.

On implementation techniques for boundary conditions

Applications to CFD

S. Turek, L. Rivkind

Institut für Angewandte Mathematik, Universität Heidelberg

Im Neuenheimer Feld 294, 69120 Heidelberg, Germany

`http://gaia.iwr.uni-heidelberg.de/~ture`

`http://www.iwr.uni-heidelberg.de/~featflow`

⇓

October 1999: **Universität Dortmund**

Application of 'standard' codes to:

1) **Time-dependent** domains/boundaries ?

2) **Complex** details $\longleftrightarrow$ Sequence of **multigrid** meshes ?

$\Downarrow$

Typical approaches:

1) **Re-meshing in each time step**

$\longrightarrow$ CPU-intensive: *Data access vs. FLOPs ???*

2a) **Coarse mesh with all details** $\rightarrow$ **3D ???**

2b) **Fine mesh $\rightarrow$ coarse meshes via coarsening**

$\longrightarrow$ Nested or non-nested ???

$\longrightarrow$ Efficiency of intergrid: *Convergence rates/CPU ???*

$\longrightarrow$ Practical experience ???

2c) **"Very coarse" mesh $\rightarrow$ local refining (Sauter)**

$\longrightarrow$ Practical experience ???

What are the problems in realistic applications???

Many highly-tuned codes are available only for globally refined variants of fixed (semi-adaptive) coarse meshes!

Computational efficiency of SPARSE Matrix-Vector applications?

Computational efficiency of meshing strategies in nonsteady simulations?

$\Downarrow$

Alternative (additional) approach:

Iterative Filtering Techniques for Fictitious Boundary Conditions

Iterative Filtering Techniques:

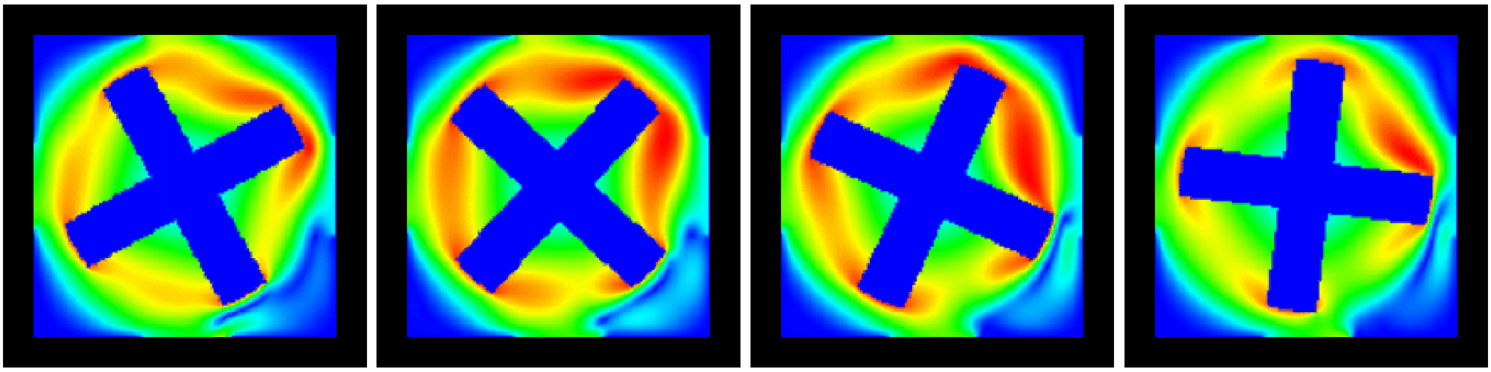
1. (Rough) boundary parametrization for large-scale structures!
2. Fine-scale structures as (level-dependent) interior objects!
3. Use projectors in iterative components to the "right" b.c.'s!

