On the Next Generation of CFD Tools

S. Turek + Feast Group

M. Altieri, Chr. Becker, S. Buijssen, S. Kilian, D. Kuzmin, F. Platte
J. Hron, P. Kotalik, W. Decheng, A. Ouazzi, R. Schmachtel, L. Rivkind...

Institut für Angewandte Mathematik & Numerik
Universität Dortmund

http://www.featflow.de

Trends in Numerics for PDEs:

'A posteriori error control/adaptive meshing'

'Iterative (parallel) solution strategies'

'Operator-splitting for coupled problems'

Improvement of Numerical Efficiency !!!

$\Updownarrow$

Trends in Processor Technology:

'Enormous improvements in **Processing Data***'*

'Much lower advances in **Moving Data***'*

1 PC in 10 years $\approx$ 1 CRAY T3E today !!!

Questions:

'Can we use this enormous computing power ?'

'Particularly: *For Numerics for PDEs ???'*

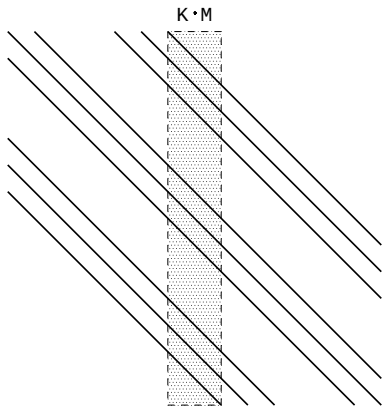
'If **not**, *how to achieve a significant percentage ?'*

$\Downarrow$

Sparse Matrix-Vector techniques (FEM)

```
      DO 10 IROW=1,N
      DO 10 ICOL=KLD(IROW),KLD(IROW+1)-1
10     Y(IROW)=DA(ICOL)*X(KCOL(ICOL))+Y(IROW)
```

Sparse Banded MV techniques (FD)



Results (MFLOP/s) for IDENTICAL matrices

2D case	N	Sparse	SB-V	SB-C	**MG-V**	**MG-C**
DEC 21264	65^2	205 (178)	538	795	370	452
'**ES40**'	1025^2	**78 (11)**	**158**	**813**	**185**	**401**
HITACHI	65^2	173 (82)	238	391	191	266
'**SR8000**'	1025^2	**144 (7)**	**226**	**390**	**200**	**267**
AMD K7	65^2	203 (195)	101	556	122	355
'**ATHLON**'	1025^2	**31 (10)**	**64**	**236**	**58**	**126**

Sparse **MV techniques**

- MFLOP/s rates far away from '**Peak Performance**'
- PC partially **faster** than processors in 'supercomputers' !!!

Sparse Banded **MV techniques**

- FEM-Simulation for **complex domains** ???

$\Downarrow$

Complete Simulation Tools much slower !!!

FEAST project

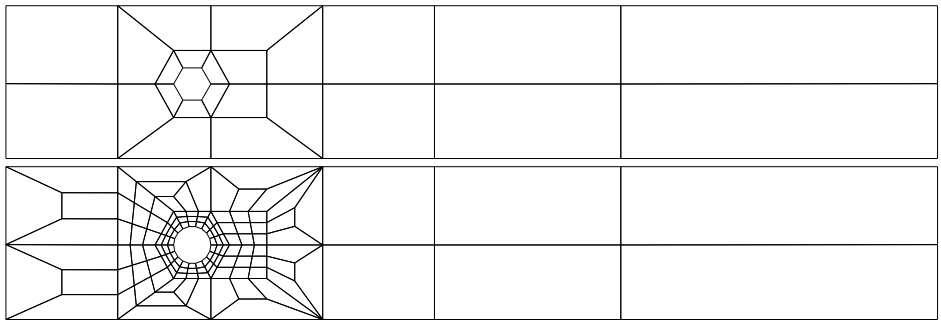
Implementation techniques and
Numerics (!) adapted to **hardware !!!**

