Concepts of

High Performance FEM Simulation

and Realization in the FEAST Software

S. Turek + FEAST GROUP

M. Altieri, Chr. Becker, S. Buijssen, S. Kilian, D. Kuzmin, ...

Institut für Angewandte Mathematik & Numerik
Universität Dortmund

`http://www.featflow.de`

Classification of PDE software:

For education: MATLAB, etc.

⇒ *'play–around' with Mathematics, 'easy' user interface*
⇒ *simple, but robust algorithms, easy applicable*
⇒ *'independent of implementation/language'*

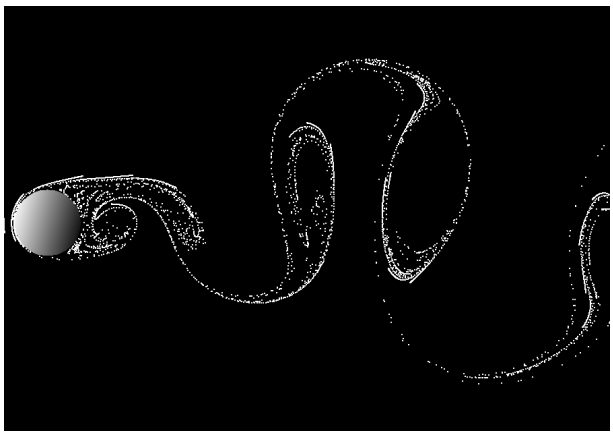
For research: FEAT, UG, DIFFPACK, KARDOS, LIMA, etc.

⇒ *open for numerical and algorithmic changes*
⇒ *flexible, but robust data structures, reusable components*
⇒ *'independent of user interface'*

For applications:

⇒ *'optimal play–together of numerics* **and** *implementation'*
⇒ *'real life' configurations, PC/Workstation/Supercomputer*
⇒ *robustness* **and** *efficiency*

'Critical' example: CFD



‘Realistic’ CFD applications:

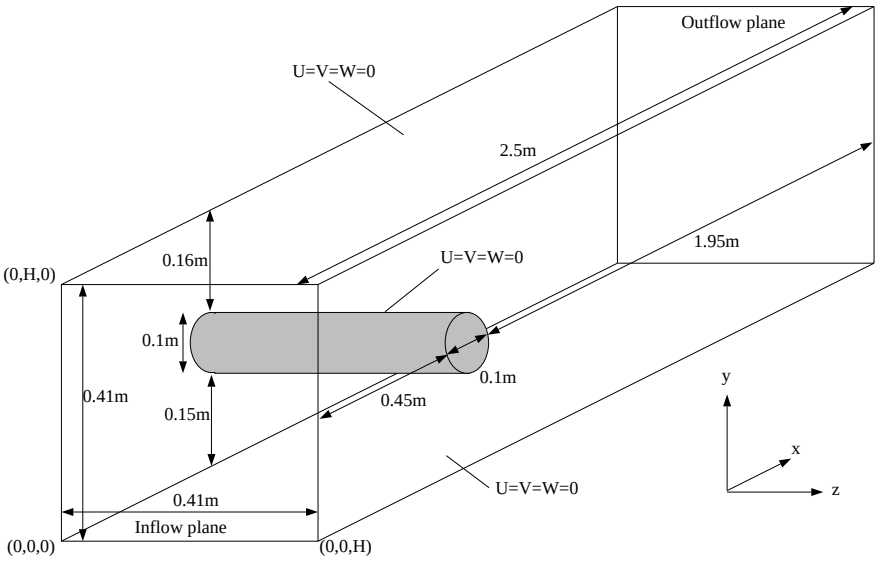
- complex domains, anisotropic meshes in space and time
- $10^3 - 10^5$ time steps $+$ $10^4 - 10^8$ unknowns
- optimal control of **user–defined** physical quantities

$\Uparrow$

Numerics
+ Algorithms
+ Implementation
+ Hardware platforms

$\Downarrow$

DFG Benchmark ”Channel flow around cylinder”



- laminar 2D + 3D
- stationary ($Re = 20$)
- nonstationary ($Re = 100$)
- **lift** c_l, **drag** c_l **???**

Example I in 3D ('Poisson Problem'):

- $100 \times 100 \times 100$ grid points $\Longrightarrow$ **Problem size:** $N = 10^6$
- **Complexity of GE:** $N^{7/3} \approx 10^{14}$ FLOPs

