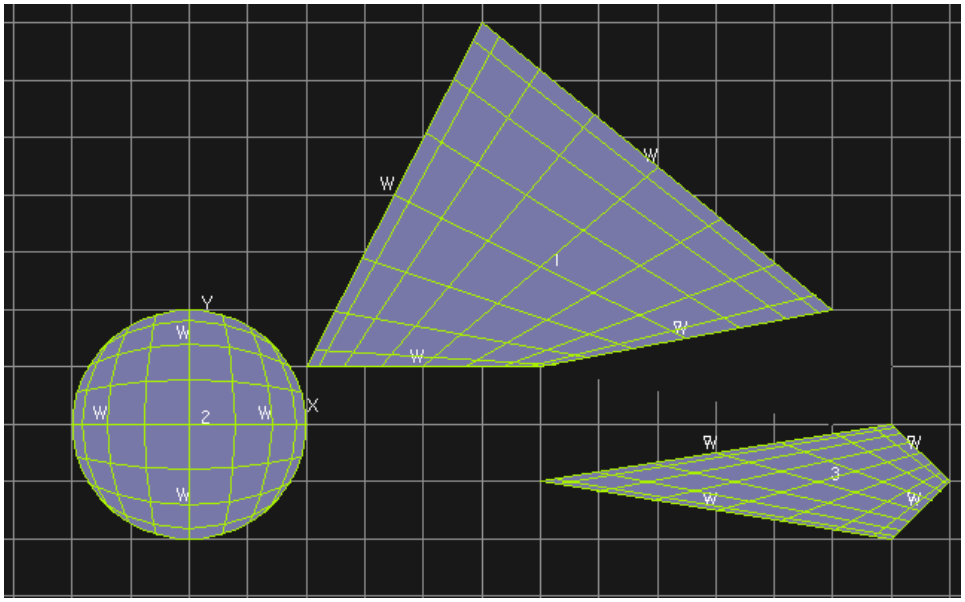
Given $\frac{\partial x_i}{\partial r_j}$, we can find $\frac{\partial r_i}{\partial x_j}$, and thus use the chain rule, e.g.,

$$\frac{\partial u}{\partial x} = \frac{\partial u}{\partial r}\frac{\partial r}{\partial x} + \frac{\partial u}{\partial s}\frac{\partial s}{\partial x}.$$

In *Nek*: "`call gradm1(ux,uy,uz,u)`" assuming `ux,uy,uz,u` properly declared.

Notes about Mapped Elements

- Best to use affine (i.e., linear) transformations in order to preserve underlying GLL spacing for stability and accurate quadrature.
- Avoid singular corners - ~180^{o} or ~0^{o}
- Avoid high-aspect-ratio cells, if possible



Multidimensional Integration

- *Given that we have Lagrangian interpolants based on GLL quadrature points, we have*

$$\begin{aligned}
\int_\Omega vu d\mathbf{x} &\approx \sum_{k=1}^{n} \rho_k v(\xi_k) u(\xi_k) \\
&= \sum_k \rho_k \left(\sum_{i=1}^{n} v_i \phi_i(\xi_k) \right) \left(\sum_{j=1}^{n} u_j \phi_j(\xi_k) \right) \\
&= \sum_k \rho_k \left(\sum_{i=1}^{n} v_i \delta_{ik} \right) \left(\sum_{j=1}^{n} u_j \delta_{jk} \right) \\
&= \sum_k \rho_k v_k u_k \;=\; \underline{v}^T B \underline{u}, \;\; b_{ij} := \delta_{ij} \rho_i.
\end{aligned}$$

- *In particular,*

$$\int_\Omega u d\mathbf{x} = \sum_k \rho_k u_k \;=\; \underline{b}^T \underline{u}, \;\; b_i := \rho_i.$$

- In Nek, this ***vector reduction*** is implemented as: *alpha = glsc2(u,bm1,n)*

Navier-Stokes Discretization Options

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \nu \nabla^2 \mathbf{u}$$
$$\nabla \cdot \mathbf{u} = 0$$

- Imposition of the constraint div **u** = 0 is a major difficulty in solving the incompressible Navier-Stokes equations, both from theoretical and implementation perspectives.
- Was not well-understood till the mid-80s (give, or take…).
- The fundamental difficulty is that the discrete operators do not commute, except under special circumstances (e.g., Fourier bases).
- Nek supports two distinct approaches:
  - Option 1 (P_N-P_{N-2}):
    - *discretize in space using compatible approximation spaces*
    - *solve coupled system for pressure/velocity*
  - Option 2 (P_N-P_N, or *splitting*):
    - *discretize in time first*
    - *take continuous divergence of momentum equation to arrive at a Poisson equation for pressure, with special boundary conditions*

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

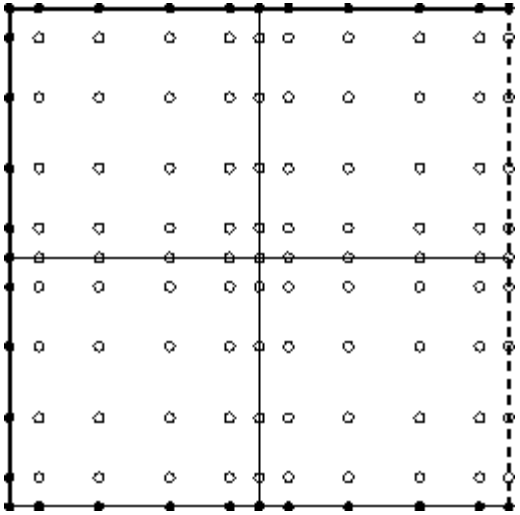
WRT: *Find* $\mathbf{u} \in X^N, p \in Y^N$ *such that:*

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

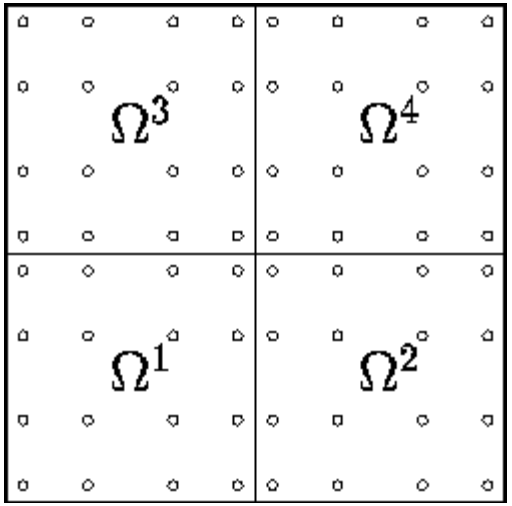
$$- (q, \nabla \cdot \mathbf{u})_G = 0 \qquad \forall\, q \in Y^N \subset L^2$$

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

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



Gauss-Lobatto Legendre points (velocity)



Gauss Legendre points (pressure)

Consistent Splitting for Unsteady Stokes

(MPR 90, Blair-Perot 93, Couzy 95)

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

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

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

- ***E*** - consistent Poisson operator for pressure, SPD
  - boundary conditions applied in velocity space
  - most compute-intensive phase

Comparison of $P_N - P_{N-2}$ and $P_N - P_N$ Options in Nek

	$P_N - P_{N-2}$	$P_N - P_N$
– SIZE:	lx2=lx1-2	lx2=lx1
– pressure:	discontinuous	continuous
– solver:	$E = DB^{-1}D^T$	A (std. Laplacian)
– preconditioner:	SEMG	Schwarz (but to be upgraded)
– free-surface	Yes	No
– ALE	Yes	No
– low Mach	No	Yes
– LES	OK	Better
– low Re	Better	OK
– var. prop.	Implicit (stress formulation)	semi-implicit
– spectrally accurate	Yes	Yes

- Nek will ensure that the problem type is compatible with the discretization choice.
- For most cases, speed is determined by the pressure solve, which addresses the fastest timescales in the system (the acoustic waves).
  - For $P_N - P_{N-2}$, the solver has been highly optimized over the last 15 years.
  - The $P_N - P_N$ version was developed by the ETH group (Tomboulides, Frouzakis, Kerkemeier) for low Mach-number combustion and has only recently been folded into the production Nek5000 code.