$\Uparrow$

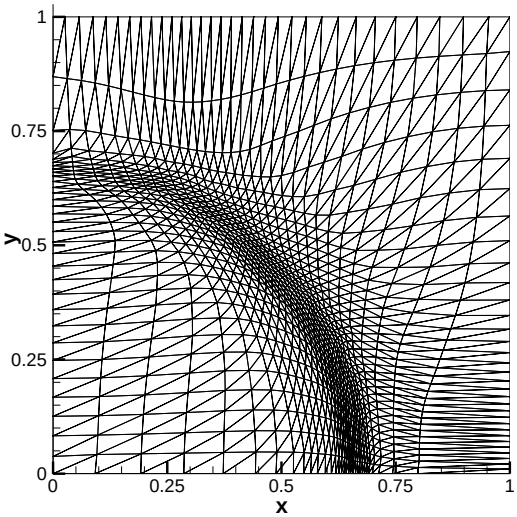
Hardware-oriented Numerics

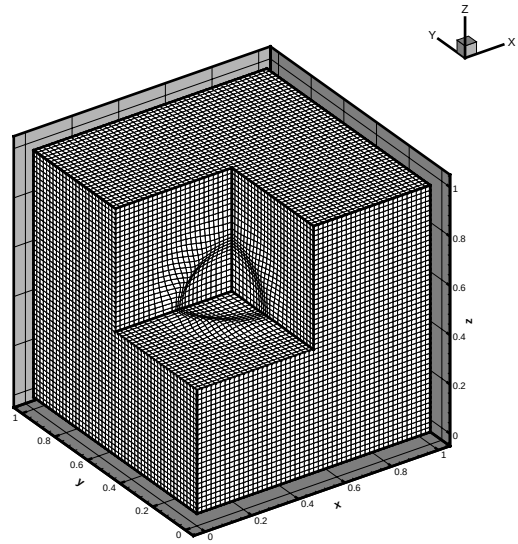
1) Concepts for adaptive meshing

1) macro-oriented adaptivity



2) patchwise adaptivity





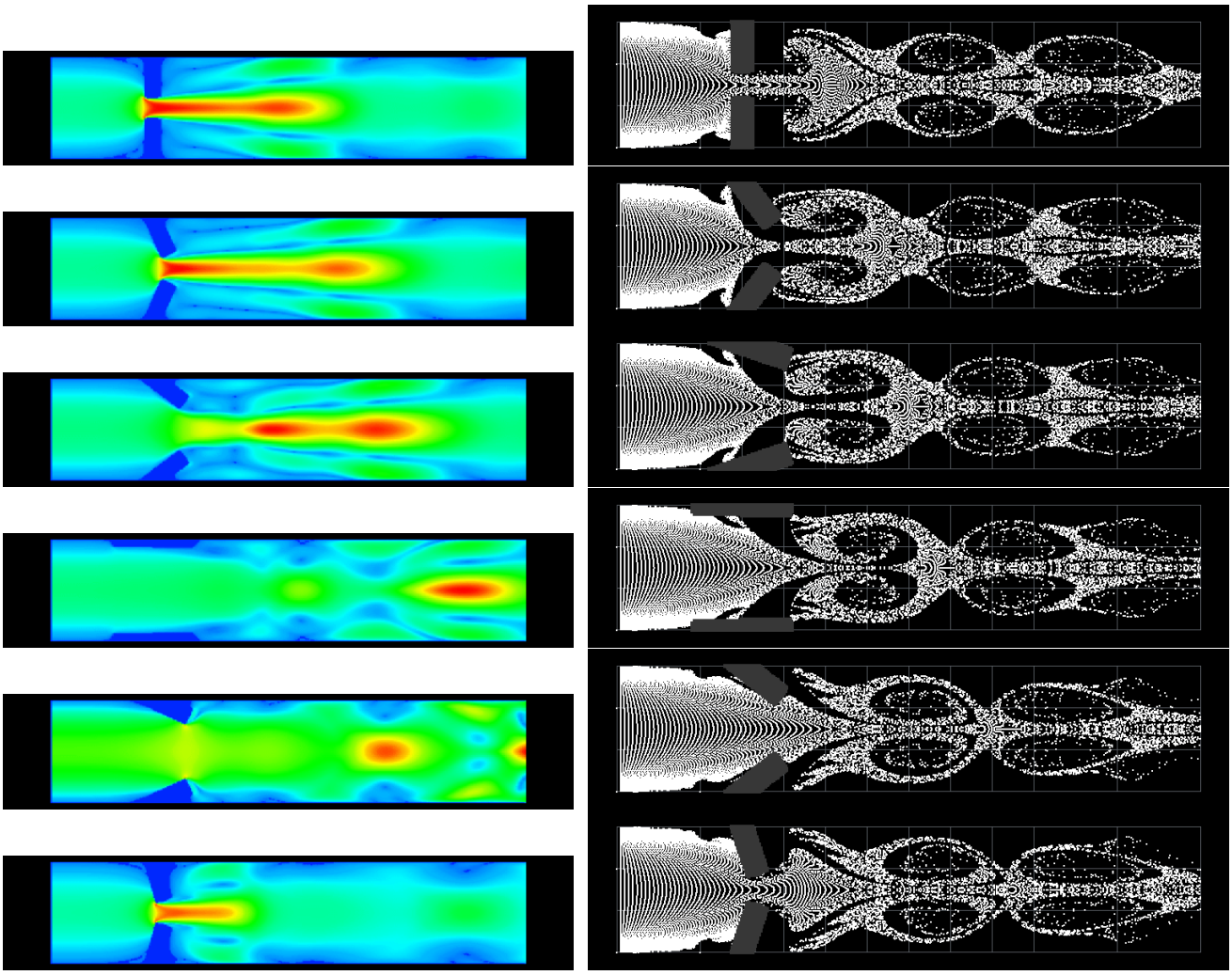
3) local adaptivity

$\rightarrow$ hanging nodes, triangles/quads,...

Example: Time-dependent Geometries

‘Fictitious Boundary Techniques’

Iterative filtering techniques on **time-independent** *grids*
without *explicit capturing of* **time-dependent** *geometries*



II) Concepts for iterative (parallel) solvers

1) Standard Multigrid

