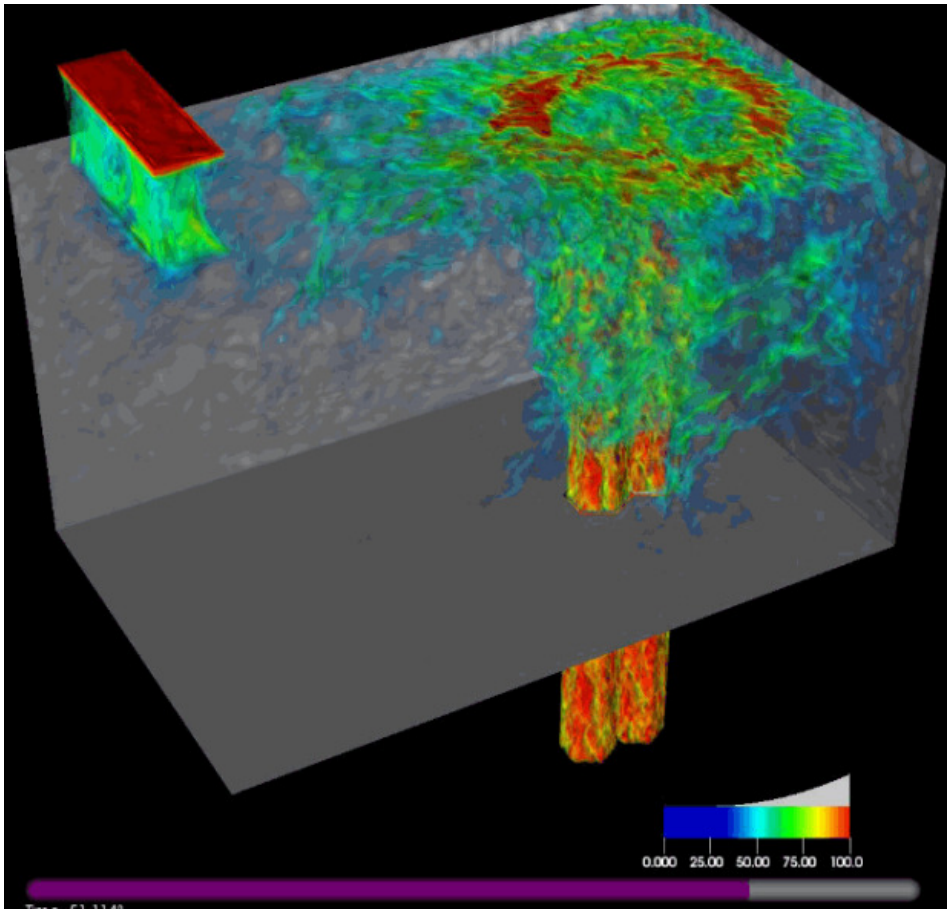
Nek5000 Tutorial

Velocity prediction, ANL MAX experiment.



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

Mathematics and Computer Science Division
Argonne National Laboratory

Presenters

- ***Paul Fischer***
  - spectral element overview
  - Nek5000
  - Prenek

- ***Aleks Obabko (& Hank Childs, LBL)***
  - VisIt overview

- Additional help
  - ***Shashi Aithal*** – Nek5000 on fusion, RANS development
  - ***Yulia Peet*** – multidomain coupling
  - ***Katie Heisey*** – automated build/test suite, example suite, mesh partitioner
  - ***Stefan Kerkemeier*** – principal software engineer

Objectives

- *Course Objectives:*
  - *Provide an overview of Nek5000 capabilities*
  - *Introduce users to Nek5000 and VisIt usage*
- *By the end of the day, you should be able to run some basic flow simulations*

Outline

- *Nek5000 capabilities*
- *Equations, timestepping, and SEM basics*
- *Workflow example*
  - *Setting initial and boundary conditions*
  - *Basic runtime analysis*
  - *Parallel / serial issues that you should understand*
- *Using VisIt to analyze results*
- *Mesh generation options*
  - *Building meshes with genbox, prenek, and morphing*
- *Walking through examples; hands on simulations*

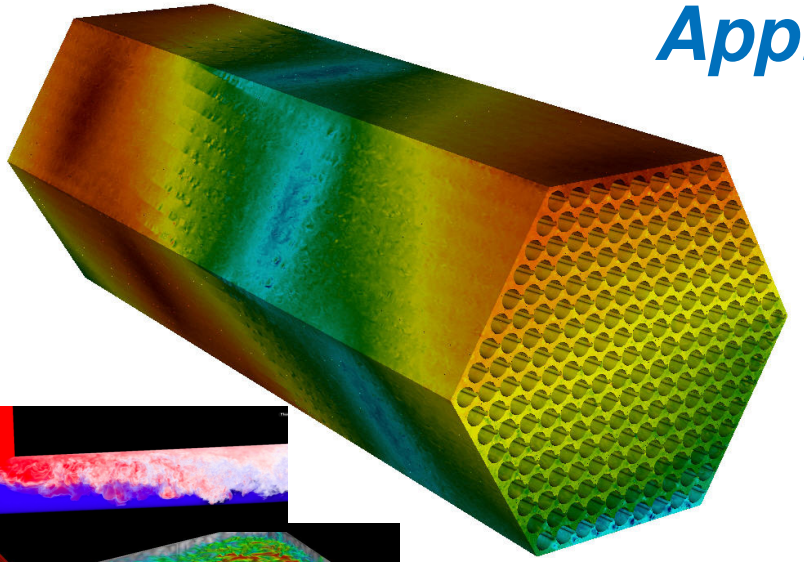
Some Resources

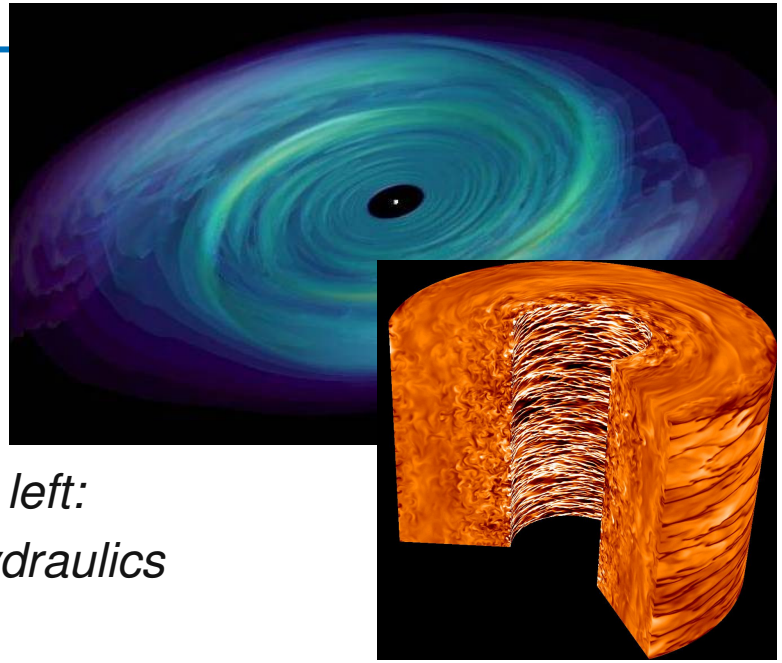
- Nek5000 wiki page (google nek5000)
- www.mcs.anl.gov/~fischer/Nek5000

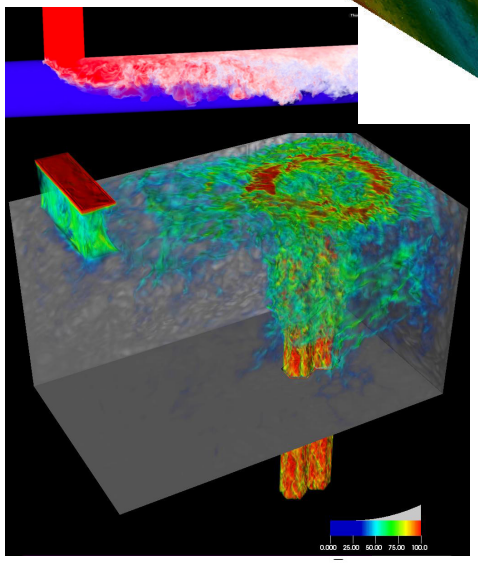
Part I

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

Applications

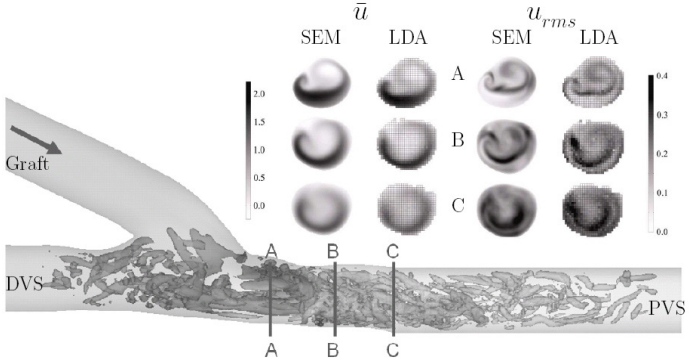


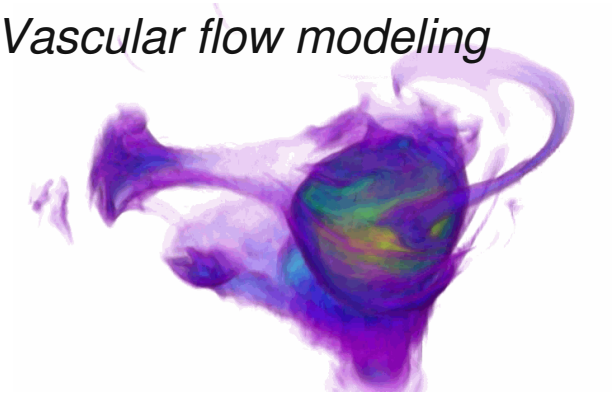


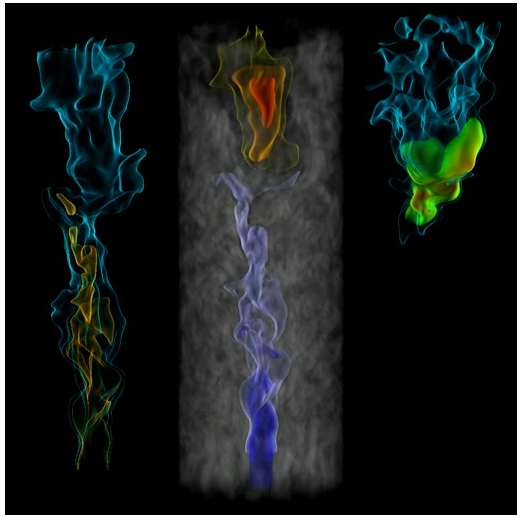


Clockwise from upper left:

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







Coarse DNS: Channel Flow at Re_b=13,000

Simulations by J. Ohlsson, KTH, Stockholm

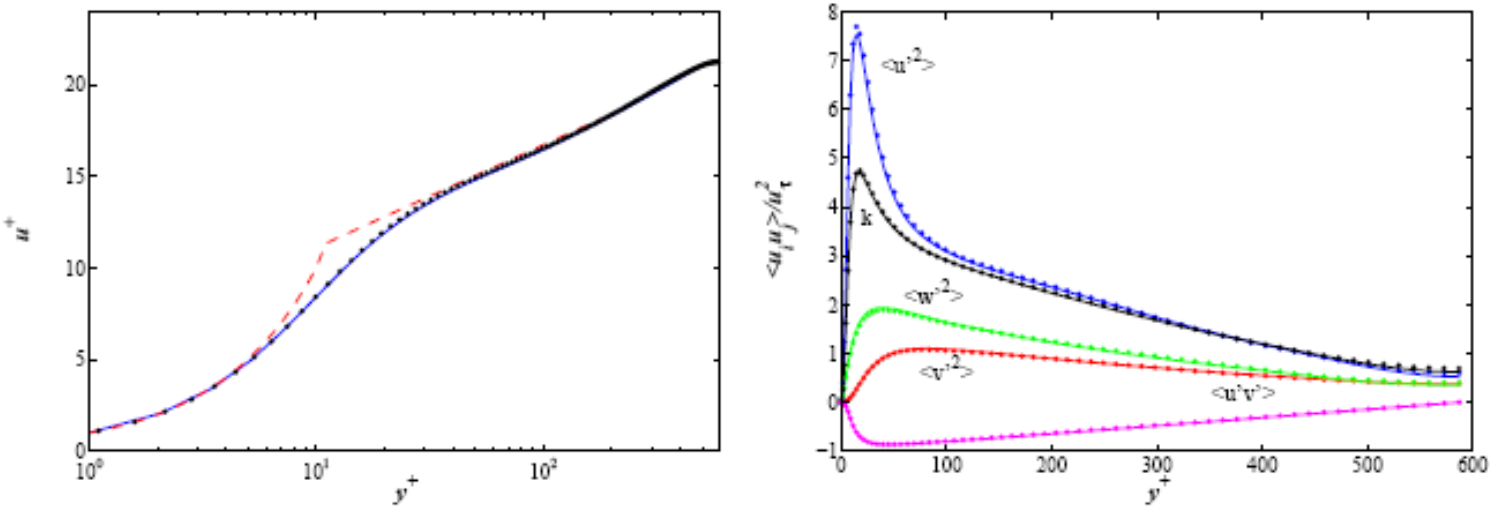
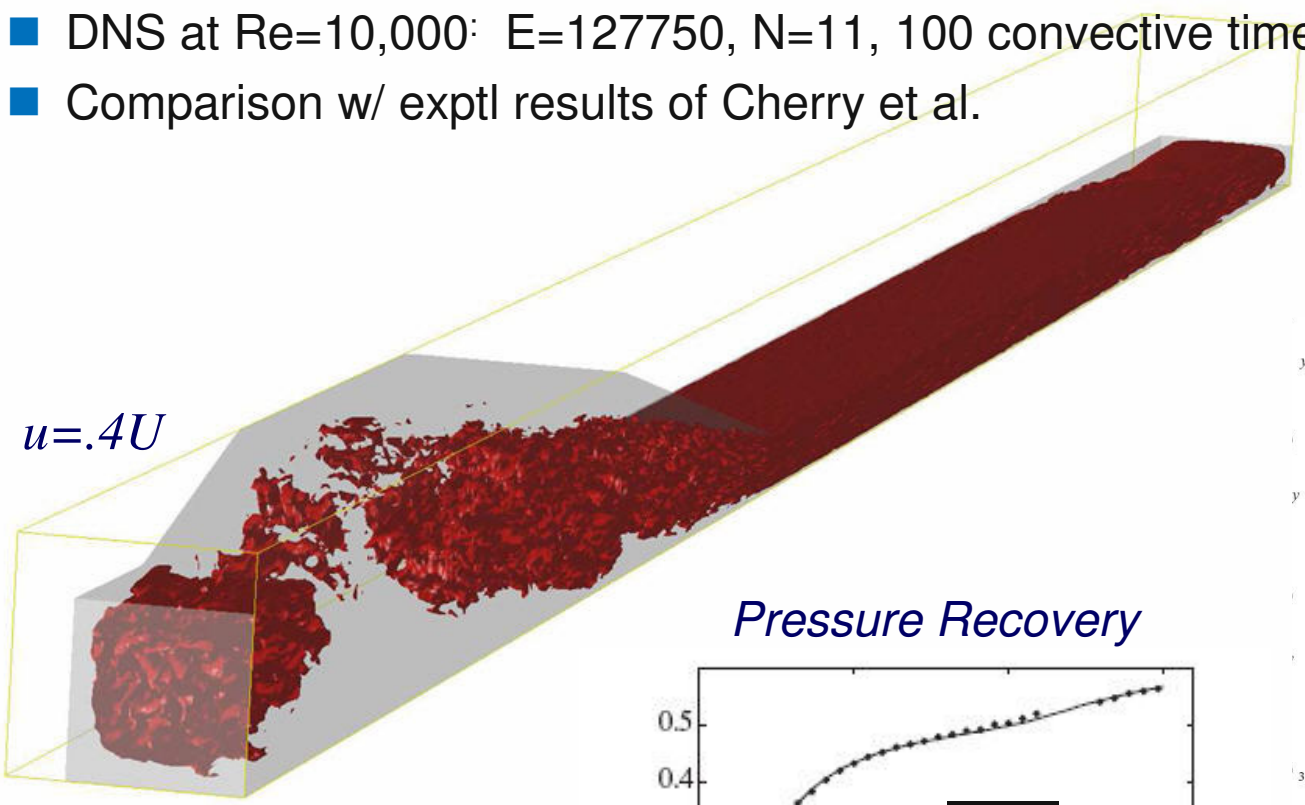