$\Downarrow$

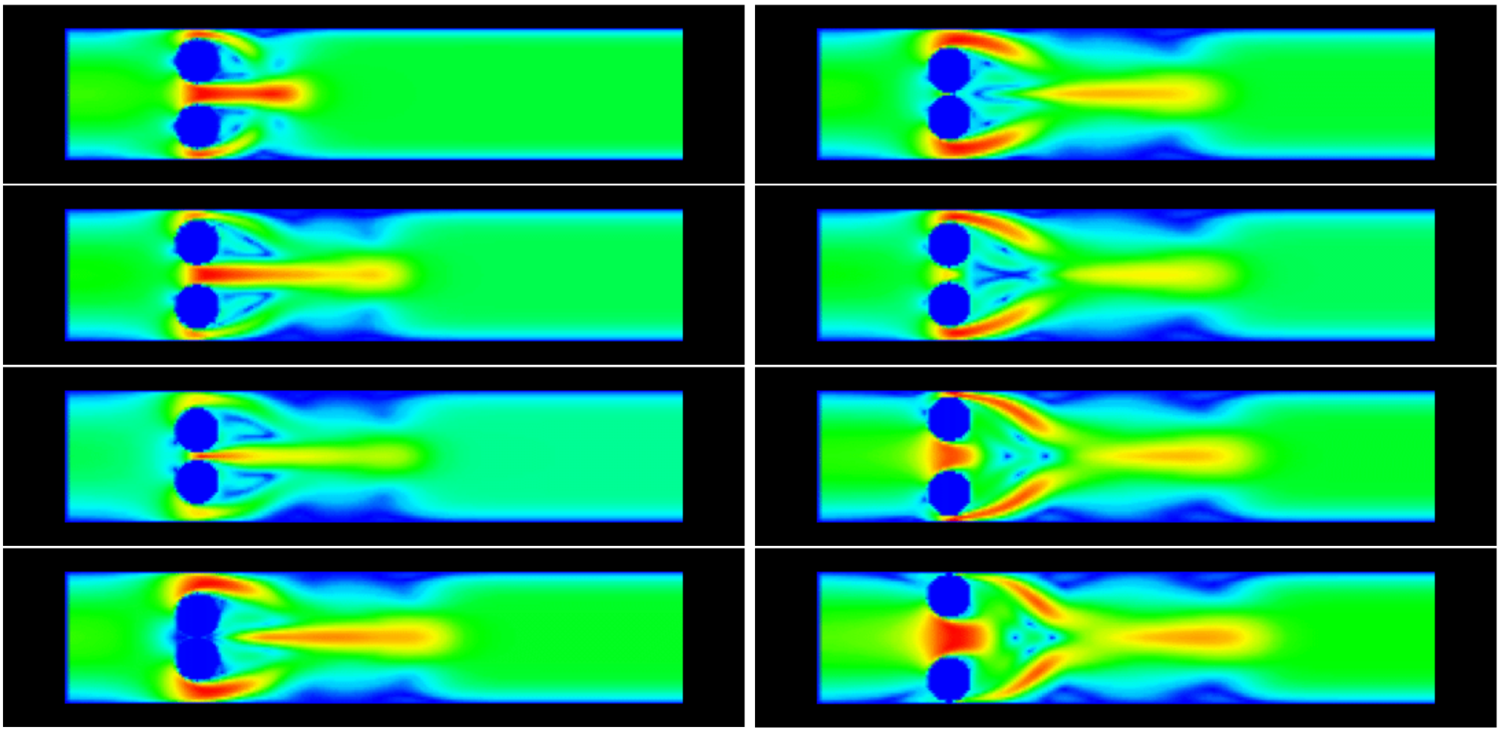
Examples for possible applications

Examples from the 'Virtual Album' I

'Rotating propeller in a 2D chamber'