- parallelization of 'recursive' smoothers (only blockwise)?
- **too few arithmetic work vs. data exchange!**

2) Standard Domain Decomposition

- good ratio for communication/arithmetic work!
- **bad convergence behaviour w.r.t. multigrid!**

$\Downarrow$

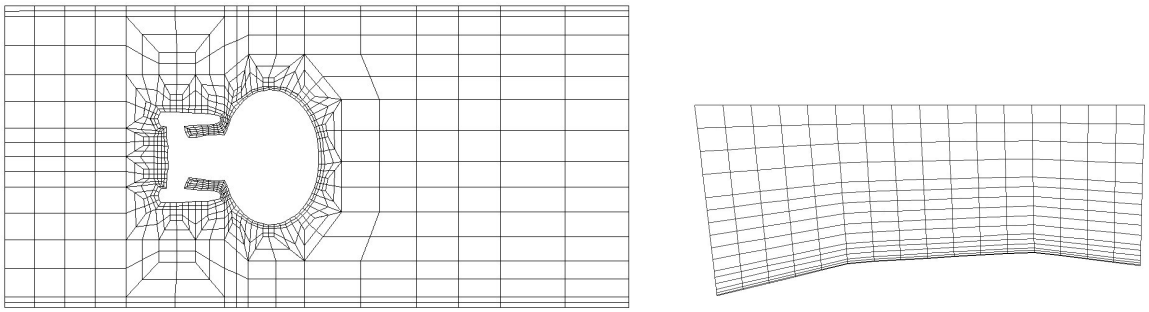
1) + 2) = ScaRC

Hide recursively all anisotropies in 'local' units! (**robustness**)

Perform all Linear Algebra tasks on 'local' units only! (**efficiency**)

Example: Realization of ScaRC in FEAST

2D decomposition and zoomed (macro) element (LEVEL 3) with locally anisotropic refinement towards the wall

