Space-Time Multigrid Techniques for Solving Time-Dependent Optimal Control Problems in CFD

Michael Köster

Michael Hinze, Stefan Turek

Institute for Applied Mathematics
TU Dortmund

DMV 2008,
15. – 19.09.2008

Distributed Control of nonstationary flow

Distributed Control for the nonstationary Navier-Stokes equation with tracking-type functional for a given z:

$$J(y,u) = \frac{1}{2}||y-z||_Q^2 + \frac{\alpha}{2}||u||_Q^2 + \frac{\gamma}{2}||y(T)-z(T)||_\Omega^2 \quad \rightarrow \quad \text{min!}$$

on $Q = \Omega \times [0,T]$ such that

$$\begin{aligned} y_t - \nu\Delta y + (y\nabla)y + \nabla p &= u \quad \text{in } Q \\ -\nabla \cdot y &= 0 \quad \text{in } Q \end{aligned} \qquad + \text{BC}$$

No constraints on the control $u \in Q$.

Distributed Control of nonstationary flow

Corresponding KKT-System:

$$
\begin{aligned}
y_t - \nu \Delta y + (y\nabla)y + \nabla p &= u && \text{in } Q \\
-\nabla \cdot y &= 0 && \text{in } Q \\
-\lambda_t - \nu \Delta \lambda - (y\nabla)\lambda + (\nabla y)\lambda + \nabla \xi &= (y - z) && \text{in } Q \\
-\nabla \cdot \lambda &= 0 && \text{in } Q \\
u &= -\tfrac{1}{\alpha}\lambda && \text{in } Q
\end{aligned}
$$

+ boundary conditions
+ initial condition
+ terminal condition $\lambda(T) = \gamma(y(T) - z(T))$ in Ω

Distributed Control of nonstationary flow

Corresponding KKT-System:

$$
\begin{aligned}
y_t - \nu \Delta y + (y\nabla)y + \nabla p &= -\tfrac{1}{\alpha}\lambda && \text{in } Q \\
-\nabla \cdot y &= 0 && \text{in } Q \\
\\
-\lambda_t - \nu \Delta \lambda - (y\nabla)\lambda + (\nabla y)\lambda + \nabla \xi &= (y - z) && \text{in } Q \\
-\nabla \cdot \lambda &= 0 && \text{in } Q
\end{aligned}
$$

Distributed Control of nonstationary flow

Corresponding KKT-System:

$$\begin{aligned} y_t - \nu\Delta y + (y\nabla)y + \nabla p + \tfrac{1}{\alpha}\lambda &= 0 \quad \text{in } Q \\ -\nabla \cdot y &= 0 \quad \text{in } Q \end{aligned}$$

$$\begin{aligned} -\lambda_t - \nu\Delta\lambda - (y\nabla)\lambda + (\nabla y)\lambda + \nabla\xi - y &= -z \quad \text{in } Q \\ -\nabla \cdot \lambda &= 0 \quad \text{in } Q \end{aligned}$$

Distributed Control of nonstationary flow

Corresponding KKT-System:

$$
\begin{aligned}
y_t + N(y)y + \nabla p + \tfrac{1}{\alpha}\lambda &= 0 \quad \text{in } Q \\
-\nabla \cdot y &= 0 \quad \text{in } Q
\end{aligned}
$$

$$
\begin{aligned}
-\lambda_t + N^*(y)\lambda + \nabla \xi - y &= -z \quad \text{in } Q \\
-\nabla \cdot \lambda &= 0 \quad \text{in } Q
\end{aligned}
$$

Distributed Control of nonstationary flow

Corresponding KKT-System:

$$\begin{pmatrix} y_t \\ -\lambda_t \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} N(y) & \frac{1}{\alpha} & \nabla & \\ -I & N^*(y) & & \nabla \\ -\nabla\cdot & & 0 & \\ & -\nabla\cdot & & 0 \end{pmatrix} \begin{pmatrix} y \\ \lambda \\ p \\ \xi \end{pmatrix} = \begin{pmatrix} 0 \\ -z \\ 0 \\ 0 \end{pmatrix}$$

$$\Rightarrow \quad \begin{pmatrix} w_t \\ 0 \end{pmatrix} + \begin{pmatrix} N(w) & \nabla \\ -\nabla\cdot & 0 \end{pmatrix} \begin{pmatrix} w \\ q \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}$$

$$\Rightarrow \quad \text{generalised Navier–Stokes equation}$$

Space–time discretisation

- Time discretisation:
  Backward Euler (later: 2nd order Crank-Nicolson, FS-θ)

$$\begin{aligned} \frac{y_i - y_{i-1}}{\Delta t} + N(y_i)y_i + \nabla p_i + \frac{1}{\alpha}\lambda_i &= \dots \\ \frac{\lambda_i - \lambda_{i+1}}{\Delta t} + N^*(y_i)\lambda_i + \nabla \xi_i - y_i &= \dots \end{aligned}$$

- Space discretisation:
  LBB-stable Finite Elements ($\tilde{Q}_1/Q_0$, later Q_2/P_1)
  on general 2D meshes (later: 3D)

Space–time discretisation

$\Rightarrow$ Space–time system

$$A(x)x = b$$

Here (for n timesteps, $x_k = x(t_k)$):

$$x = (\underbrace{y_0, \lambda_0, p_0, \xi_0}_{x_0 = x(t_0)}, \ \ldots, \ \underbrace{y_n, \lambda_n, p_n, \xi_n}_{x_n = x(t_n)})$$

