

# PyFR: High-Order Accurate Cross-Platform Petascale Computational Fluid Dynamics with Python

F. D. Witherden<sup>1</sup> and P. E. Vincent<sup>2</sup>

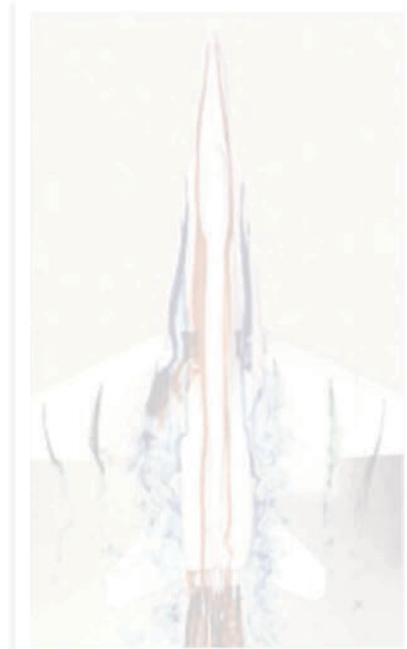
<sup>1</sup>Department of Aeronautics and Astronautics, Stanford University

<sup>2</sup>Department of Aeronautics, Imperial College London

AMS Seminar Series,  
NASA Ames Research Center, 31<sup>st</sup> May 2016

# Motivation

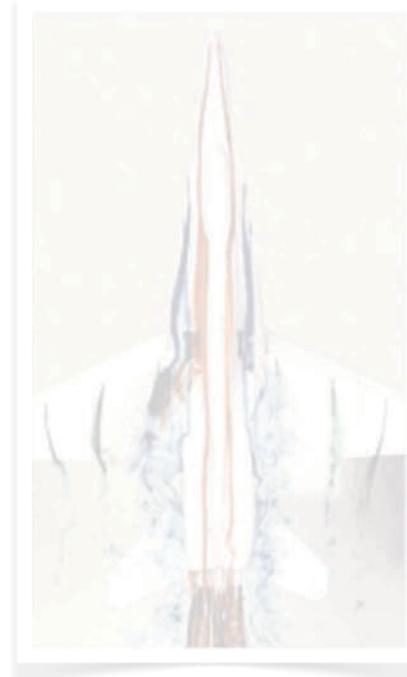
# Motivation



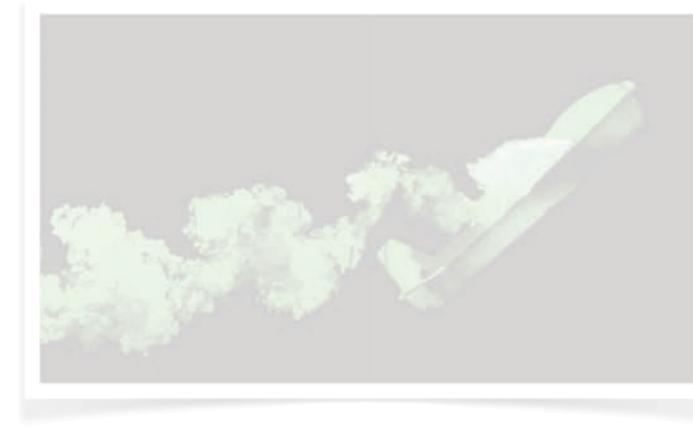
Current industry standard CFD tools  
have limited capabilities



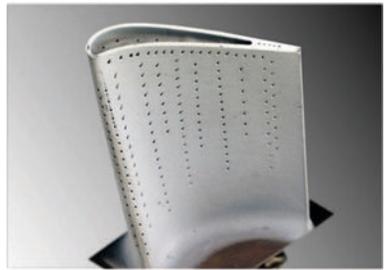
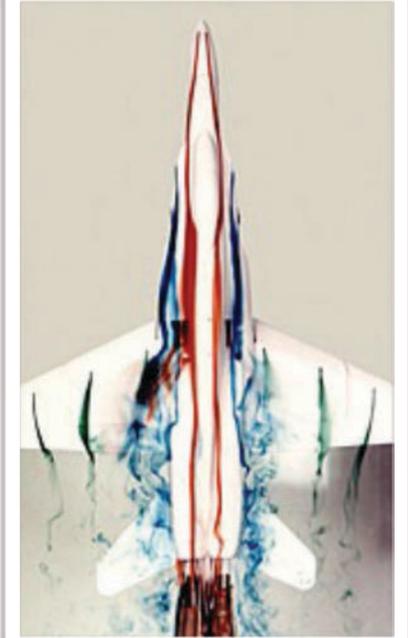
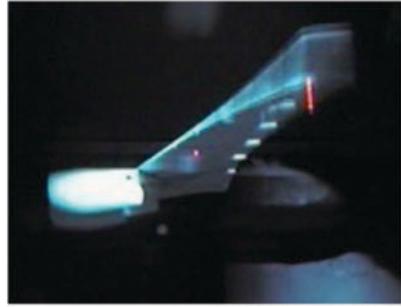
# Motivation



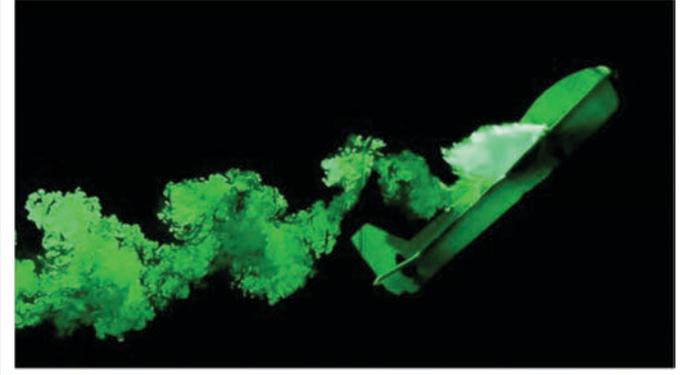
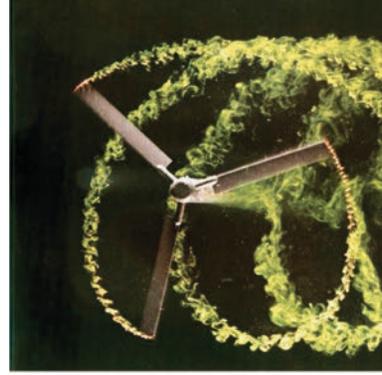
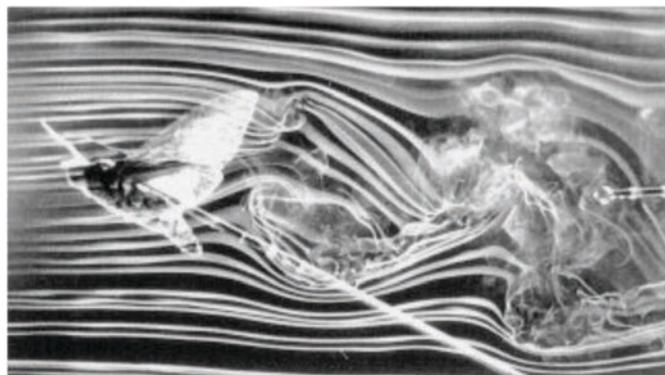
Technology is decades old and designed for solving steady flow problems (using RANS approach)