Figure 15: Turbulent channel flow simulations at $Re_\tau = 590$ with polynomial order 15 (a resolution of 288 in the homogenous directions and 192 in the wall-normal direction) showing a) mean velocity profile and b) Reynolds stresses together with the turbulent kinetic energy as defined above. In this 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.

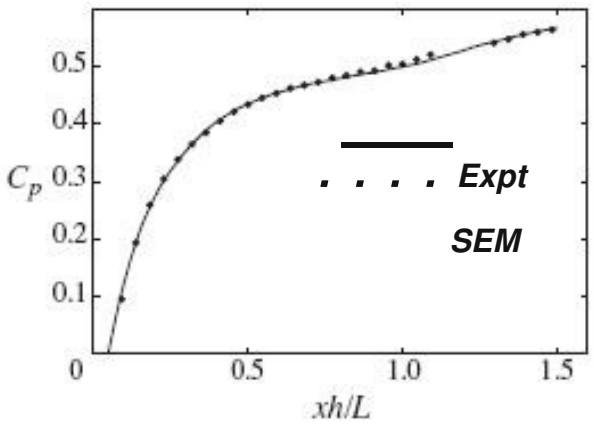
Separation in an Asymmetric Diffuser

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

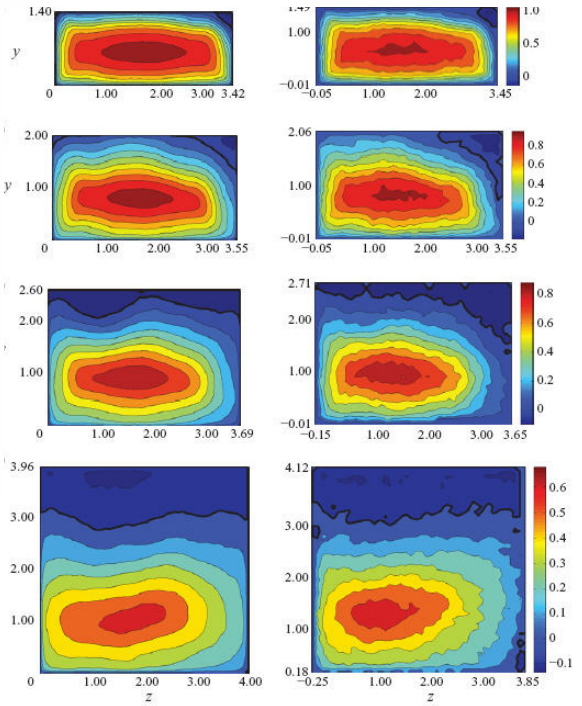
- Flow separation and recovery
- DNS at Re=10,000: E=127750, N=11, 100 convective time units
- Comparison w/ exptl results of Cherry et al.



Pressure Recovery



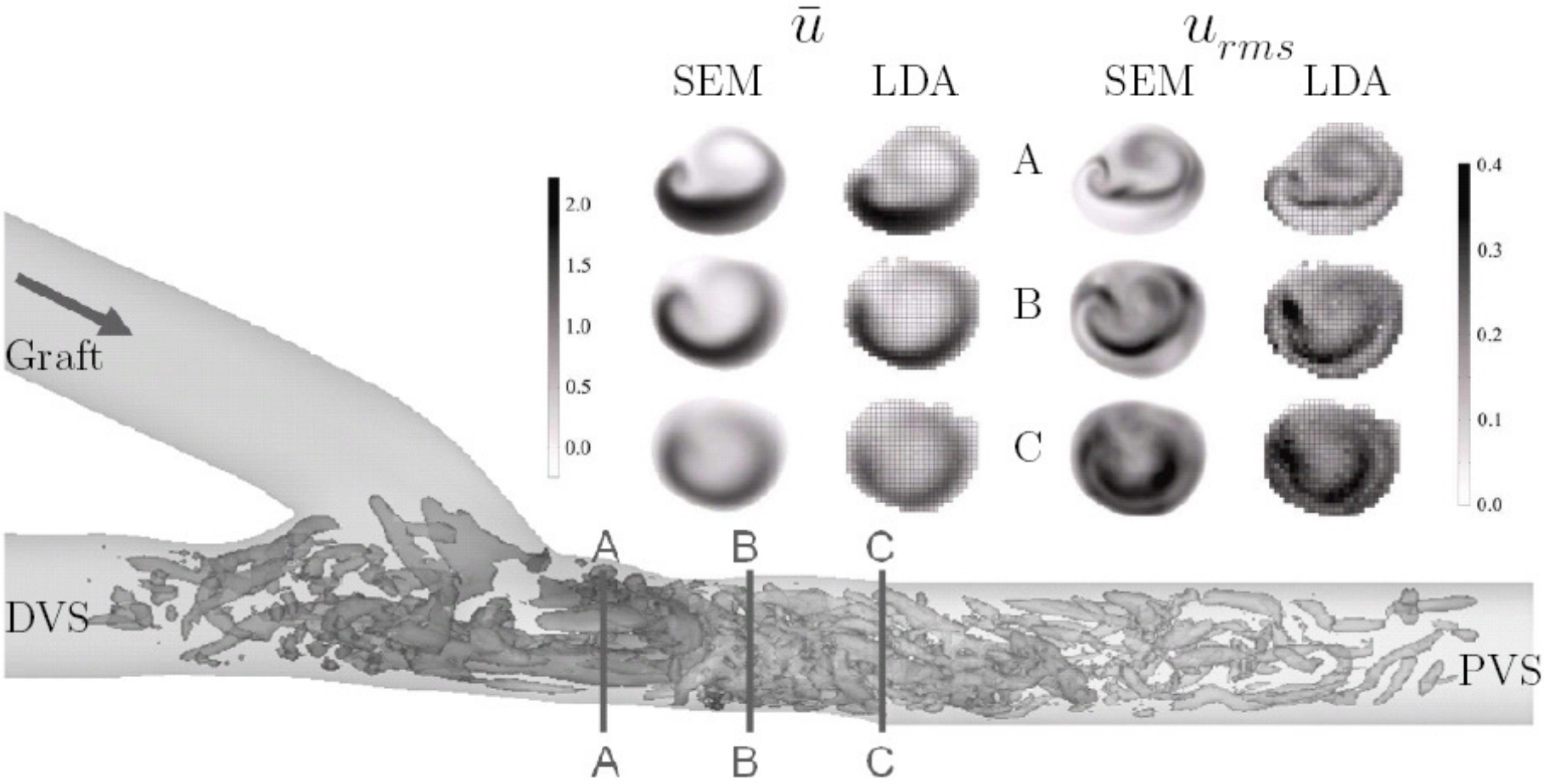
Axial Velocity



SEM *expt.*

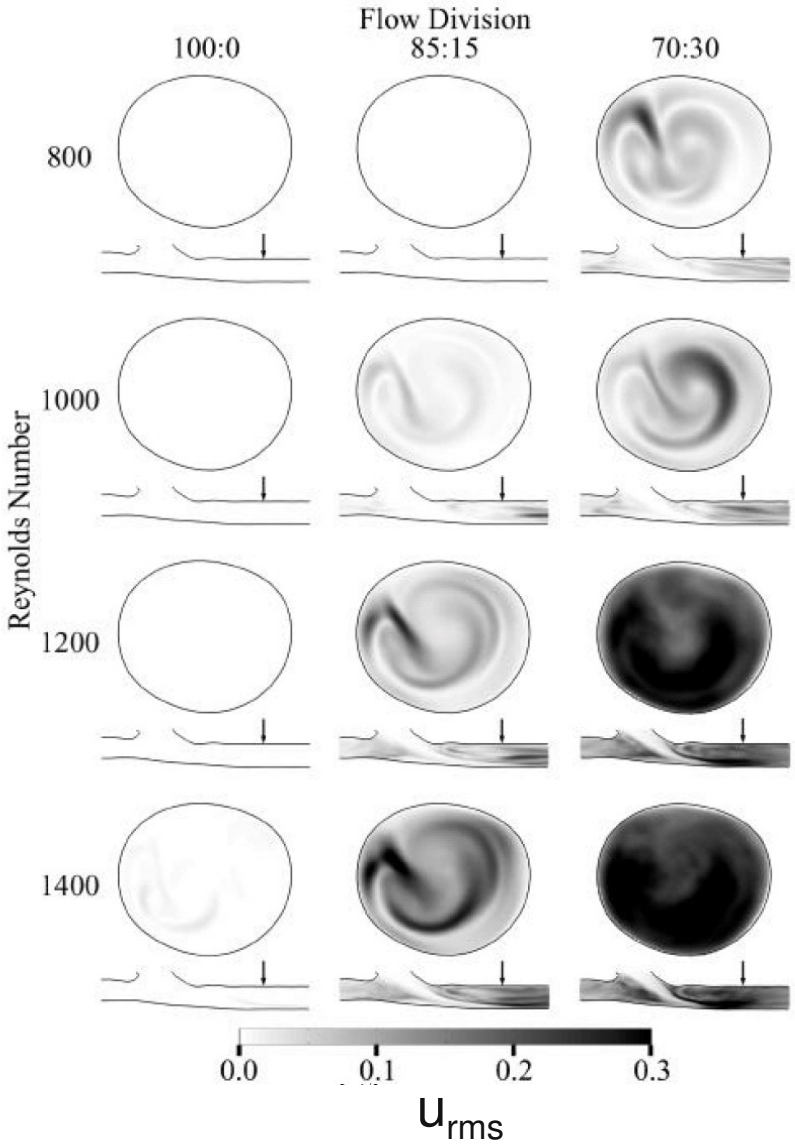
Low Re Turbulence in Complex Domains

Arteriovenous graft flow @ Re=1200



Loth, F., Bassiouny, Ann. Rev. Fluid Mech. (2008)

Influence of Reynolds Number and Flow Division on u_{rms}



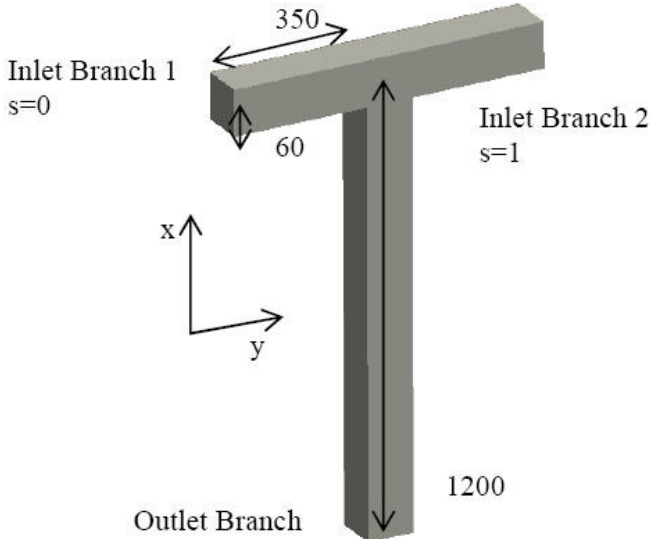
Validated simulations allow prediction of the relative influences of flow division and Reynolds number on transition to turbulence in arteriovenous grafts.