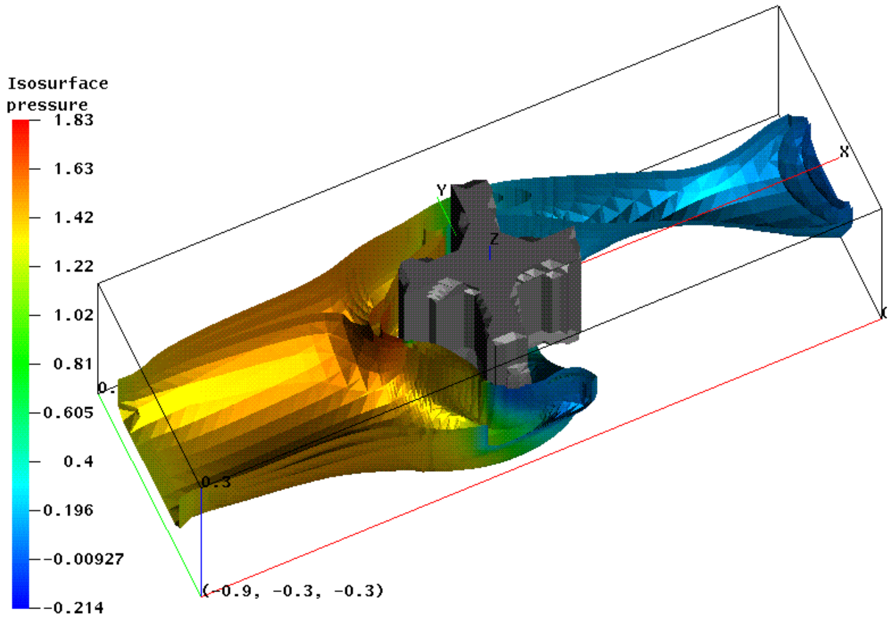


'2D channel flow around moving balls'