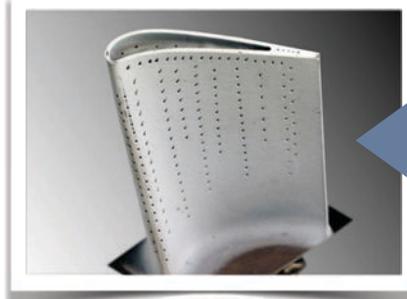
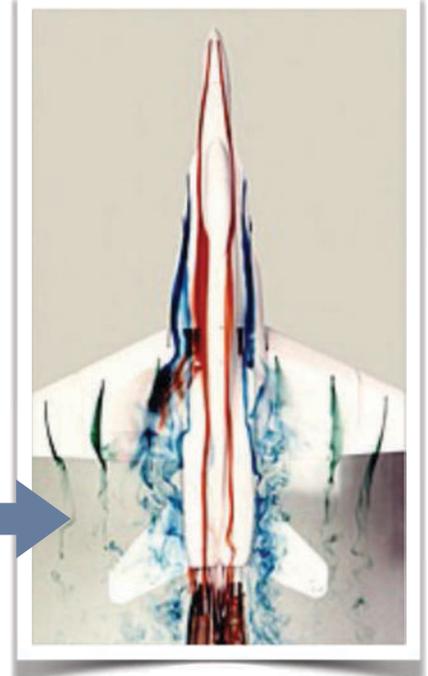
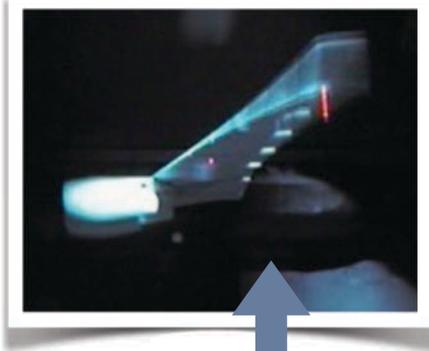
# Motivation



Technology is decades old and designed for solving steady flow problems (using RANS approach)



# Motivation

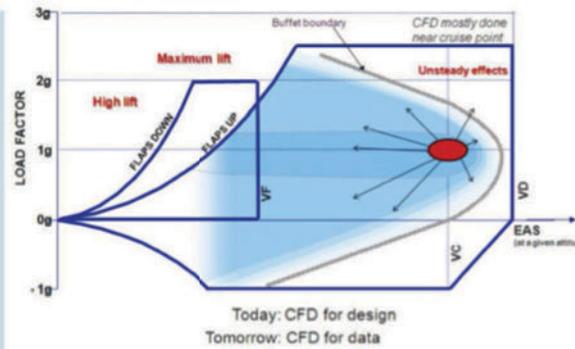
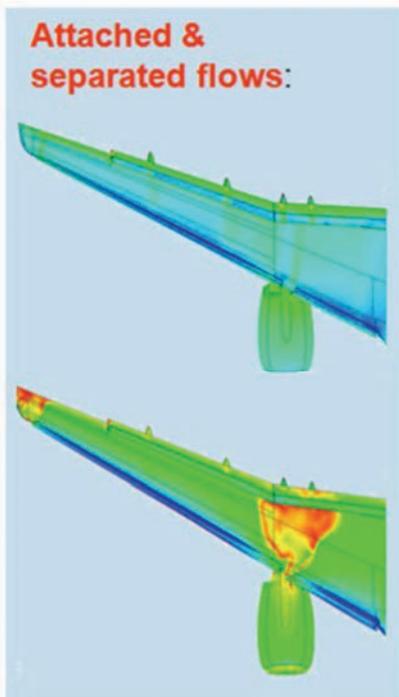


Need to expand the 'industrial CFD envelope'

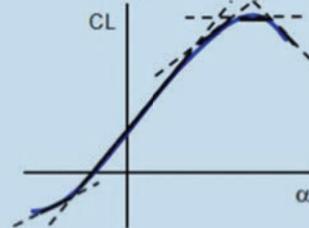


# Motivation

## Airbus Needs – expanding the envelope



Non-linearity:



**All configurations:**

Clean



Airbrakes out



High lift



# Motivation

- “reliable use of CFD has remained confined to a small but important region of the operating design space due to the inability of current methods to reliably predict turbulent separated flows” [2]

# Motivation

- Objective of our research is to advance industrial CFD capabilities from their current 'RANS plateau'

# Motivation

- Plan to achieve this by leveraging benefits of (and synergies between) high-order **Flux Reconstruction (FR)** methods for unstructured grids and massively-parallel **modern hardware platforms**

# Motivation

Flux Reconstruction  
+  
Modern Hardware



# Algorithms

- **Flux Reconstruction (FR)** approach to high-order methods was first proposed by Huynh in 2007 [3]
- **High-order** accurate in space
- **Works on unstructured grids**

# Algorithms

- So ...

High Accuracy + Complex Geometry

# Algorithms

- Consider **1D** scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = 0$$

- Divide **1D** domain into elements

$$\Omega = \bigcup_{n=0}^{N-1} \Omega_n \quad \bigcap_{n=0}^{N-1} \Omega_n = \emptyset \quad \Omega_n = \{x \mid x_n < x < x_{n+1}\}$$



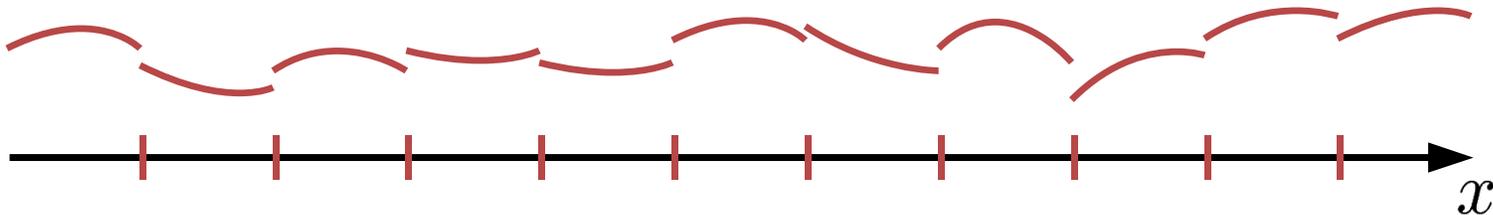
# Algorithms

- Consider **1D** scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = 0$$

- Represent solution by order **k** piecewise **discontinuous** polynomials in each element

$$u^\delta = \bigoplus_{n=0}^{N-1} u_n^\delta \approx u$$



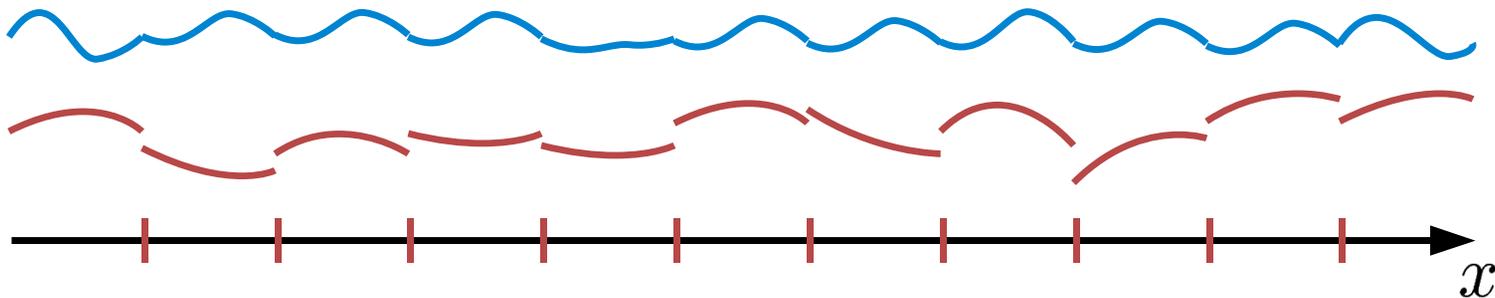
# Algorithms

- Consider 1D scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = 0$$

- Represent flux by order  $k+1$  piecewise continuous polynomials within each element