Nek5000 LES Validation: T-Junction Studies

E. Merzari ANL

Square T-junction simulation and comparison with experimental data

– *20 M points, first point at $y^+ < 1$, $Re_{out} = 7000$*



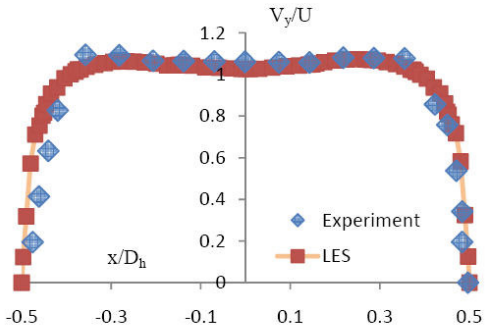


Fig. 6 Reynolds Averaged y-velocity at z=0, y=-30, values normalized by the Mixing Bulk Velocity

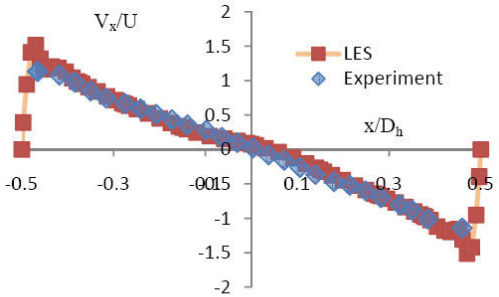


Fig. 8 Reynolds Averaged x-velocity at z=0, y=-30, values normalized by the Mixing Bulk Velocity

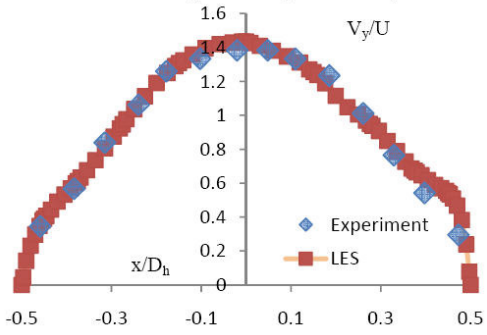


Fig. 7 Reynolds Averaged y-velocity at z=0, y=-90, values normalized by the Mixing Bulk Velocity

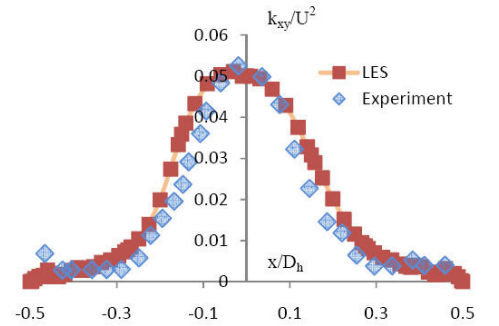


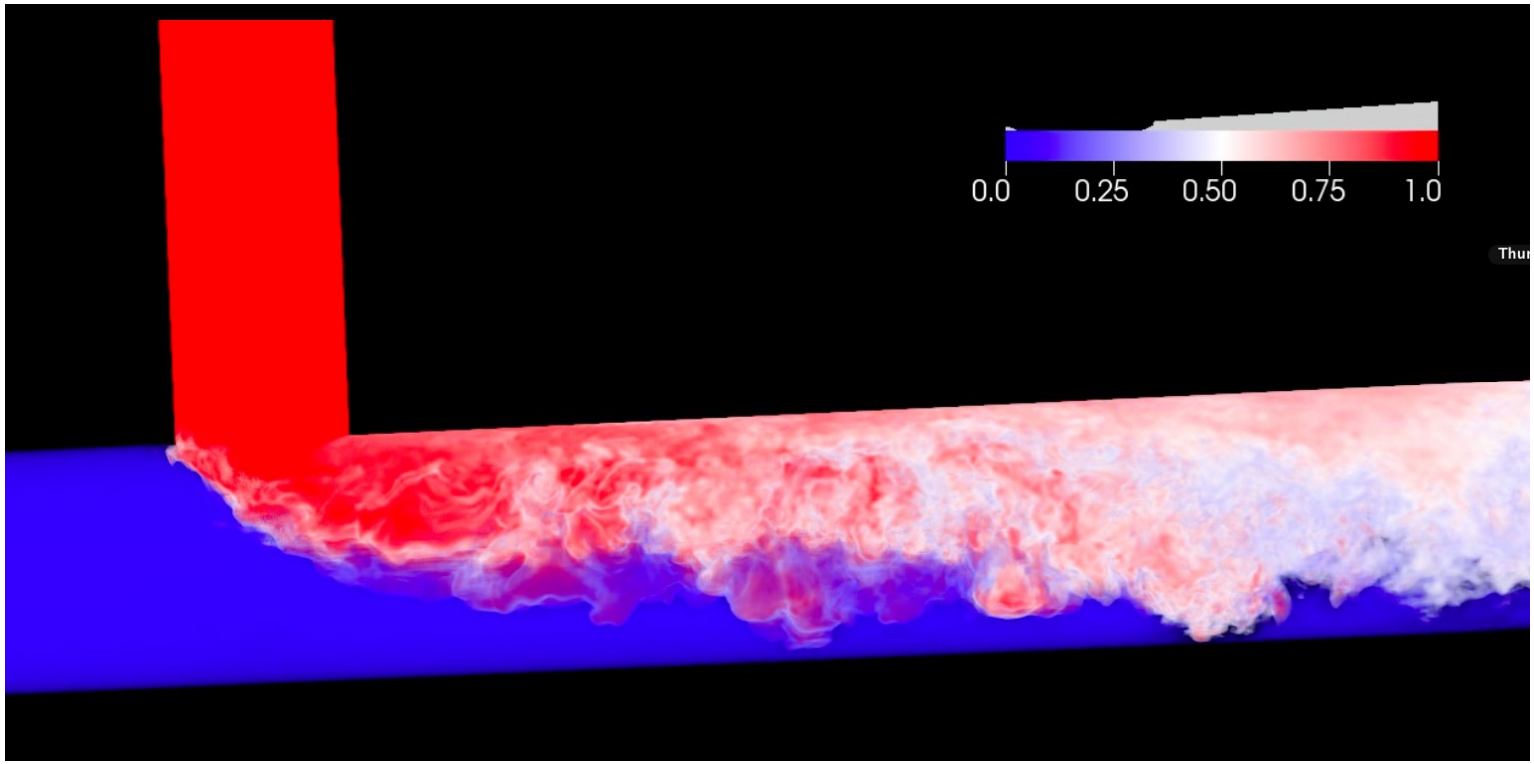
Fig. 11 Reynolds Averaged Turbulent kinetic energy at z=0, y=-90, values normalized by the square of the Mixing Bulk Velocity

[1] *Merzari et al., Proper Orthogonal Decomposition of the flow in a T-junction, Proc. ICAPP (2010)*

[2] *Hirota et al., Exptl Study on Turbulent Flow and Mixing in Counter-Type T-junction, J. Therm. Sci. & Tech. 3, 157 – 58 (2008)*

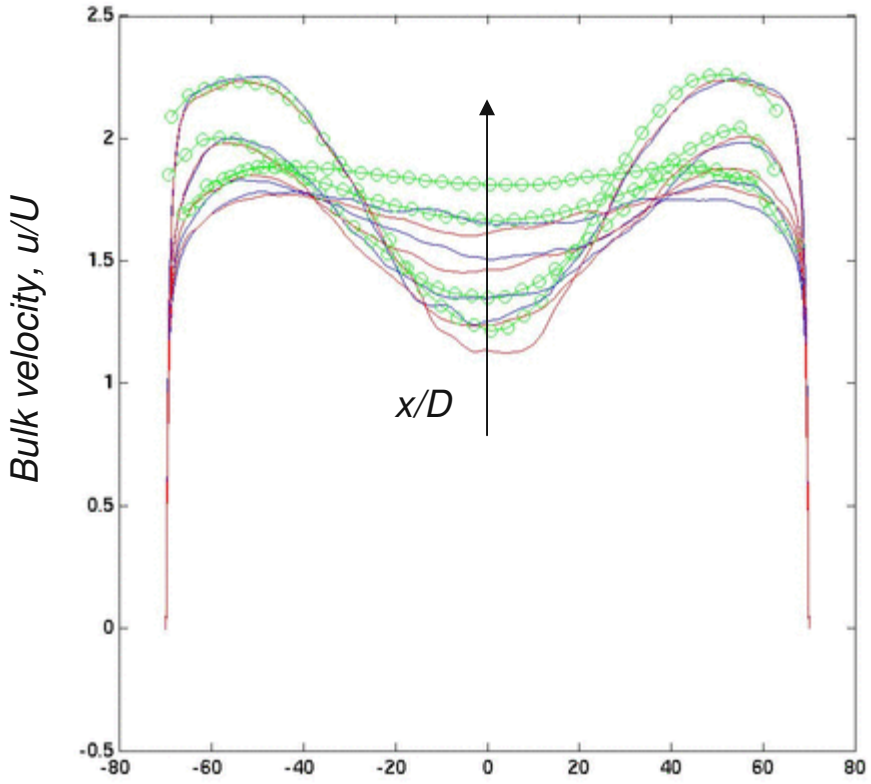
NEA/OECD Blind T-Junction Benchmark

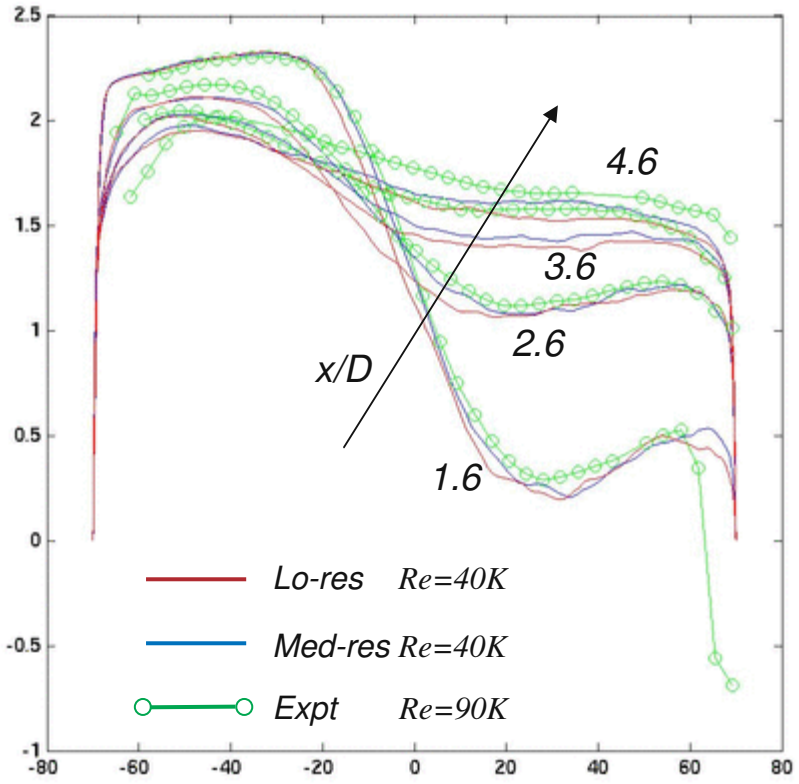
- Thermal striping experiment with hot/cold inlets at Re ~ 10^5
- Inlet velocity and temperature data provided by Vattenfall.
- Of 29 entries, Nek5000 submission ranked 1st and 6th, respectively, in temperature and velocity prediction (CFD4NRS 2010)



Velocity Comparison Downstream of T-junction

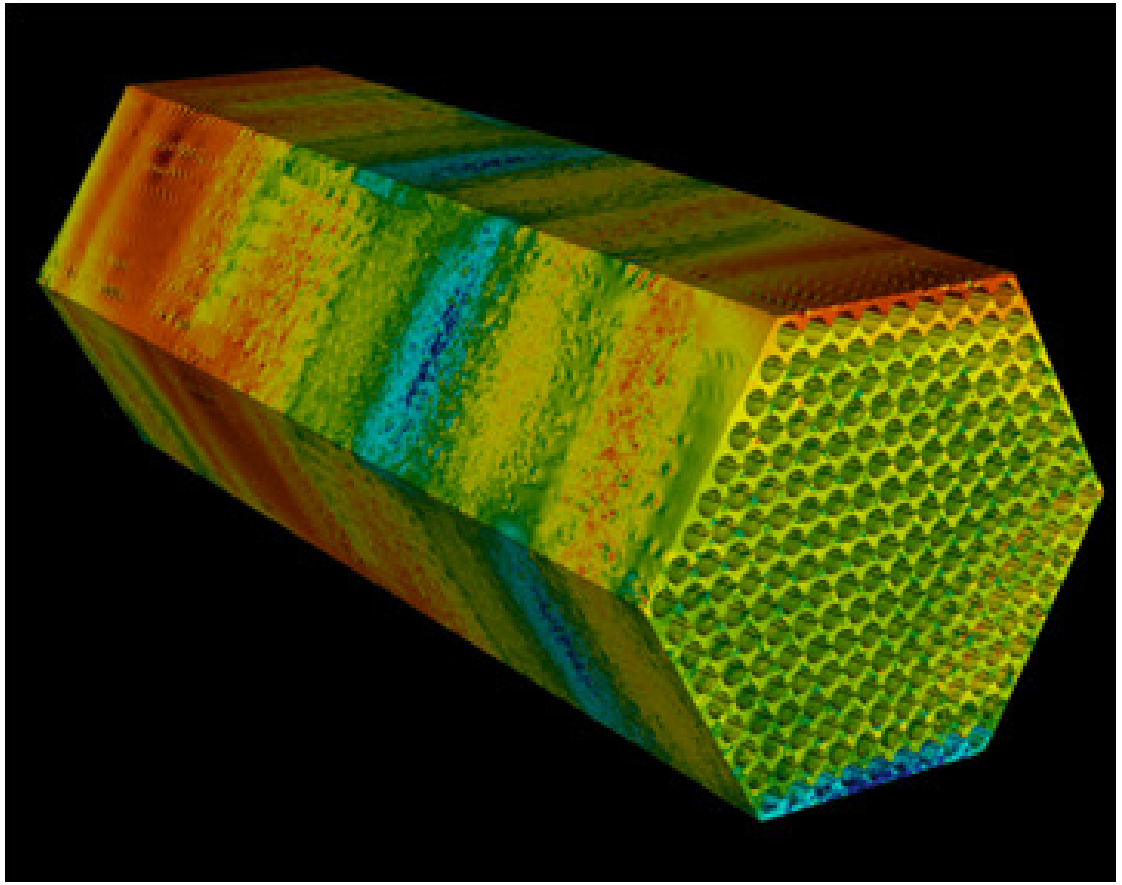
- Medium resolution results are in excellent agreement at x=1.6 & 2.6
- Experiment (Re=90K) exhibits more rapid recovery of profile than simulation (Re=40K)