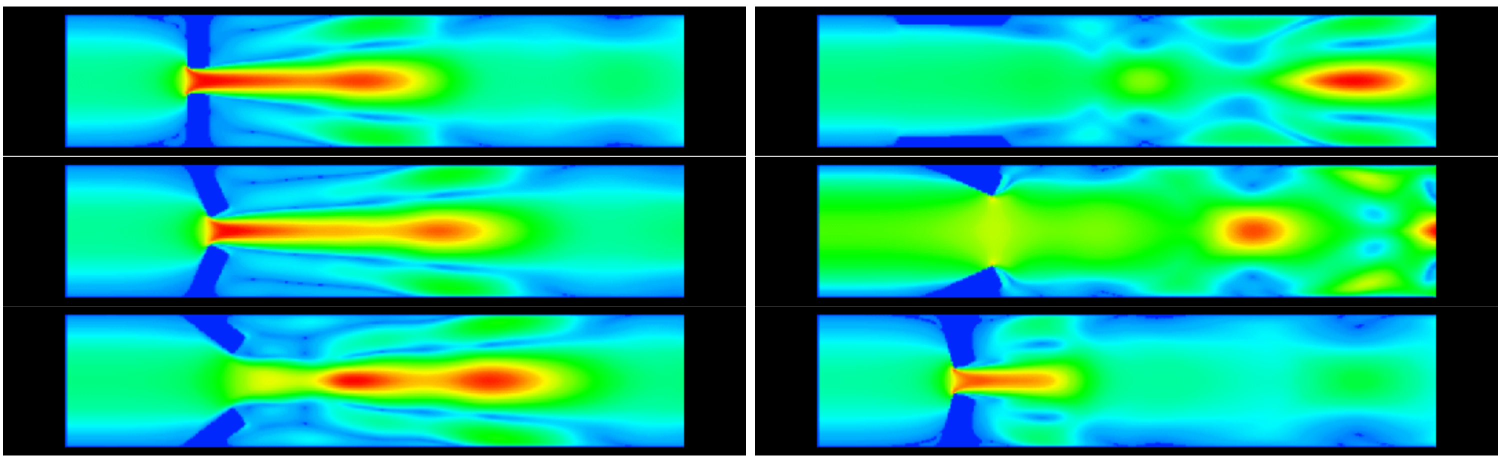


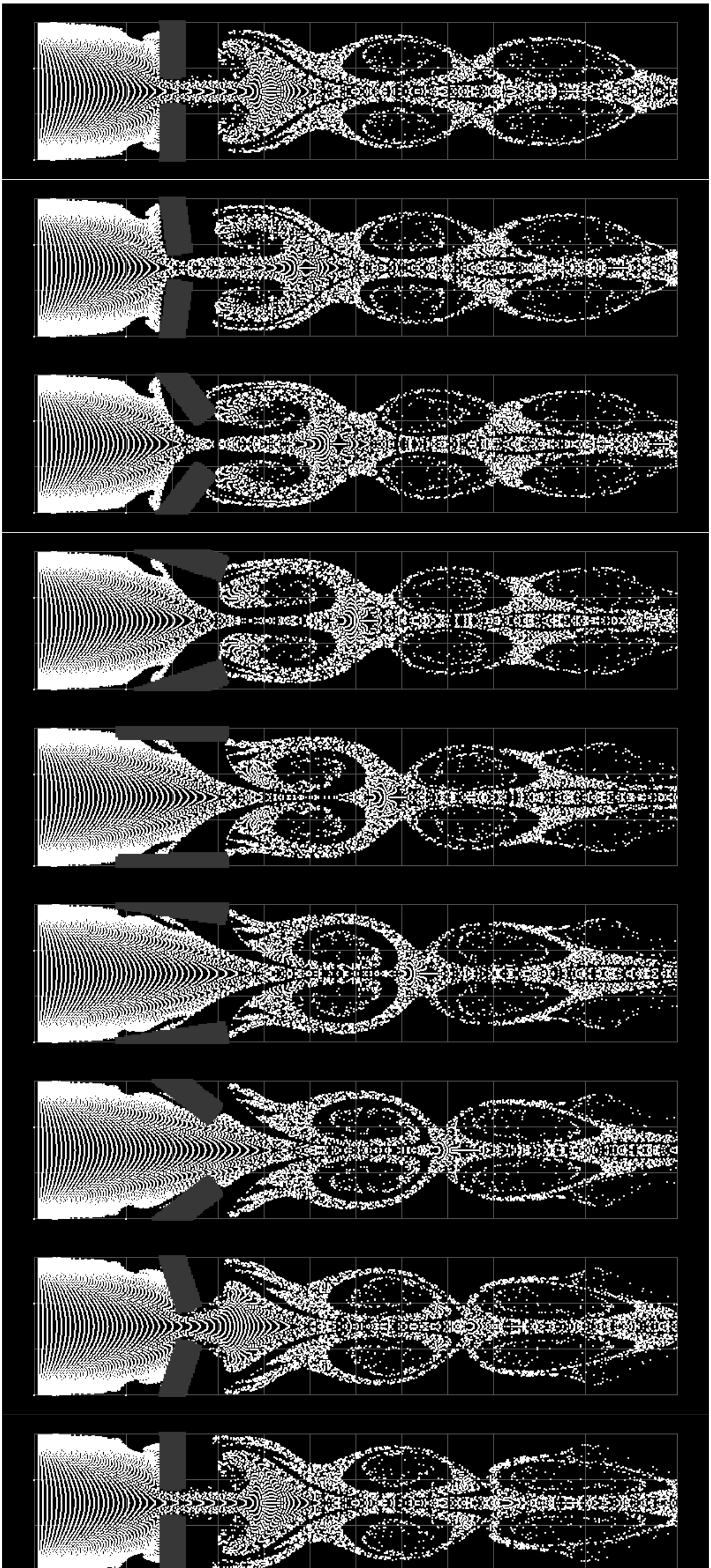
Examples from the ‘Virtual Album’ II

‘3D channel flow around rotating propeller’



‘2D channel flow around moving flaps I’





B.c.'s in iterative FEM codes:

1. **Fully explicit**

   $\rightarrow$ Eliminate rows/columns in matrix (**Dirichlet b.c.'s**)!

   $\rightarrow$ Eliminate corresponding components of solution/r.h.s.!