100 sec on a 1 TFLOP/s computer

- **Complexity of (opt.) MG:** $1,000N \approx 10^9$ FLOPs

100 sec on a 10 (!!!) MFLOP/s computer

Example II in 3D ('Poisson Problem'):

- $1,000^3$ grid points $\Longrightarrow$ **Problem size:** $N = 10^9$
- **Complexity of GE:** $N^{7/3} \approx 10^{21}$ FLOPs

10^6 **sec on a 1 P(!!!)FLOP/s computer**

- **Complexity of (opt.) MG:** $1,000N \approx 10^{12}$ FLOPs

1,000 sec on a 1 (!!!) GFLOP/s computer

$\Downarrow$

Huge improvements by modern Numerics !!!

- Discretization techniques
- **Iterative solvers** (Krylov-space, **MG**)

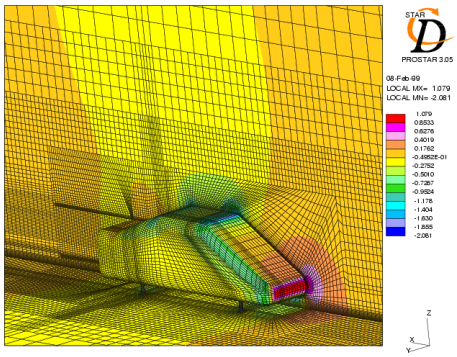
Typical results for 'realistic' applications:

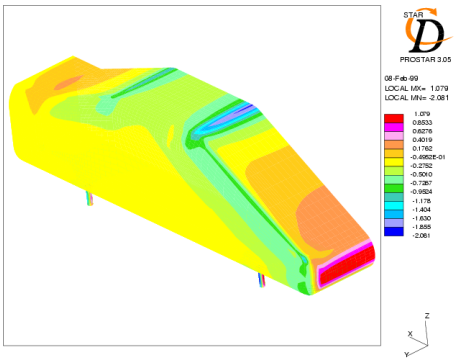
- **STAR-CD** ($k-\epsilon$), **500,000 cells** (by DaimlerChrysler)
- SGI Origin2000 (**6 processors**), **6.5 h** (CPU)

$\Uparrow$

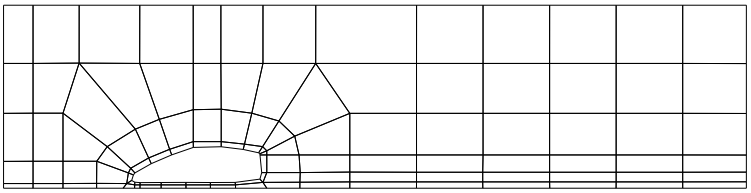
'Optimal' multigrid $\sim$ **1 sec/subproblem ???**

Quantity	**Experiment**	**Simulation**	**Difference**
Drag ('c_w')	0.165	0.317	**92 %**
Lift ('c_a')	-0.083	-0.127	**53 %**





- **MV** multiplication in **FEATFLOW** (F77 !!!)



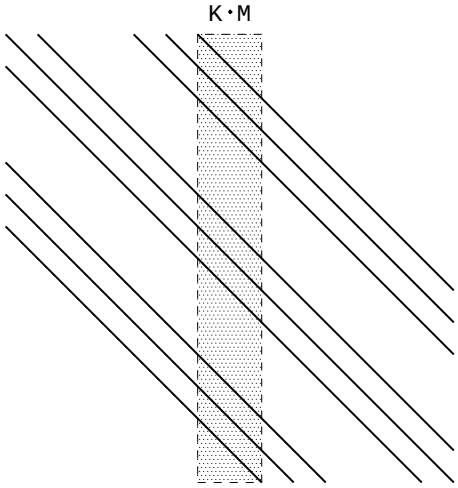
Computer	#Unknowns	**CM**	**TL**	**STO**
SUN E450 ($\sim$ 250 MFLOP/s)	13,688	22	20	**19**
	54,256	17	15	**13**
	216,032	16	14	**6**
	862,144	16	15	**4**

Sparse Matrix-Vector techniques

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

→ Standard technique in most **FEM** or **FV** codes
→ Storage of '**non-zero**' matrix elements only (*CSR, CSC, ...*)
→ Access via **index vectors, linked lists, pointers**, etc

