

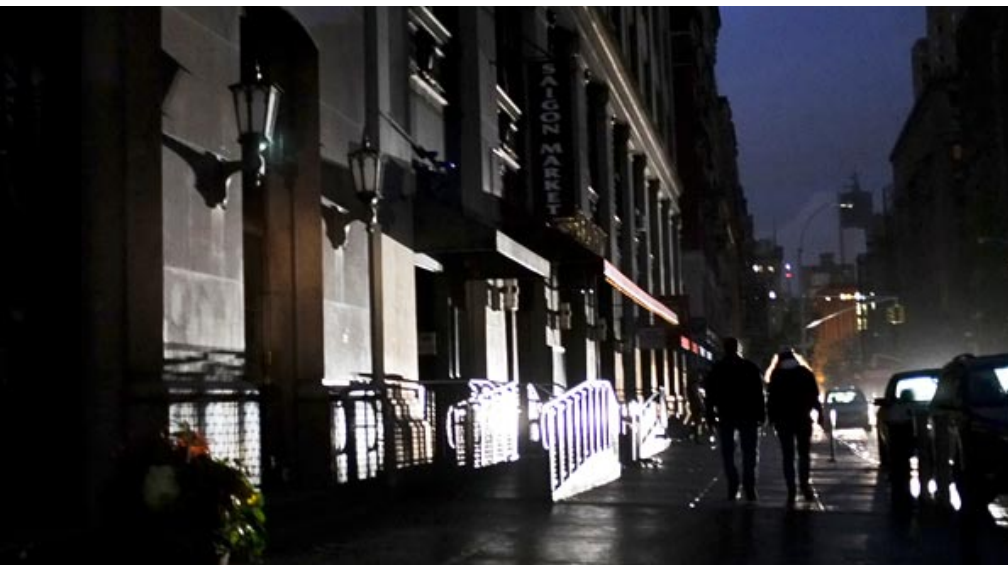
Advanced computation in fluid mechanics

Introduction

Fluid dynamics around us?



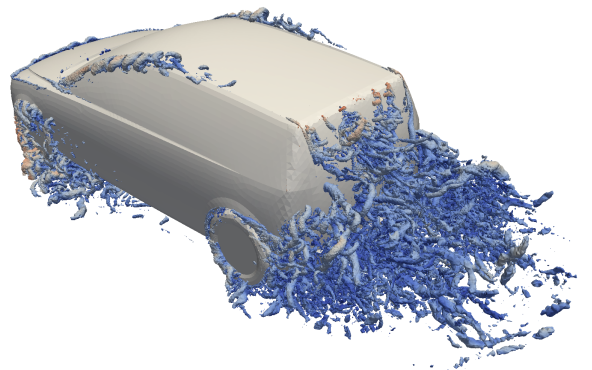
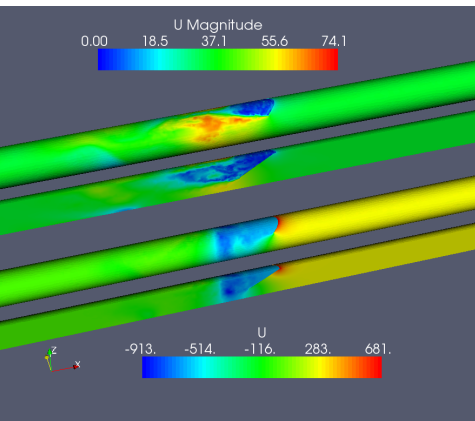
Vädrets vinnare. Tack vare avancerade matematiska formler blir träffsäkerheten i meteorologernas varningar bättre för varje år. Här uppdaterar James Franklin, på National Hurricane Center, amerikanerna om orkanen Sandy i en tv-sändning.



Vehicles, airplanes

Goal: simulation and virtual prototyping

- Aerodynamic forces
- Vibrations and wing flutter
- Noise generation: landing gear, exhaust system
- **Exterior/interior turbulent air flow, fluid-structure interaction, aero-acoustics**



Americas cup

Mike Drummond, BMW Oracle's
Racing Design Director:

"The entire aerodynamic project
has been based on numerical
simulations without wind tunnel."



Navier-Stokes equations

$$\rho(\dot{u} + (u \cdot \nabla)u) = \nabla \cdot \sigma + \rho f, \quad (x, t) \in \Omega \times I$$

$$\nabla \cdot u = 0, \quad (x, t) \in \Omega \times I$$

$\Omega \subset \mathbb{R}^3$, $I = (0, T)$, $\Gamma = \partial\Omega$, **velocity** u , **density** ρ , **force** f

Cauchy stress tensor $\sigma = \sigma_{ij}$ with $i, j = 1, 2, 3$

$$\sigma_{ij} = \frac{1}{3}\sigma_{kk}\delta_{ij} + \left(\sigma_{ij} - \frac{1}{3}\sigma_{kk}\delta_{ij}\right) \equiv -p\delta_{ij} + s_{ij}$$

Navier-Stokes equations

Newtonian fluid: $\sigma_{ij} = -p\delta_{ij} + 2\mu\epsilon_{ij} + \lambda\nabla \cdot u\delta_{ij}$

μ - dynamic viscosity, λ - second coeff. of viscosity

Strain rate tensor: $\epsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)$

$\nabla \cdot u = 0 \quad \Rightarrow \quad$ Stokes law: $\sigma_{ij} = -p\delta_{ij} + 2\mu\epsilon_{ij}$

$2\nabla \cdot \epsilon(u) = \Delta u + \nabla(\nabla \cdot u) \Rightarrow$ Navier-Stokes equations

$$\dot{u} + (u \cdot \nabla)u - \nu\Delta u + \nabla p = f, \quad (x, t) \in \Omega \times I$$

$$\nabla \cdot u = 0, \quad (x, t) \in \Omega \times I$$

Kinematic viscosity $\nu = \mu/\rho$, kinematic pressure p

Reynolds number

$$u = u^*U, \quad x = x^*L, \quad t = t^*T, \quad p = p^*P$$
$$\frac{\partial}{\partial t} = \frac{1}{T} \frac{\partial}{\partial t^*}, \quad \nabla = \frac{1}{L} \nabla^*, \quad \Delta = \frac{1}{L^2} \Delta^* \quad \Rightarrow$$

$$\frac{U}{T} \frac{\partial u^*}{\partial t^*} + \frac{U^2}{L} (u^* \cdot \nabla^*) u^* - \frac{\nu U}{L^2} \Delta^* u^* + \frac{P}{L} \nabla^* p^* = 0$$

$Re = UL/\nu$ measures convective vs viscous effects:

For $Re > 1$ choose convective time scale $T = L/U$

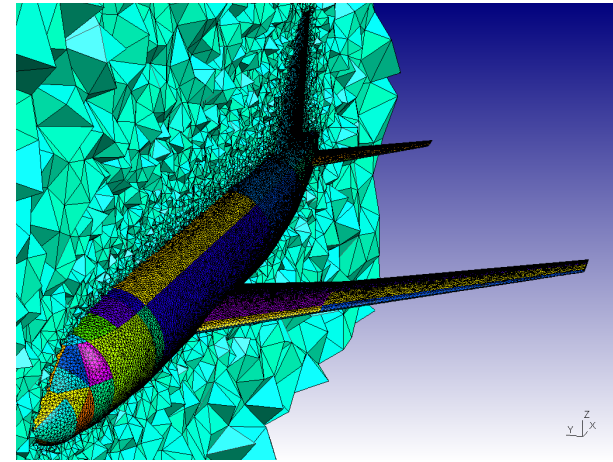
$$\dot{u} + (u \cdot \nabla)u - Re^{-1} \Delta u + \nabla p_c = 0 \quad (Re \rightarrow \infty \Rightarrow \text{Euler})$$

For $Re < 1$ choose viscous time scale $T = L^2/\nu$

$$\dot{u} + Re(u \cdot \nabla)u - \Delta u + \nabla p_v = 0 \quad (Re \rightarrow 0 \Rightarrow \text{Stokes})$$

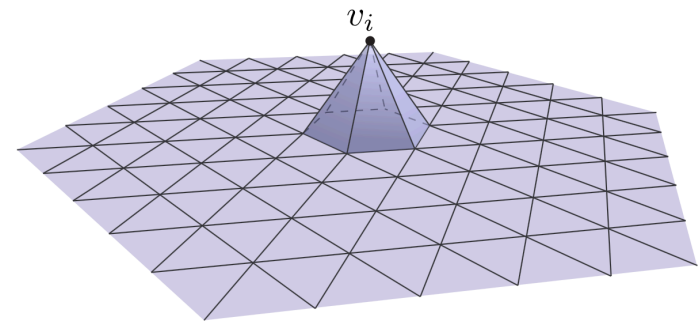
Solving the NS equations

- Finite Element Method (FEM)
- Finite Volume Method (FVM)
- Finite Difference Method (FDM)
- Mesh free/particle methods