Space–time discretisation

$A(x)$ has a very special structure! E.g. for 2 timesteps:

$$A(x)x = \left(\begin{array}{cc|cc|cc} M & -B & & & & \\ -M\ \ NST^* & -B & -\frac{M}{\Delta t} & & & \\ \hline -B^T & 0 & & & & \\ -B^T & 0 & & & & \\ \hline -\frac{M}{\Delta t} & & NST\ \ \frac{M}{\alpha} & -B & & \\ & & -M\ \ NST^* & -B & -\frac{M}{\Delta t} & \\ \hline & & -B^T & 0 & & \\ & & -B^T & 0 & & \\ \hline & & -\frac{M}{\Delta t} & & NST\ \ \frac{M}{\alpha} & -B \\ & & & & -\frac{\gamma M}{\Delta t}\ \ NST^* & -B \\ \hline & & & & -B^T & 0 \\ & & & & -B^T & 0 \end{array}\right) \left(\begin{array}{c} y_0 \\ \lambda_0 \\ p_0 \\ \xi_0 \\ \hline y_1 \\ \lambda_1 \\ p_1 \\ \xi_1 \\ \hline y_2 \\ \lambda_2 \\ p_2 \\ \xi_2 \end{array}\right)$$

$\longrightarrow$ Sparse, (block) tridiagonal system

Design of a One-Shot solver

- Nonlinearity: Newton method for quadratic convergence

$$x^{i+1} = x^i + A'^{-1}(x^i)(b - A(x^i)x^i)$$

- Linear subproblems: space-time Multigrid solver

  $\longrightarrow$ Convergence rates independent of refinement level of the space-time mesh

Space–time multigrid ingredients

Essential multigrid components:

- Mesh hierarchy for a space-time cylinder $Q = \Omega \times [0, T]$:
  Choose arbitrary space-time coarse mesh and refine!
- Prolongation/Restriction in space + time.
  Combination of FE in space & FD in time.
- An efficient smoother! E.g. with $\tilde{A} := A'(x^j)$:

$$v^{j+1} = v^j + \omega P^{-1}(b - \tilde{A}v^j), \qquad j = 1, ..., \mathrm{NSM}$$

What to use as preconditioner P?

Space–time multigrid ingredients

Essential multigrid components:

- Mesh hierarchy for a space-time cylinder $Q = \Omega \times [0, T]$:
  Choose arbitrary space-time coarse mesh and refine!
- Prolongation/Restriction in space + time.
  Combination of FE in space & FD in time.
- An efficient smoother! E.g. with $\tilde{A} := A'(x^j)$:

$$v^{j+1} = v^j + \omega P^{-1}(b - \tilde{A}v^j), \qquad j = 1, ..., \mathrm{NSM}$$

What to use as preconditioner P?

Space–time discretisation and preconditioner

$\tilde{A}$ in compressed form (omitting B and B^T here):

$$\left(\begin{array}{cc|cc|cc|c} M & & & & & & \\ -M & NST^* & & -\frac{M}{\Delta t} & & & \\ \hline -\frac{M}{\Delta t} & & NST & \frac{1}{\alpha}M & & & \\ & & -M & NST^* & & -\frac{M}{\Delta t} & \\ \hline & & -\frac{M}{\Delta t} & & NST & \frac{1}{\alpha}M & \\ & & & & -M & NST^* & -\frac{M}{\Delta t} \\ \hline & & & & & \dots & \dots \end{array}\right)$$

$\Rightarrow$ Block-Jacobi preconditioner:

$$P := P_{Jac} := \left(\begin{array}{cc|cc|cc|c} M & & & & & & \\ -M & NST^* & & & & & \\ \hline & & NST & \frac{1}{\alpha}M & & & \\ & & -M & NST^* & & & \\ \hline & & & & NST & \frac{1}{\alpha}M & \\ & & & & -M & NST^* & \\ \hline & & & & & & \dots \end{array}\right)$$

Space–time discretisation and preconditioner

$\tilde{A}$ in compressed form (omitting B and B^T here):

$$\left(\begin{array}{cc|cc|cc|c} M & & & & & & \\ -M & NST^* & & -\frac{M}{\Delta t} & & & \\ \hline -\frac{M}{\Delta t} & & NST & \frac{1}{\alpha}M & & & \\ & & -M & NST^* & & -\frac{M}{\Delta t} & \\ \hline & & -\frac{M}{\Delta t} & & NST & \frac{1}{\alpha}M & \\ & & & & -M & NST^* & -\frac{M}{\Delta t} \\ \hline & & & & & \dots & \dots \end{array}\right)$$

$\Rightarrow$ Block-Jacobi preconditioner:

$$P := P_{Jac} := \left(\begin{array}{cc|cc|cc|c} M & & & & & & \\ -M & NST^* & & & & & \\ \hline & & NST & \frac{1}{\alpha}M & & & \\ & & -M & NST^* & & & \\ \hline & & & & NST & \frac{1}{\alpha}M & \\ & & & & -M & NST^* & \\ \hline & & & & & & \dots \end{array}\right)$$

Space–time discretisation and preconditioner

$\tilde{A}$ in compressed form (omitting B and B^T here):

$$\left(\begin{array}{cc|cc|cc|c}
M & & & & & & \\
-M & NST^* & & -\frac{M}{\Delta t} & & & \\
\hline
-\frac{M}{\Delta t} & & NST & \frac{1}{\alpha}M & & & \\
 & & -M & NST^* & & -\frac{M}{\Delta t} & \\
\hline
 & & -\frac{M}{\Delta t} & & NST & \frac{1}{\alpha}M & \\
 & & & & -M & NST^* & -\frac{M}{\Delta t} \\
\hline
 & & & & & \dots & \dots
\end{array}\right)$$

$\Rightarrow$ Symmetric Gauß-Seidel-like preconditioner $P := P_{SGS}$.