$$f^\delta = \bigoplus_{n=0}^{N-1} f_n^\delta \approx f$$



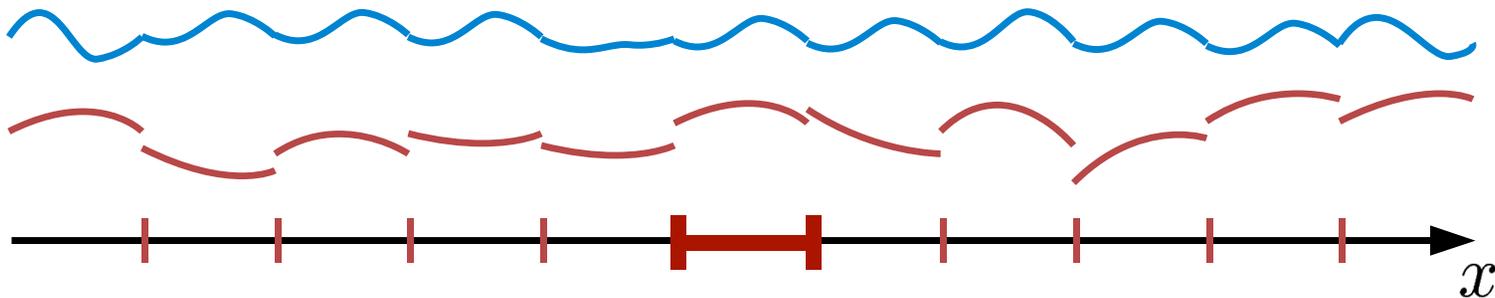
# Algorithms

- Consider 1D scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = 0$$

- Represent flux by order  $k+1$  piecewise continuous polynomials within each element

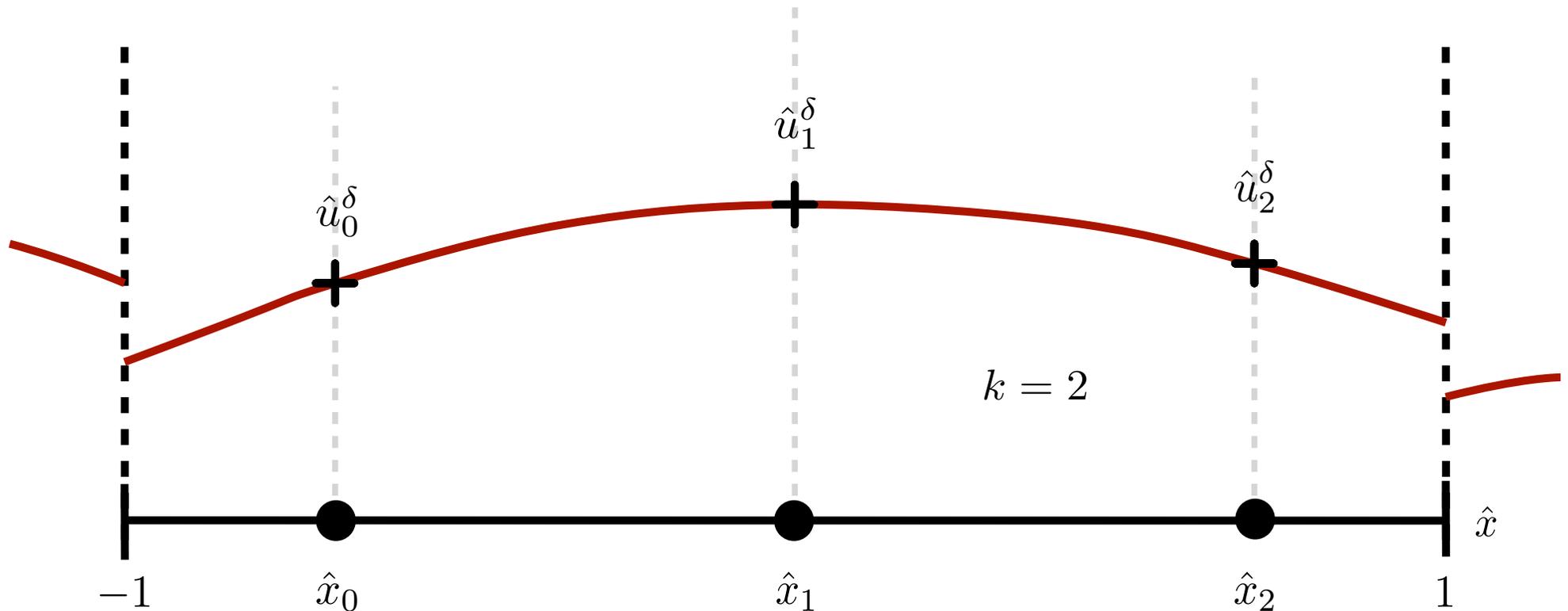
$$f^\delta = \bigoplus_{n=0}^{N-1} f_n^\delta \approx f$$



# Algorithms

- Represent order  $k$  solution within standard element using a **nodal basis**

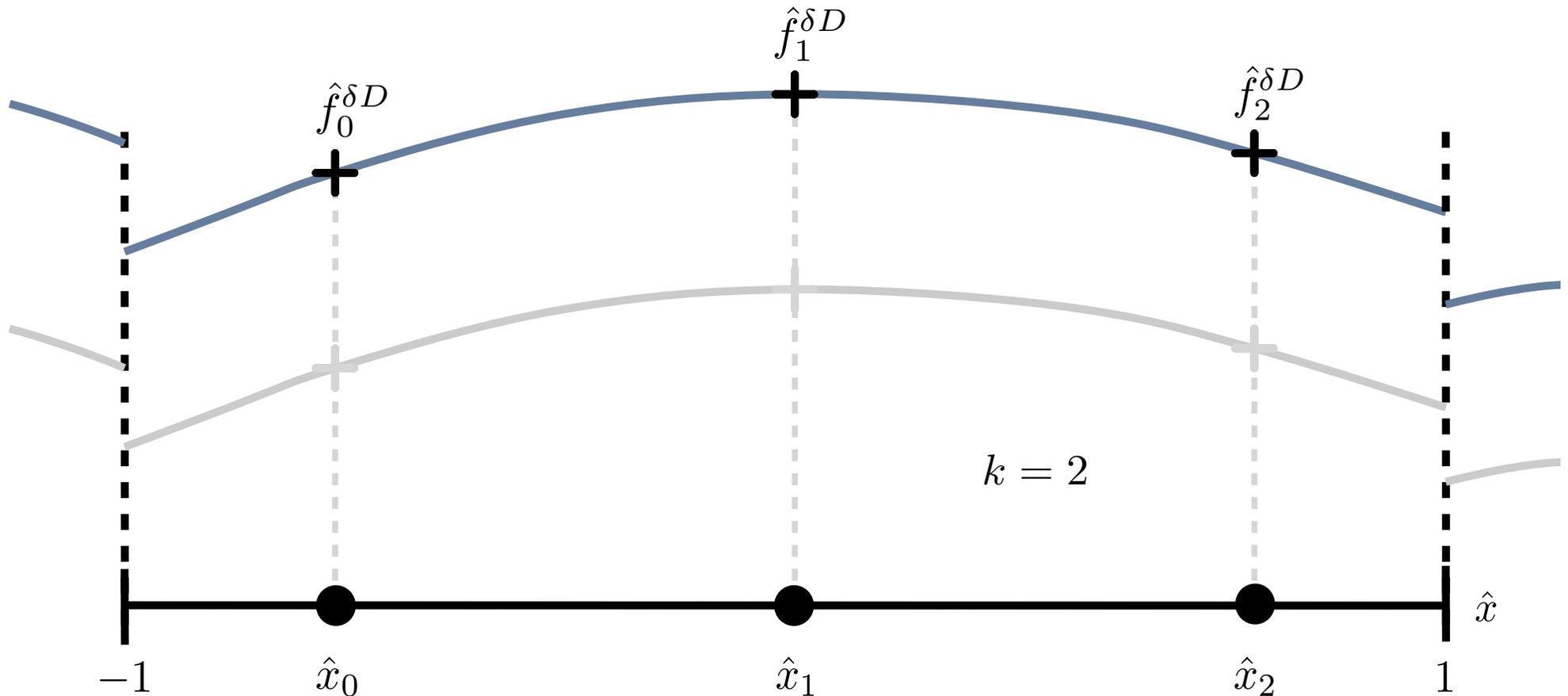
$$\hat{u}^\delta = \sum_{i=0}^k \hat{u}_i^\delta l_i$$