2. **Semi-implicit** (∼ **Standard**)

   $\rightarrow$ Modify rows in matrix only $\Rightarrow$ **Identity** matrix!

   $\rightarrow$ Prescribe correct values for components of solution/r.h.s.!

   $\rightarrow$ NO ELIMINATION $\Rightarrow$ components will NOT change!!!

3. **Fully implicit**

   $\rightarrow$ let the assembled matrix $\Rightarrow$ natural b.c.'s!

   $\rightarrow$ Prescribe correct values for components of solution/r.h.s.!

   $\rightarrow$ Apply iterative sub-step!

   $\rightarrow$ Prescribe correct values for components of solution/r.h.s.!

   $\rightarrow$ NO ELIMINATION $\Rightarrow$ components DO change!!!

Advantages of Iterative Filtering Techniques:

- Use of standard codes with fixed (semi-adaptive) meshes only!
- Same matrix for different b.c.'s ⇒ RAM/CPU!
- Preserve computational efficiency for nonsteady domains!
- Superconvergence through (locally) orthogonal meshes!
- High Performance ⇒ FEAST, SPARSE BANDED BLAS!

⇓

Questions:

1. Multigrid convergence ???
2. Accuracy near "real" boundaries ???
   ⟶ lift, drag, pressure distribution

⇓

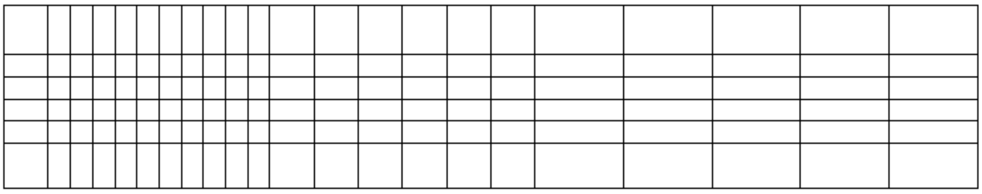
FEATFLOW: CC2D

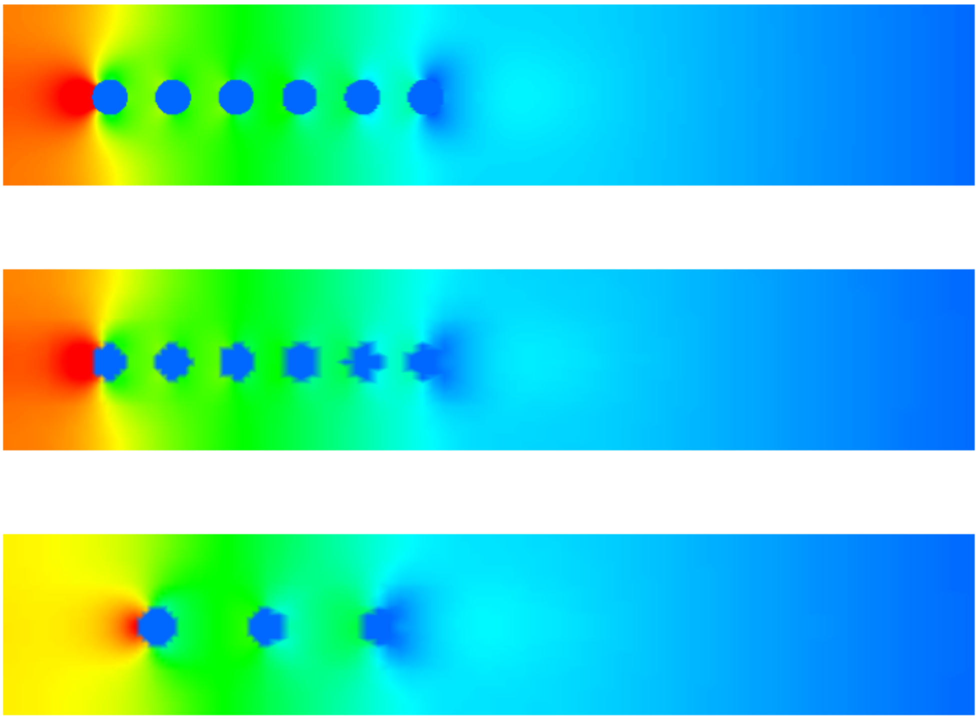
1. rotated multilinear/constant $\tilde{Q}_1/Q_0$
2. fully coupled solver with Vanka-like smoother
3. adaptive upwinding/streamline-diffusion