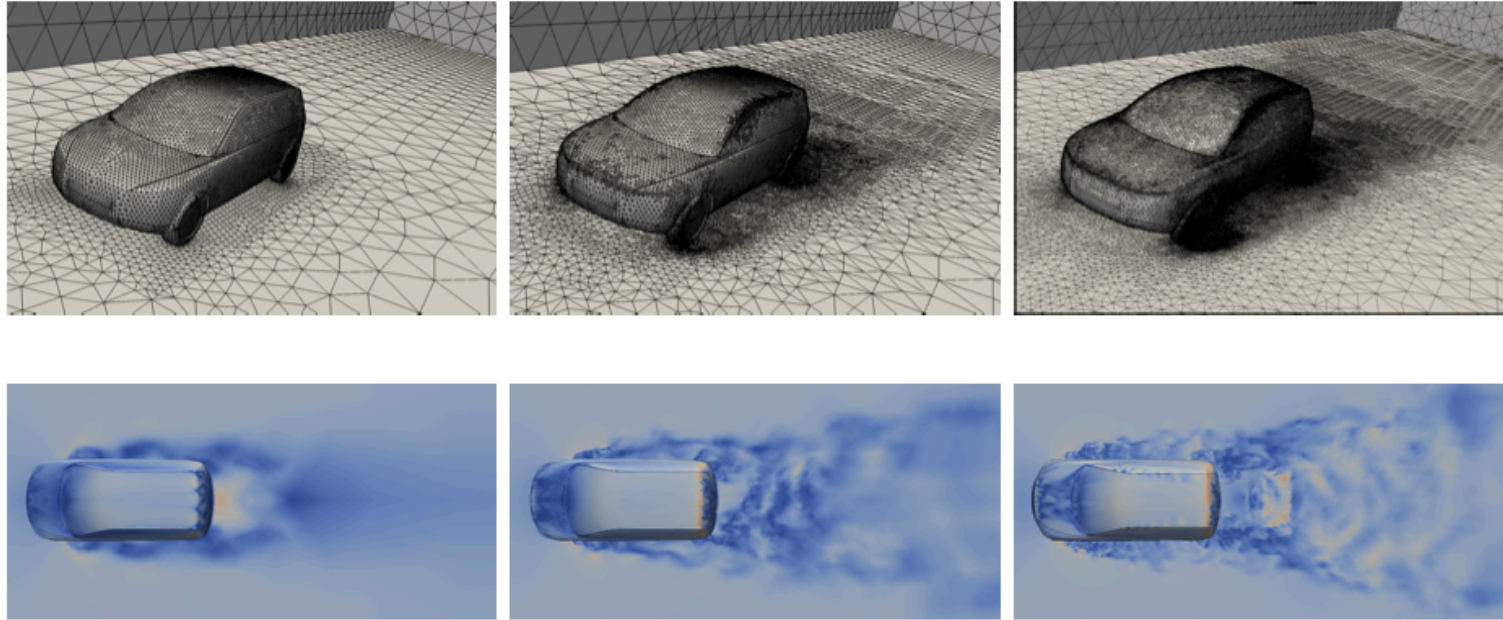


[Onera]

- Cost scales with number of degrees of freedom (#dofs)
- How to choose optimal mesh (minimize #dofs)



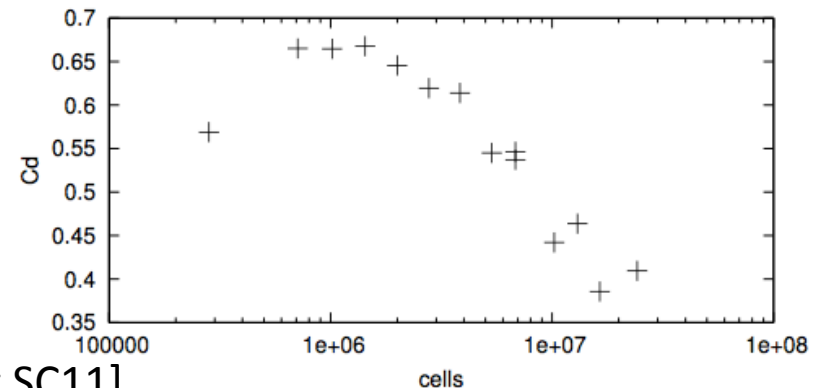
Adaptive finite element methods



[Geometric model from of Volvo Cars]

Aerodynamic drag

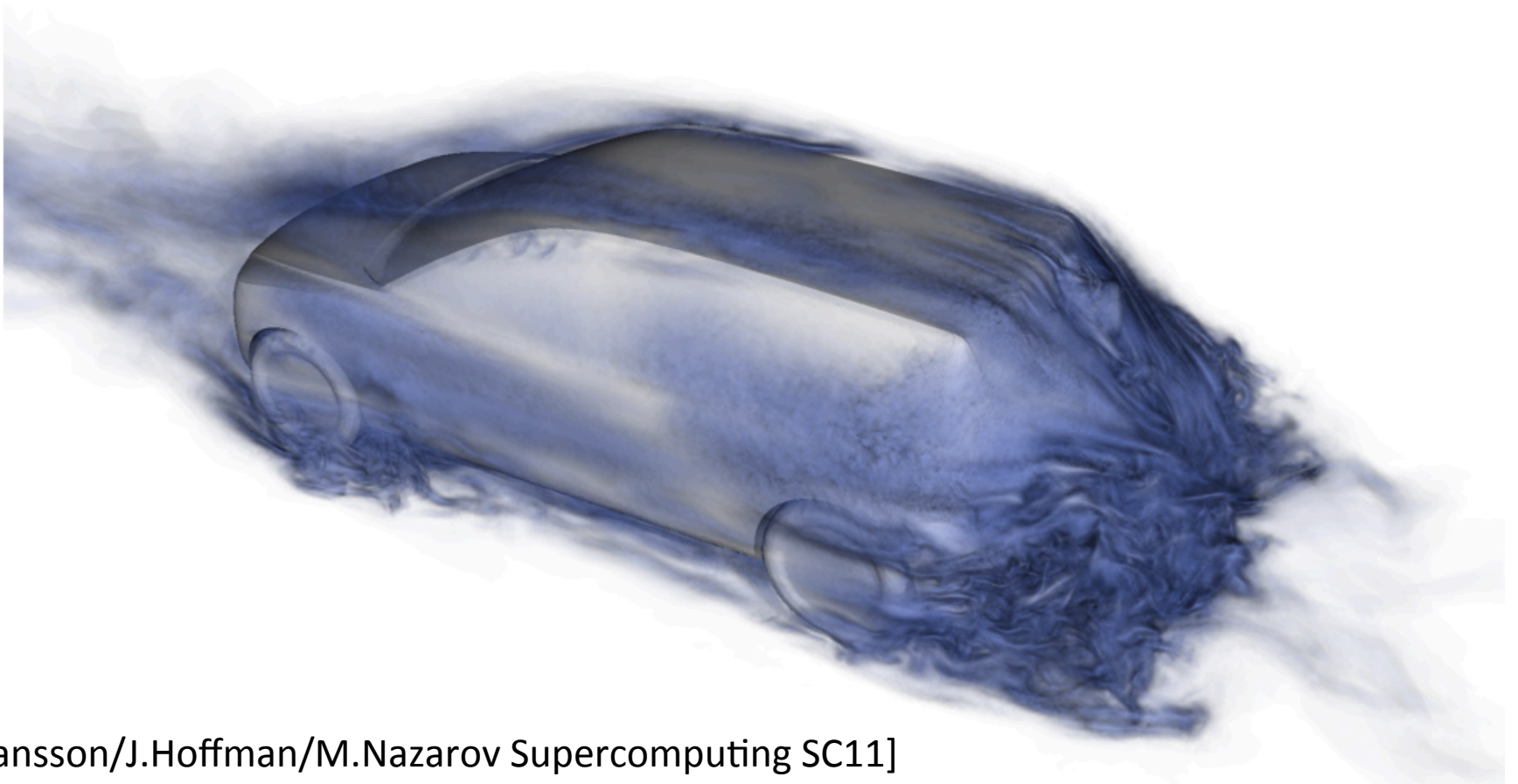
- ▶ Refine w.r.t error in drag
- ▶ Reference value $C_D = 0.359$



Full Car model (VRAK)