# Algorithms

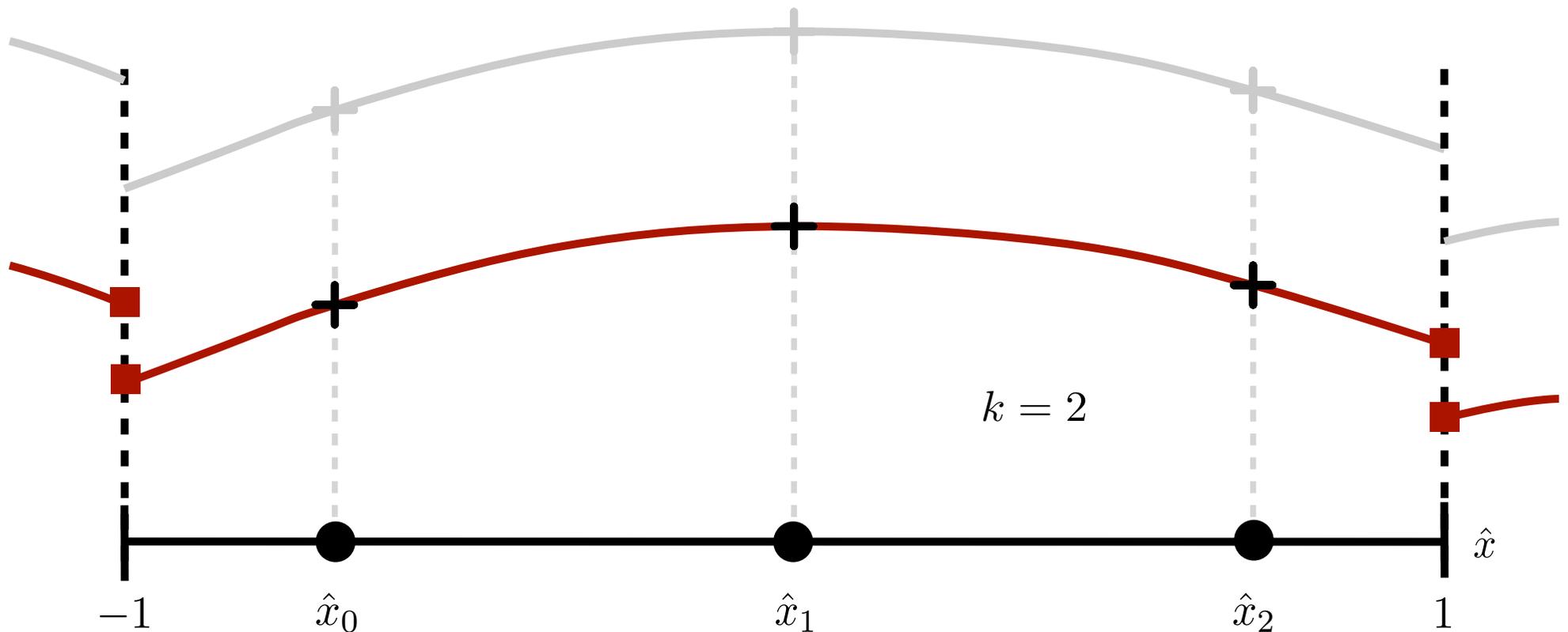
- Reconstruct discontinuous flux

$$\hat{f}^{\delta D} = \sum_{i=0}^k \hat{f}_i^{\delta D} l_i$$



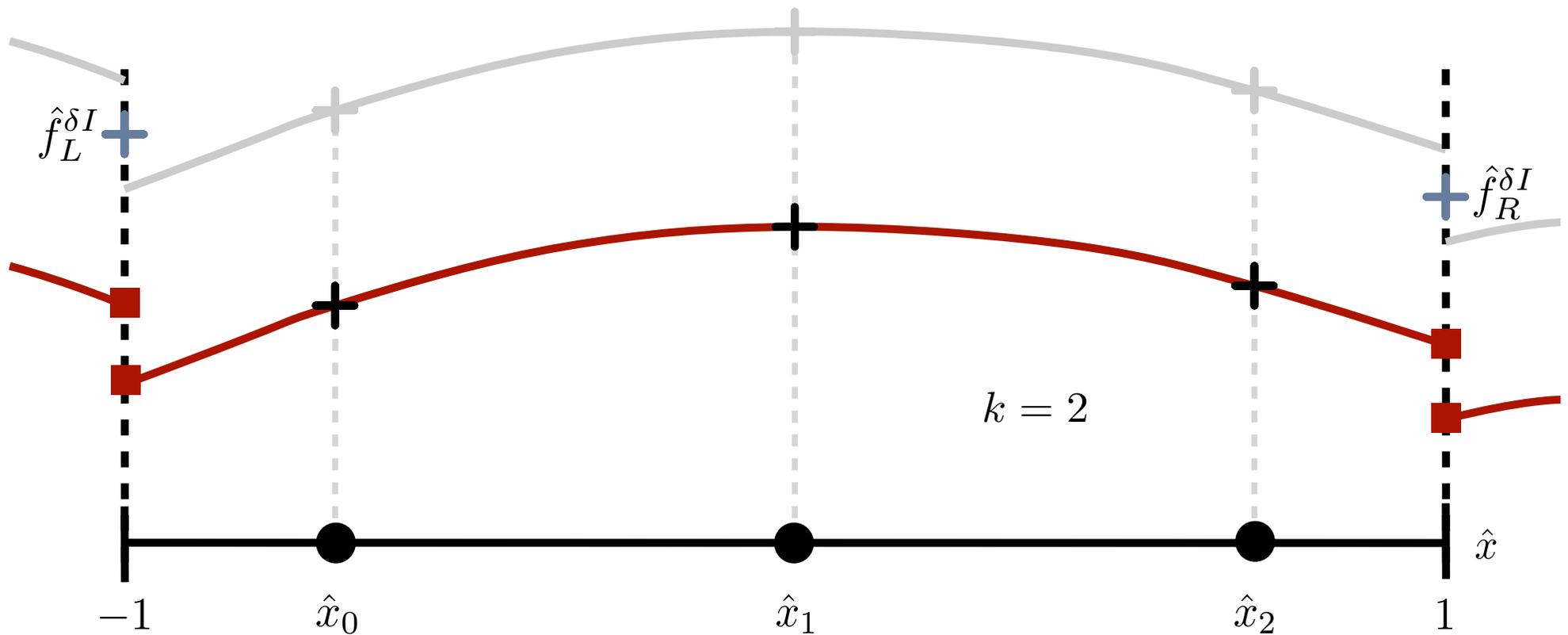
# Algorithms

- Evaluate solution at element boundaries