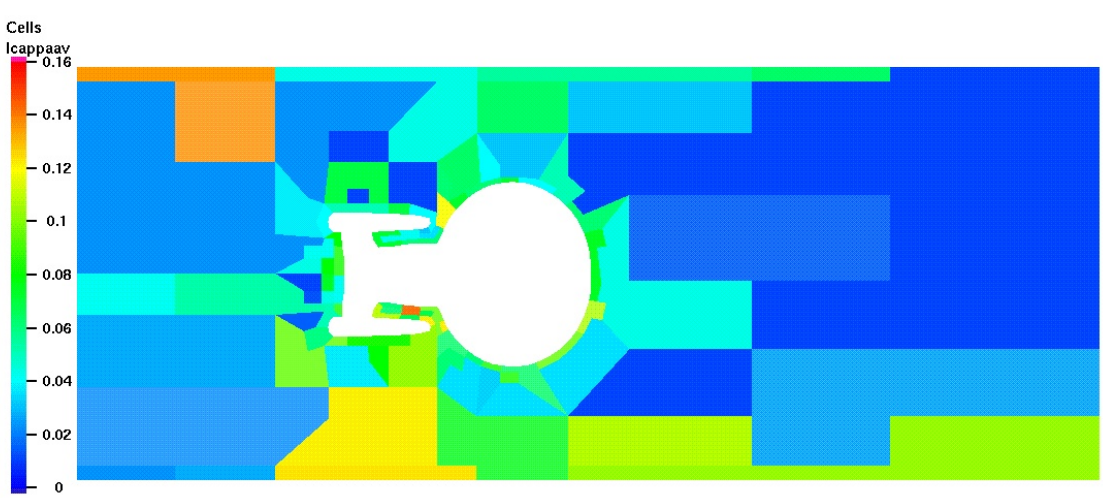


ScaRC-CG solver (smoothing steps: 1 global ScaRC ; 1 local 'MG-TriGS') for locally (an)isotropic refinement

Global (parallel) convergence rates

$\#NEQ$	Dirichlet 'Velocity'		Neumann 'Pressure'	
	$AR \approx 10$	$AR \approx 10^6$	$AR \approx 10$	$AR \approx 10^6$
$843,776$	0.17 (**8**)	0.17 (**8**)	0.20 (**9**)	0.17 (**8**)
$3,375,104$	0.18 (**9**)	0.19 (**9**)	0.22 (**10**)	0.22 (**10**)
$13,500,416$	0.19 (**9**)	0.18 (**9**)	0.23 (**10**)	0.23 (**10**)

Local convergence rates (for $AR \approx 10^6$)



III) MPSC: Incompressible Flow Solvers

Results with FEATFLOW

Computer	total time	'memory intensive'	'floating point'
IBM SP2 (160 MHz)	**2748**	1587 (**58%**)	1161 (**42%**)
PC PII (400 MHz)	**4927**	2401 (**49%**)	2526 (**51%**)
IBM SP2 (66 MHz)	**5970**	3824 (**64%**)	2146 (**36%**)

- **'Discrete Projection':** Burgers + Pressure Poisson
- Streamline-Diffusion FEM discretization in 3D
- Implicit adaptive time stepping
- *Faster solvers useless!*

$\Downarrow$

How to increase the 'floating point' intensive parts?

Larger time steps!

How???