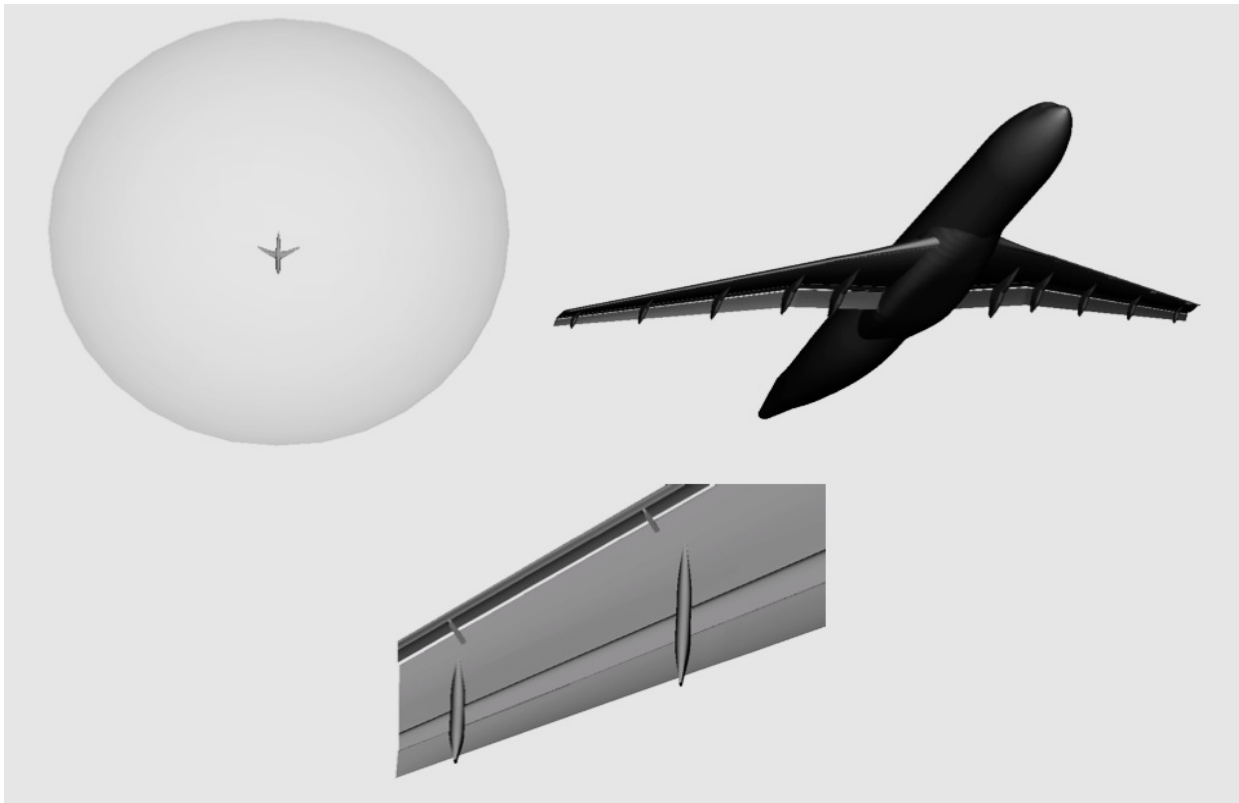
Dual solution

The solution characterize sensitivity in the output (drag)