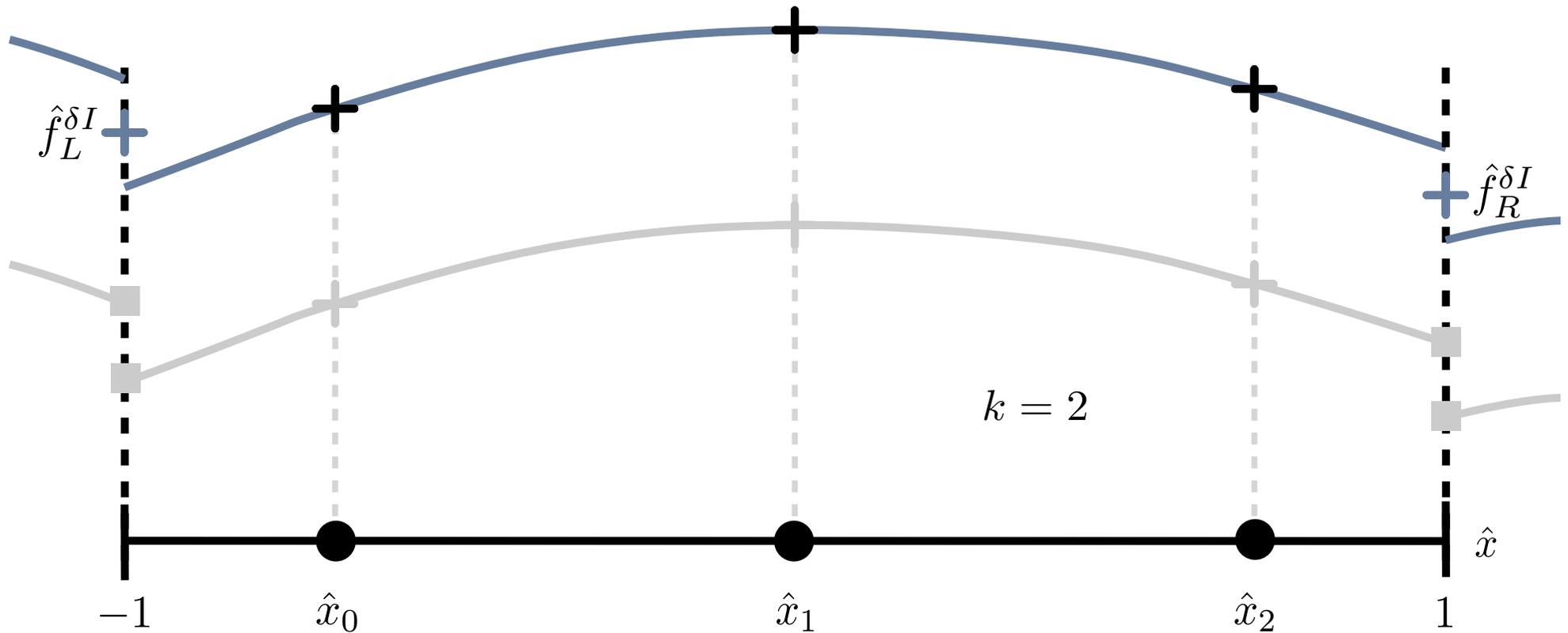
# Algorithms

- Calculate **common** interface fluxes



# Algorithms

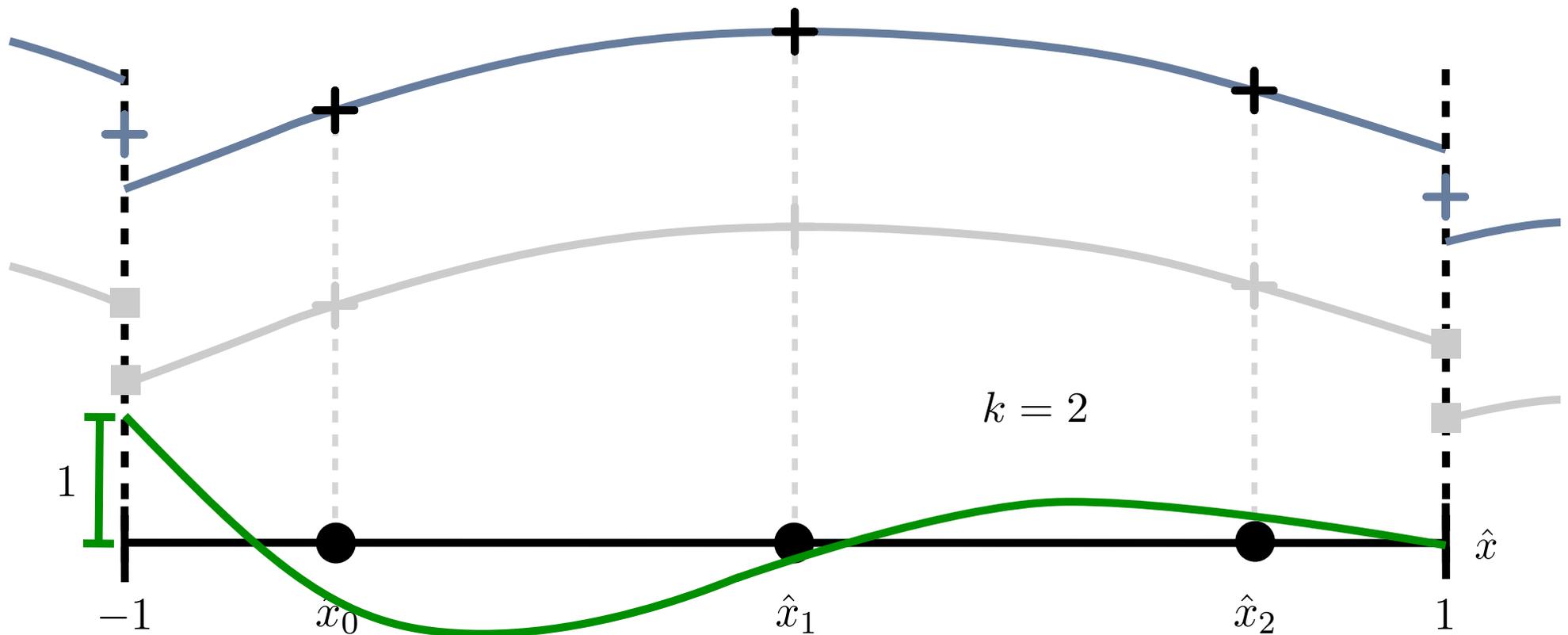
- Calculate common interface fluxes



# Algorithms

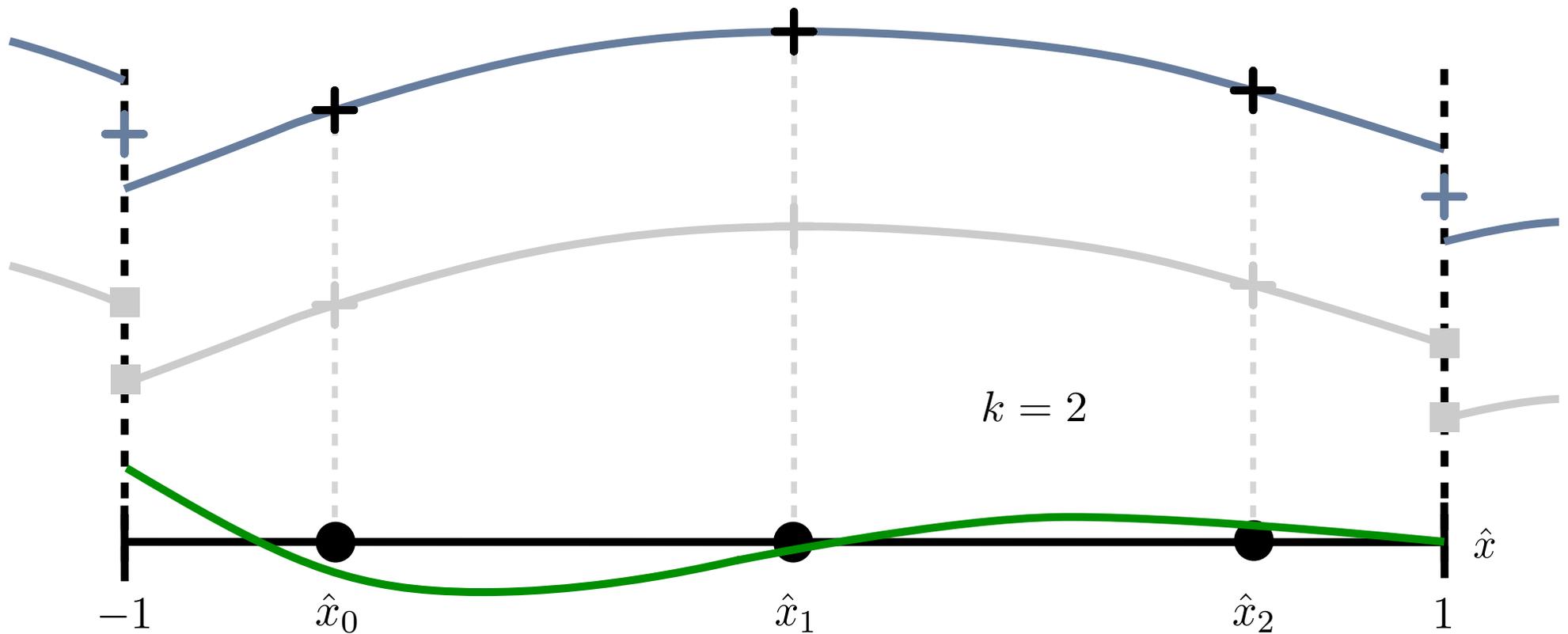