Sparse Banded MV techniques



→ Typical for **FD** codes on '**tensorproduct meshes**'
→ Storage of '**non-zero**' elements in **bands/diagonals**
→ MV multiplication '**bandwise**' and '**window-oriented**'
→ Sparse Banded Blas (⇒ **FEAST !!!**)

Results on locally structured meshes

3D case	N	SPARSE	SBB-V	SBB-C	**MG-V**	**MG-C**
DEC 21264	17^3	164 (150)	446	765	342	500
(500 MHz)	33^3	64 (54)	240	768	233	474
'**DS20**'	65^3	**72 (24)**	**249**	**713**	**196**	**447**
IBM RS6K/597	17^3	86 (81)	179	480	171	368
(160 MHz)	33^3	81 (16)	170	393	152	300
'**SP2**'	65^3	**81 (8)**	**178**	**393**	**150**	**276**
INTEL PII	17^3	29 (28)	56	183	48	136
(400 MHz)	33^3	29 (24)	53	139	47	116
'**LOW COST**'	65^3	**29 (19)**	**54**	**125**	**45**	**101**

Sparse **MV techniques**

- MFLOP/s rates far away from '**Peak Performance**'
- Depending on **problem size + numbering**
- '**Old**' (IBM PWR2) partially **faster** then '**new**' (IBM P2SC)
- PC partially **faster** than processors in 'supercomputers' !!!

⇓

'Most adaptive codes should run on PC's'

Sparse Banded Blas **MV techniques**

- 'Supercomputing' power gets visible (up to **800 MFLOP/s**)
- **Warning:** Hard work !!! ($\rightarrow$ Sparse Banded Blas)
- However: **'Optimal'** (local) multigrid solver possible !!!

$\Uparrow$

1997 National Technology Roadmap for Semiconductors						
Year	1997	1999	2003	2006	2009	2012
Transistors/chip	**11M**	21M	76M	200M	520M	**1.4B**

1 Billion Transistors ($\times$ **100 !**)

10 Ghz clock rate ($\times$ **20 !**)

$\Downarrow$

1 PC 'faster' than **complete** *CRAY T3E today !*

Memory access ???

Processors are very 'sensible' supercomputers ???

FEAST project

Numerics and **implementation** techniques adapted to **hardware !!!**

$\Uparrow$

Precise knowledge of **processor** characteristics for different tasks in **MG** for **FEM** spaces

$\Uparrow$

(Collection of) **FEAST INDICES**

$\Downarrow$

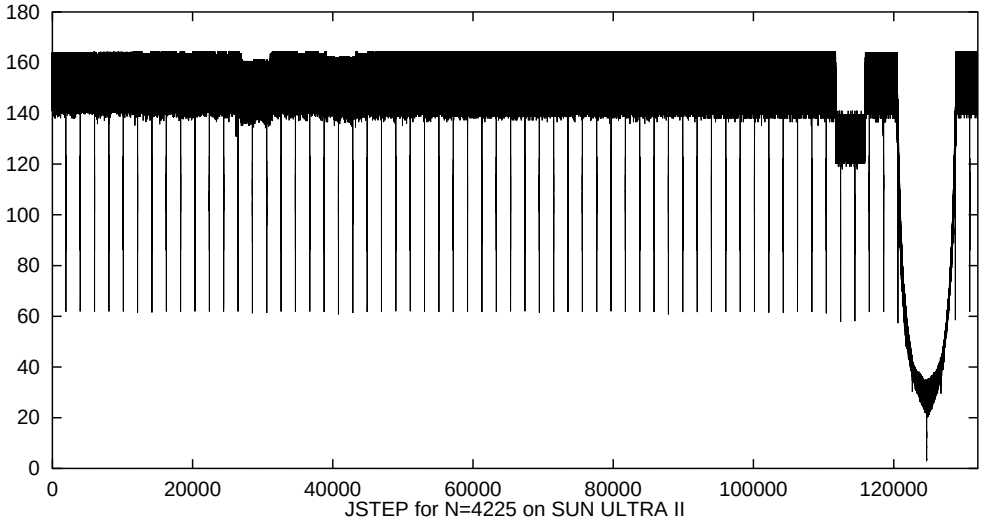
1) Get the optimum !!!