In every timestep:

$$\begin{pmatrix} -\frac{M}{\Delta t} & 0 \\ 0 & 0 \end{pmatrix} v_{i-1} + \begin{pmatrix} NST & \frac{1}{\alpha}M \\ -M & NST^* \end{pmatrix} v_i + \begin{pmatrix} 0 & 0 \\ 0 & -\frac{M}{\Delta t} \end{pmatrix} v_{i+1} = \dots$$

$\longrightarrow v_i$

Space–time preconditioner

Essential: To use $P^{-1}_{Jac/SGS}$, apply for every timestep:

$$\left(\begin{array}{cc|cc} NST & \frac{1}{\alpha}M & B & \\ -M & NST^* & & B \\ \hline B^T & & 0 & \\ & B^T & & 0 \end{array}\right)^{-1}$$

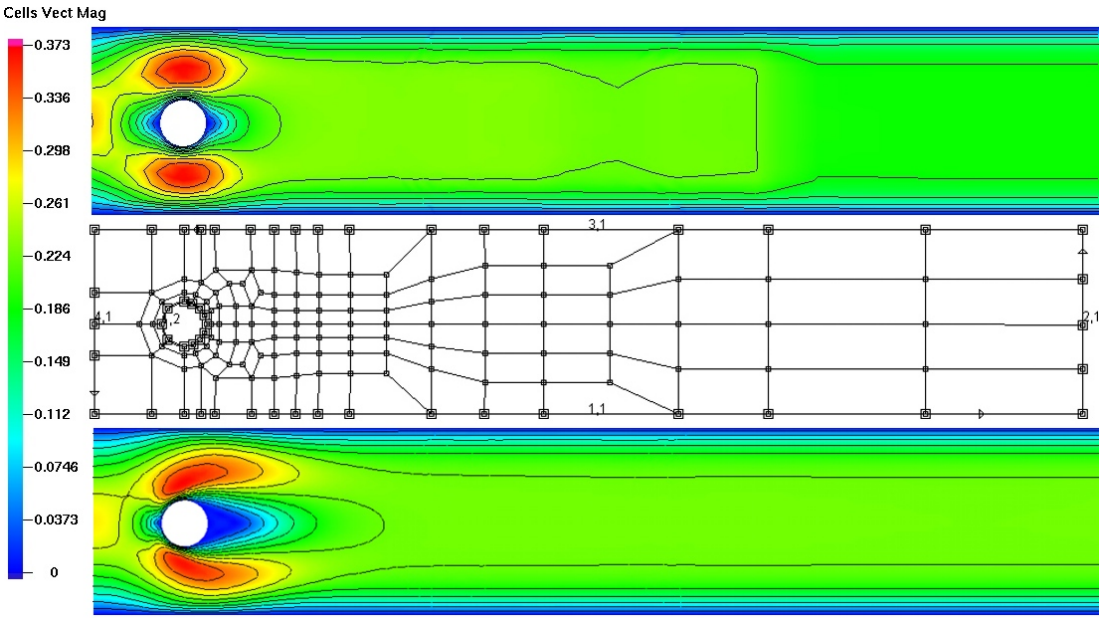
Generalised Navier–Stokes $\Rightarrow$ use efficient CFD techniques!

- Multigrid in space
- Smoother: mod. LPSC (VANKA-like)
  $\rightarrow$ Primal and dual coupled!

} **FeatFlow!**

Numerical example: Flow-around-cylinder

- Target flow z: Stokes flow, $t \in [0, 1]$, starting from rest



Stokes at $t = 1$

Mesh

uncontrolled Nav.St.

- Optimal control problem: Navier–Stokes, $Re = 20$
- Coarse mesh: 1404 DOF's in space, 5 timesteps, $\Delta t := 0.2$
  $\Rightarrow$ 8 424 DOF's, $\times 8$ per level

Numerical example: Flow-around-cylinder

Convergence of the Newton solver

		Simulation				Optimisation	
Δt	Space-Lv.	#NL	#MG	⌀#NL	⌀#MG	#NL	#MG
1/20	3	63	312	3	16	4	24
1/40	4	123	709	3	18	4	14
1/80	5	246	1589	3	20	4	13

- Space-time MG gained 2 digits per step
- Space-preconditioner gained 2 digits per step

Numerical example: Flow-around-cylinder

Convergence of the Newton solver

Δt	Space-Lv.	Time sim.	Time opt.	Time opt / Time sim.
1/20	3	27	917	34.0
1/40	4	209	4346	20.7
1/80	5	2227	55174	24.8

Optimisation is $C \approx 20 - 30\times$ more expensive than simulation
$\rightarrow$ independent of the refinement level

$\rightarrow$ Aim: Factor 10-15 due to ”code optimisation”!