- Define order  $k+1$  'correction function'

$$g_L(-1) = 1, \quad g_L(1) = 0,$$



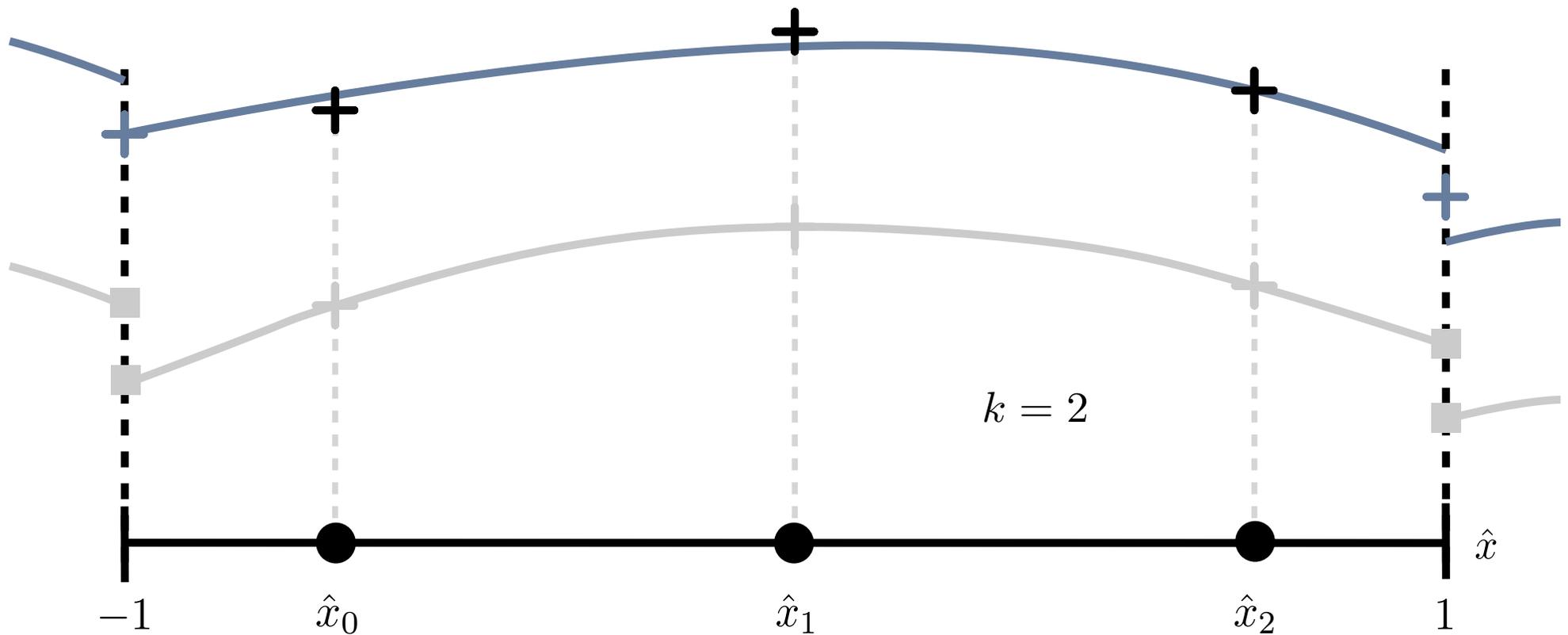
# Algorithms

- Scale this correction function ...



# Algorithms

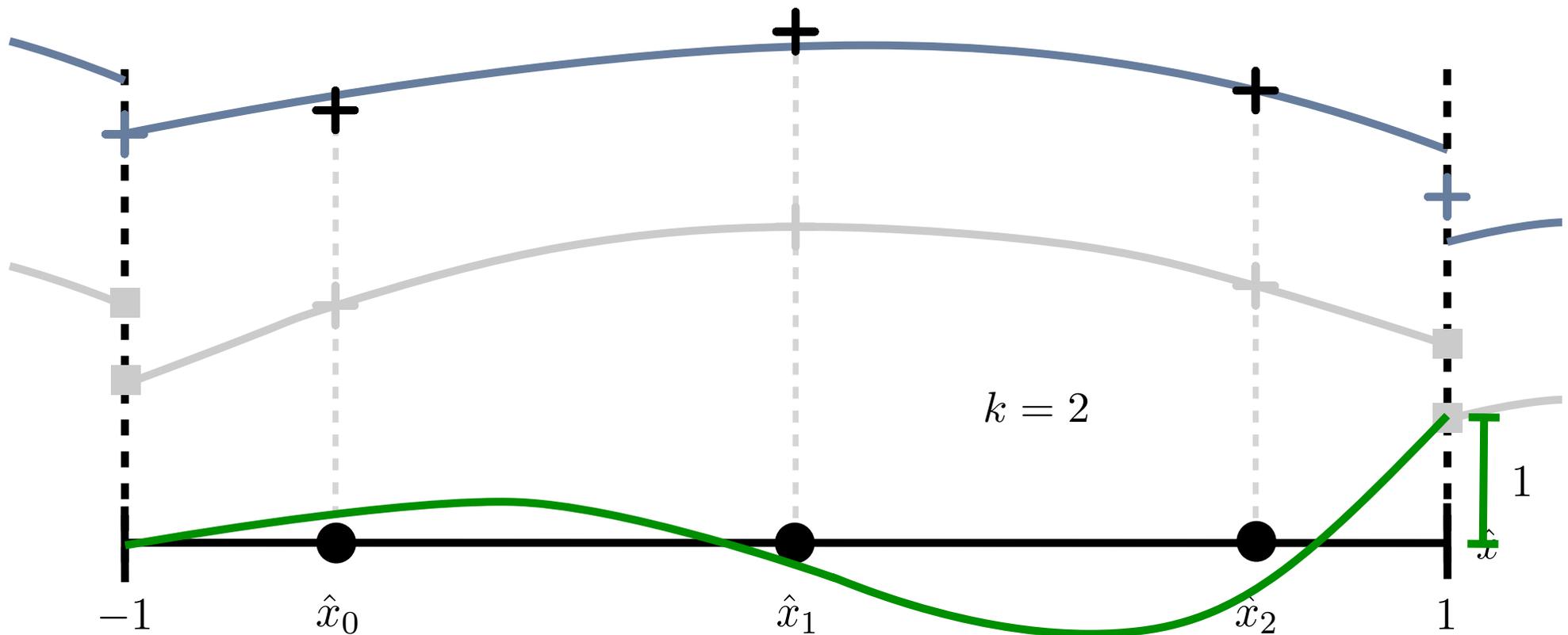
- ... and add it to the discontinuous flux