Navier-Stokes Boundary Conditions

- A few key boundary conditions are listed below.

cbc	name	condition
v	velocity	specified in .usr
V	velocity	specified in .rea
W	wall	$\mathbf{u} = 0$
O	outflow	$\frac{\partial \mathbf{u}}{\partial \mathbf{n}} = 0,\ p = 0$
SYM	symmetry	$\frac{\partial u_t}{\partial \mathbf{n}} = 0,\ u_n = 0$
P	periodic	$\mathbf{u}(\mathbf{x}) = \mathbf{u}(\mathbf{x} + \mathbf{L})$

- There are many more, particularly for moving walls, free surface, etc.

- Special conditions include:
  - Recycling boundary conditions (special form of “v”)
  - Accelerated outflow to avoid incoming characteristics

Thermal Boundary Conditions

- A few key boundary conditions are listed below.

cbc	name	condition
t	temperature	specified in .usr
T	temperature	specified in .rea
I	insulated	$\frac{\partial T}{\partial \mathbf{n}} = 0$
f	flux	$k\frac{\partial T}{\partial \mathbf{n}} = f$
c	Newton cooling	$k\frac{\partial T}{\partial \mathbf{n}} = h(T - T_\infty)$
O	outflow	$\frac{\partial T}{\partial \mathbf{n}} = 0$
P	periodic	$T(\mathbf{x}) = T(\mathbf{x} + \mathbf{L})$

Filtering, Dealiasing, and LES

- LES, or more properly, subgrid scale (sgs) stress models invariably rely on dissipation to remove energy that otherwise piles up at the gridscale. WHY?

Stabilizing High-Order Methods

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

Some options:

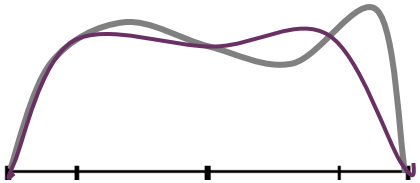
- high-order upwinding (e.g., DG, WENO)
- bubble functions
- spectrally vanishing viscosity
- *filtering*
- *dealiasing*

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)

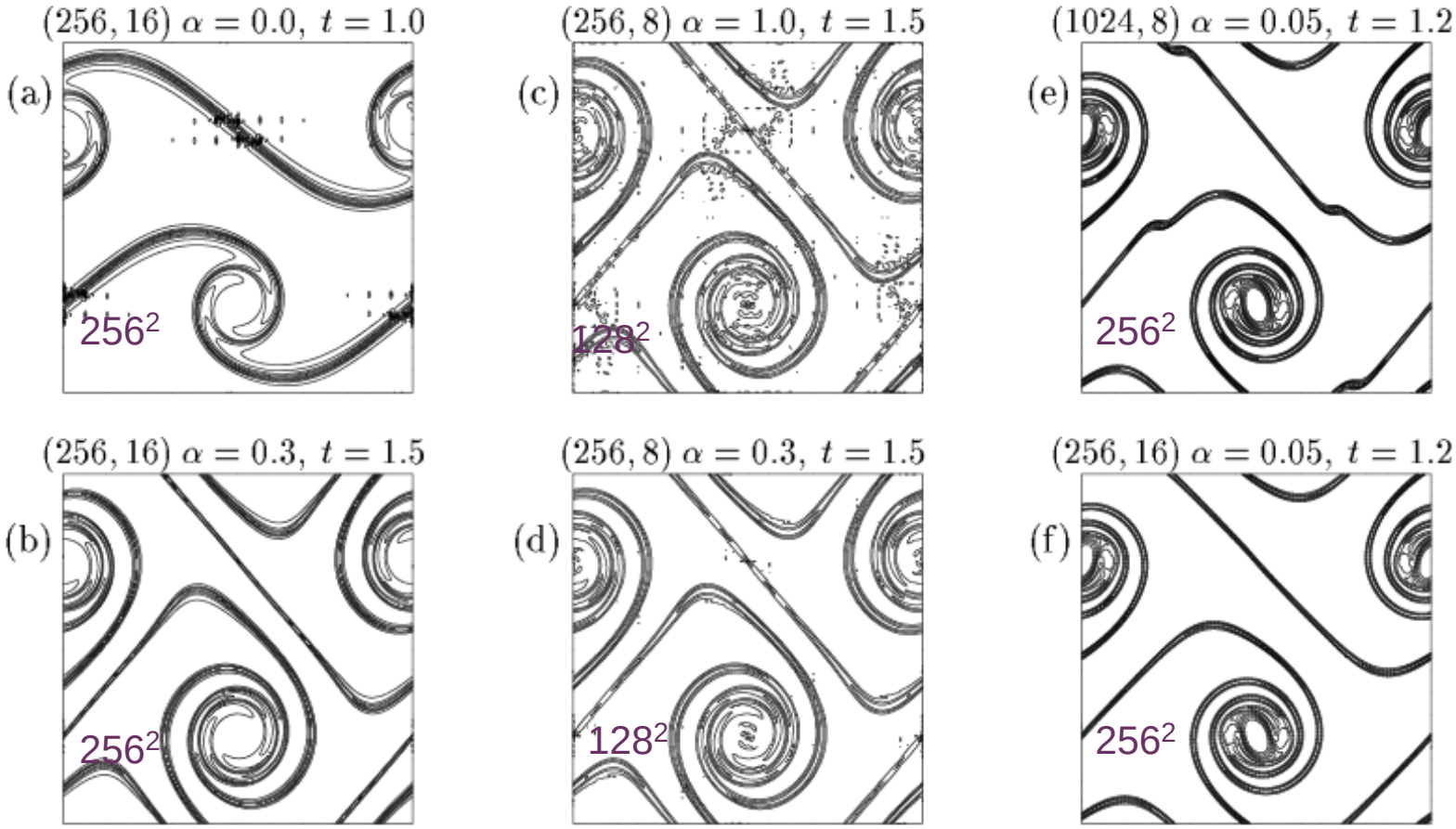
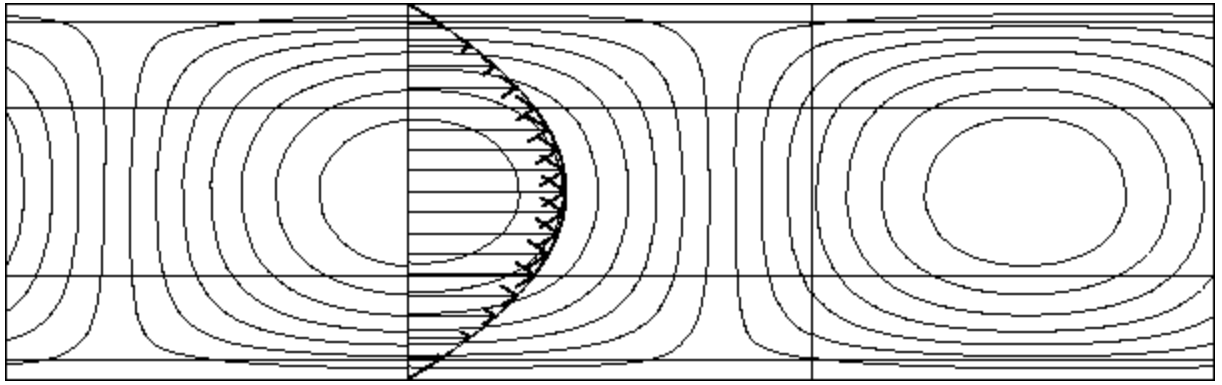


Figure 1: Vorticity for different (K, N) pairings: (a–d) $\rho = 30$, $Re = 10^5$, contours from -70 to 70 by 140/15; (e–f) $\rho = 100$, $Re = 40,000$, contours from -36 to 36 by 72/13. (cf. Fig. 3c in [4]).