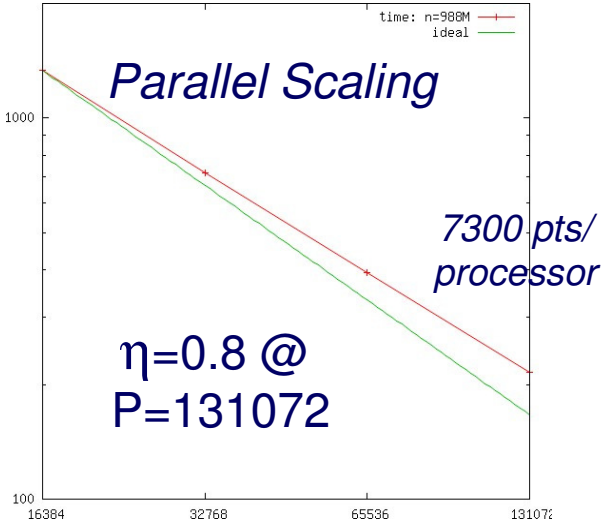


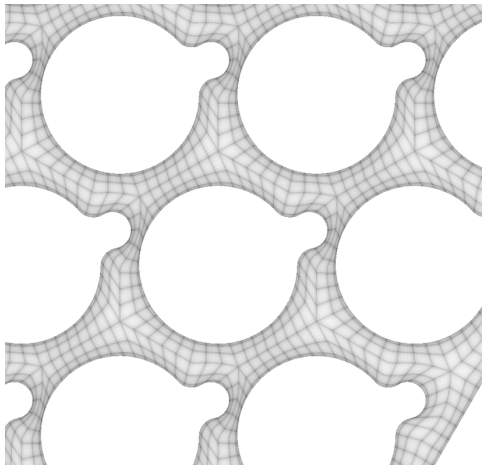


Parallel Scaling: Subassembly 217 Wire-Wrapped Pins

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





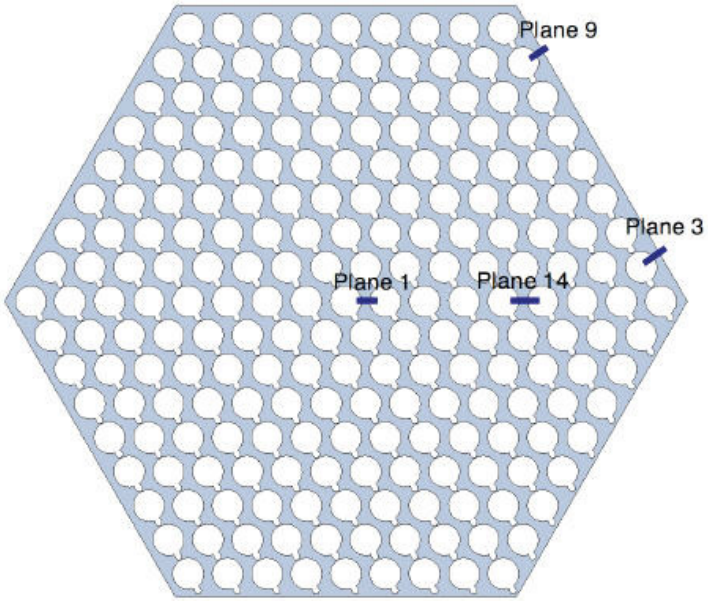


www.mcs.anl.gov/~fischer/sem1b

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)

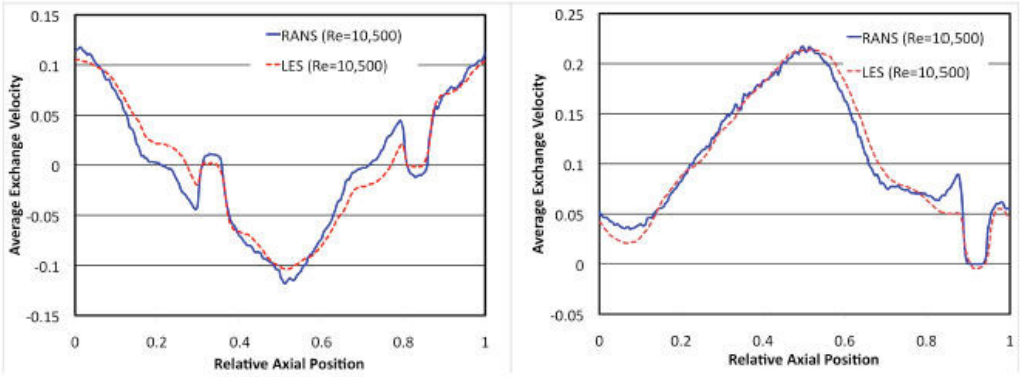


Figure 7. Comparison of predicted normal velocity profile for (a) plane group 1 and (b) plane group 3. RANS data is shown in blue and LES data is shown in red.

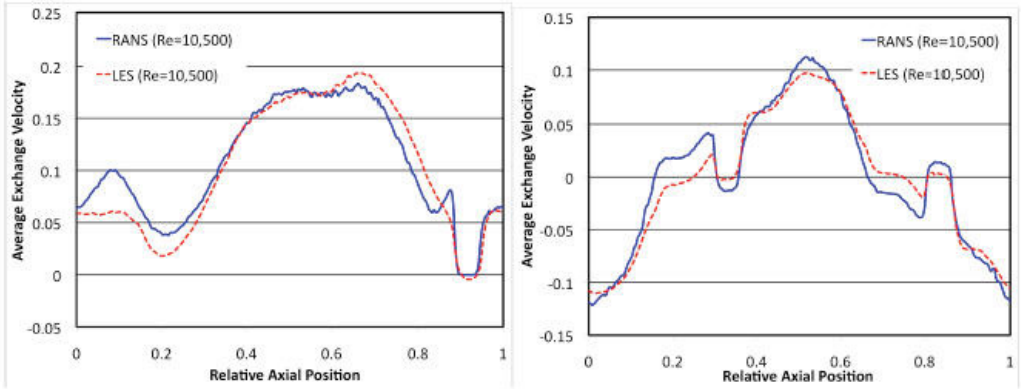


Figure 8. Comparison of predicted normal velocity profile for (a) plane group 9 and (b) plane group 14. RANS data is shown in blue and LES data is shown in red.

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

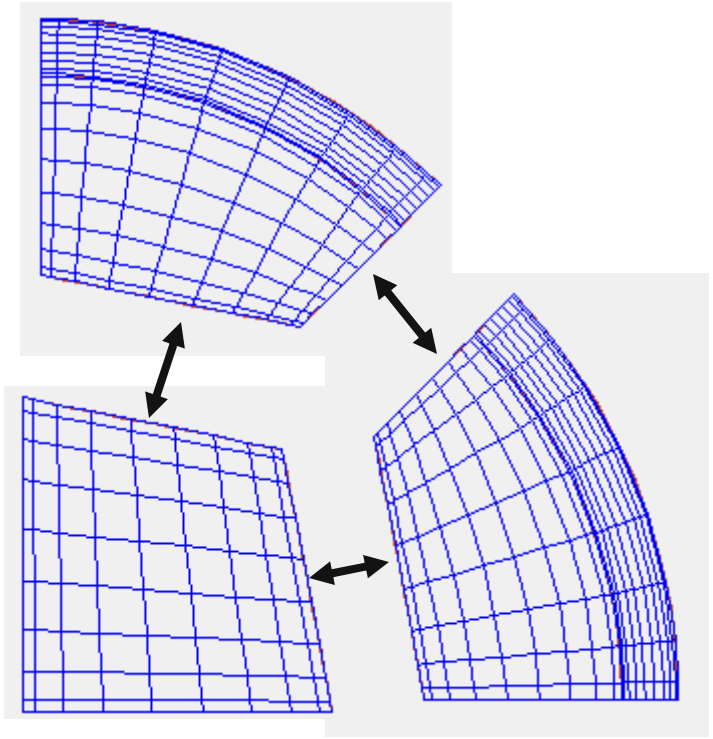
- Based on high-order spectral element method (Patera '84, Maday & Patera '89)
  - Started as Nekton 2.0. First 3D SEM code. (F., Ho, & Ronquist, '86-'89)

- 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)

Spectral Element Overview

- High-order FEM featuring
  - Minimal numerical dispersion/dissipation (N^{th} order accuracy, N=5-15, typ.)
  - Loosely coupled elements (C^0 continuity between elements)
  - Tightly coupled dofs within elements (full stiffness matrices – ***never*** formed)



- Standard domain decomposition + message-passing based parallelism
- Iterative solvers imply local work with dense operators, followed by data exchanges to update interface values

Why 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 ~ (λ / L)$\mathcal{E}_f$, << 1

- High-order methods most efficiently deliver small dispersion errors
  (Kreiss & Oliger 72, Gottlieb et al. 2007)

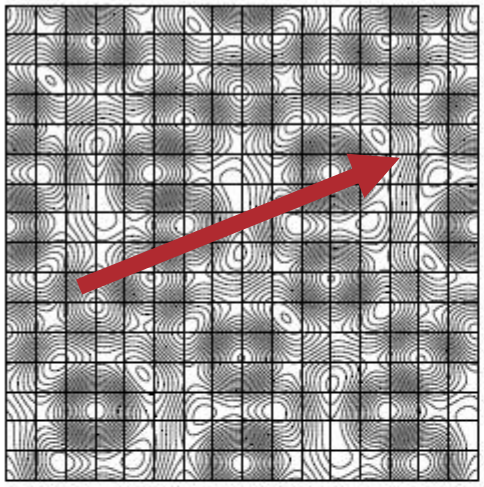
Spectral Element Convergence: Exponential with N

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

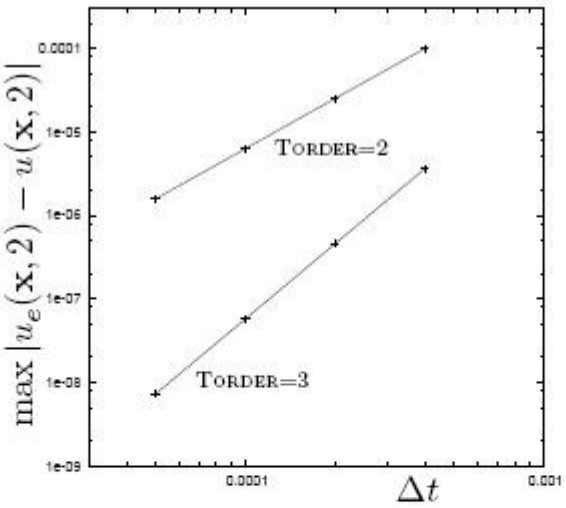
  $\cos(mx)\cos(ny)$ $\cos(mx)\sin(ny)$ $\sin(mx)\cos(ny)$ $\sin(mx)\sin(ny)$

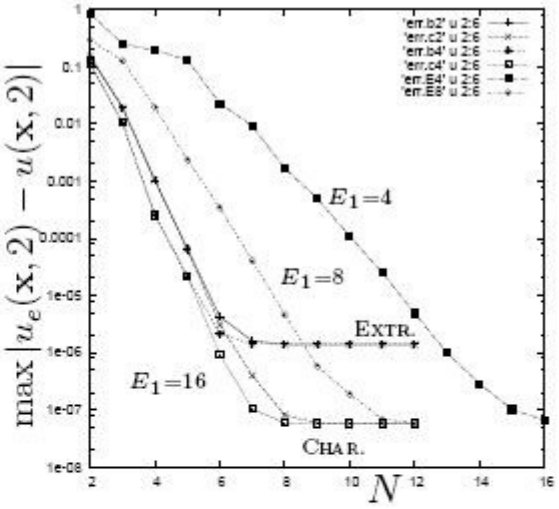
  with $m^2+n^2=\lambda^2$. [O. Walsh '92]

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

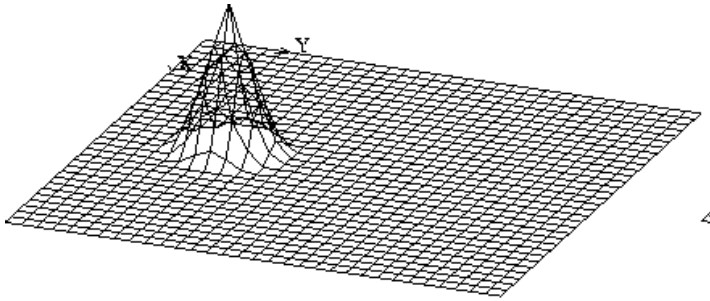


$\bar{U}=(16,5)$

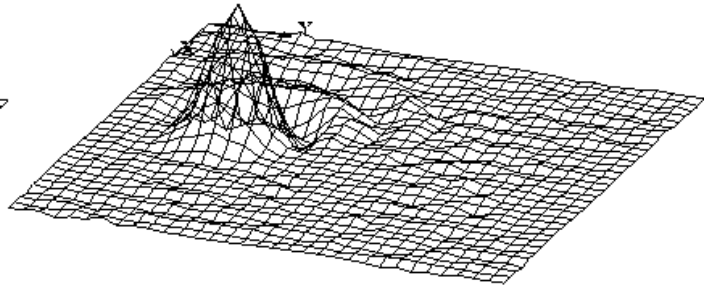




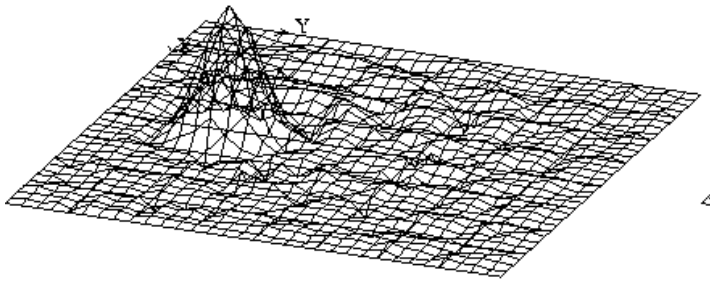
SEM Excellent transport properties, even for non-smooth *solutions*