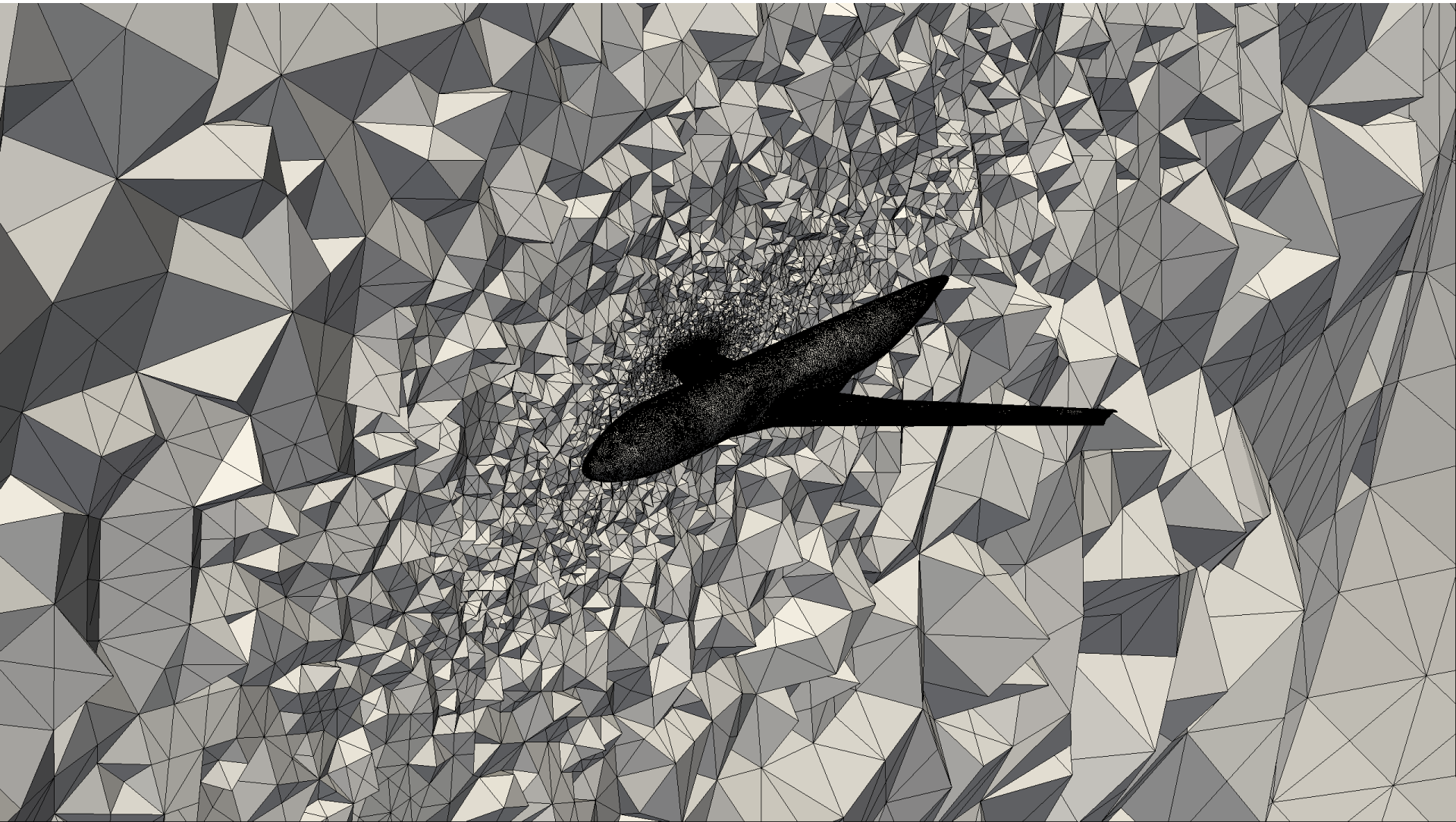


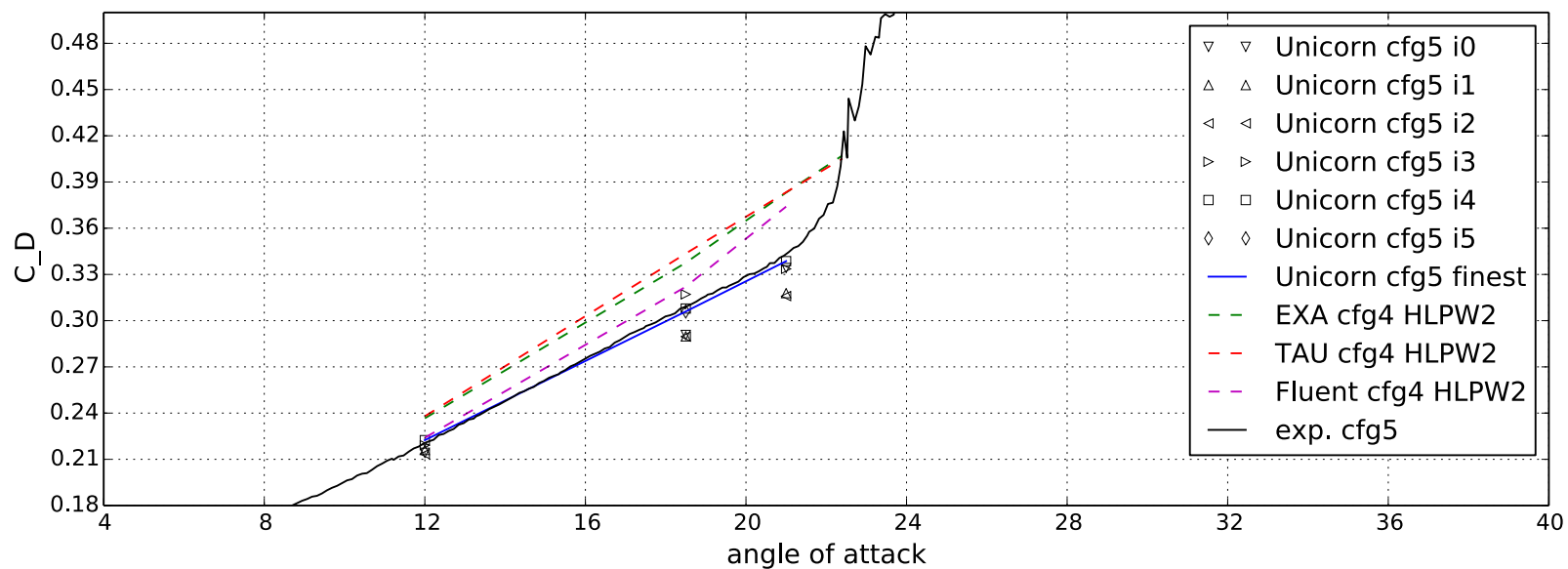
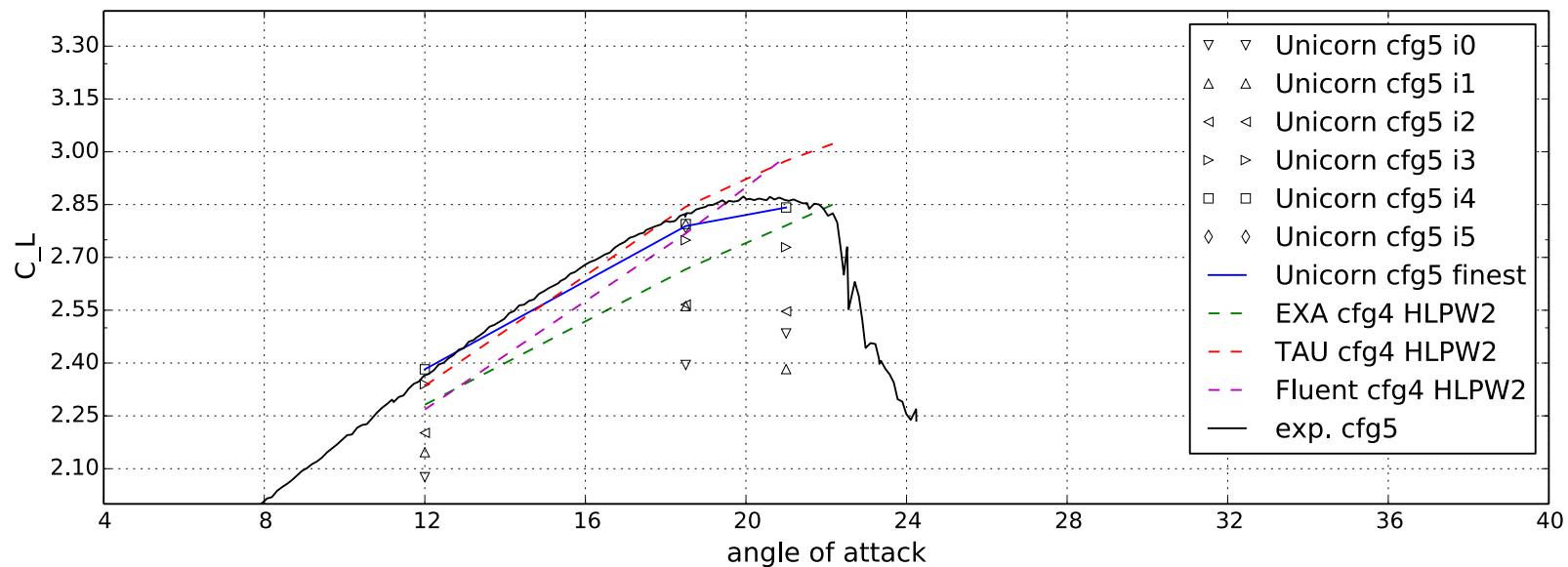
AIAA HighLift Benchmark

[J.Hoffman,/J.Jansson/N.Jansson/R.Vilela De Abreu, AIAA SciTech, 2014]

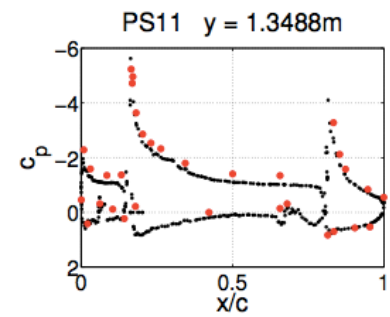
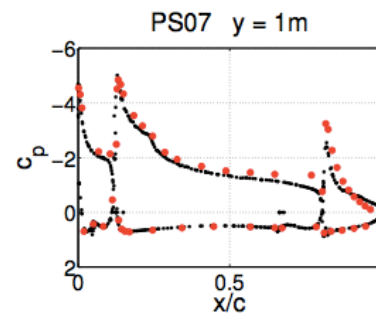
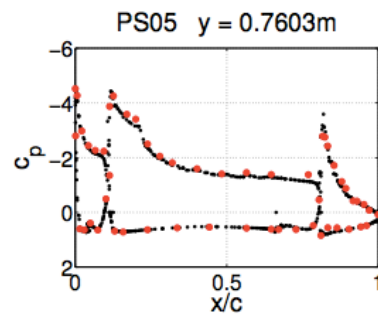
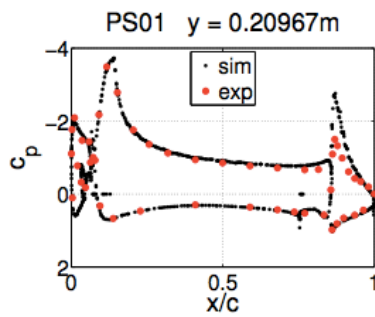
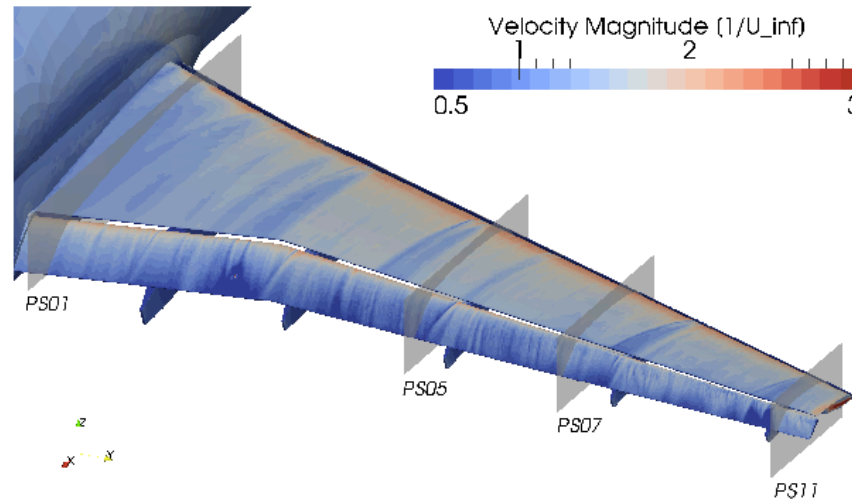