Numerical example: Flow-around-cylinder

Convergence of the Newton solver

Δt	Space-Lv.	Time sim.	Time opt.	Time opt / Time sim.
1/20	3	27	917	34.0
1/40	4	209	4346	20.7
1/80	5	2227	55174	24.8

Optimisation is $C \approx 20 - 30\times$ more expensive than simulation
$\rightarrow$ independent of the refinement level

$\rightarrow$ Aim: Factor 10-15 due to ”code optimisation”!

Numerical example: Flow-around-cylinder

Convergence of the solver for different α and γ

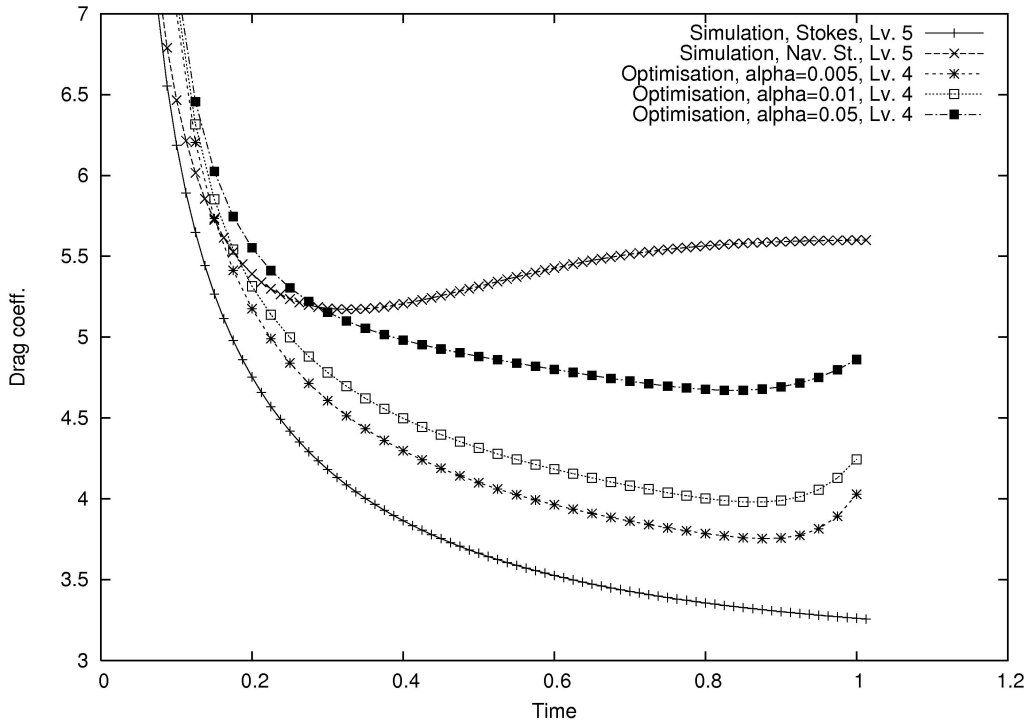
α	γ	#NL	#MG
0.05	0	5	16
	0.1	5	16
	0.5	5	16
0.01	0	5	15
	0.1	5	15
	0.5	5	37
0.005	0	5	22
	0.1	5	16
	0.5	5	72

$\Delta t = 1/40$,
Space-level 4

$\Rightarrow$ Smaller α + larger $\gamma \rightarrow$ worse convergence rates

Numerical example: Flow-around-cylinder

Solution quality: Analysis of the drag

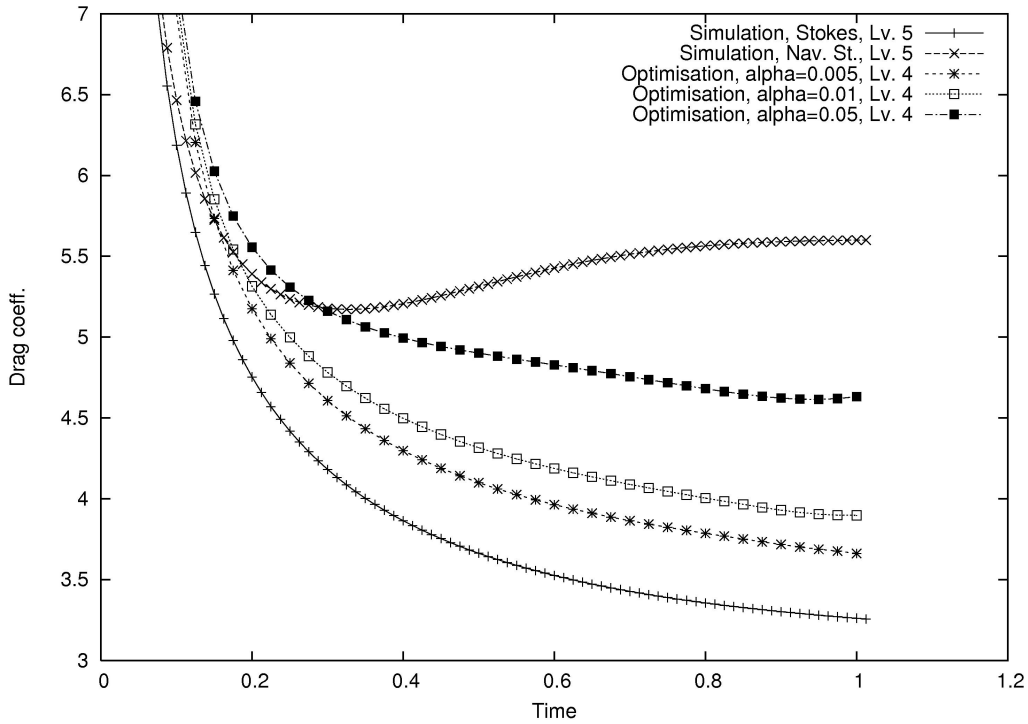


$\alpha = 0.05,\ 0.01,\ 0.005,$
$\gamma = 0.00$

$$C_D = \frac{2}{(\frac{2}{3}U_{\max})^2 \cdot 0.1} \int_\Gamma (\sigma n) n_x d\Gamma$$