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*

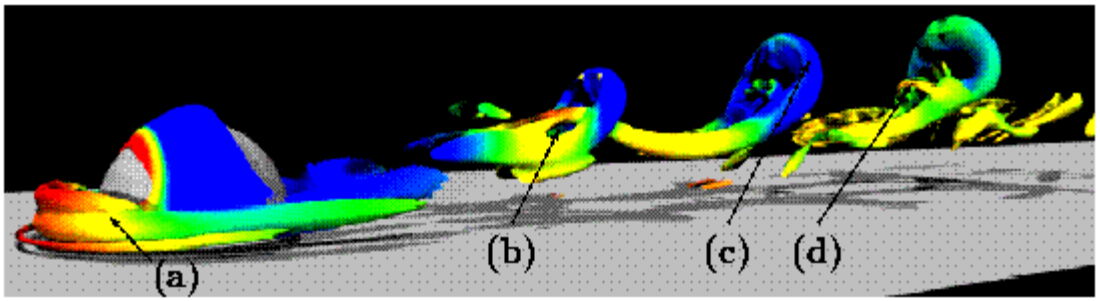
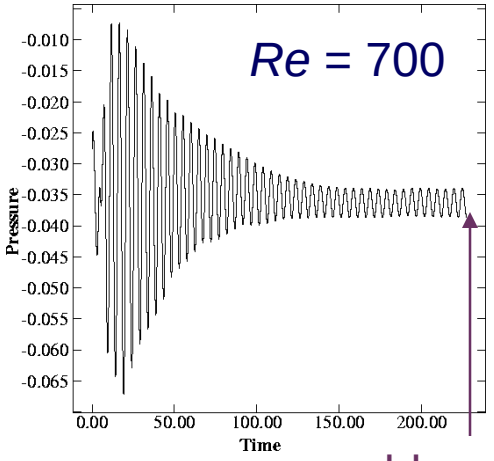
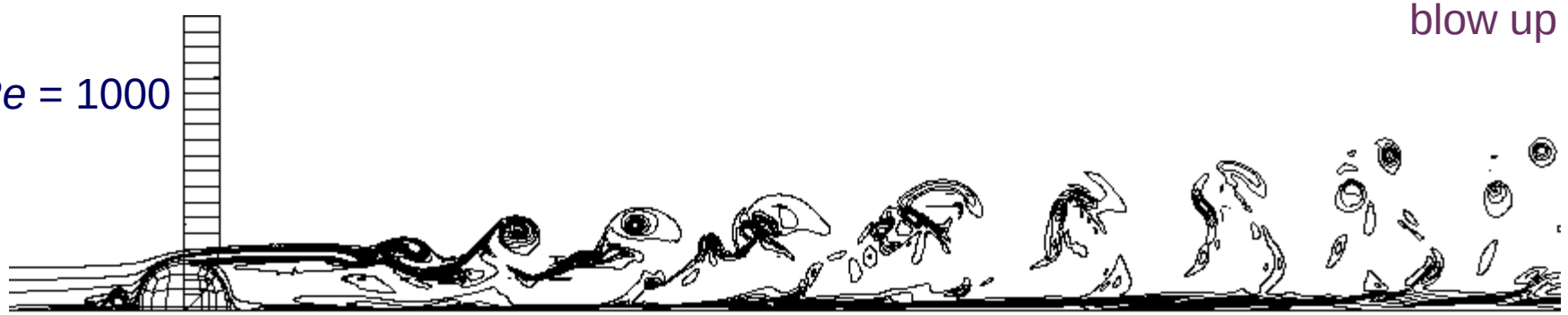


Figure 1: Principal vortex structures identified by $\lambda_2 = -1$ isosurfaces at $Re_\lambda = 760$: standing horseshoe vortex (a), interlaced tails (b), hairpin head (c), and bridge (d). Colors indicate pressure. (K=1021, $N = 15$).