Optimized mesh : aspect ratio ca. 10^6



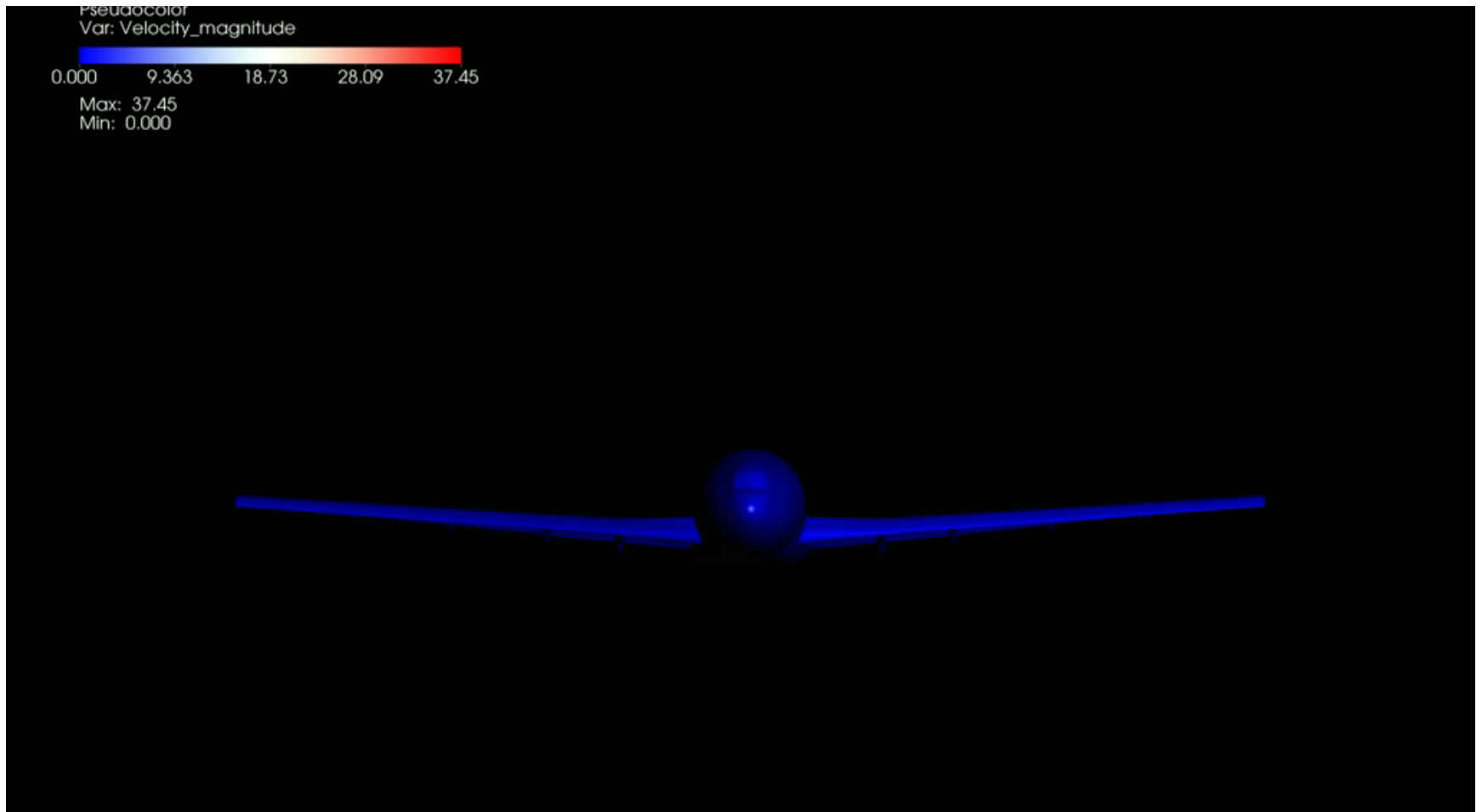


HiLiftPW-2 : $\alpha_{oa} = 12$



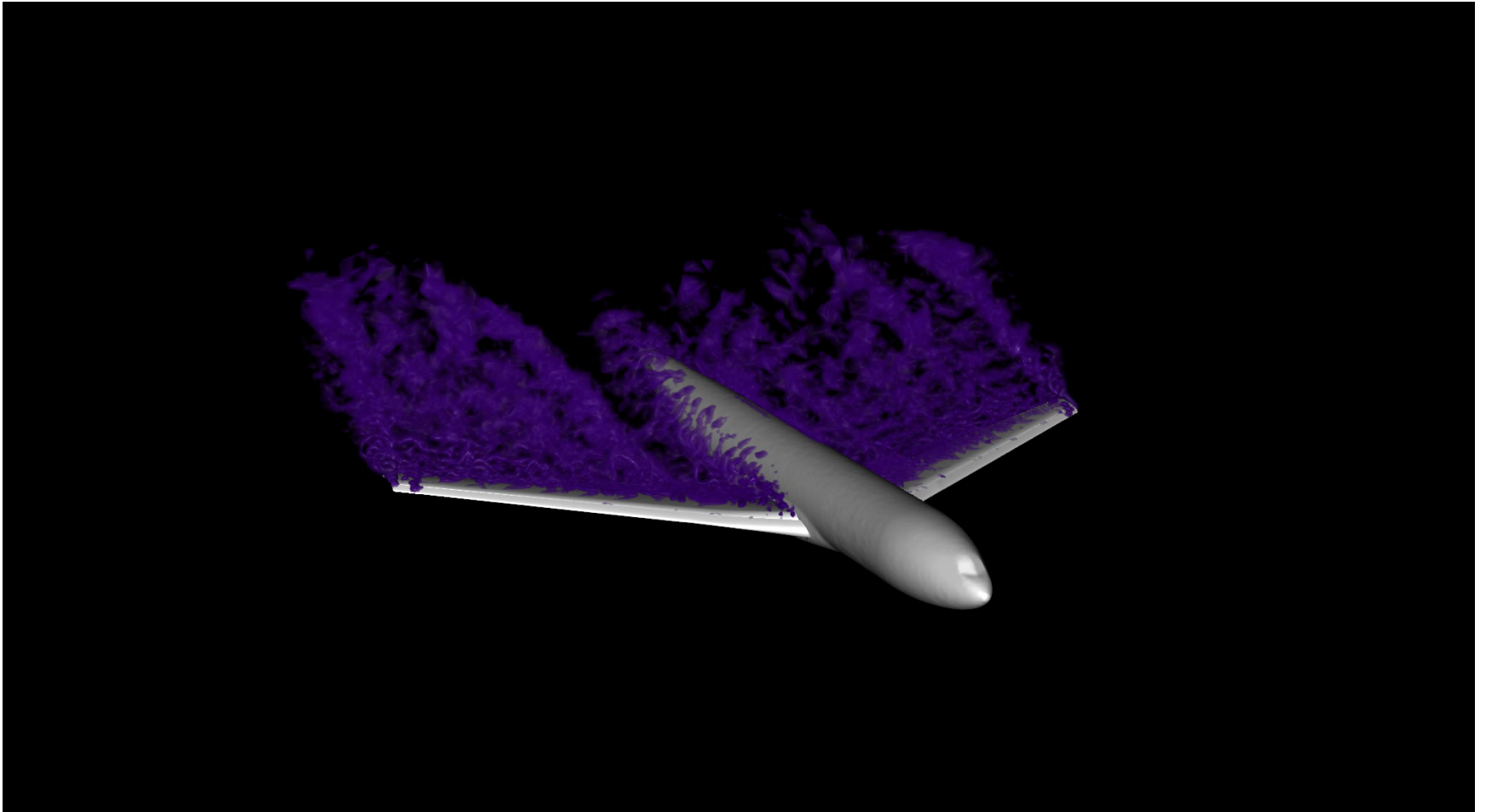
Visualization

(Rendering: 250GB data, 2 days, 80GB, 16 cores)



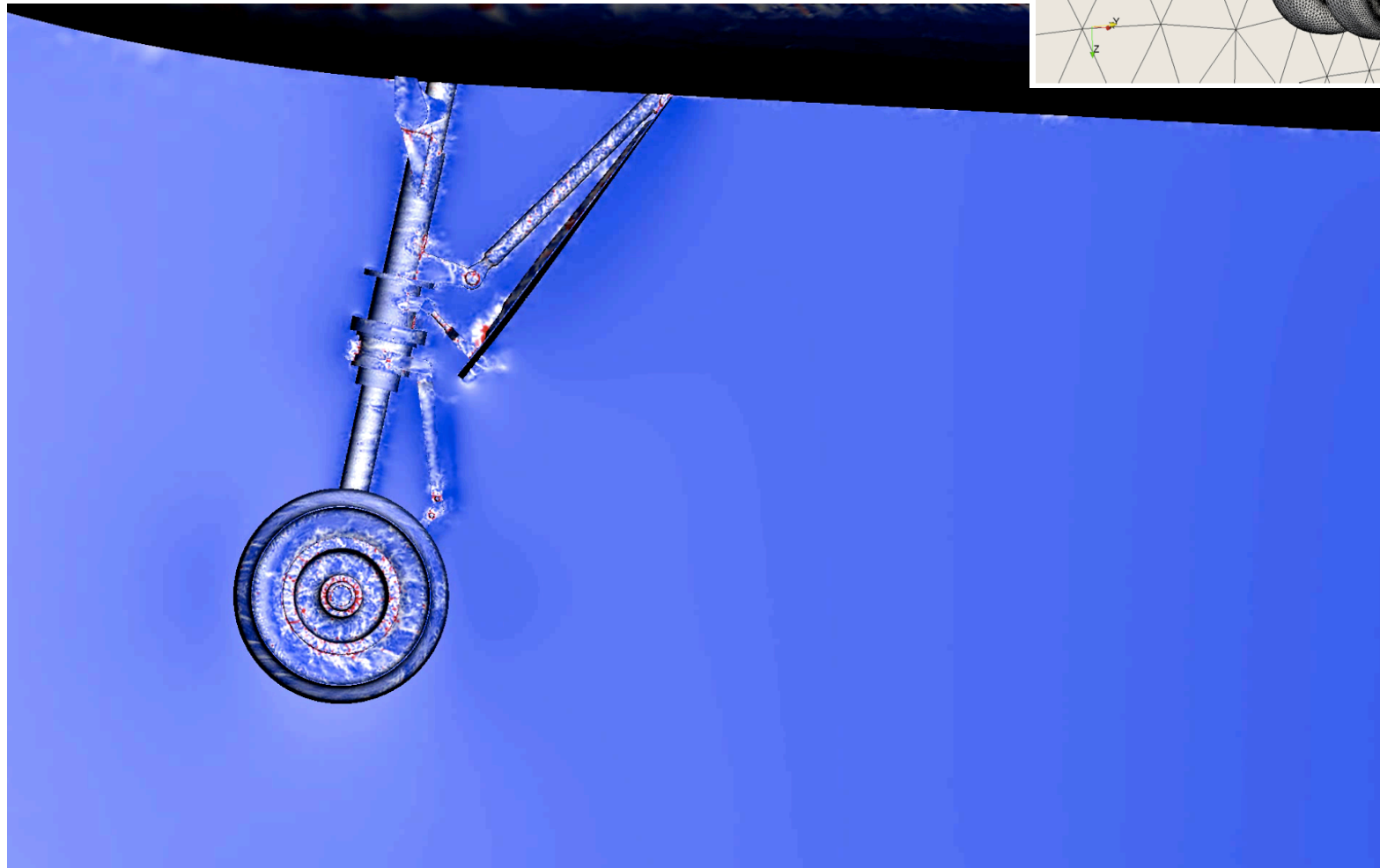
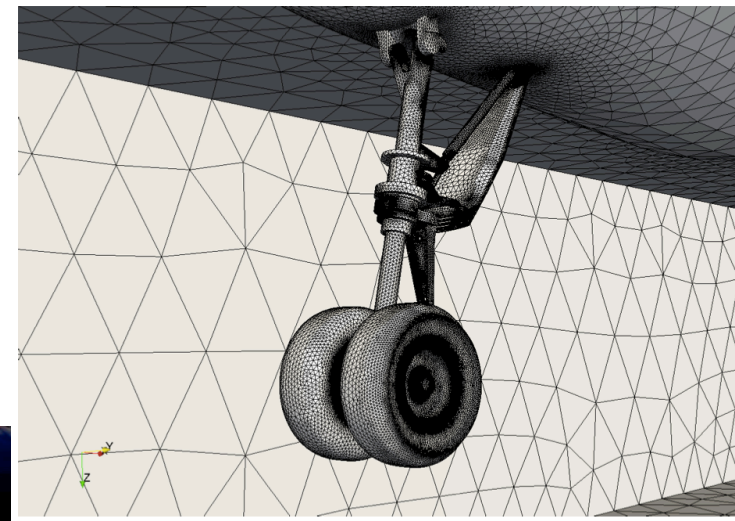
Visualization