Numerical example: Flow-around-cylinder

Solution quality: Analysis of the drag

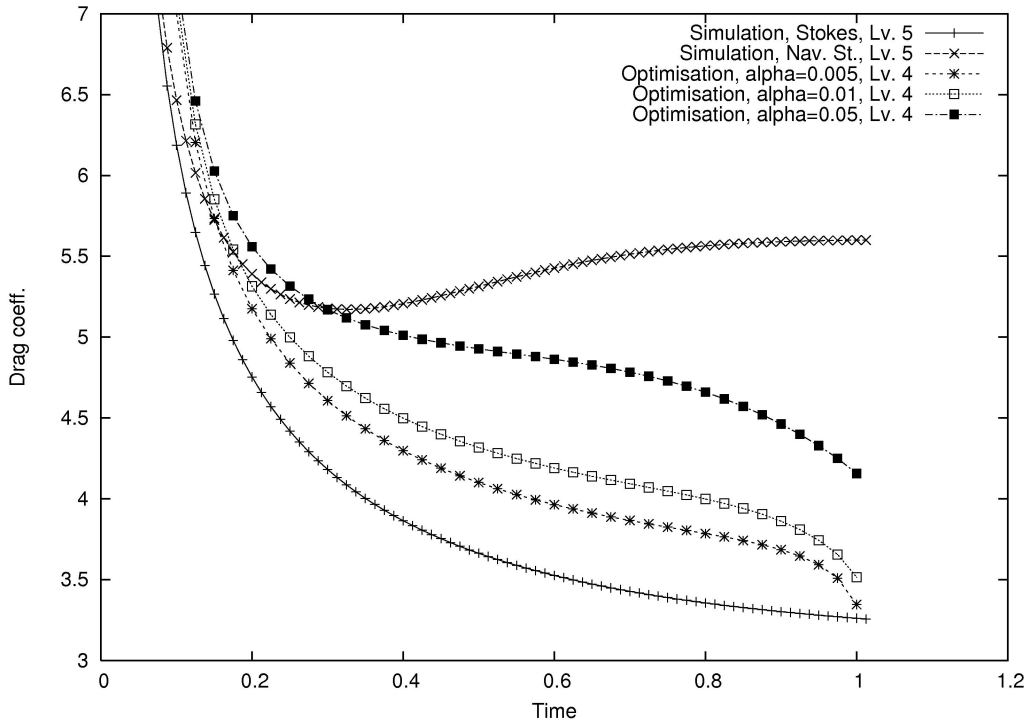


$\alpha = 0.05,\ 0.01,\ 0.005,$
$\gamma = 0.1$

$$C_D = \frac{2}{(\frac{2}{3}U_{\max})^2 \cdot 0.1} \int_\Gamma (\sigma n) n_x d\Gamma$$

Numerical example: Flow-around-cylinder

Solution quality: Analysis of the drag



$\alpha = 0.05,\ 0.01,\ 0.005,$
$\gamma = 0.5$

$$C_D = \frac{2}{(\frac{2}{3}U_{\max})^2 \cdot 0.1} \int_\Gamma (\sigma n) n_x d\Gamma$$

Outlook

Further steps:

- Q_2/P_1 + Crank-Nicolson
  $\longrightarrow$ higher accuracy, larger timesteps, fewer unknowns
- FEM-stabilisation (EO-FEM/interior penalty)
  $\longrightarrow$ higher RE numbers
- improved smoothers for the space-time problem
- incorporate constraints
- include algorithms for memory management

Space–time discretisation

Structure of $A(x)$:

$$
\left(\begin{array}{cc|cc|cc|cc}
M & & -B & & & & & \\
-M & \frac{M}{\Delta t}+N^* & & -B & -\frac{M}{\Delta t} & & & \\
\hline
-B^T & & 0 & & & & & \\
 & -B^T & & 0 & & & & \\
\hline
-\frac{M}{\Delta t} & & \frac{M}{\Delta t}+N & \frac{1}{\alpha}M & -B & & & \\
 & & -M & \frac{M}{\Delta t}+N^* & & -B & -\frac{M}{\Delta t} & \\
\hline
 & & -B^T & & 0 & & & \\
 & & & -B^T & & 0 & & \\
\hline
 & & -\frac{M}{\Delta t} & & \frac{M}{\Delta t}+N & \frac{1}{\alpha}M & -B & \\
 & & & & -\frac{\gamma}{\Delta t}M & \frac{M}{\Delta t}+N^* & & -B \\
\hline
 & & & & -B^T & & 0 & \\
 & & & & & -B^T & & 0
\end{array}\right)
$$