Numerical Efficiency

lev	circle	channel
3	0.12	0.13
4	0.11	0.11
5	0.13	0.11
6	0.10	0.10

MG(STOKES)





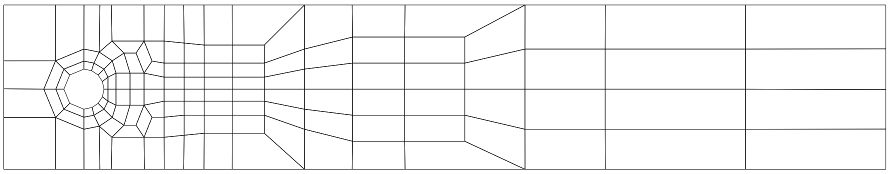
lev	NLEV	6
3	6/11	6/11
4	6/11	6/11
5	6/11	6/11
6	6/11	6/11

NNL/NMG
RE = 20

Numerical Accuracy I

DFG Benchmark 'Flow around cylinder': Steady case

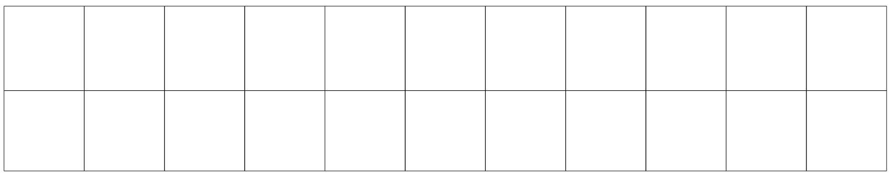
CIRC1



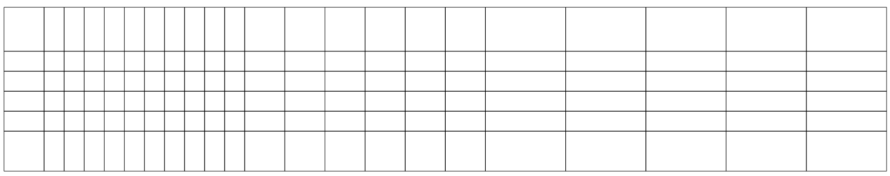
CIRC2

CIRC3

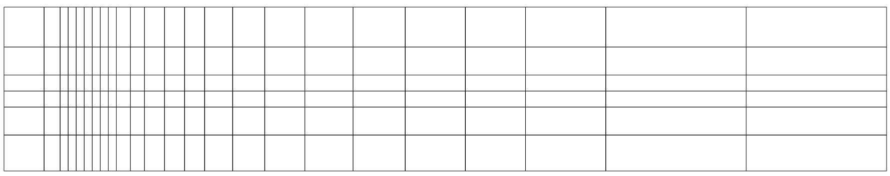
CHAN1



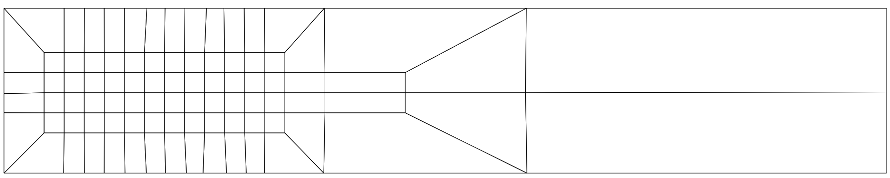
CHAN2



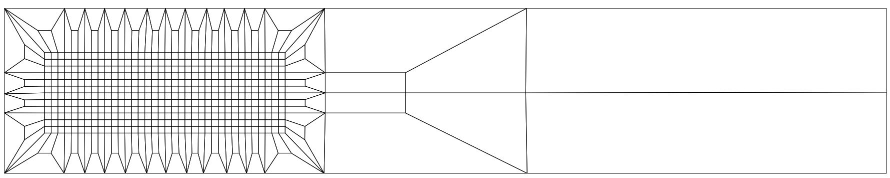
CHAN3



CHAN4



CHAN5



5 % Accuracy

	CIRC1	CIRC2	CIRC3	CHAN1	CHAN2	CHAN3	CHAN4	CHAN5