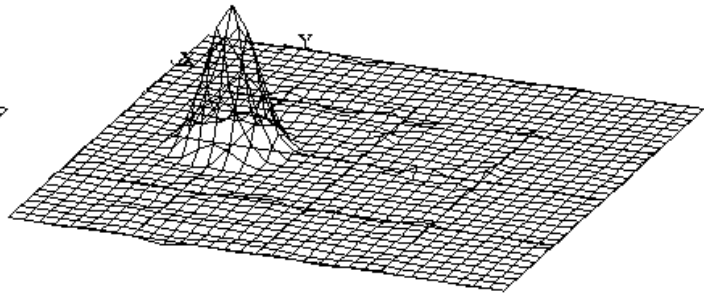
Initial Condition



$K_1 = 16,\ N = 2$



$K_1 = 8,\ N = 4$



$K_1 = 4,\ N = 8$

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*
  - *3^{rd}-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

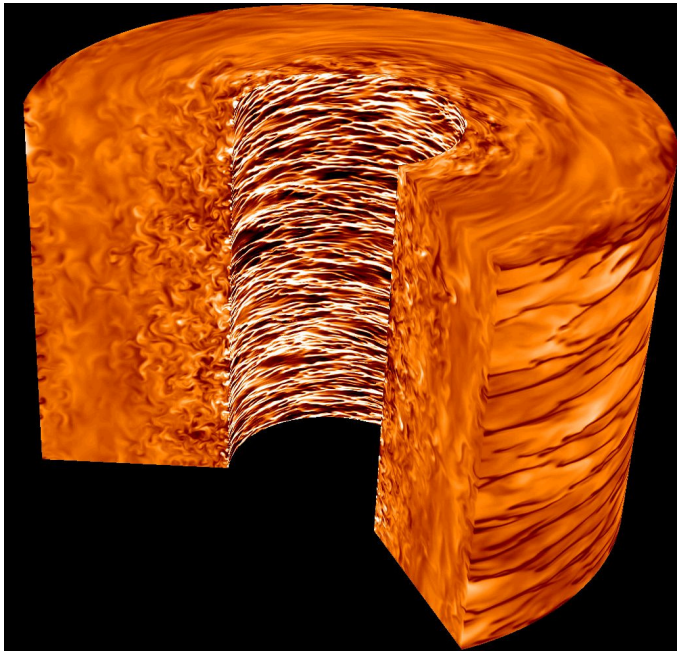
  > 150 journal articles & > 60 users worldwide

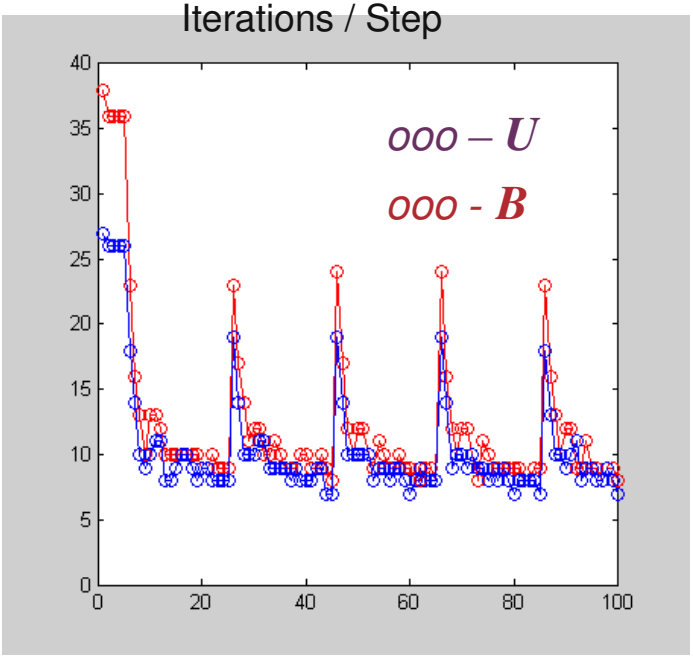
  > 400 tests after each build to ensure verified source

  (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***



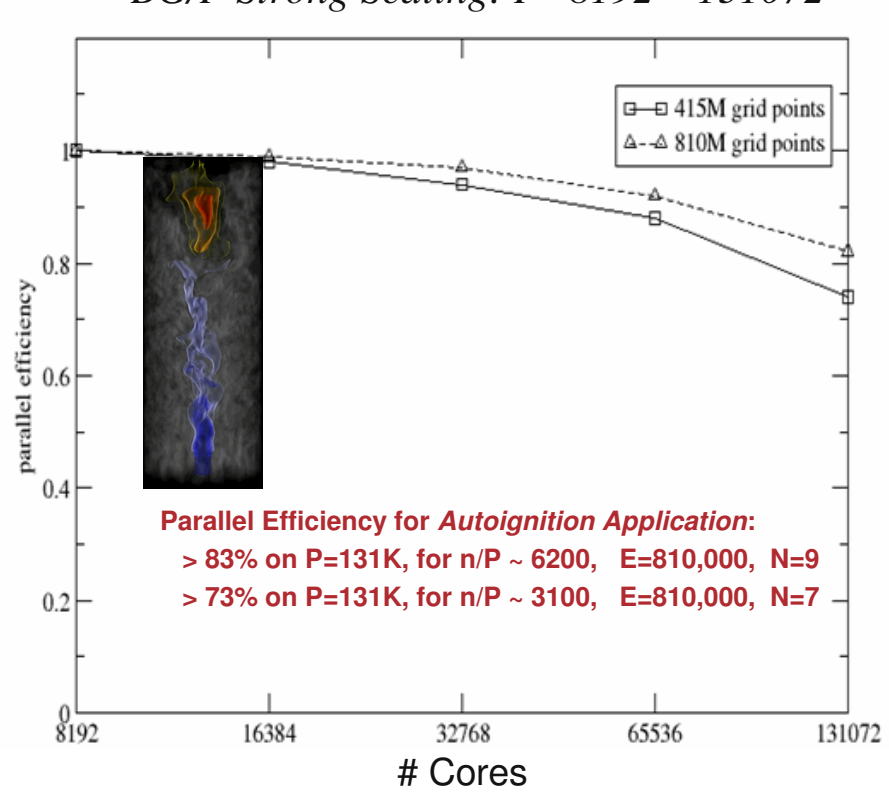


Scaling to P=262144 Cores

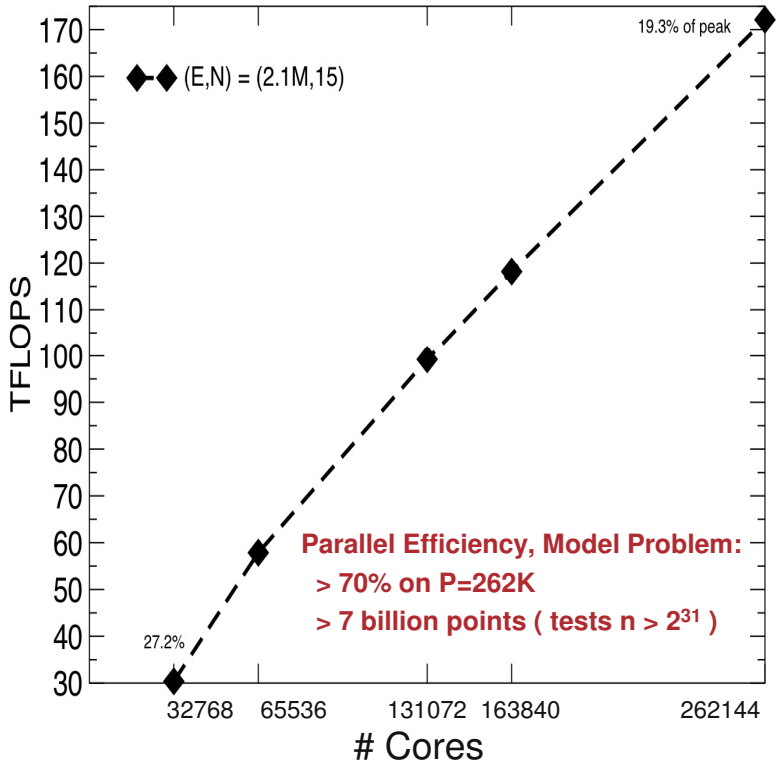
Stefan Kerkemeier
ETHZ / ANL

- 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

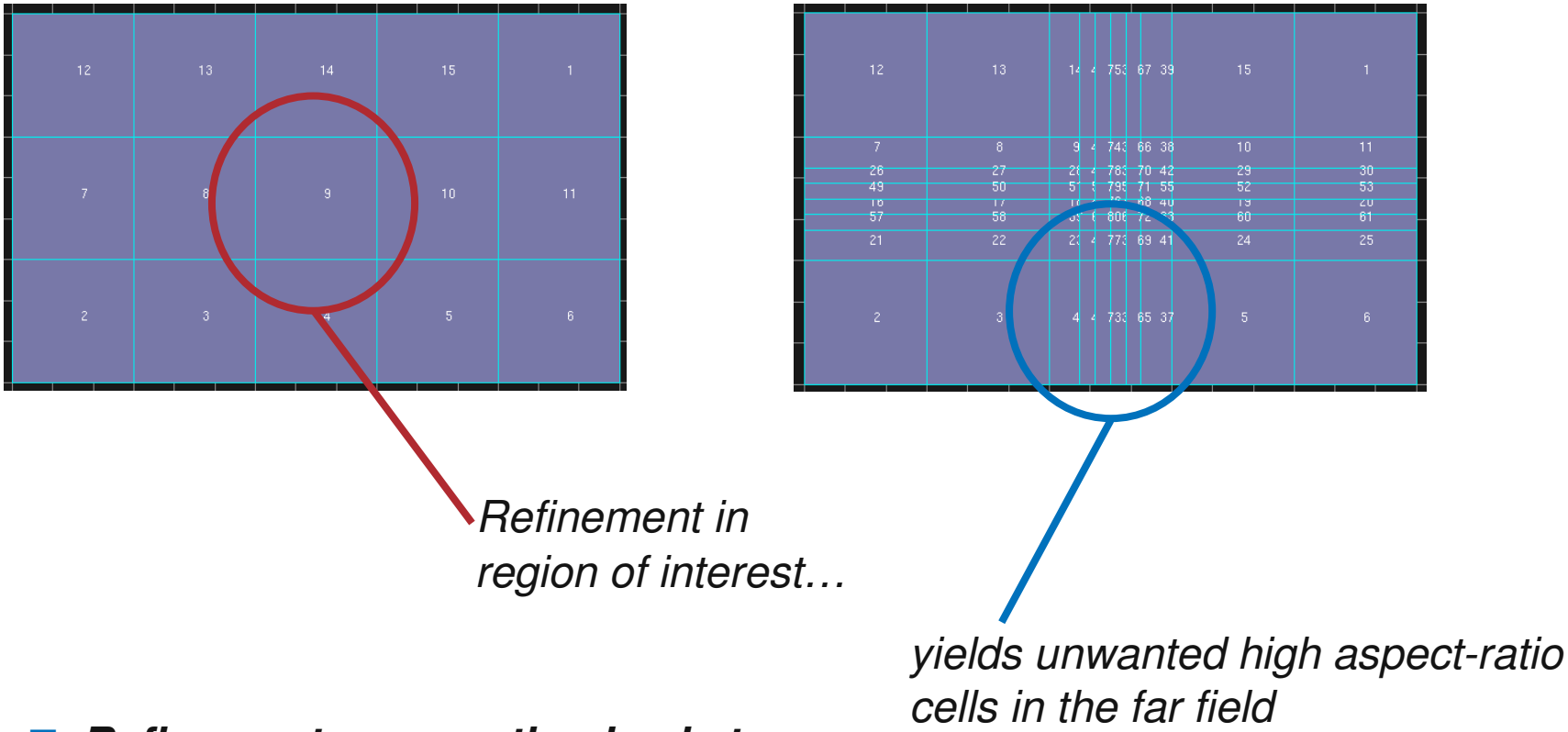


Limitations of Nek5000

- ***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

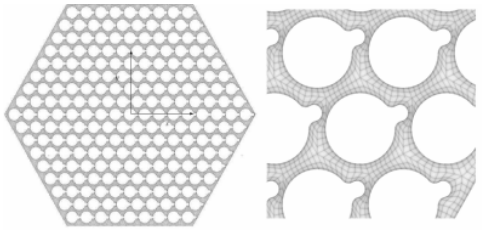
A common refinement scenario (somewhat exaggerated):

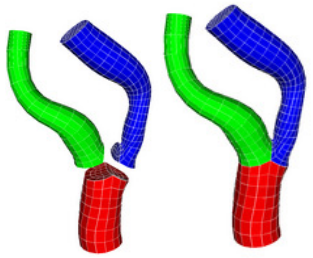


- ***Refinement propagation leads to***
  - unwanted elements in far-field
  - high aspect-ratio cells that are detrimental to iterative solver performance (F. JCP'97)

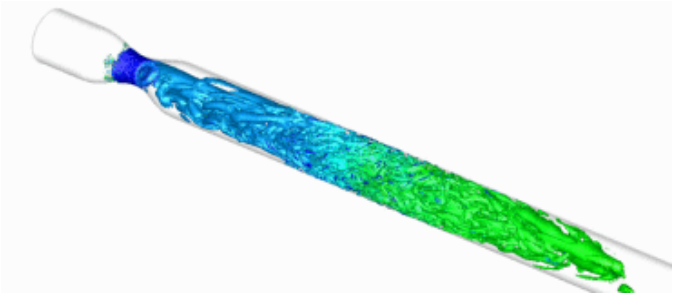
Some Meshing Options

- *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*
- *Workflow example*
  - *Setting initial and boundary conditions*
  - *Basic runtime analysis*
  - *Parallel / serial issues that you should understand*
- *Using VisIt to analyze results*
- *Mesh generation options*
  - *Building meshes with genbox, prenek, and morphing*
- *Walking through examples; hands on simulations*

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 $\mathbf{B}$

Incompressible MHD, Elsasser Variables

$$\mathbf{z}_+ := \mathbf{u} + \mathbf{B}, \qquad \mathbf{z}_- := \mathbf{u} - \mathbf{B}$$

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

$$\nabla \cdot \mathbf{z}_+ = 0$$

$$\frac{\partial \mathbf{z}_-}{\partial t} - \frac{1}{Re}\nabla^2 \mathbf{z}_- + \nabla q = - \mathbf{z}_+ \cdot \nabla \mathbf{z}_-,$$

$$\nabla \cdot \mathbf{z}_- = 0$$

— A pair of Oseen problems: z_- convects z_+ , z_+ convects z_-

— Similar form for Re ^= Rm exists.