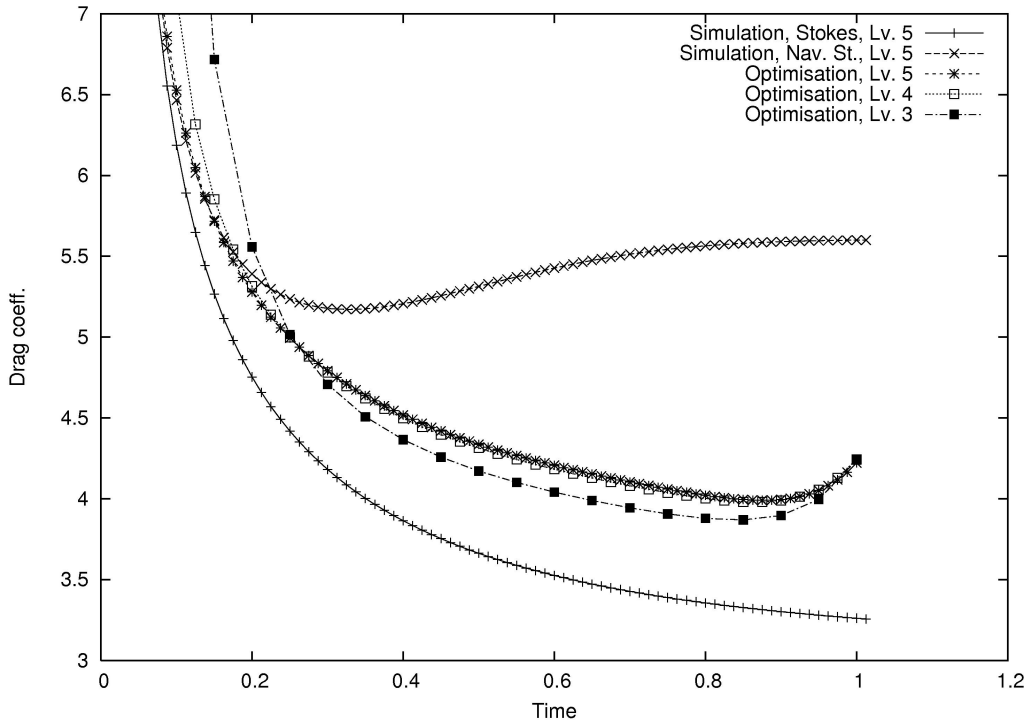
Space–time discretisation

Structure of $A'(x)$:

$$
\left[\begin{array}{cc|cc|cc|cc}
M & & -B & & & & & \\
-M+R_0 & \frac{M}{\Delta t}+N^* & & -B & -\frac{M}{\Delta t} & & & \\
\hline
-B^T & & 0 & & & & & \\
& -B^T & & 0 & & & & \\
\hline
-\frac{M}{\Delta t} & & \frac{M}{\Delta t}+N^{**} & \frac{1}{\alpha}M & -B & & & \\
& & -M+R_1 & \frac{M}{\Delta t}+N^* & & -B & -\frac{M}{\Delta t} & \\
\hline
& & -B^T & & 0 & & & \\
& & & -B^T & & 0 & & \\
\hline
& & -\frac{M}{\Delta t} & & \frac{M}{\Delta t}+N^{**} & \frac{1}{\alpha}M & -B & \\
& & & & -\frac{\gamma}{\Delta t}M+R_2 & \frac{M}{\Delta t}+N^* & & -B \\
\hline
& & & & -B^T & & 0 & \\
& & & & & -B^T & & 0
\end{array}\right]
$$

Numerical example: Flow-around-cylinder

Solution quality: Analysis of the drag

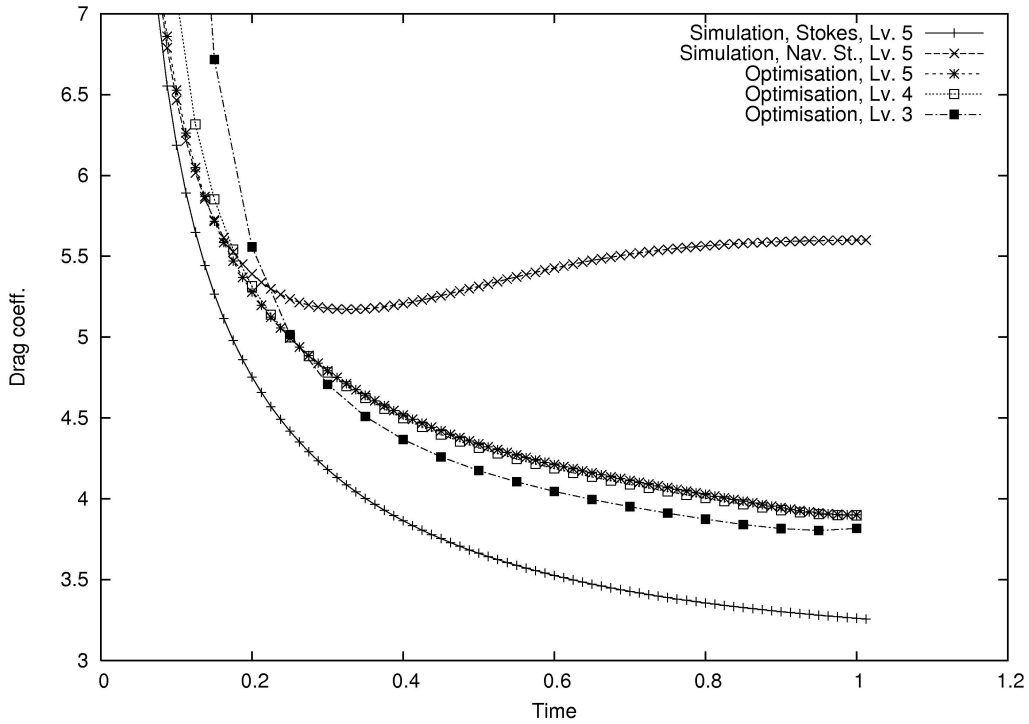


$$\alpha = 0.01,$$
$$\gamma = 0.00$$

$$C_D = \frac{2}{(\frac{2}{3}U_{\max})^2 \cdot 0.1} \int_\Gamma (\sigma n) n_x d\Gamma$$