# Algorithms

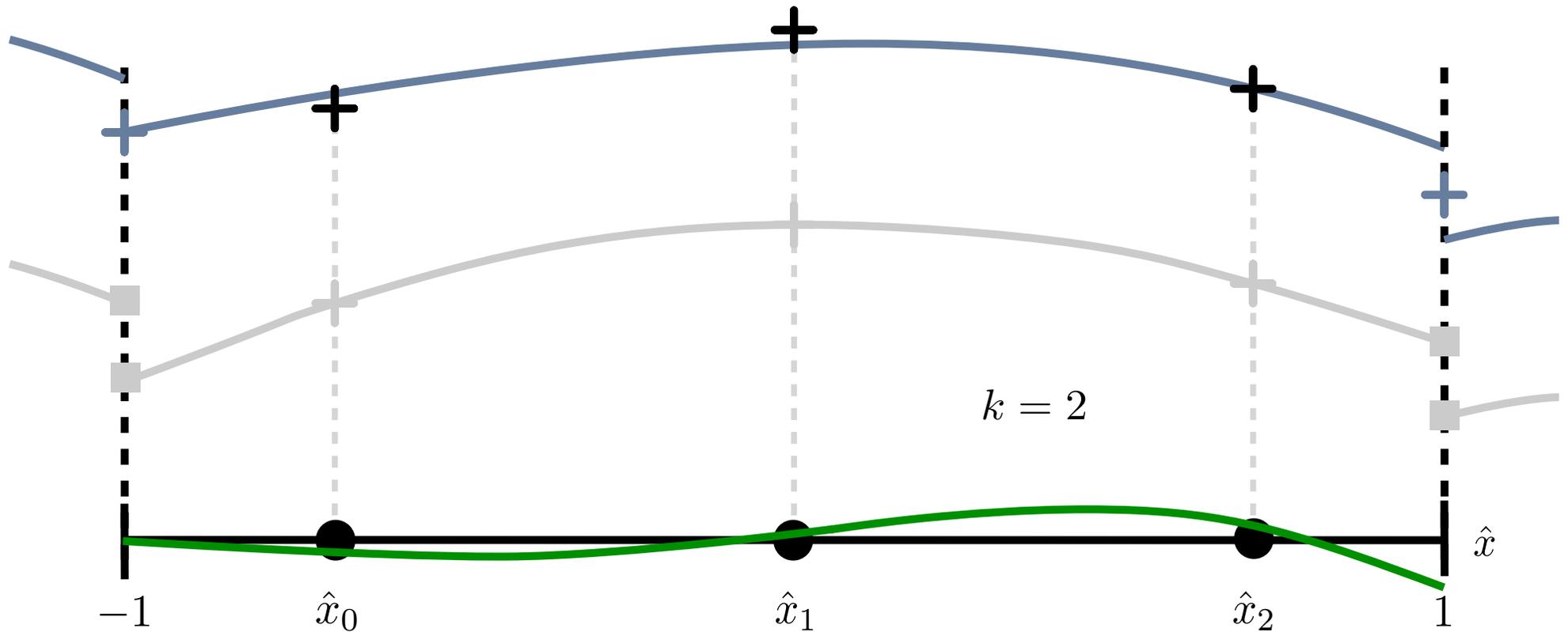
- Repeat from the right ...

$$g_R(-1) = 0, \quad g_R(1) = 1, \quad g_L(\hat{x}) = g_R(-\hat{x})$$



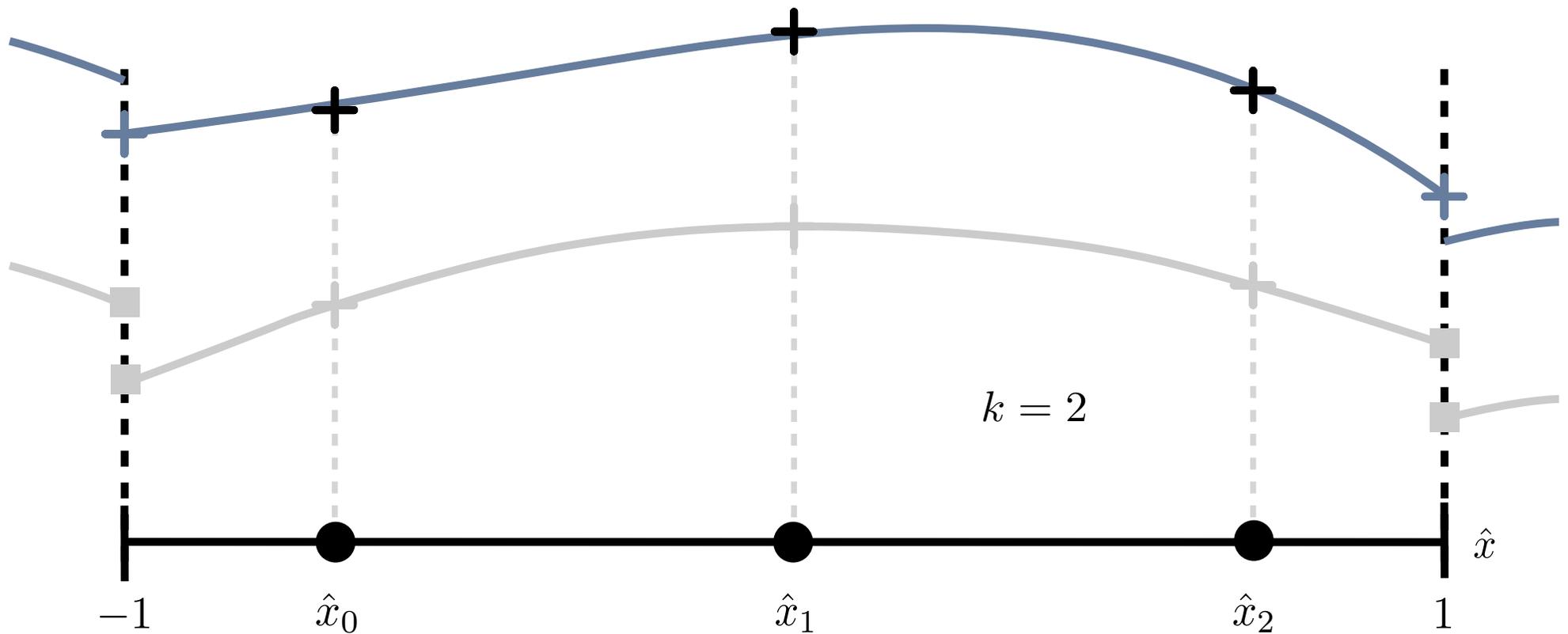
# Algorithms

- Repeat from the right ...