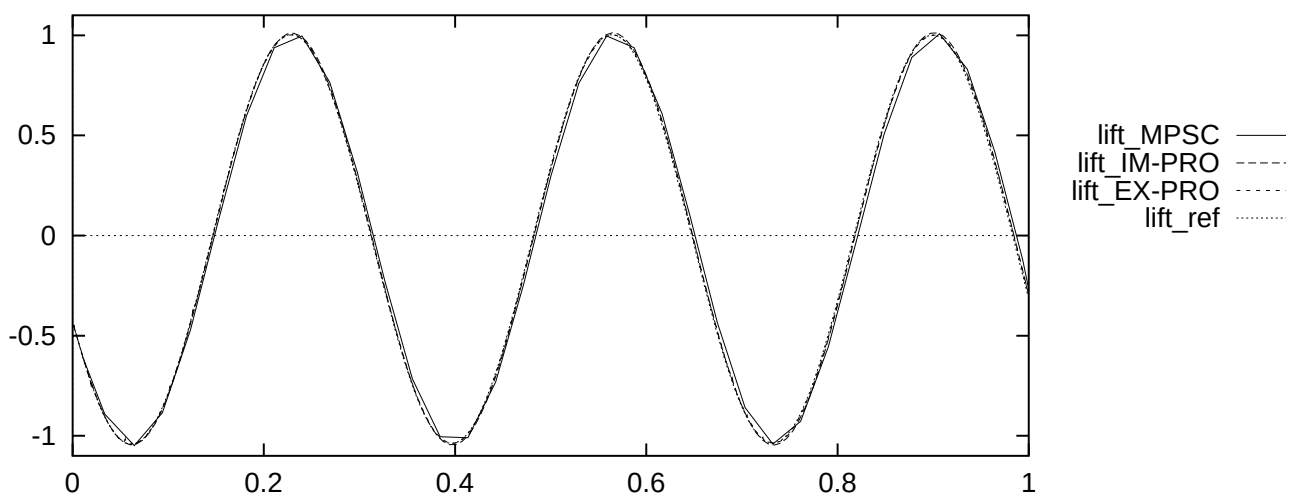
Stronger velocity-pressure coupling:

'Perform more arithmetic work with the once assembled matrices and vectors'!

$\Downarrow$

Multilevel Pressure Schur Complement

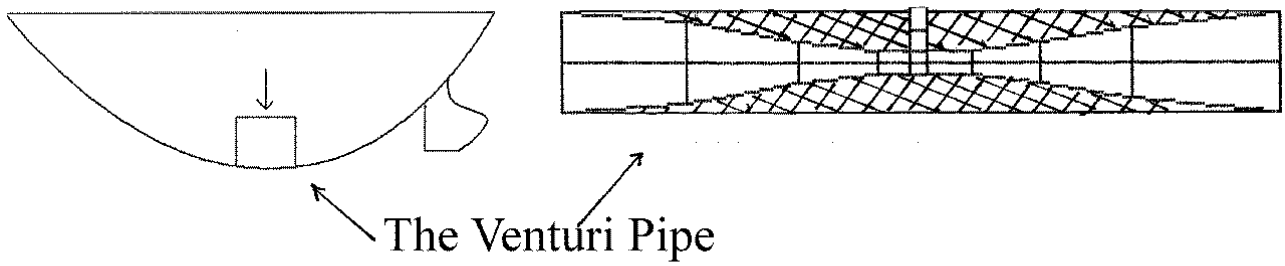
CPU (solvers)	Method	#NT
14,358 (**81%**)	fully impl. MPSC	**39**
42,679 (**51%**)	semi-impl. DPM	**165**
64,485 (**54%**)	semi-expl. DPM	**889**