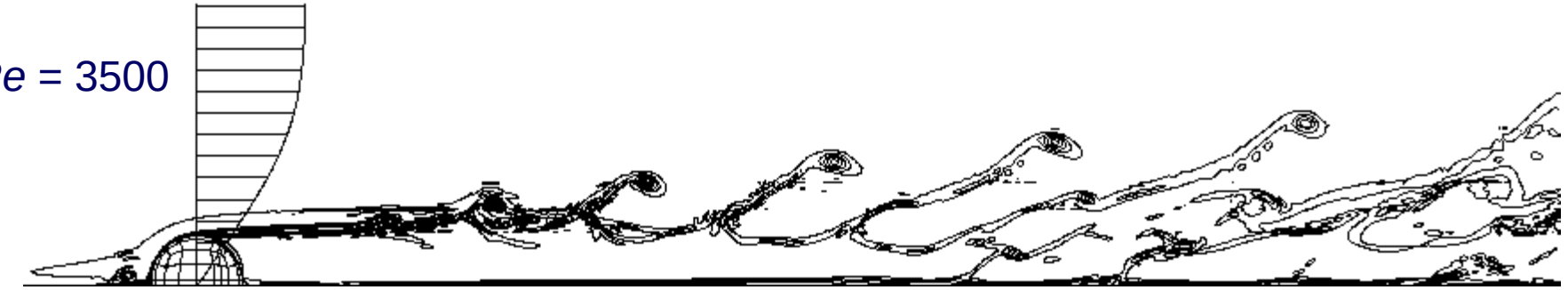


blow up

Re = 1000

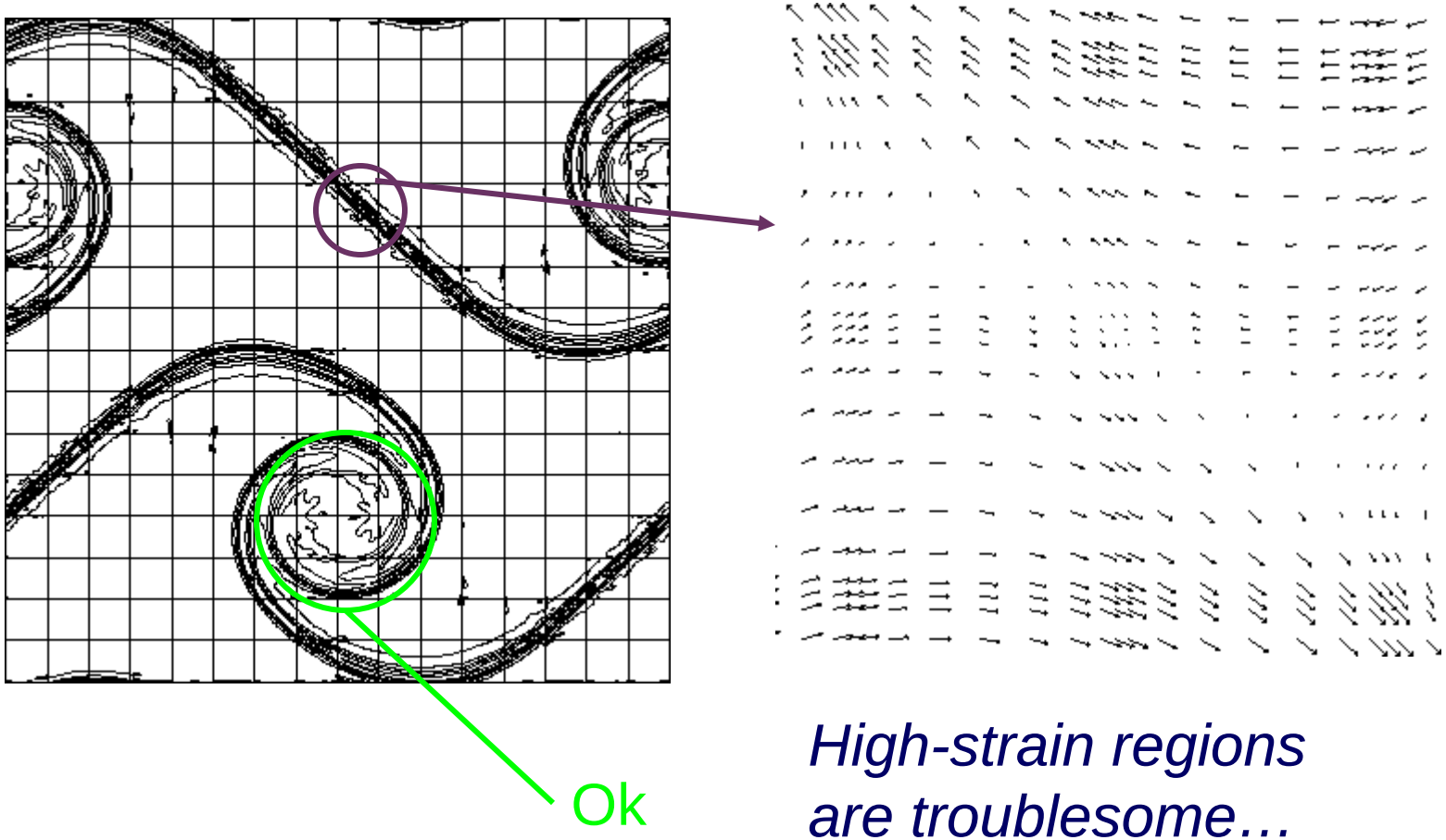


Re = 3500



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

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

$$C_{ij} = \int_\Omega \phi_i \, \mathbf{c} \cdot \nabla \phi_j \, dV$$

$$= -\int_\Omega \phi_j \, \mathbf{c} \cdot \nabla \phi_i \, dV - \int_\Omega \phi_j \phi_j \nabla \cdot \mathbf{c} \, dV$$

$$= \text{skew symmetric, if } \nabla \cdot \mathbf{c} \equiv 0.$$

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

Discrete problem should never blow up.

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

Weighted residual formulation vs. spectral element method:

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

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

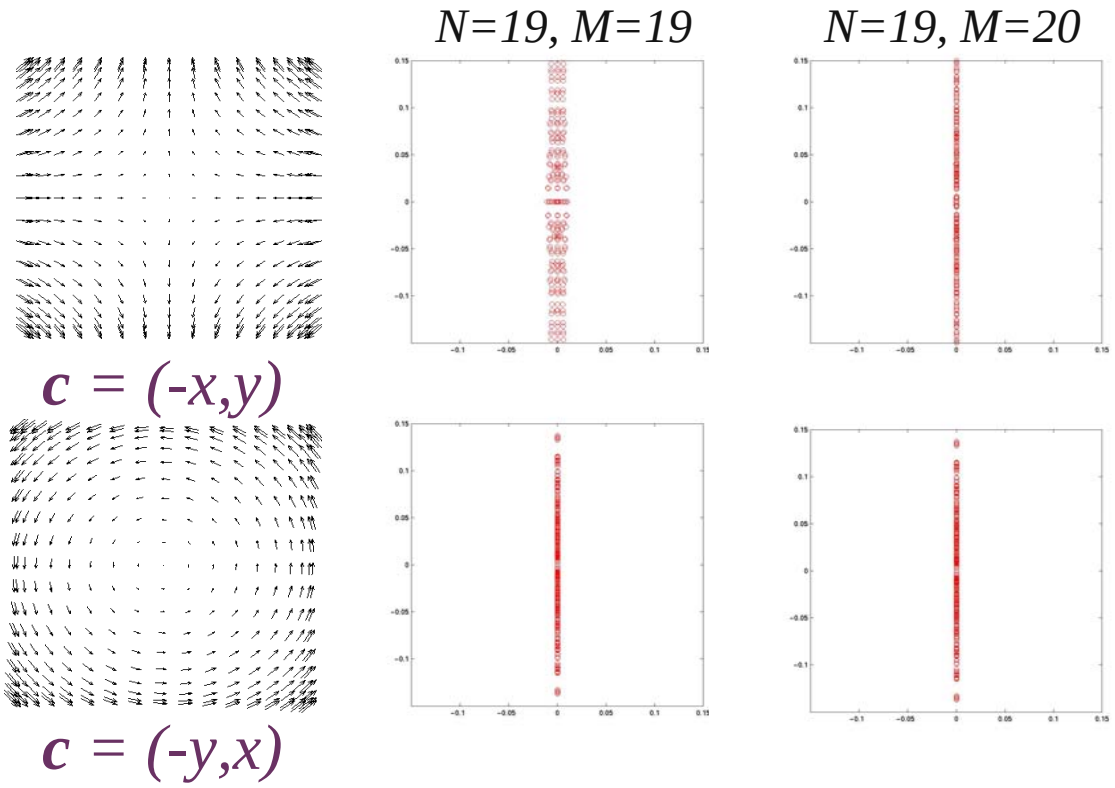
This suggests the use of over-integration (dealiasing) to ensure that skew-symmetry is retained

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

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

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.