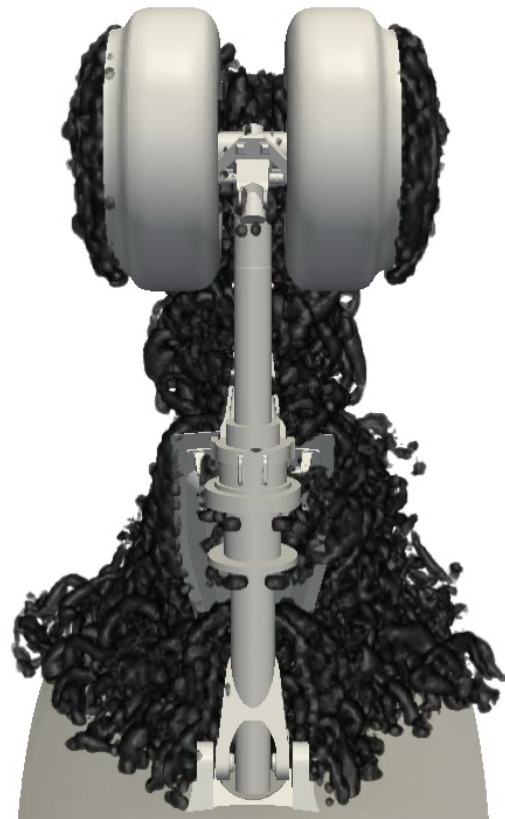
(Rendering: 250GB data, 2 days, 80GB, 16 cores)



FEM Fluid Mechanics

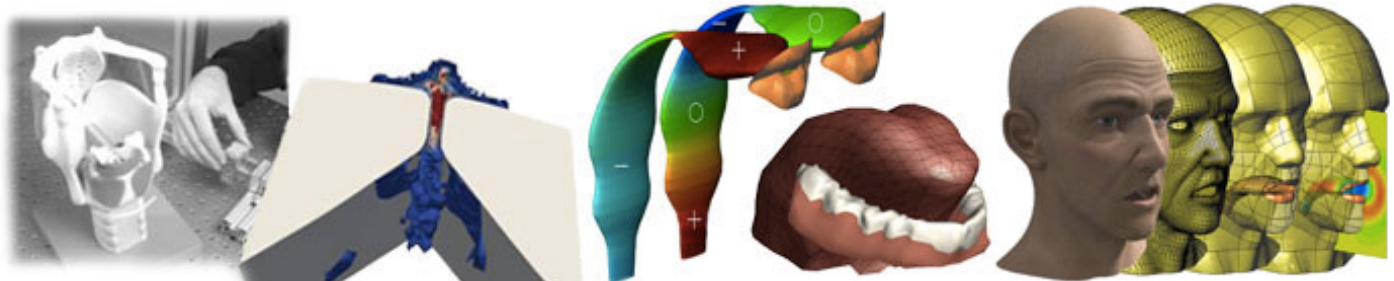
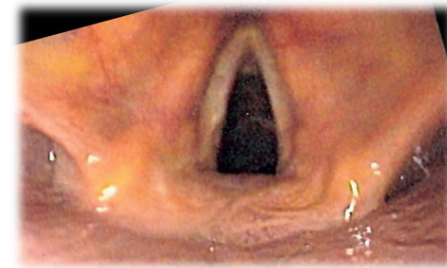
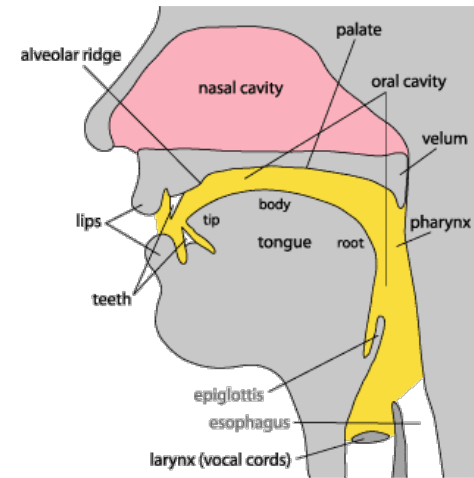
[Vilela De Abreu/Jansson/Hoffman BANC-II 2012, AIAA 2012]



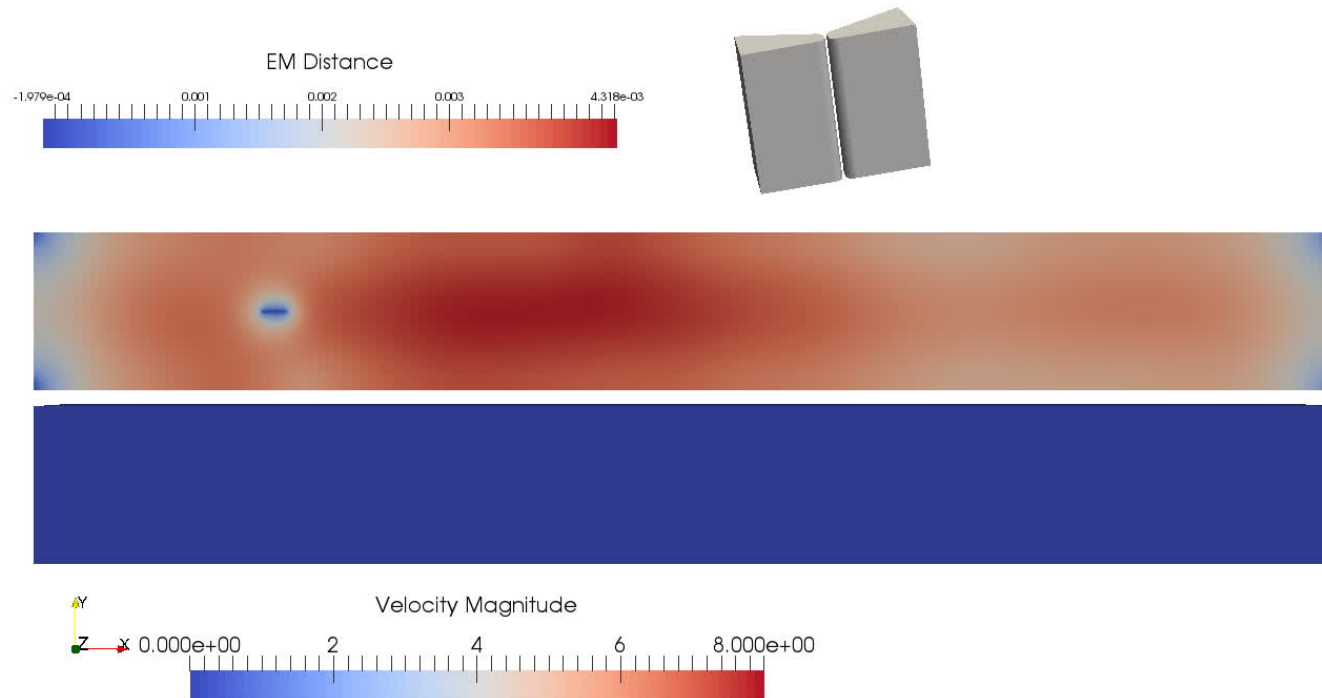


Eunison – simulating the human voice

- Goal: unified simulation of the human voice
- Turbulent flow FSI through vocal folds
- Acoustic propagation in vocal tract
- HPC implementation in FEniCS software
- Future and Emerging Technologies (FET)
- Partners: KTH, FAU Erlangen, Grenoble CNRS-GIPSA, Barcelona CIMNE, La Salle
- Collaborators: ArtiSynth group
- www.eunison.eu