— A reasonable starting point for LES development…

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)*

MHD Time Advancement

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

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

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

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

1. Compute nonlinear contributions (explicit, in Elsasser form, dealiased)
2. Solve well-conditioned Helmholtz problems for u_i^n, $i=1,3$
3. Filter u_i^n
4. Solve consistent Poisson problem for p^n
5. Compute div-free correction of u_i^n
6. Repeat 2. – 4. for B_i^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:

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

which is equivalent to

$$B\frac{d\underline{u}}{dt}\Big|_{t^n} + C\underline{u}\Big|_{t^n} \;=\; -\nu A\underline{u}\Big|_{t^n}.$$

BDF2/EXT2 Example

$$B\frac{d\underline{u}}{dt}\bigg|_{t^n} + C\underline{u}\bigg|_{t^n} = -\nu A\underline{u}\bigg|_{t^n}$$

Evaluate each term at t^n according to convenience:

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

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

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

BDF2/EXT2 Example

Rearrange,

$$H\underline{u}^n = \frac{1}{2\Delta t}\left(4B\underline{u}^{n-1} - B\underline{u}^{n-2}\right) + 2C\underline{u}^{n-1} - C\underline{u}^{n-2} + O(\Delta t^2),$$

with $H := \frac{3}{2\Delta t}B + \nu A$.

H is well-conditioned (Δt and ν small) and symmetric positive definite.

For the SEM, B is *diagonal* $\longrightarrow$ H *diagonally dominant*.

Stability of ABk, BDFk/EXTk Timesteppers

- Derived from model problem: $\frac{du}{dt} = \lambda u$

- Crucially, the chosen schemes encompass part of the imaginary axis. Important for high Reynolds number flows.

Stability Regions in the $\lambda \Delta t$ Plane

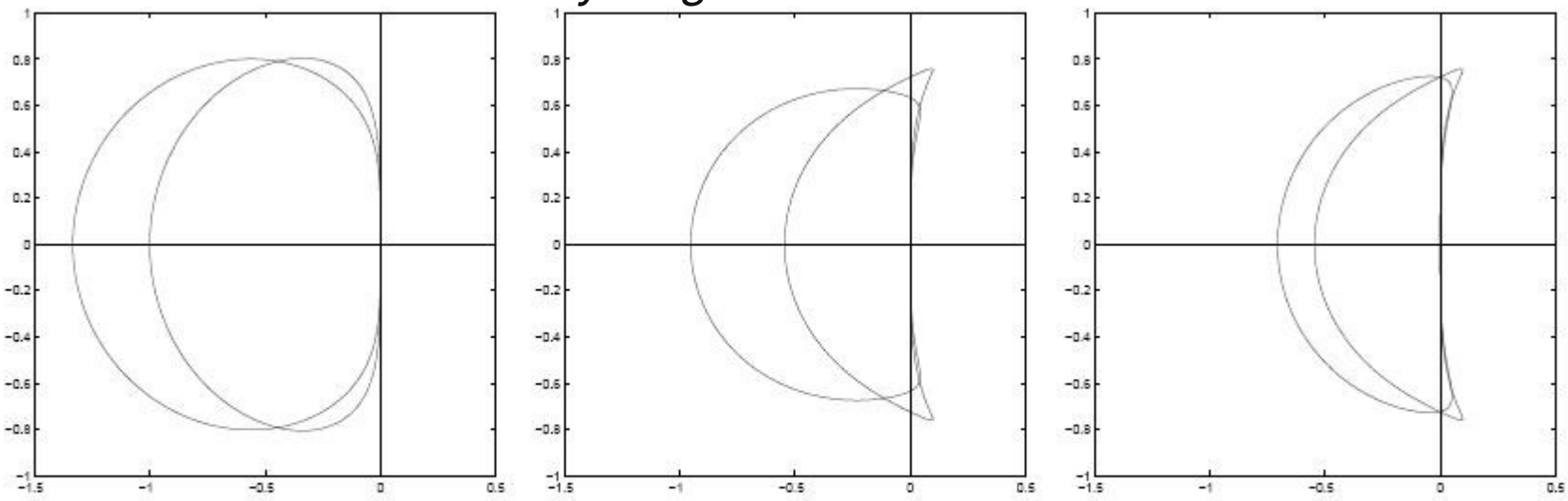


Figure 1: Stability regions for (left) AB2 and BDF2/EXT2, (center) AB3 and BDF3/EXT3, and (right) AB3 and BDF2/EXT2a.

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)

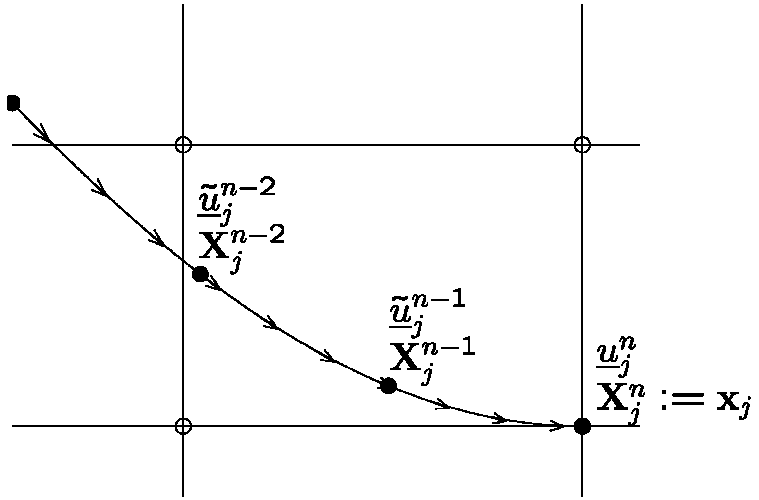
Characteristics Timestepping

- 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 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:

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

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

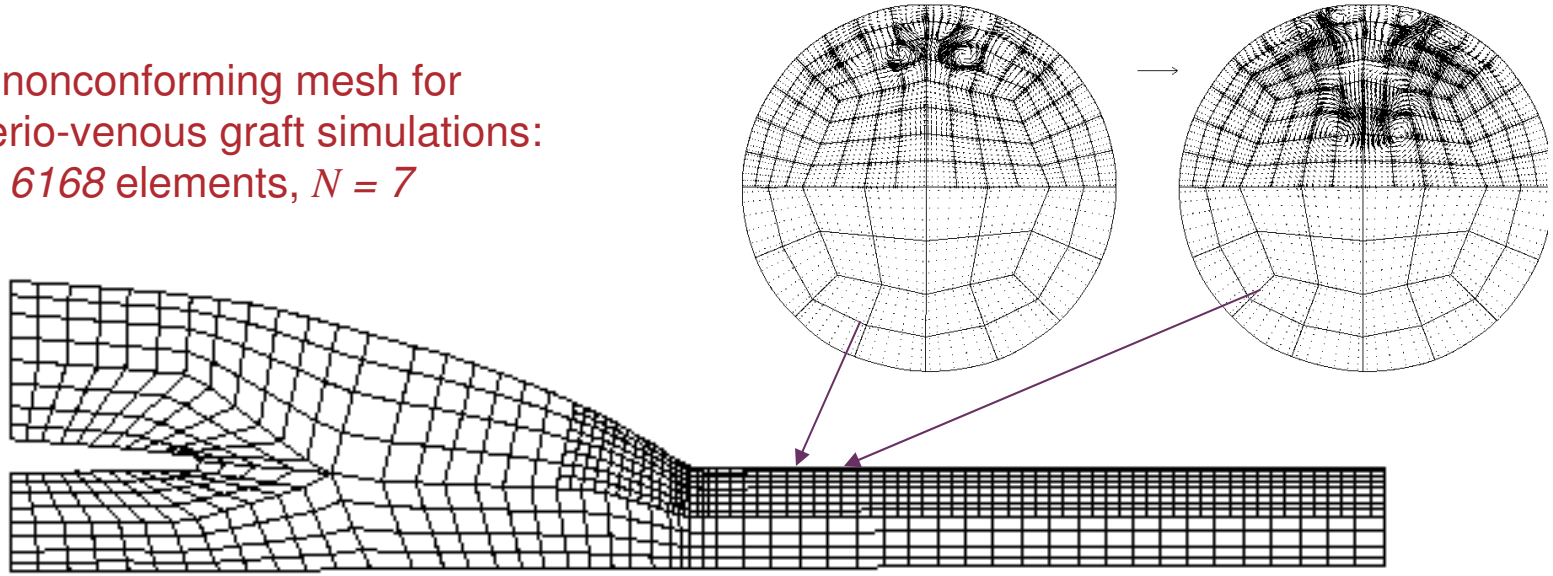
which is solved via explicit timestepping with $\Delta s \sim \Delta t_{CFL}$

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

SEM Function Representation

- **SEM choices for ϕ_j :**
  - High-order polynomials on each element
  - Compactly supported (sparse matrices, highly parallel)
  - Stable Lagrangian interpolants:
    - *Basis coefficients are also grid-point values*
      - Easy to implement boundary conditions
      - Grid-points chosen to be Gauss-Lobatto-Legendre quadrature points: *diagonal mass matrix and low-cost operator evaluation*
  - Local tensor-product bases:
    - *ijk indexing* (*low storage & minimal indirect addressing*)
    - *Matrix-free fast tensor-product operator evaluation:* (Orszag '80)
      - memory is O(n), work is O(nN) – ***Not O(nN^3) !!***

How to get to high-order? Step 1: 1D

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

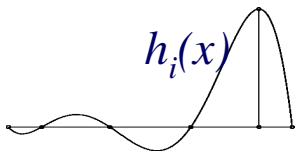
- Stable high-order basis for N[th]-order polynomial approximation space:
  - poor choices:

    $h_i(x) = x^i$

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

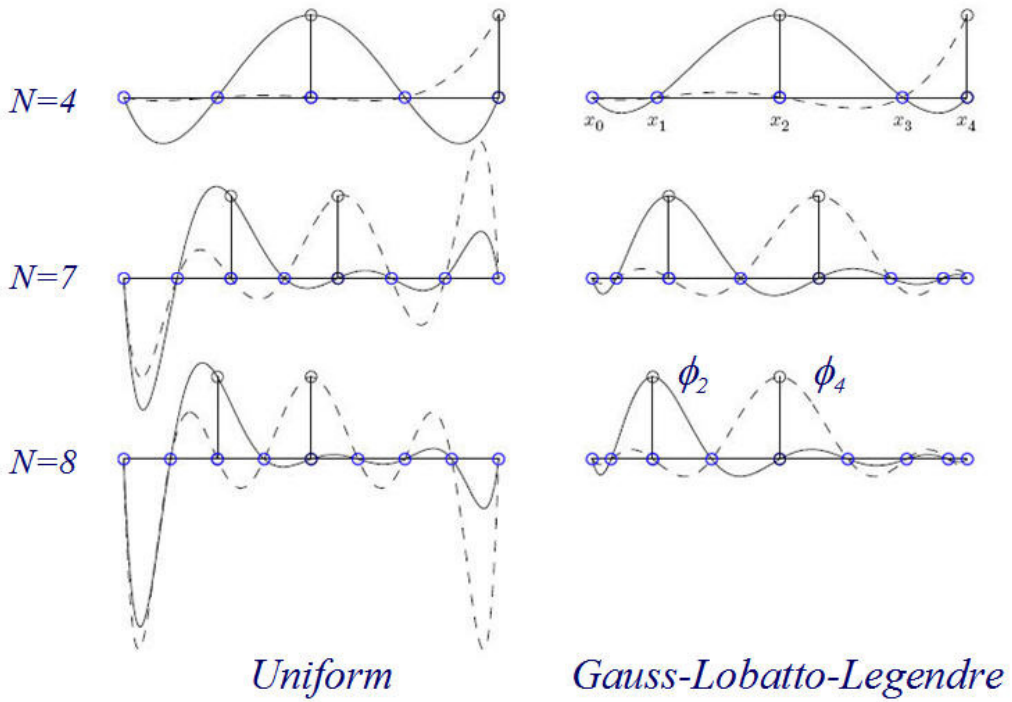
    $h_i(x) = L_i(x)$ (any orthogonal polynomial)

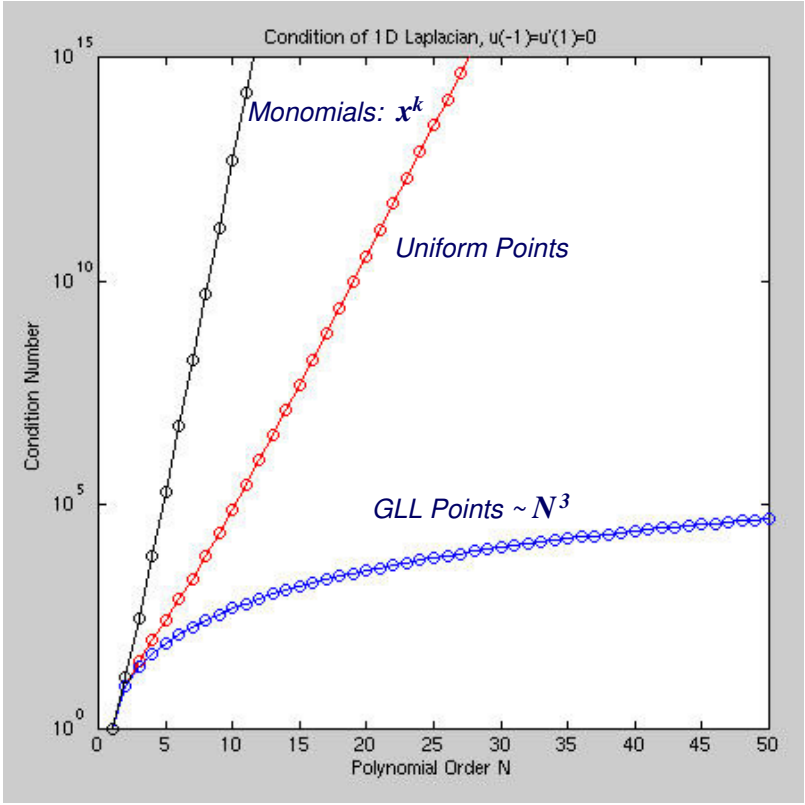
    $h_i(x) =$ Lagragian interpolant on Gauss points



Condition Number of 1D Stiffness Matrix

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





How to get to high-order? Step 2: 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….)