$\Downarrow$

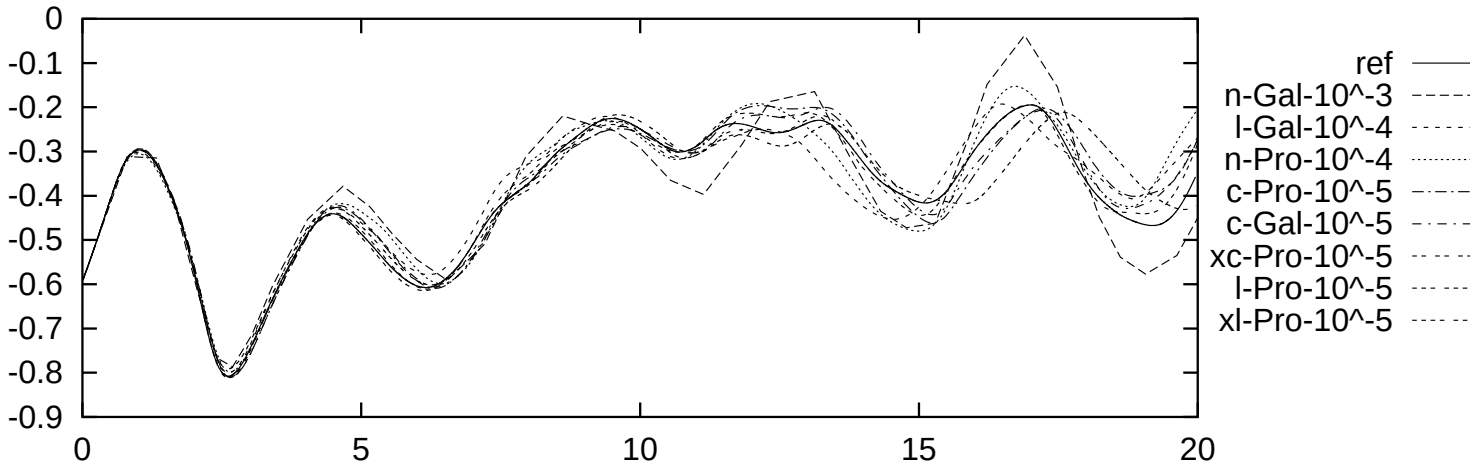
'Ready' for SPARSE BANDED BLAS
'Ready' for Galerkin-type error control

Test: Flow through Venturi Pipe



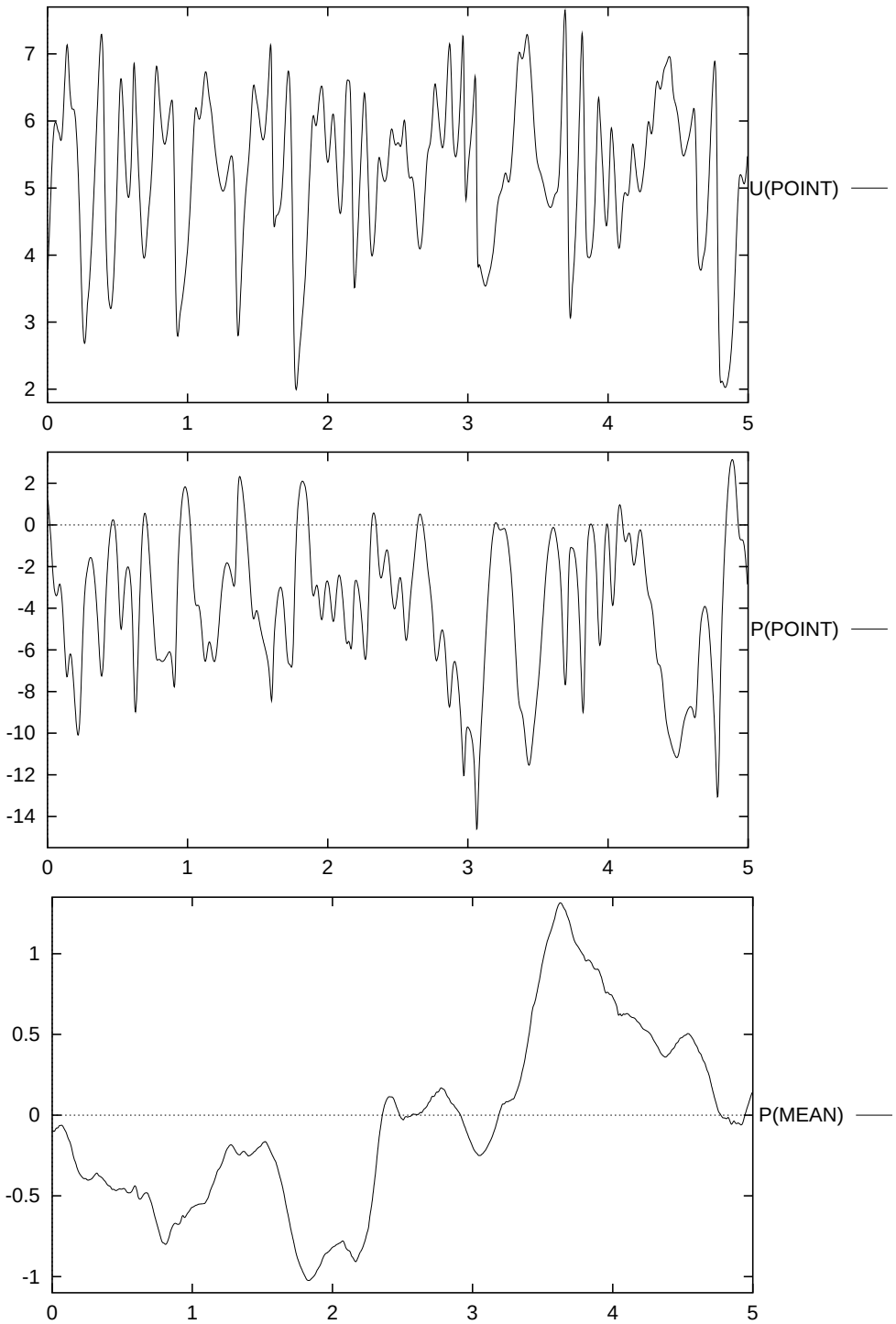
Adaptive time step control for **velocity** (global, l_2) and **flux** (through upper device **!!!**)