U1	2080	5632	32768	5632	8448	9216	5632	9088
U2	2080	352	2048	352	2112	2304	1408	9088
P	2080	1408	8192	352	2112	2304	5632	36352
ΔP	2080	1408	2048	5632	2112	2304	5632	9088

1 % Accuracy

	CIRC1	CIRC2	CIRC3	CHAN1	CHAN2	CHAN3	CHAN4	CHAN5
U1	8320	90112	131072	360448	135168	36864	90112	145408
U2	2080	1408	8192	1408	2112	9216	1408	9088
P	8320	22528	131072	1408	2112	9216	90112	581632
ΔP	33280	22528	32768	360448	135168	9216	90112	145408

Results for low Re numbers:

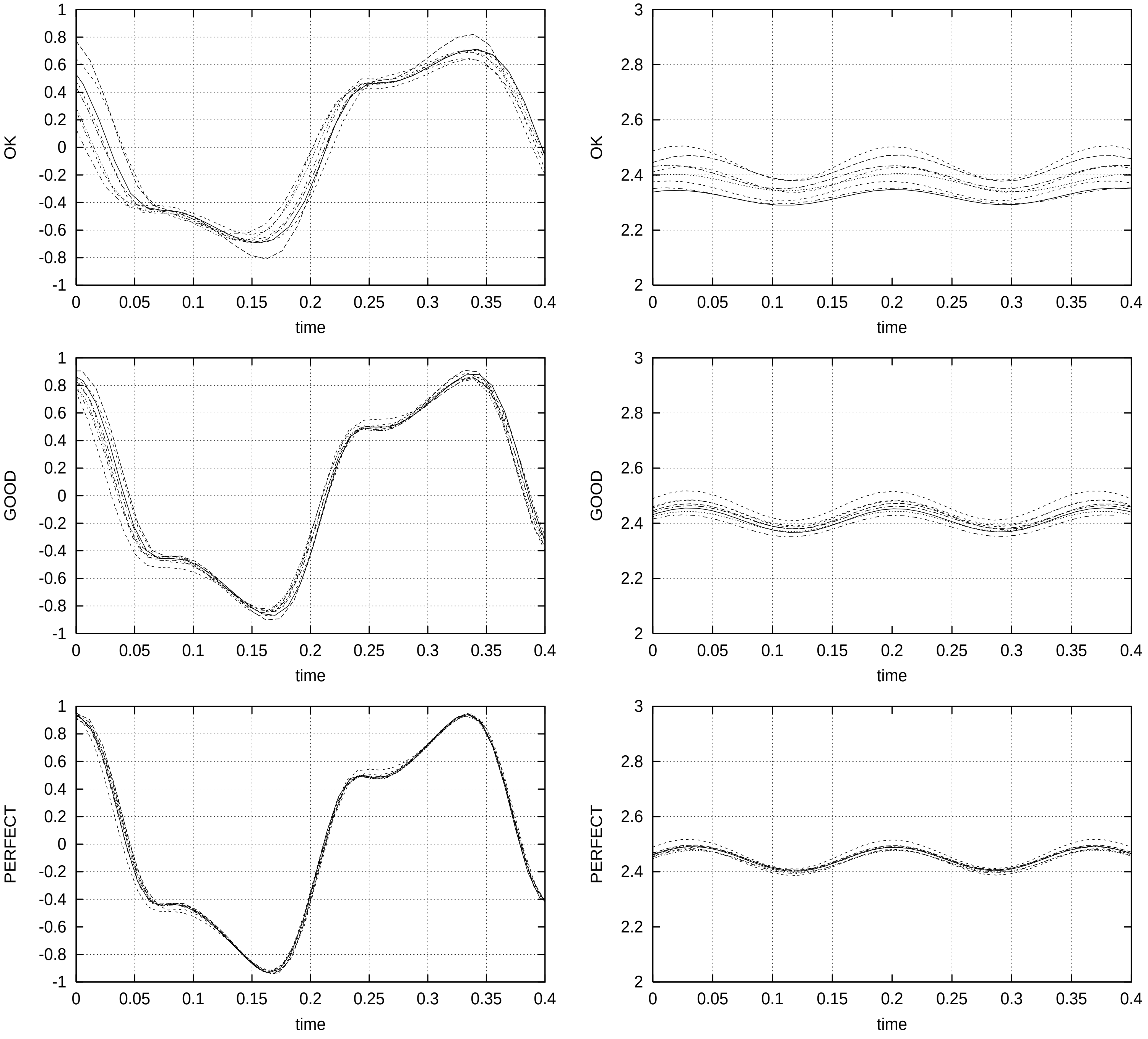
- Globally uniform meshes good for **non-b.c. values** (U2,P)
- Local grid adaptation needed for **b.c. value** (ΔP)
- Combination (U1) ???