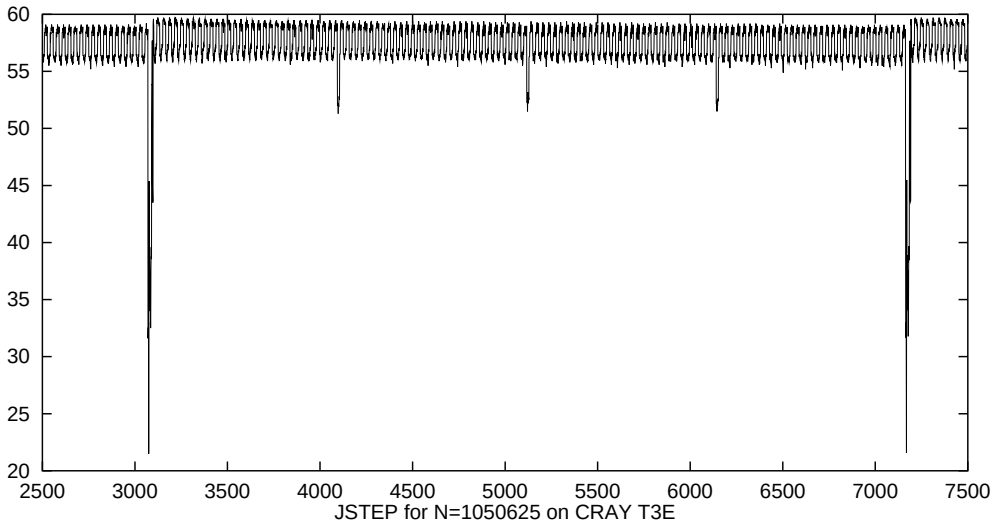
2) Prevent from dramatic performance losses !!!

Examples for performance losses:

Memory management

```
      DOUBLE PRECISION DX(8000000)
      DO 220 I = 1,ITER(J)
220     CALL DAXPY(N(J),ALPHA,DX(1),INCX,DX(1+N(J)+JSTEP),INCX)
```





⇓

Cache associativity ???

Compiler I

GNU (EGCS F77) vs. vendor-optimized compilers ???

DEC 21164 PC/LINUX, in Cache!!!

ULTRIX F77: INDEX **59** *- MV-C* **830** *MFLOP/s !!!*
EGCS F77: INDEX **37** *- MV-C* **196** *MFLOP/s !!!*

Compiler II

F90 vs. F77 compilers ???

SUN ULTRA II, in Cache!!!

F77: INDEX **49** *- MV-C* **404** *MFLOP/s !!!*
F90: INDEX **24** *- MV-C* **64** *MFLOP/s !!!*

Programming Languages

C vs. F77 compilers (SUN E3500) ???

$N = 1025^2$	F77	C
DAXPY-C	30	14
DAXPY-V	16	12
DAXPY-I	8	7

$N = 65^2$	F77	C
DAXPY-C	228	62
DAXPY-V	166	42
DAXPY-I	109	36

'My' conclusions:

Everybody can/must (!) test the own computational performance!

Most recent (FEM) codes are NOT slower on PC's than on 'High Performance' processors!

→ **However:** there is 'supercomputing power' available!!!

Compromise between 'reusable flexibility' and 'machine-dependent efficiency' is required!

Modern Numerics has to consider recent and future hardware trends!

→ **(macro-oriented) adaptivity** concepts !!!

→ ScaRC as improved **MG/DD** solver !!!

→ **MPSC:** Balance of **FLOPs** and **data access**

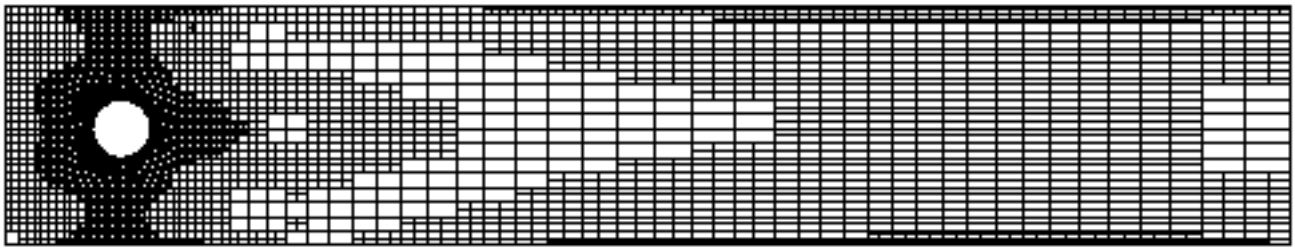
⇓

'Hardware-Oriented Numerics'