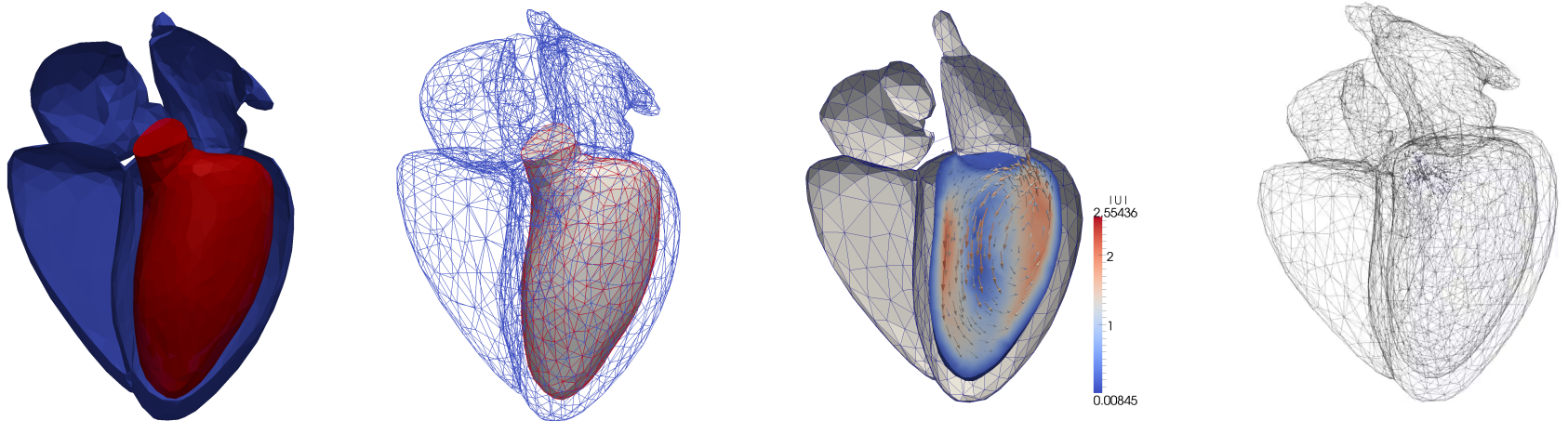
FSI simulation of vocal folds phonation



4-chamber heart model

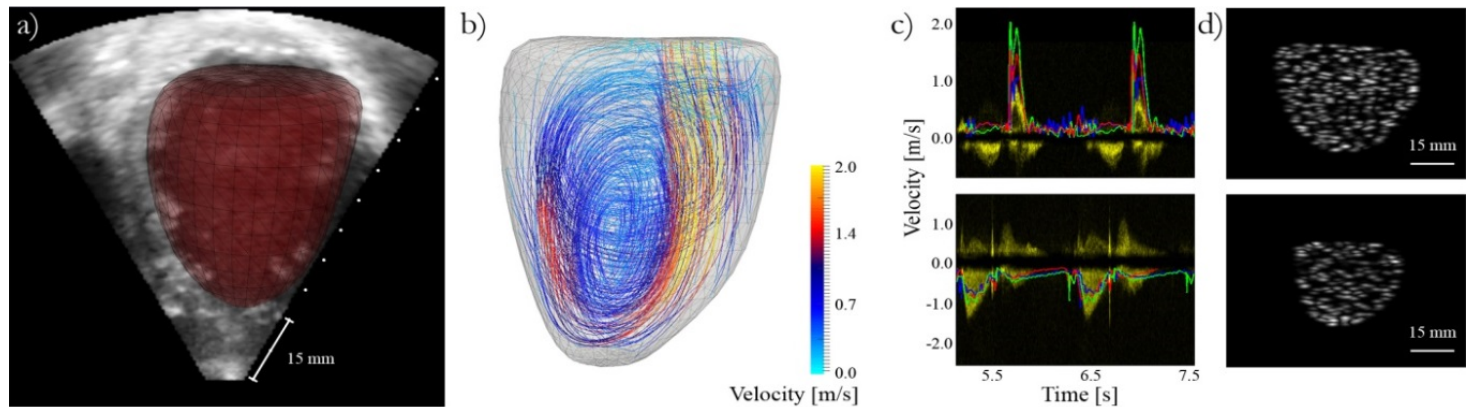
[J.Spühler/C.Fru/A.Larcher/J.Jansson/JH, 2015]

- 4-chamber heart model from ultrasound data [Philips]
- Extraction of left ventricle FEM model [KTH]
- FEM simulation of patient-specific blood flow



Patient specific simulation

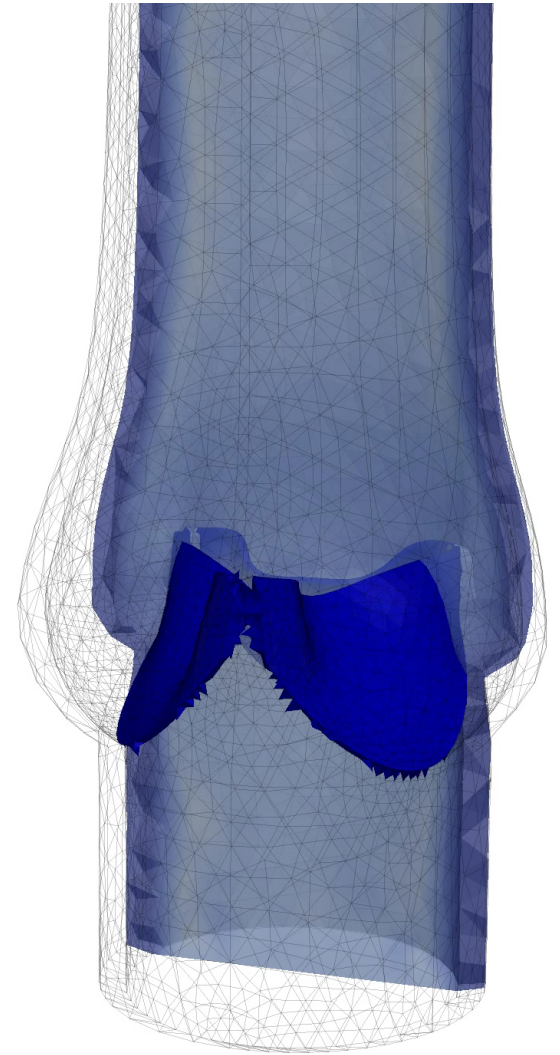
[Spühler/Larsson/Larsson/Hoffman, 2015]



Routine 4D TTE -> Segmentation -> Simulation -> Validation

Aortic valves

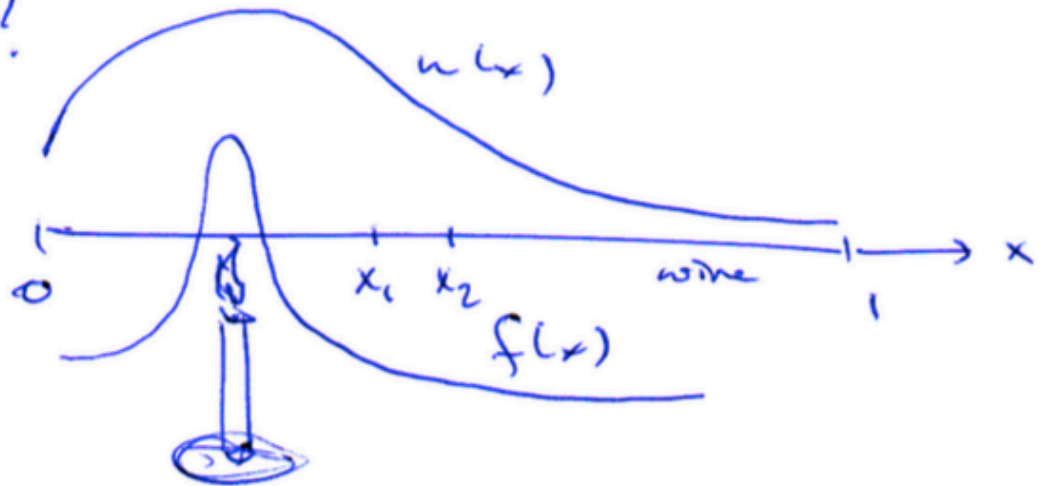
- Anatomy and geometry [Thubrikar, 1990]
- Simplified pulsating inflow



1D FEM

Heat conduction in a thin heat-conducting wire in $[0,1]$, heated by heat source $f(x)$.

What is the stationary distribution of the temperature $u(x)$?