$$|f(\mathbf{u}_h^k(t_n), p_h^k(t_n)) - f(\mathbf{u}(t_n), p(t_n))| \leq TOL$$



	Meth.	#NT	**Flux**		**Pressure**	
			l_2	mean	l_2	mean
	n-MPSC	**38**	15%	1%	19%	1%
O	l-MPSC	**222**	11%	2%	13%	0%
K	n-DPM	**263**	11%	2%	14%	3%
	l-DPM	**649**	4%	0%	5%	0%
	xl-DPM	**6941**	11%	0%	14%	0%

Test: Perpendicular inflow into pipe

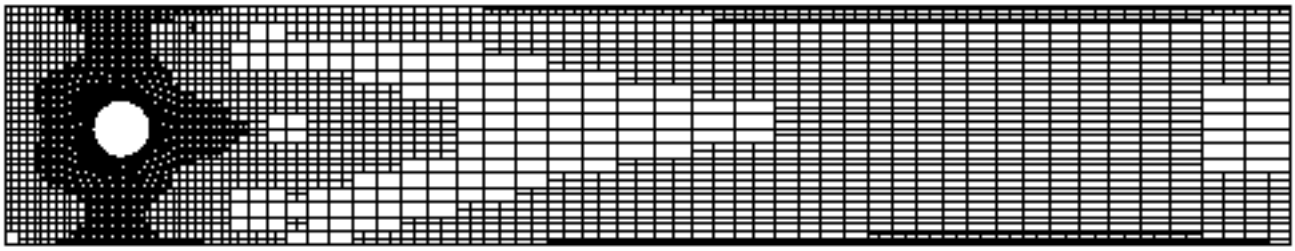


Rigorous adaptive error control of **mean values** ('wall pressure') in **space/time**?!?

Hardware-Oriented Numerics for PDEs:

1) *Patch-oriented adaptivity*

'Many' (local) **tensorproduct grids** (S-B BLAS)
'Few' (local) **unstructured** *grids* (SPARSE)



2) *Generalized MG-DD solver:* ScaRC

Exploit locally 'regular' structures **(efficiency)**
Recursive 'clustering' of anisotropies **(robustness)**

'Strong local solvers improve global convergence !'

3) *Navier-Stokes solvers: Full Galerkin MPSC*

Develop Navier-Stokes solvers with <u>*more*</u> *'arithmetic' than 'memory intensive' operations*

'Exploit locally regular structures !!!'

Consequences for FEATFLOW:

Prototypical version of FEAST is available

Prototypical version of FEASTFLOW is expected for 2002/3

DVD version of FEATFLOW + 'Virtual Album' (full) in 2001

'FEATFLOW spring school' in 2002

'Last' FEATFLOW version in 2002 with many applications