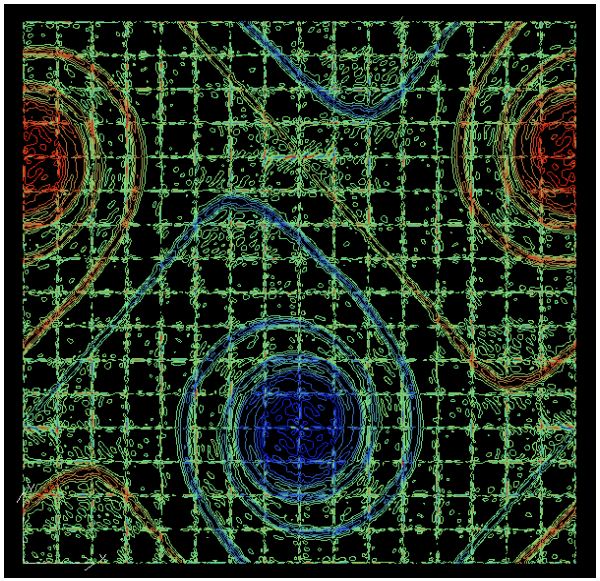


Stabilization Summary

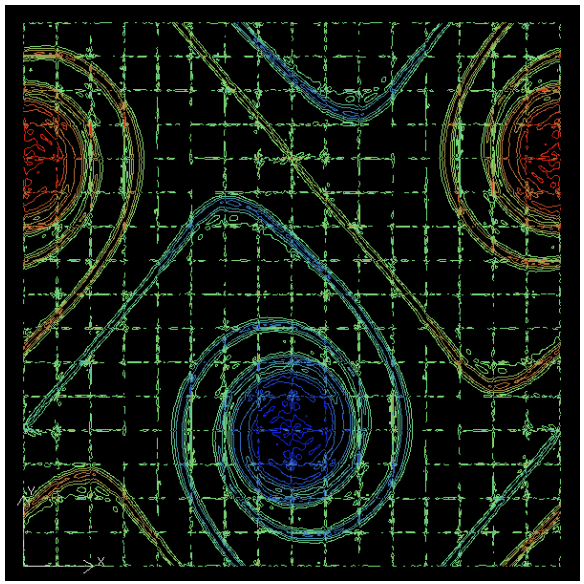
- Filtering acts like well-tuned hyperviscosity
  - Attacks only the fine scale modes (that, numerically speaking, shouldn't have energy anyway…)
  - Can precisely identify which modes in the SE expansion to suppress (unlike differential filters)
  - Does not compromise spectral convergence
- Dealiasing 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

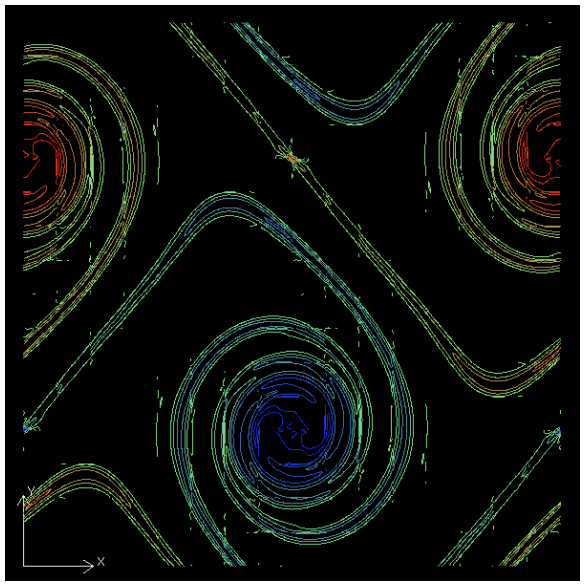
$\nu = 0$, no filter



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



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



Linear Solvers

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, $\boldsymbol{A}$

Pressure Solution Strategy: $\mathbf{E\underline{p}^n = \underline{g}^n}$

1. *Projection: compute best approximation from previous time steps*
   - *Compute* $\underline{p}^*$ *in span*$\{ \underline{p}^{n-1}, \underline{p}^{n-2}, \ldots , \underline{p}^{n-l} \}$ *through straightforward projection.*
   - *Typically a 2-fold savings in Navier-Stokes solution time.*
   - *Cost: 1 (or 2) matvecs in E per timestep*
2. *Preconditioned CG or GMRES to solve*

$$E\,\Delta\underline{p} = \underline{g}^n - E\,\underline{p}^*$$

Initial Guess for $\mathbf{A\underline{x}^n = \underline{b}^n}$ *via Projection* (~~A=E, SPD)~~

Given $\cdot\ \underline{b}^n$

$\cdot\ \{\tilde{\underline{x}}_1, \ldots, \tilde{\underline{x}}_l\}$ satisfying $\tilde{\underline{x}}_i^T A \tilde{\underline{x}}_j = \delta_{ij}$,

$\cdot$ Set $\bar{\underline{x}} := \Sigma \alpha_i \tilde{\underline{x}}_i, \quad \alpha_i = \tilde{\underline{x}}_i^T \underline{b}$ (best fit solution)

$\cdot$ Set $\Delta \underline{b} := \underline{b}^n - A\bar{\underline{x}}$

$\cdot$ Solve $A\Delta \underline{x} = \Delta \underline{b}$ to $tol\ \epsilon$ (black box solver)

$\cdot\ \underline{x}^n := \bar{\underline{x}} + \Delta \underline{x}$

$\cdot$ If $(l = l_{\max})$ then

$\tilde{\underline{x}}_1 = \underline{x}^n / ||\underline{x}^n||_A$

$l = 1$

else

$\tilde{\underline{x}}_{l+1} = (\Delta \underline{x} - \Sigma \beta_i \tilde{\underline{x}}_i)/(\Delta \underline{x}^T A \Delta \underline{x} - \Sigma \beta_i^2)^{\frac{1}{2}}, \quad \beta_i = \tilde{\underline{x}}_i A \Delta \underline{x}$

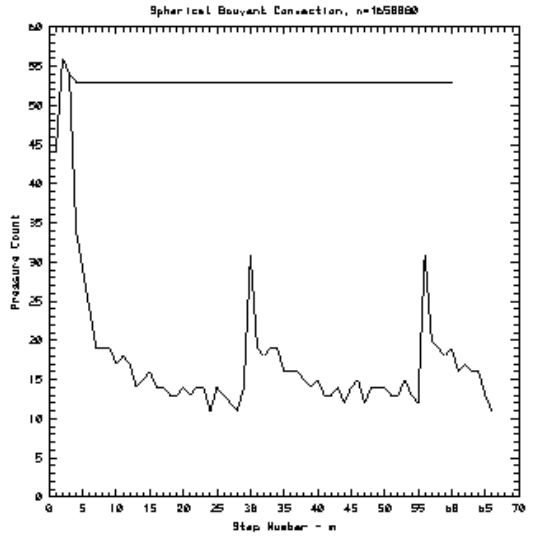
$l = l + 1$

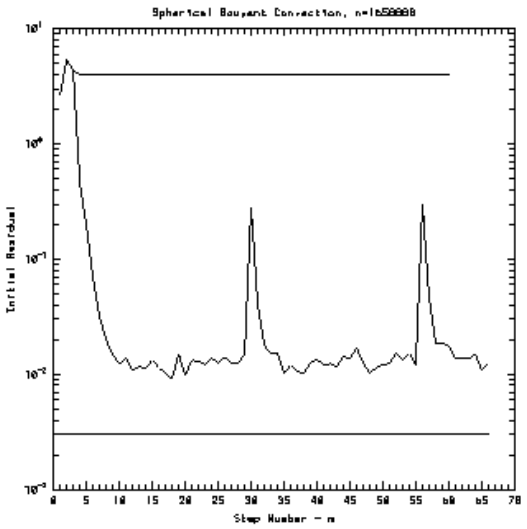
endif

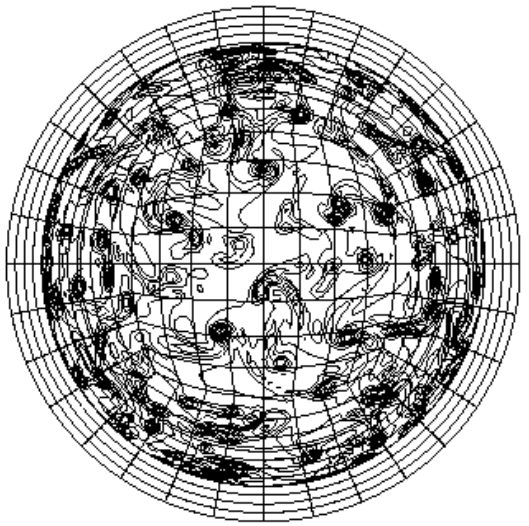
Initial guess for $\mathbf{E}\underline{\mathbf{p}}^{\mathbf{n}} = \mathbf{g}^{\mathbf{n}}$ *via projection onto previous solutions*