$\Downarrow$

CIRC1, CIRC2 + CHAN3

Numerical Accuracy II

DFG Benchmark 'Flow around cylinder': Nonsteady case



U2 ΔP

'OK'

	CIRC1	CIRC2	CIRC3	CHAN1	CHAN2	CHAN3	CHAN5
U2	8320	22528	32768	22528	8448	9216	9088
ΔP	8320	22528	32768	22528	8448	9216	9088

'GOOD'

	CIRC1	CIRC2	CIRC3	CHAN1	CHAN2	CHAN3	CHAN5
U2	33280	90112	131072	90112	33792	36864	36352
ΔP	33280	22528	131072	90112	33792	36864	36352

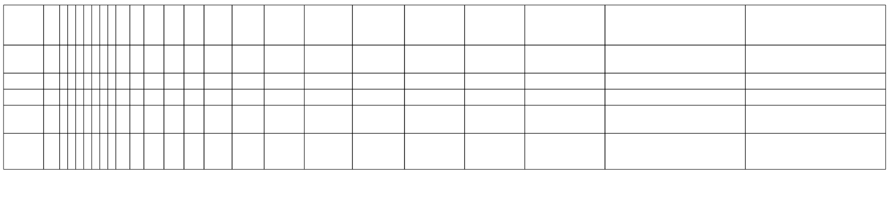
'PERFECT'

	CIRC1	CIRC2	CIRC3	CHAN1	CHAN2	CHAN3	CHAN5
U2	133120	360448	524887	360448	135168	147456	145408
ΔP	133120	90112	131072	360448	135168	147456	145408

$\Downarrow$

Results for medium Re numbers:

- Combination of local and global adaptation necessary
- No explicit advantage of 'body-fitted' meshes



- **But:** Local adaptivity ??? Rigorous error control ???

Conclusions:

- Many standard codes can be modified for nonstationary domains and complex geometries via **Iterative Filtering Techniques**
- **Projection techniques** and **locally orthogonal** meshes give good quality with excellent computational performance

⇓

Nice tools for qualitatively good simulations of complex (nonsteady) configurations !!!

+

Quantitatively accurate results via (locally) blockstructured meshes and local adaptivity !!!

- Combination with **hierachical data, solver and matrix structures** provides computational and numerical efficiency

⇓

FEAST + SPARSE BANDED BLAS

- Application to **free boundaries**, **multi-phase flow** and **fluid-structure coupling** (→ Glowinski)

⇓

FEATFLOW