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

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

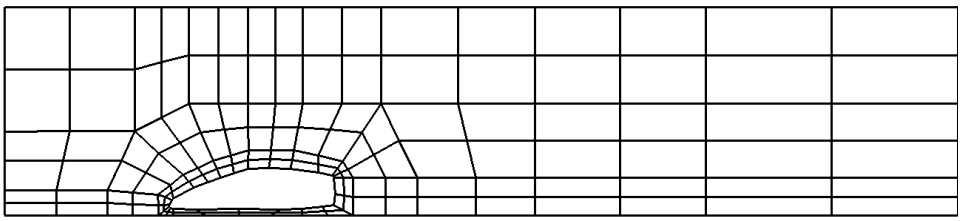
'Strong local solvers improve global convergence !'

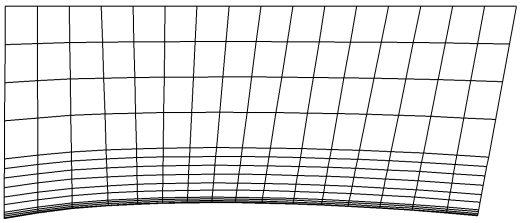
3) *Navier-Stokes solvers: Full Galerkin MPSC*

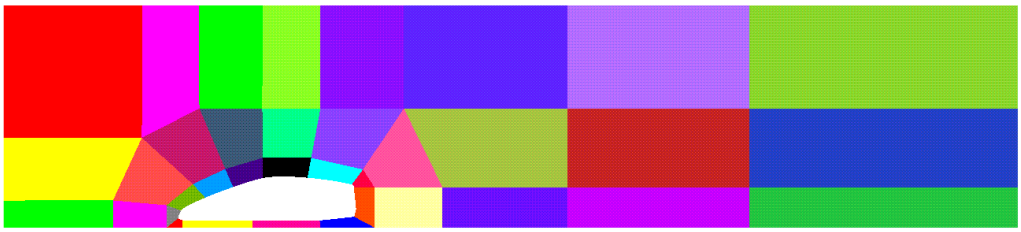
Develop Navier-Stokes solvers with more 'arithmetic' than 'memory intensive' operations

'Exploit local regularity !!!'

Example: Automotive Aerodynamics (DaimlerChrysler)







Parallel 'two level ScaRC' for the Pressure–Poisson problem

+

Adaptivity via modifying local tensorproduct-like meshes

	N_l	$ScaRC_g$
$h_{min} \approx 10^{-4}$ ('I')	32	0.03
	64	0.04
	128	0.04
$h_{min} \approx 10^{-8}$ ('II')	32	0.03
	64	0.04
	128	0.04

'I'	N_l	TGS_l	'II'	TGS_l
$AR \approx 1$	32	0.05	$AR \approx 1$	0.05
	64	0.06		0.06
	128	0.06		0.06
$AR \approx 10$	32	0.01	$AR \approx 10^5$	0.01
	64	0.01		0.01
	128	0.01		0.01

'Robustness + Accuracy'

MFLOP/s rates of Sparse Banded Blas

	NEQ	MV-V	MV-C	TGS-V	TGS-C	MGJ-V	MGJ-C	MGT-V	MGT-C
Cray T3E	65^2	82	391	59	143	96	391	84	281
(600 MHz)	257^2	72	293	49	118	75	194	70	171
'**MPP**'	1025^2	47	282	35	86	54	163	**51**	**142**
Cray T90	65^2	855	414	51	52	842	450	183	162
(440 MHz)	257^2	932	887	124	125	912	698	275	257
'**Vector**'	1025^2	939	1091	223	215	930	950	**422**	**418**
SUN E3500	65^2	298	520	67	70	262	382	158	188
(400 MHz)	257^2	271	588	64	69	252	408	153	190
'**Shared**'	1025^2	51	224	39	65	62	153	**57**	**117**
AMD K7	65^2	182	522	110	159	188	444	167	317
(700 MHz)	257^2	82	283	55	127	87	189	83	175
'**LINUX**'	1025^2	68	237	51	98	65	131	**63**	**125**

'E ciency'

High Performance FEM/FV Simulation:

(International) Cooperation for the design + realization of flexible software for education, research and industrial applications!