ICASE TR '93; CMAME '98

- $\| \underline{p}^n - \underline{p}^* \|_E = O(\Delta t^l) + O(\varepsilon_{\text{tol}})$
- two additional mat-vecs per step
- storage: $2+l_{max}$ vectors
- results with/without projection (1.6 million pressure nodes)





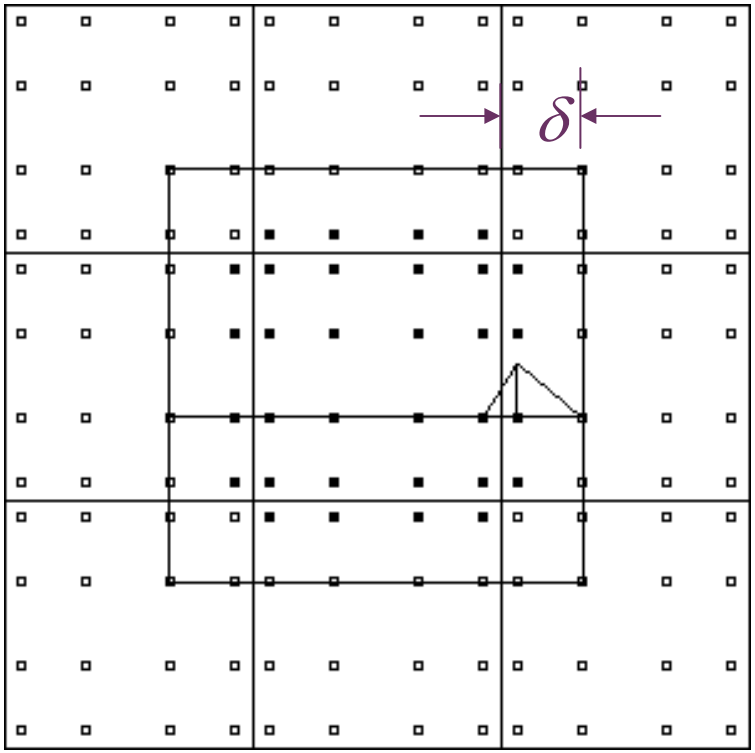


- 4 fold reduction in iteration count, 2 – 4 in typical applications

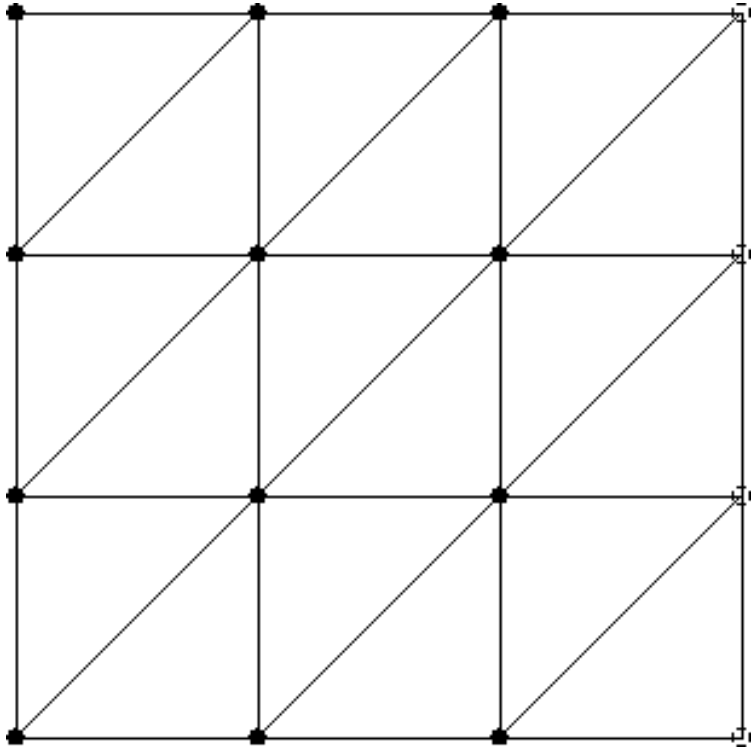
Overlapping Additive Schwarz Preconditioner / Smoother

(Dryja & Widlund 87, Pahl 93, PF 97, FMT 00)

$$\underline{z} = M\underline{r} = \sum_{e=1}^{E} R_e^T A_e^{-1} R_e \underline{r} + R_0^T A_0^{-1} R_0 \underline{r}$$



***Local Overlapping Solves**: FEM-based Poisson problems with homogeneous Dirichlet boundary conditions, A_e .*

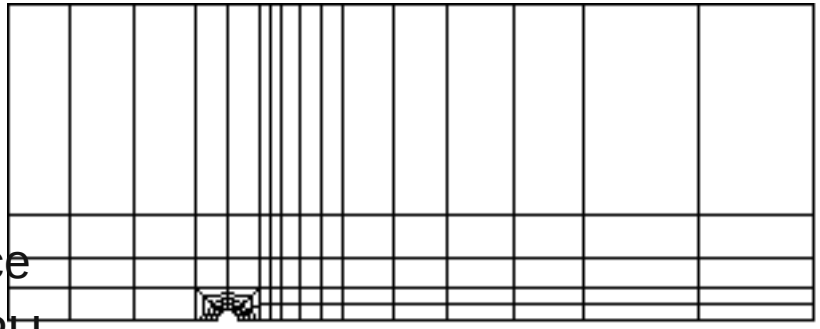


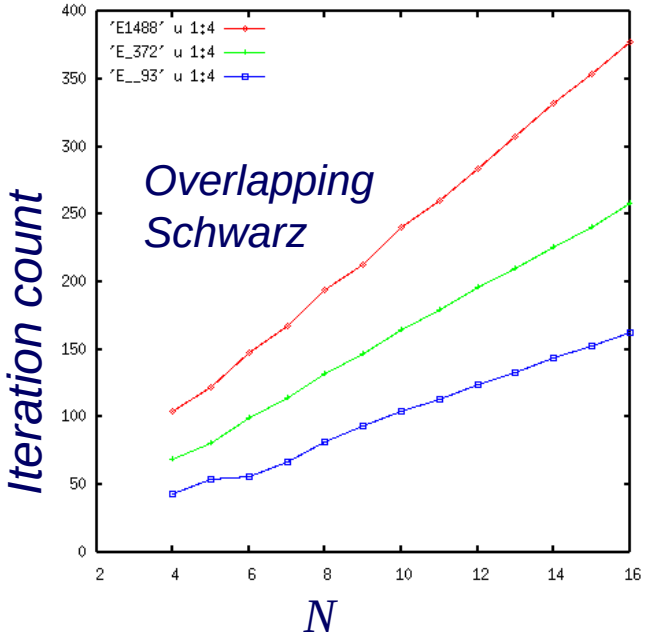
***Coarse Grid Solve**: Poisson problem using linear finite elements on entire spectral element mesh, A_0 (GLOBAL).*