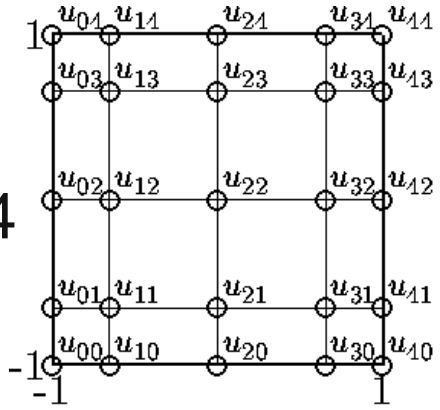
Extension to 2D

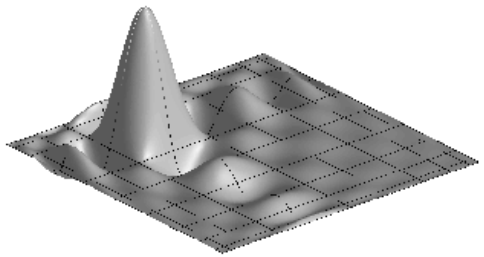
Nodal bases on the Gauss-Lobatto-Legendre points:

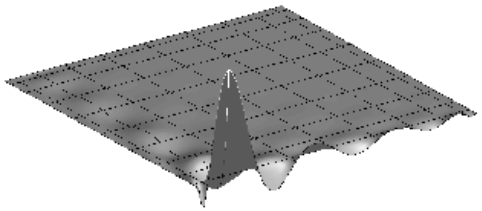
$$u(x,y) = \sum_{i=0}^{N} \sum_{j=0}^{N} u_{ij} h_i(x) h_j(y), \quad h_i(\xi_p) = \delta_{ip}, \quad h_i \in \mathbf{P}_N$$

basis coefficients

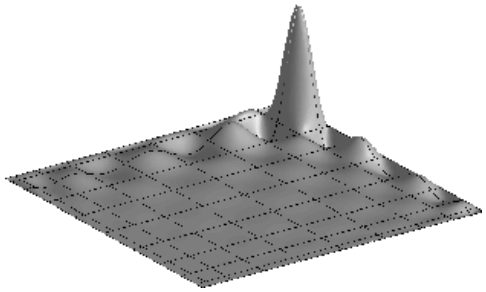
N=4







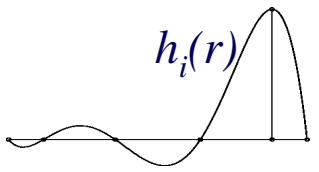
N=10



Matrix-Matrix Based Derivative Evaluation

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

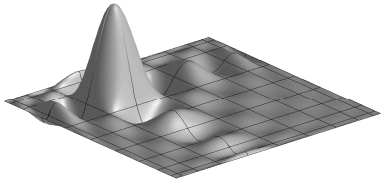
$$u(r,s) = \sum_{i=0}^{N} \sum_{j=0}^{N} u_{ij} h_i(r) h_j(s), \quad h_i(\xi_p) = \delta_{ip}, \; h_i \in \mathbb{P}_N$$



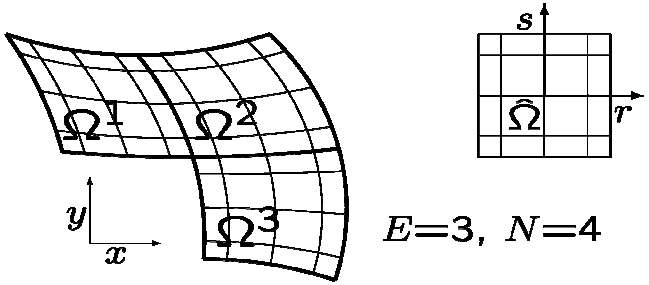
allows derivatives to be evaluated as matrix-matrix products:

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

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

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

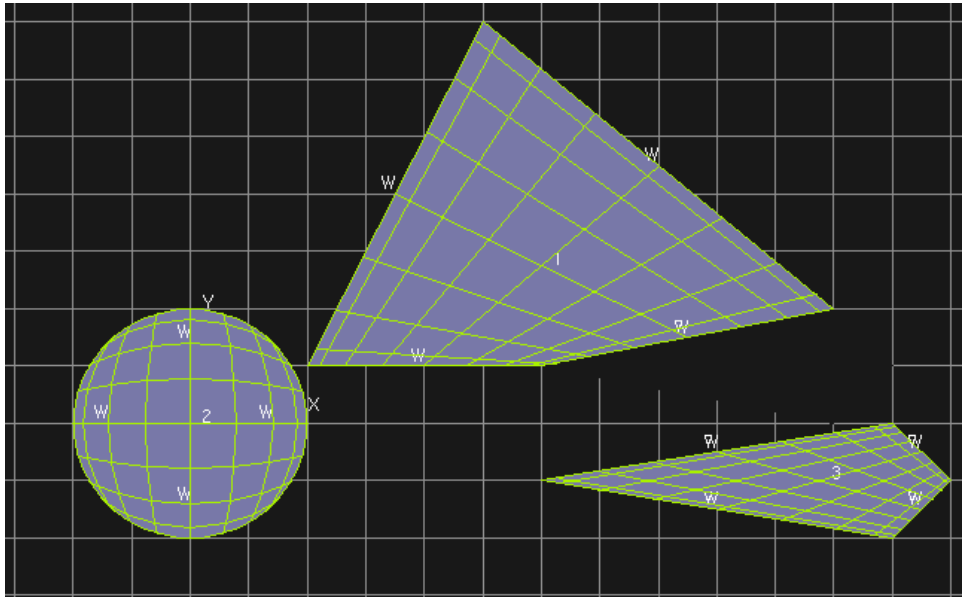
Given $\frac{\partial x_i}{\partial r_j}$, we can find $\frac{\partial r_i}{\partial x_j}$, and thus use the chain rule, e.g.,

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

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*

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

- *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)*

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 in R^d is only $O\,(N^{d+1})$ not $O\,(N^{2d})$ [*Orszag '80*]
- Memory access is 7 x number of points (G_{rr}, G_{rs}, etc., are *diagonal*)
- Work is dominated by matrix-matrix products involving D_r, D_s, etc.

Generic SEM Operator Evaluation

- Spectral element coefficients stored on element basis ($\underline{u}_L$ not $\underline{u}$)

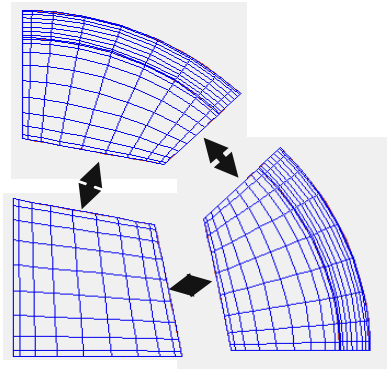
$$\underline{w} = A\underline{u} = Q^T A_L Q\underline{u}, \qquad \underline{w}_L := Q\underline{w}, \qquad \underline{u}_L := Q\underline{u}$$

$$\underline{w}_L = QQ^T A_L \underline{u}_L$$

local work (matrix-matrix products)

nearest-neighbor (gather-scatter) exchange

$$A_L := \begin{bmatrix} A^1 & & & \\ & A^2 & & \\ & & \ddots & \\ & & & A^E \end{bmatrix}$$



- Decouples complex physics (A_L) from communication (QQ^T)

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)

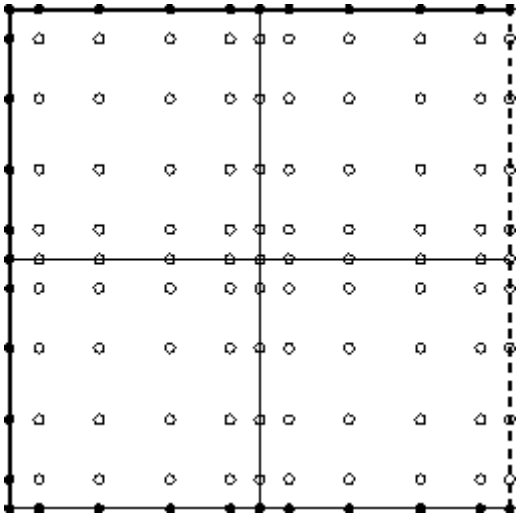
WRT: *Find* $\mathbf{u} \in X^N, p \in Y^N$ *such that:*

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

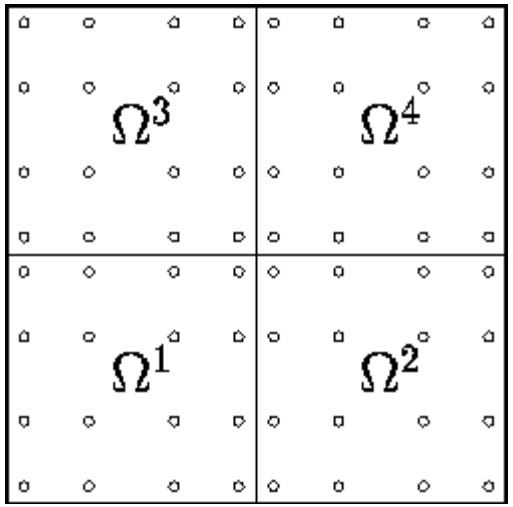
$$- (q, \nabla \cdot \mathbf{u})_G = 0 \qquad \forall\, q \in Y^N \subset L^2$$

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	$T(\mathbf{x}) = T(\mathbf{x} + \mathbf{L})$

Part 3

Workflow Example

Outline

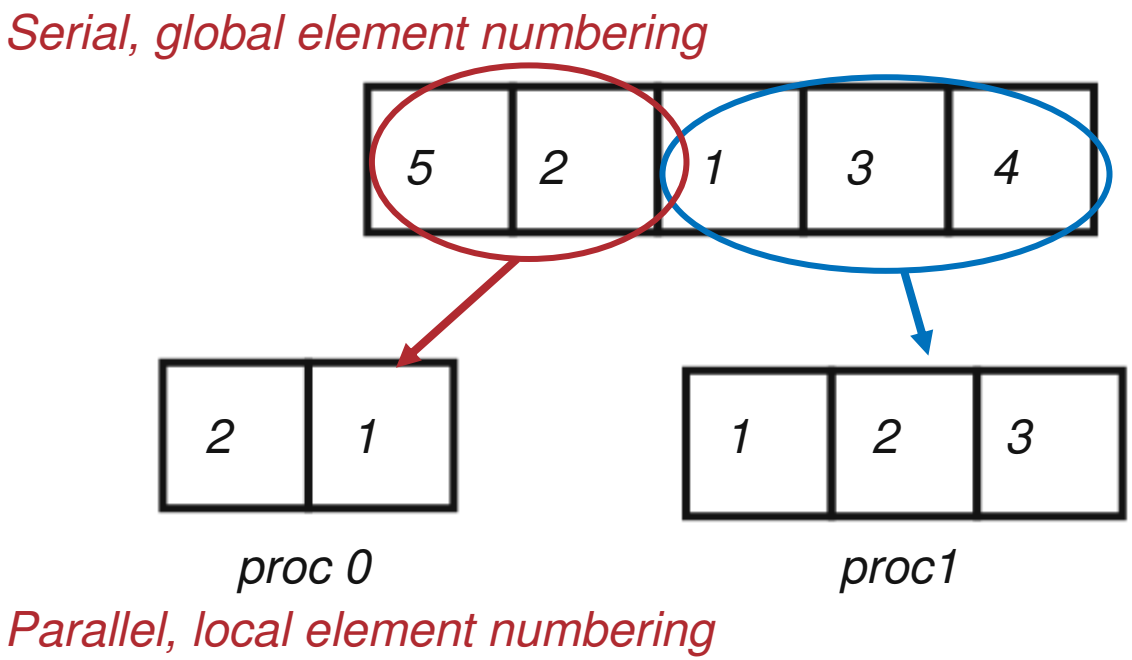
- *Nek5000 capabilities*
- *Equations, timestepping, and SEM basics*
- *Workflow example*
  - *Parallel / serial issues that you should understand*
  - *Setting initial and boundary conditions*
  - *Basic runtime analysis*
- *Using VisIt to analyze results*
- *Mesh generation options*
  - *Building meshes with genbox, prenek, and morphing*
- *Walking through examples; hands on simulations*