Numerical example: Flow-around-cylinder

Solution quality: Analysis of the drag

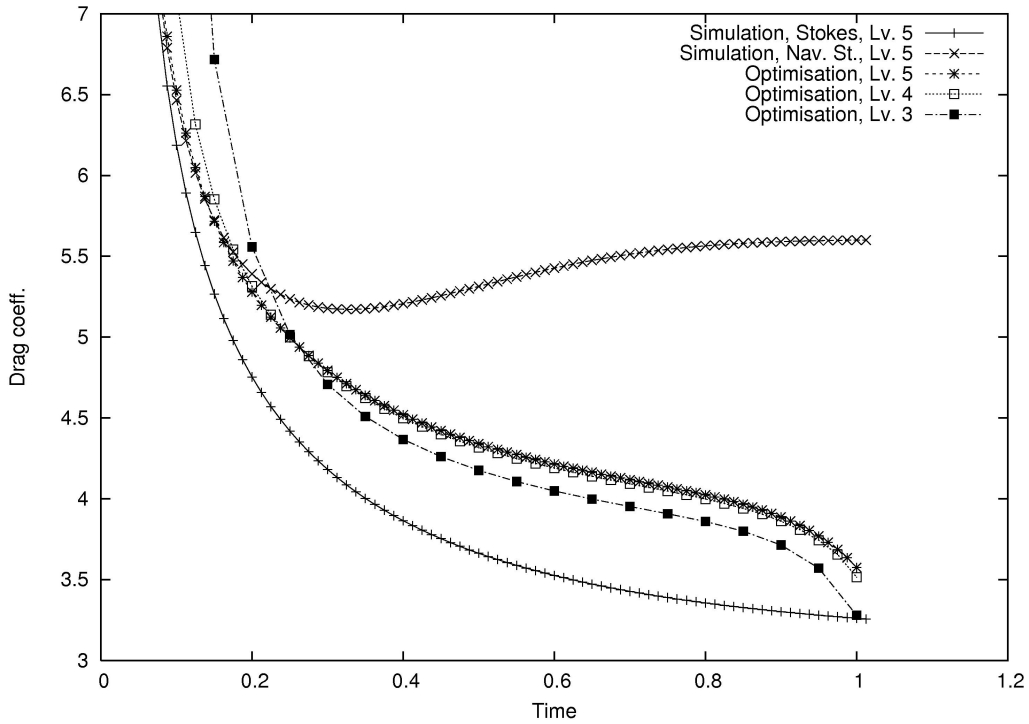


$$\alpha = 0.01,$$
$$\gamma = 0.1$$

$$C_D = \frac{2}{(\frac{2}{3}U_{\max})^2 \cdot 0.1} \int_\Gamma (\sigma n) n_x d\Gamma$$

Numerical example: Flow-around-cylinder

Solution quality: Analysis of the drag



$$\alpha = 0.01,$$
$$\gamma = 0.5$$

$$C_D = \frac{2}{(\frac{2}{3}U_{\max})^2 \cdot 0.1} \int_\Gamma (\sigma n) n_x d\Gamma$$

Numerical example: Flow-around-cylinder

Convergence of the solver for different ν

		$\nu = 1/1000$		$\nu = 1/250$	
Δt	Space-Lv.	#NL	#MG	#NL	#MG
1/10	2	4	36	4	13
1/20	3	4	24	3	8
1/40	4	4	14	3	7
1/80	5	4	13	3	8

$\Rightarrow$ Stabilisation and stronger smoothers necessary for higher RE numbers