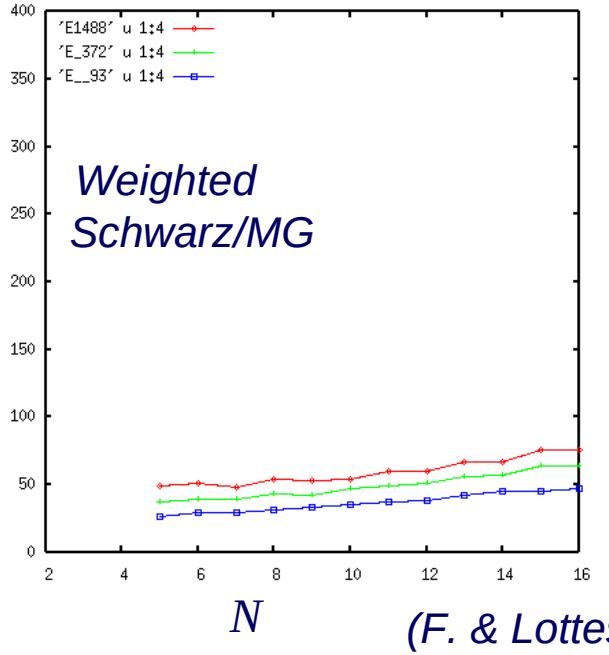
E-*Based Schwarz vs. SEMG for* High-Aspect Ratio *Elements*

- Base mesh of E=93 elements
  - Quad refine to generate E=372 and E=1488 elements,
  - N=4,…,16
  - SEMG reduces E and N dependence
  - *2.5 X reduction* in Navier-Stokes CPU time for N=16

2D Navier-Stokes Model Problem



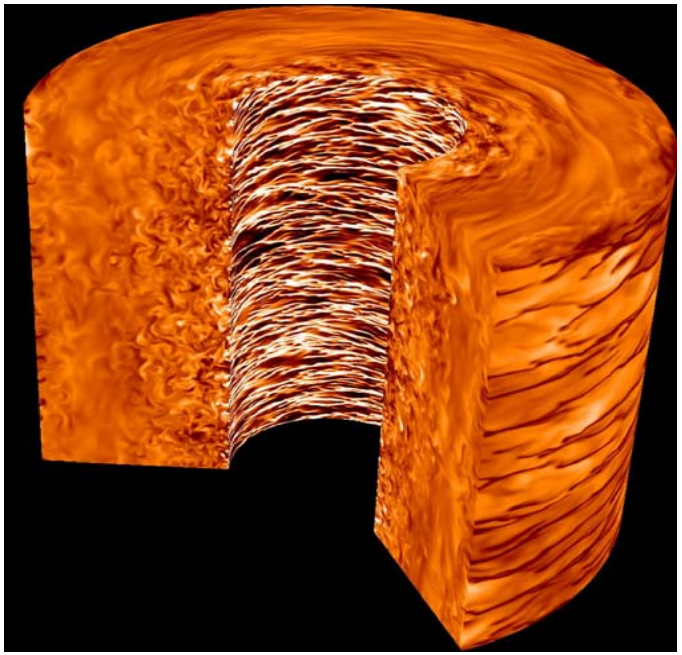


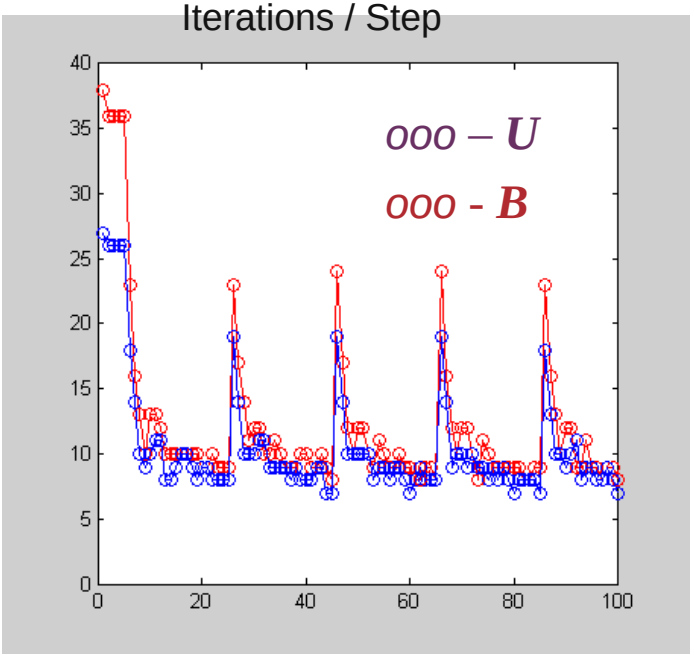


(F. & Lottes, 2004)

Solver Performance: Hybrid Schwarz-Multigrid

- *Magneto-rotational instability* *(Obabko, Cattaneo & F.)*
  - *E=140000, N=9 (n = 112 M), P=32768 (BG/L)*
  - *~ 1.2 sec/step*
  - *~ 8 iterations / step for U & B*
  - ***Key is to have a scalable coarse-grid solver***





AMG vs. XX^T Performance

Navier-Stokes Solution time break down for n=120 M, n_c = 417000

Case/P	Total	QQ^T	Coarse	all_reduce()
x4096	1994	125	1180	1.2
a4096	1112	125	192	1.4
b4096	846	126	25	1.
8192	460	88	22	1.
16384	266	64	20	1.

- Cannot consider XX^T on larger problems.
- "a4096" case is relies on pairwise + all_reduce
  - First version (not shown, pairwise-only), was not much faster than XX^T. Why?

Outline

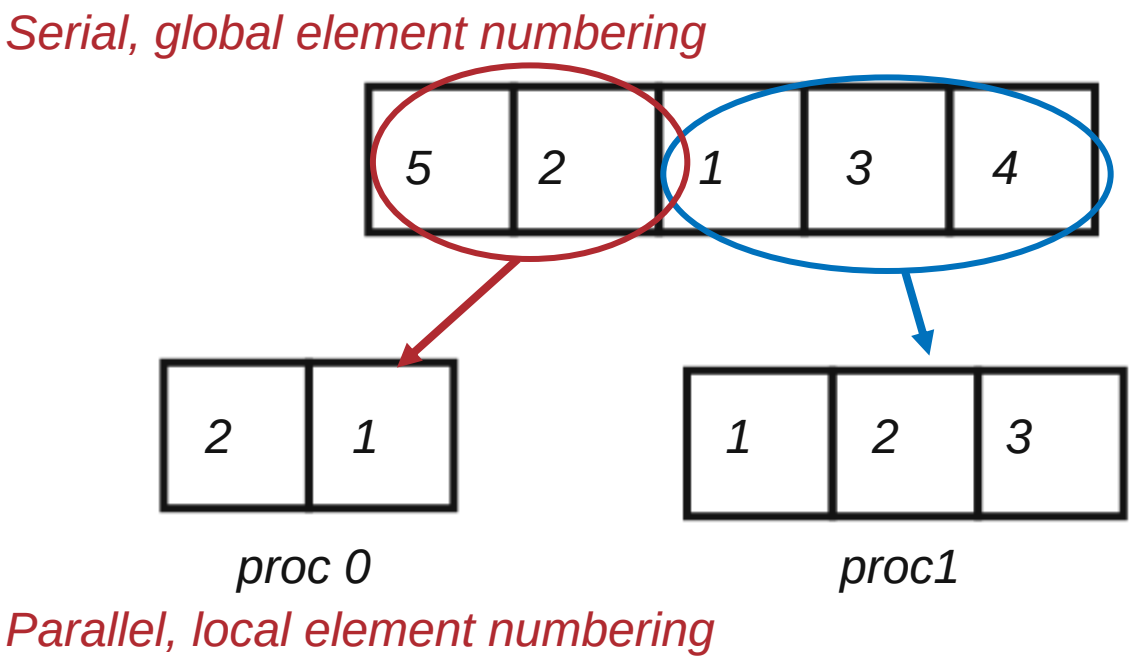
- *Nek5000 capabilities*
- *Equations, timestepping, and SEM basics*
- *Parallel / serial issues that you should understand*