Serial / Parallel Issues

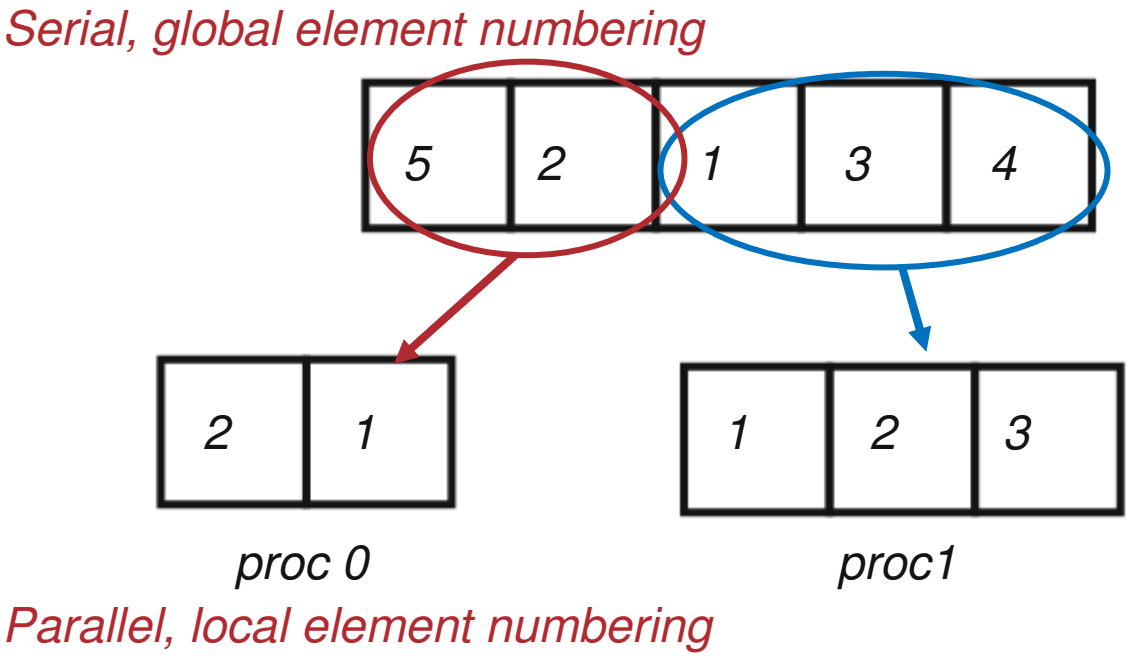
- *Locally, the SEM is **structured.***
- *Globally, the SEM is **unstructured**.*
- *Vectorization and serial performance derive from the structured aspects of the computation.*
- *Parallelism and geometric flexibility derive from the unstructured, element-by-element, operator evaluation.*
- *Elements, 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



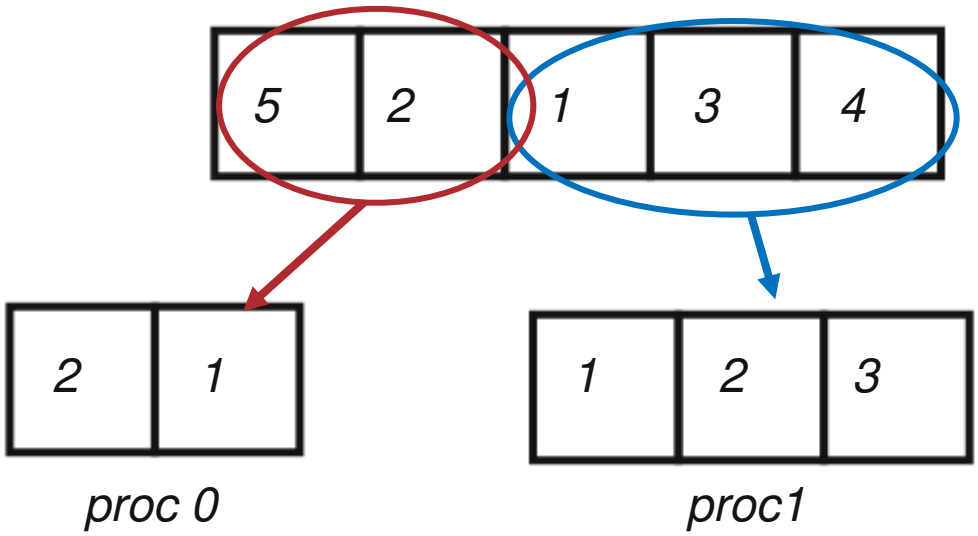
Parallel Structure



- 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()!!

Structure of .usr file

- *Let's look at a file!*

Structure of .rea file

- *Let's look at Kovasznay example…*

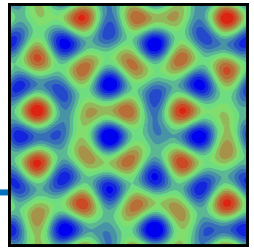
Starting Nek5000 on Fusion

- *Install source and build tools*
  - *ssh to fusion.lcrc.anl.gov*
  - *Add +pgi-9.0 to your .soft file and "resoft"*
  - *svn co https://svn.mcs.anl.gov/repos/nek5 nek5_svn*
  - *cd nek5_svn/trunk/tools and specify compiler in "maketools"*

    F77="pgf77"

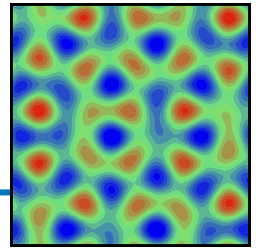
    CC="pgcc"
  - *maketools all*

Running First Case: Eddy Problem



- *cd ~nek5_svn/examples; mkdir t1; cd t1; cp ../eddy/* .*
- *cp ~nek5_svn/trunk/nek/makenek .*
- *makenek eddy_uv*
- *nekb eddy_uv 1 (runs on 1 node = 8 cores)*
  - *Results output to:*
    - logfile – stdout:
      - *timestepping info, computed errors, etc.*
    - eddy_uv.fld01,…,eddy_uv.fld12
      - *velocity & pressure distributions (binary)*

A quick peek at the data



- *Type "postx &", then*

	click	***type***	*comment*
1.	SET TIME	12	load fld12
2.	SET QUANTITY		
3.	VORTICITY		
4.	PLOT		

- *Final error is in* eddy_uv.fld11

- *To check the error:*

	click	***type***	*comment*
1.	SET TIME	11	load fld11
2.	SET QUANTITY		
3.	VELOCITY		
4.	MAGNITUDE		
5.	PLOT		

Eddy Example

- *Q: What does the error look like with outflow inflow/boundary conditions?*

- *A:*
  - *Make a new mesh*
  - *Change the bcs in .rea and .usr files*
  - *Look at the error*

- *To build the new mesh, we'll use genbox*

genbox

genbox

- genbox provides a simple way to generate a basic box mesh comprising an nel_x x nel_y x nel_z array of elements, or a composite mesh with several boxes.

- It uses an existing base mesh as input to specify parameters, etc. and generates a new set of elements and associated boundary conditions.

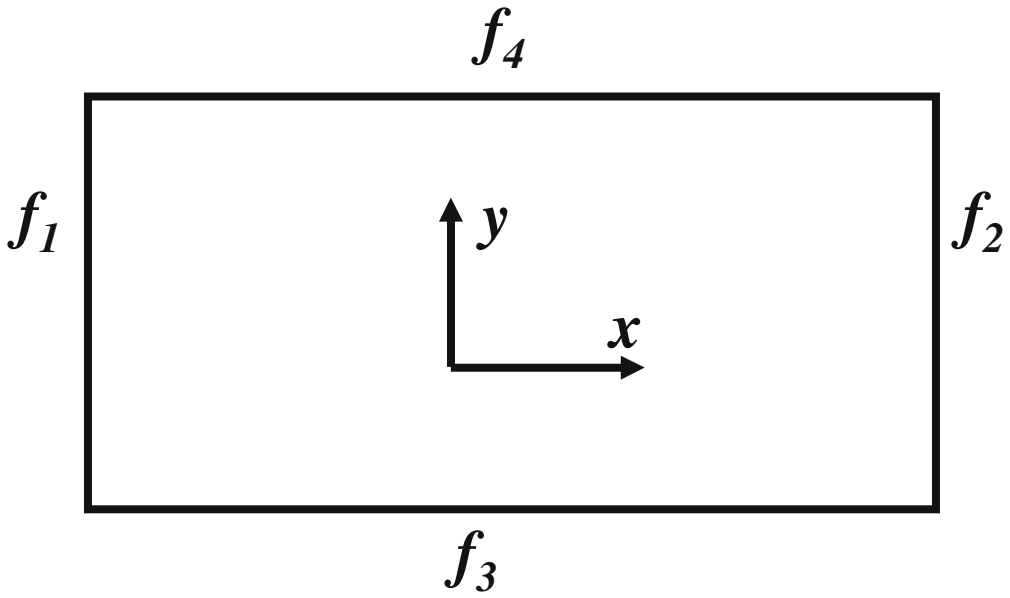
- The output is “box.rea”

- One can then run “genmap”

- Assuming the code is already compiled with an appropriate .usr file, one can then run Nek5000

genbox

- genbox geometry (2D) – uses a symmetric face ordering



- BC: v ,O ,W ,SYM, , yields
  - f1: “velocity”
  - f2: “outflow”
  - f3: “wall”
  - f4: “symmetry”

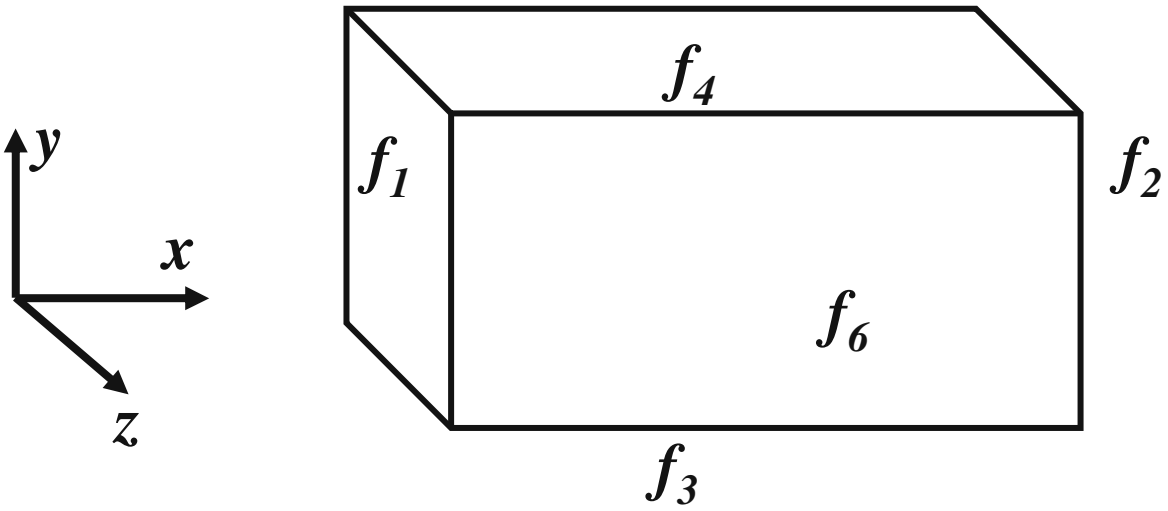
genbox example, 2D

- genbox generates a 2D or 3D input file “box.rea”

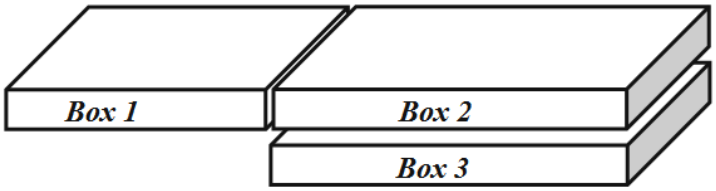
```
#
#  This is a 2D demo file for nek5_svn/trunk/tools/genbox
#
#     --  It produces an 8x5 mesh for channel flow
#         with inflow/outflow boundary conditions
#
#     --  base.rea is assumed to be an existing case
#         w/ parameters of interest already set
#
#
base.rea
2                       spatial dimension
1                       number of fields
#
#  First (and only) box:
#
Box 1
-8  5                       nelx,nely,nelz for Box 1
0.  4.    1.                x0 x1 ratio
-1. -.8 -.4  .4  .8 1.      y0 y1 ... yn
v  ,O  ,W  ,W  ,            BCs  (3 characters each!!)
```

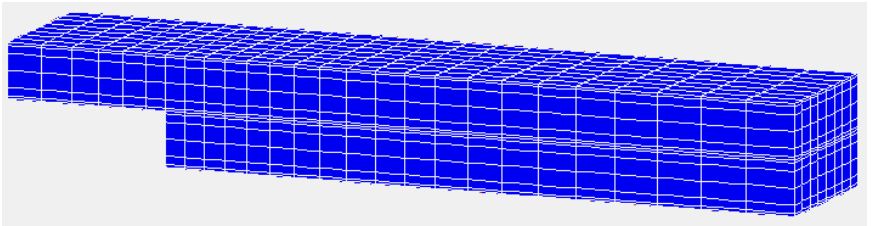
genbox, 3D

- genbox face ordering in 3D:



Multibox Case: Backward Facing Step





```
#  Box file for simple backward-facing step mesh
3d.rea
3                       spatial dimension (if negative dump .re2 file)
1                       number of fields
#
BOX 1
-6 6 -8                       nelx,nely,nelz (negative --> auto spacing)
-3 0 .9                       x_0,  x_n, ratio
0 .05 .20 .5 .8 .95 1.        y_0...y_n
0  3 1.0                      z_0,  z_n, ratio
v  ,   ,W  ,W  ,P  ,P         BCs:   west,east,bottom,top,back,front
BOX 2
-20 6 -8                      nelx,nely,nelz
0  12  1.05                   x_0,  x_n, ratio
0 .05 .20 .5 .8 .95 1.        y_0...y_n
0  3 1.0                      z_0,  z_n, ratio
   ,O  ,   ,W  ,P  ,P         BCs:   west,east,bottom,top,back,front
BOX 3
-20 6 -8                      nelx,nely,nelz
0  12  1.05                   x_0,  x_n, ratio
-1 -.95 -.8 -.5 -.2 -.05 0    y_0...y_n
0  3 1.0                      z_0,  z_n, ratio
W  ,O  ,W  ,   ,P  ,P         BCs:   west,east,bottom,top,back,front
```

- BCs for internal faces are blank
- Use additional boxes for more control over mesh grading, etc.

genbox conventions

- # indicates comment
- If nelx (y, or z) > 0, user provides $x_0,\ldots,x_{nelx}$ in ascending order, possibly on multiple lines
- If nelx (y, or z) < 0, user provides $x_0 < x_{nelx}$, and *ratio,* so that domain $[x_0, x_{nelx}]$ is partitioned into nelx subdomains, with $dx_{i+1} = ratio*dx_i$
- If ndim < 0, genbox generates .rea and .re2 (binary) file [*new convention*]
- “B” or “b” for Box indicates a box descriptor follows
- “C” or “c” for Circle indicates a circle descriptor (currently supported?)
- BCs *must* be 3 characters (including blanks) !
- Base input file must match dimension (2D or 3D) of the given case