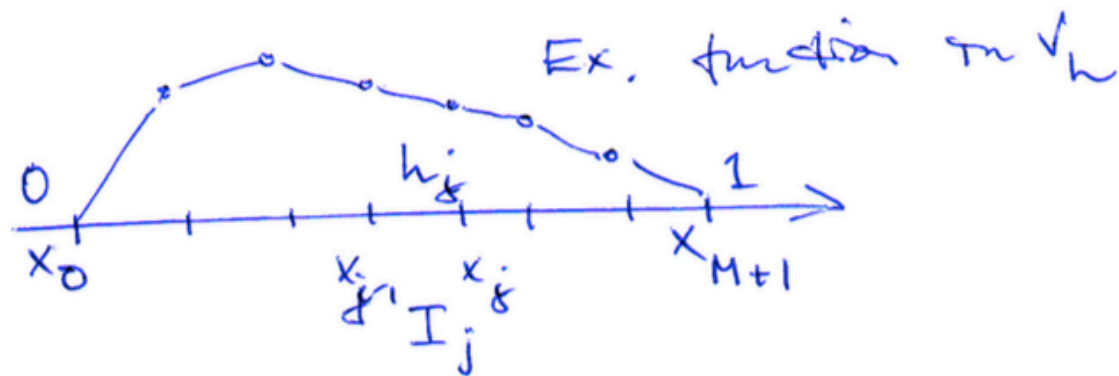


$q(x)$ - heat flux in the direction of positive x-axis

Find numerical approx. of D.E.

$$(*) \quad \begin{cases} -u'' = f & \text{on } (0,1) \\ u(0) = u(1) = 0 \end{cases}$$

Piecewise linear approximation: on mesh $\mathcal{T}_h$



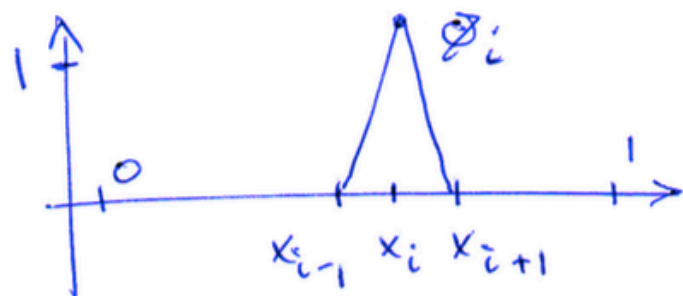
Divide $[0,1]$ into subintervals $I_j = (x_{j-1}, x_j)$

of length $h_j = x_j - x_{j-1}$; $\mathcal{T}_h = 0 = x_0 < x_1 < \dots < x_{n+1} = 1$

$V_h = \left\{ \text{set of all cont. p.w. lin. functions on } \Omega \text{ that} \right\}$
 are zero at $x=0$ and $x=1$.

V_h finite dimensional vector space of dimension M
 (M degrees of freedom : nodes)

Basis of V_h : hat functions $\{\phi_j\}_{j=1}^M$



$$\phi_i(x) = \begin{cases} 0 & x \notin [x_{i-1}, x_{i+1}] \\ \frac{x - x_{i-1}}{x_i - x_{i-1}} & x \in [x_{i-1}, x_i] \\ \frac{x - x_{i+1}}{x_i - x_{i+1}} & x \in [x_i, x_{i+1}] \end{cases}$$

Any function $v(x)$ on V_h can be written

$$v(x) = \sum_{j=1}^M v(x_j) \phi_j(x) = \sum_{j=1}^M \phi_j(x) \phi_j(x)$$

$U(x) = \sum_{j=1}^n \alpha_j \phi_j(x)$ is determined by $\{\alpha_j\}_{j=1}^n$

Determine $\{\alpha_j\}_{j=1}^n$ from (G)!

$$(G) \quad \int_0^1 U' v' dx = \int_0^1 f v dx \quad \forall v \in V_h$$

$$\Rightarrow \int_0^1 \left(\sum_{j=1}^n \alpha_j \phi_j(x) \right)' \phi_i' dx = \int_0^1 f \phi_i dx, \quad i=1, \dots, n$$

($\{\phi_i\}_{i=1}^n$ basis of $V_h \Rightarrow$ suff. to check for basis)

$$\Rightarrow \sum_{j=1}^n \alpha_j \int_0^1 \phi_j' \phi_i' dx = \int_0^1 f \phi_i dx \quad i=1, \dots, n$$

Corresponds to linear system of equations: $A\alpha = b$

```

# Begin demo

from dolfin import *

# Create mesh and define function space
mesh = UnitSquareMesh(32, 32)
V = FunctionSpace(mesh, "Lagrange", 1)

# Define Dirichlet boundary (x = 0 or x = 1)
def boundary(x):
    return x[0] < DOLFIN_EPS or x[0] > 1.0 - DOLFIN_EPS

# Define boundary condition
u0 = Constant(0.0)
bc = DirichletBC(V, u0, boundary)

# Define variational problem
u = TrialFunction(V)
v = TestFunction(V)
f = Expression("10*exp(-(pow(x[0] - 0.5, 2) + pow(x[1] - 0.5, 2)) / 0.02)")
g = Expression("sin(5*x[0])")
a = inner(grad(u), grad(v))*dx
L = f*v*dx + g*v*ds

# Compute solution
u = Function(V)
solve(a == L, u, bc)

# Save solution in VTK format
file = File("poisson.pvd")
file << u

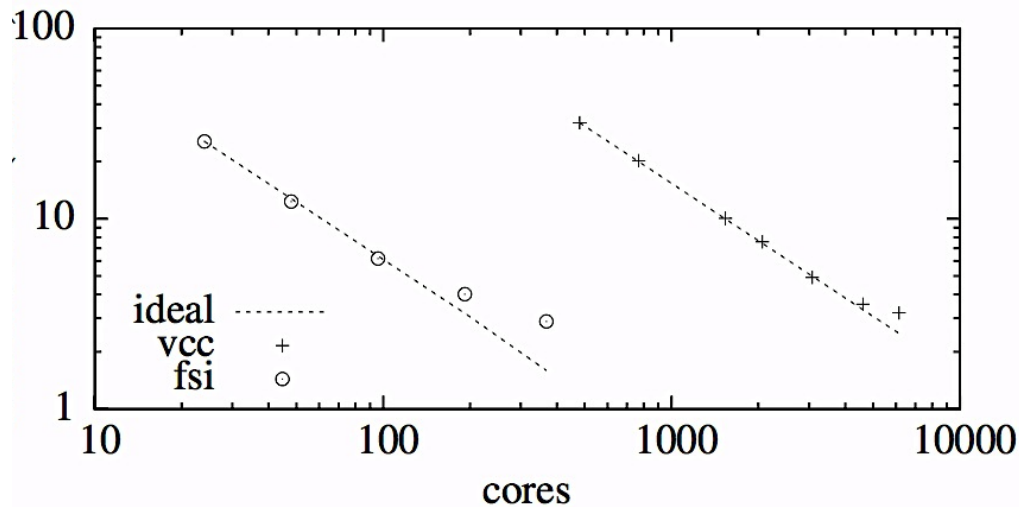
# Plot solution
plot(u, interactive=True)

```

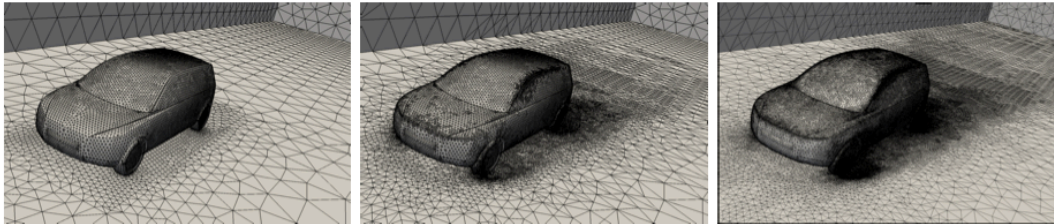
FEniCS open source FEM software



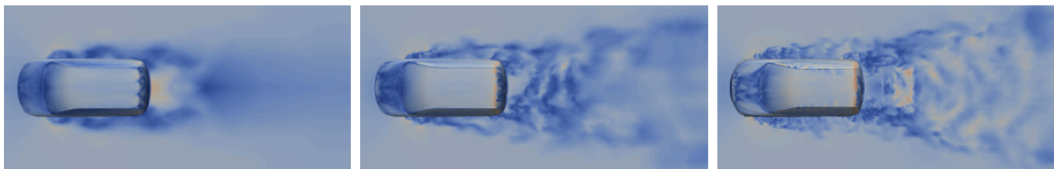
[N.Jansson/J.Hoffman/M.Nazarov Supercomputing SC11, 2011]



Optimal scaling up to tens of thousands of cores



Volvo Cars VRAK geometry model



KTH Department of Computational Science and Technology (CST)

- High Performance Computing: leads exascale FP7 projects, PDC delivers HPC capacity in Sweden/SNIC, Europe/PRACE
- Visualization and Human Computer Interaction: HPCViz runs the Visualization Studio VIC (research/education/industry)
- Mathematical Modeling and Simulation: analysis, methods, algorithms, open source software, applications, industry