Serial / Parallel Issues

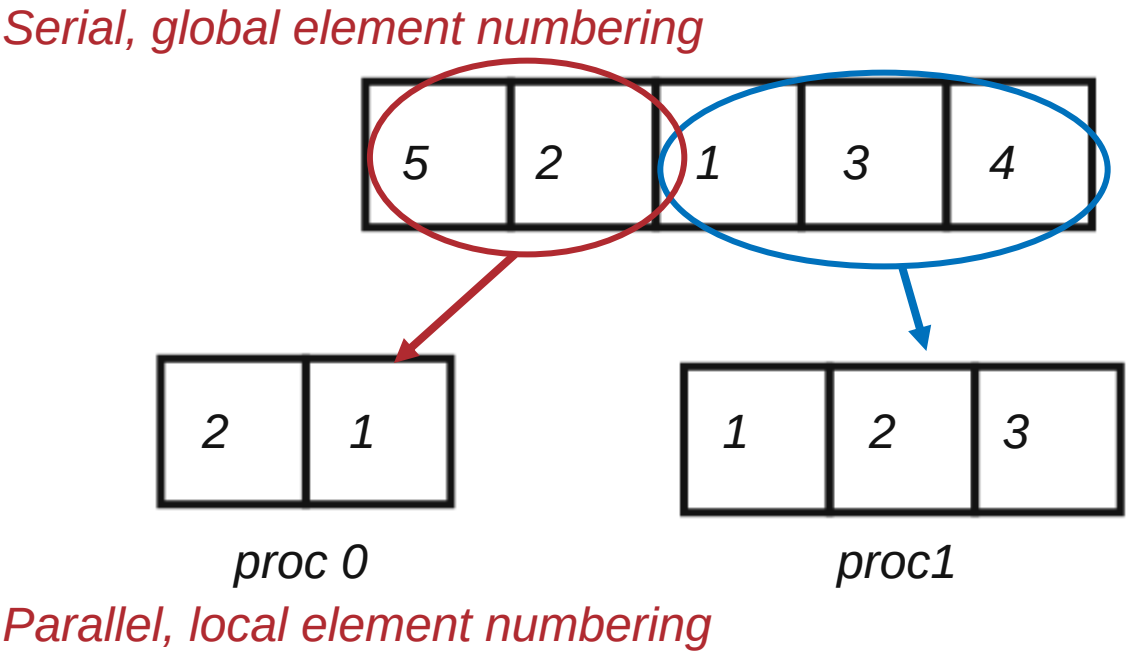
- *Locally, the SEM is **structured.***
- *Globally, the SEM is **unstructured**.*
- *Vectorization and serial performance derive from the structured aspects of the computation.*
- *Parallelism and geometric flexibility derive from the unstructured, element-by-element, operator evaluation.*
- *Elements, or groups of elements are distributed across processors, but an element is never subdivided.*

Parallel Structure

- Elements are assigned in ascending order to each processor



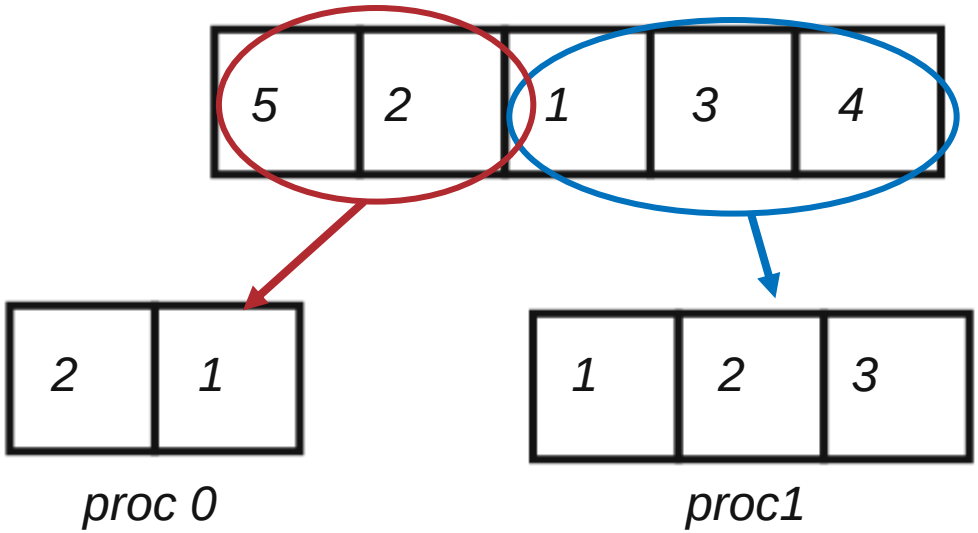
Parallel Structure



- For the most part, don't care about global element ***numbering***
  - *(We'll show some examples where one might)*

- *Key point is that,*
  - *on proc 0, nelt=2* *(nelt = # elements in temperature domain)*
  - *on proc 1, nelt=3* *(nelv = # elements in fluid domain, usually = nelt)*

Parallel Structure



- Arrays that distinguish which processor has which elements:
  - proc 0
    - *nelt=2*
    - *lglel=(2,5)*

  proc 1

  nelt=3

  lglel=(1,3,4)

- Common arrays (scaling as nelgt, but only two such arrays):
  - gllel=(1,1,2,3,2), gllnid=(1,0,1,1,0)

Serial Structure

- All data contiguously packed (and quad-aligned):

  real u(lx1,ly1,lz1,lelt)

  - *Indicates that u is a collection of elements, e=1,…,Nelt =< lelt, each of size* $(N+1)^d$*, d=2 or 3*

Serial / Parallel Usage

- A common operation (1st way…)

```
s=0
do e=1,nelv
do iz=1,nz1
do iy=1,ny1
do ix=1,nx1
     s=s+u(ix,iy,iz,e)
enddo,…,enddo
```

- Parallel Version

```
s=0
do e=1,nelv
do iz=1,nz1
do iy=1,ny1
do ix=1,nx1
     s=s+u(ix,iy,iz,e)
enddo,…,enddo

s=glsum(s,1)
```

Serial / Parallel Usage

- A common operation (2nd way…)

```
n=nx1*ny1*nz1*nelv
s=0
do i=1,n
     s=s+u(i,1,1,1)
enddo
```

- Parallel Version

```
n=nx1*ny1*nz1*nelv
s=0
do i=1,n
     s=s+u(i,1,1,1)
enddo

s=glmax(s,1)
```

Serial / Parallel Usage

- A common operation (3rd way…)

  n=nx1*ny1*nz1*nelv

  s=glsum(u,n)

- Parallel Version

  n=nx1*ny1*nz1*nelv

  s=glsum(u,n)

  – If you want a ***local*** max:

  s=vlsum(u,n)

  – Note: Important that every processor calls glmax()!!

Considerations as you move forward

- For performance:
  - Optimal tuning for mxm ?
  - What about the pressure solve?
    - *Projection on?*

- For stability:
  - Dealiasing on?
  - Filtering on?

- Open issues we expect to address in the next 12-24 months:
  - SEMG for the P_N-P_N formulation
  - (near) monotone advection schemes

Thank You!