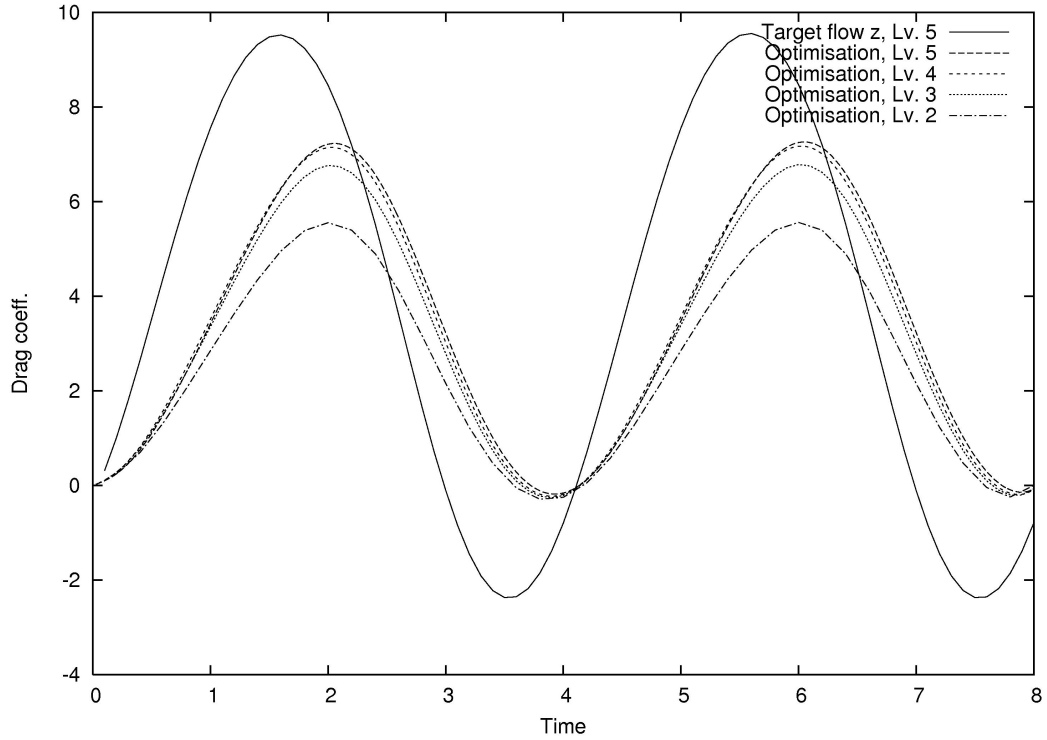
Numerical example:

Flow-around-cylinder with pulsating BC

- Domain: Flow-around-cylinder (like above), $T = 8$.
- Coarse mesh:
  1404 DOF's in space, 20 timesteps with $\Delta t = 0.4$.
- Target flow z: Parabolic inflow with max. velocity

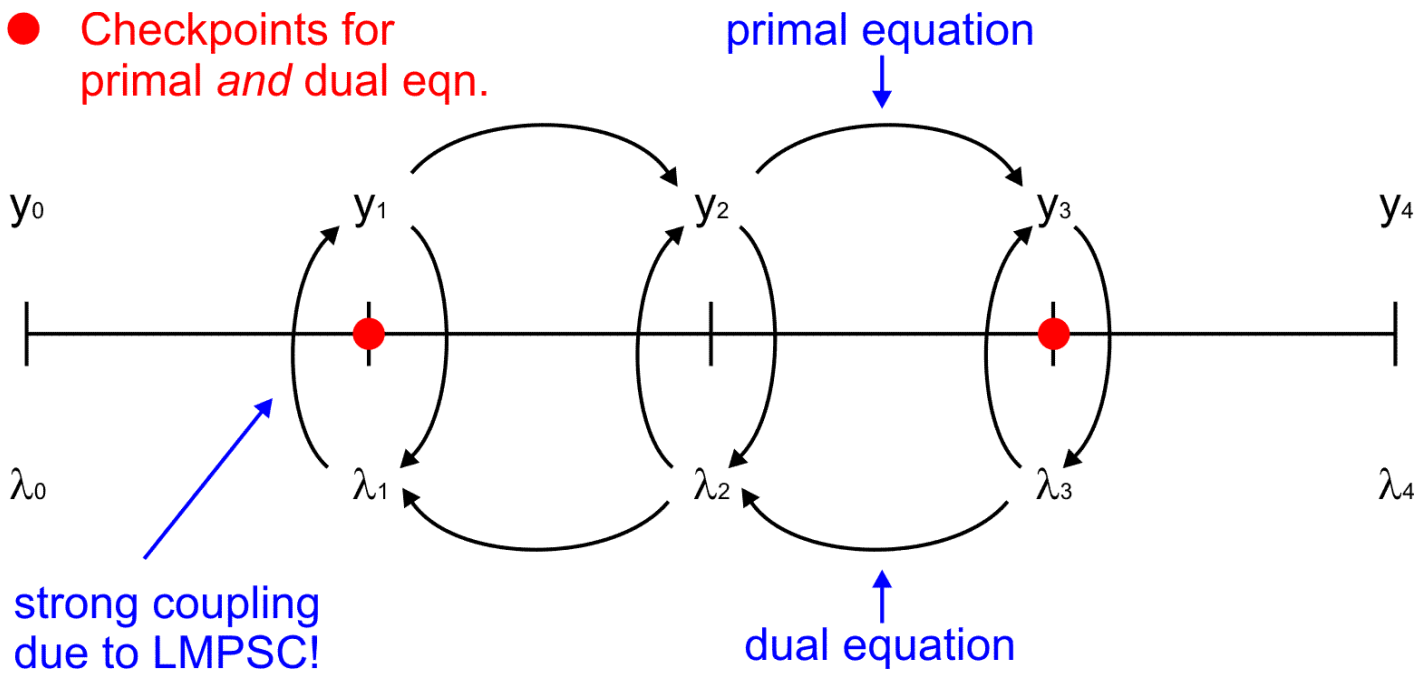
$$U_{\max}(t) := 0.15 \cdot \left(1 + \sin(\frac{t+3}{2}\pi)\right)/2$$

- Optimisation: Natural boundary conditions at inflow



$$C_D = \frac{2}{(\frac{2}{3}0.15)^2 \cdot 0.1} \int_\Gamma (\sigma n) n_x d\Gamma$$

Checkpointing in the One-shot approach



- Checkpoints $\longrightarrow$ nonlinear subproblems of the same kind.
- High computational costs necessary for recomputation $\longrightarrow$ due to strong coupling by LMPSC!