# Algorithms

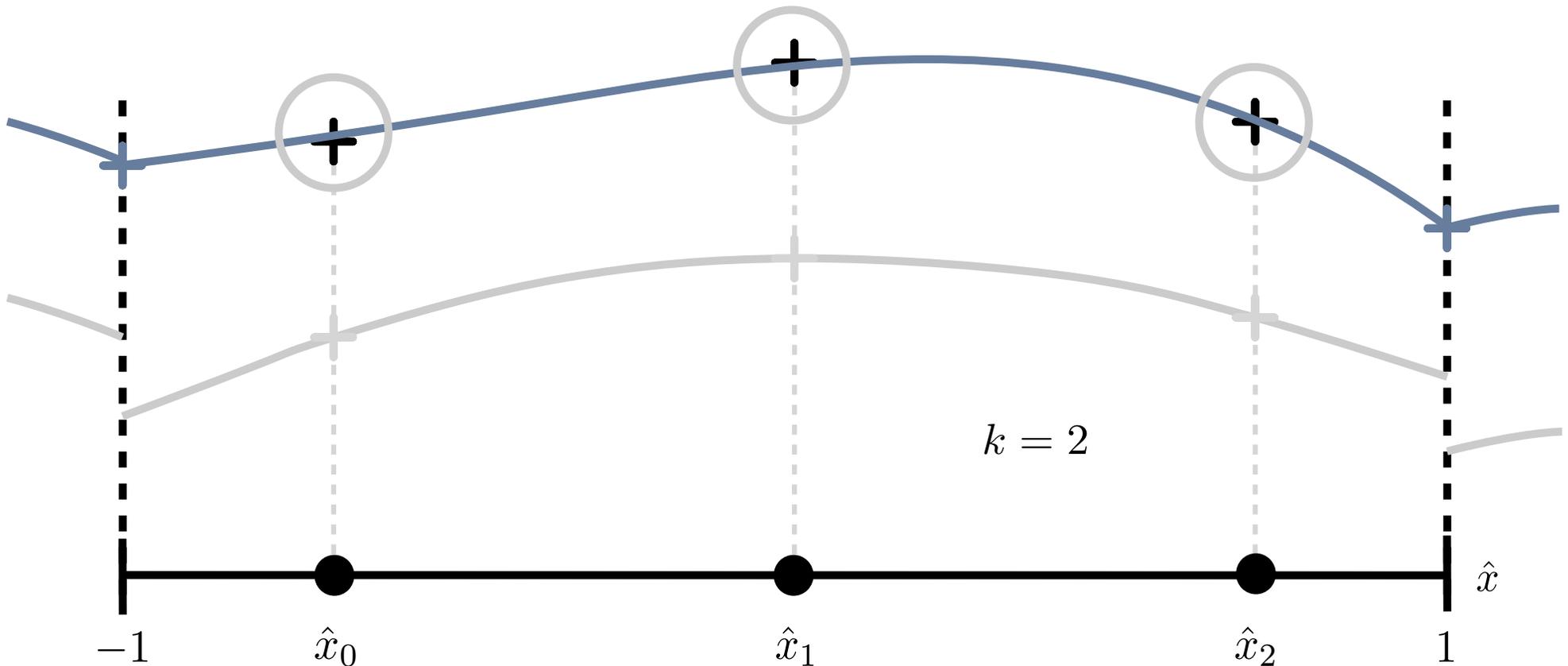
- Repeat from the right ...



# Algorithms

- Evaluate divergence of the continuous flux at the solution points

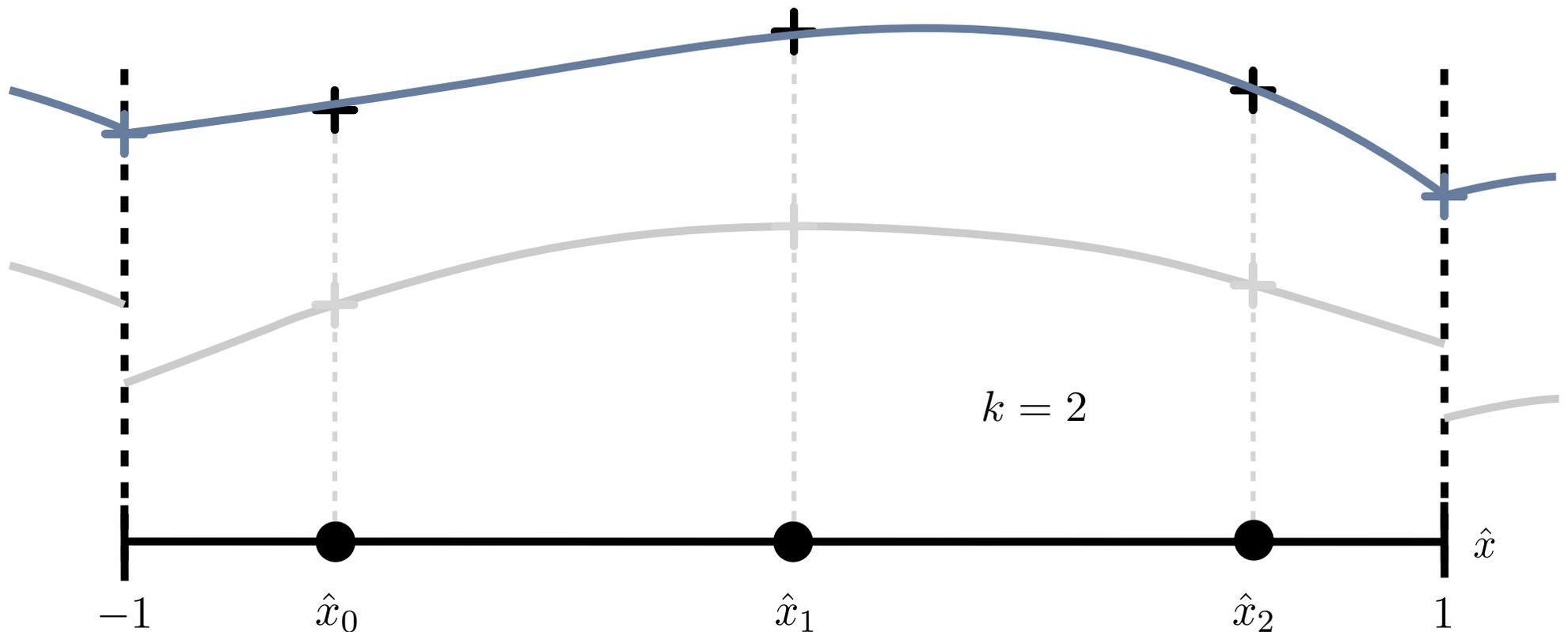
$$\frac{d\hat{u}_i^\delta}{dt} = -\frac{\partial f^\delta}{\partial \hat{x}}(\hat{x}_i)$$



# Algorithms

- And advance the solution in time ...

$$\frac{d\hat{u}_i^\delta}{dt} = -\frac{\partial f^\delta}{\partial \hat{x}}(\hat{x}_i)$$



# Algorithms

- Nature of FR scheme depends on location of solution points, interface flux, correction function
- Can recover a wide range of schemes via judicious choice of correction function [4]
- A multi-parameter family of provably stable FR schemes have previously been identified [5][6]

[4] H.T. Huynh. A flux Reconstruction Approach to High-Order Schemes Including Discontinuous Galerkin Methods. AIAA Paper 2007-4079. 2007

[5] P. E. Vincent, P. Castonguay, A. Jameson. A New Class of High-Order Energy Stable Flux Reconstruction Schemes. Journal of Scientific Computing. 2011

[6] P. E. Vincent, et al. An Extended Range of Stable-Symmetric-Conservative Flux Reconstruction Correction Functions. Computer Methods in Applied Mechanics and Engineering. 2015

# Hardware

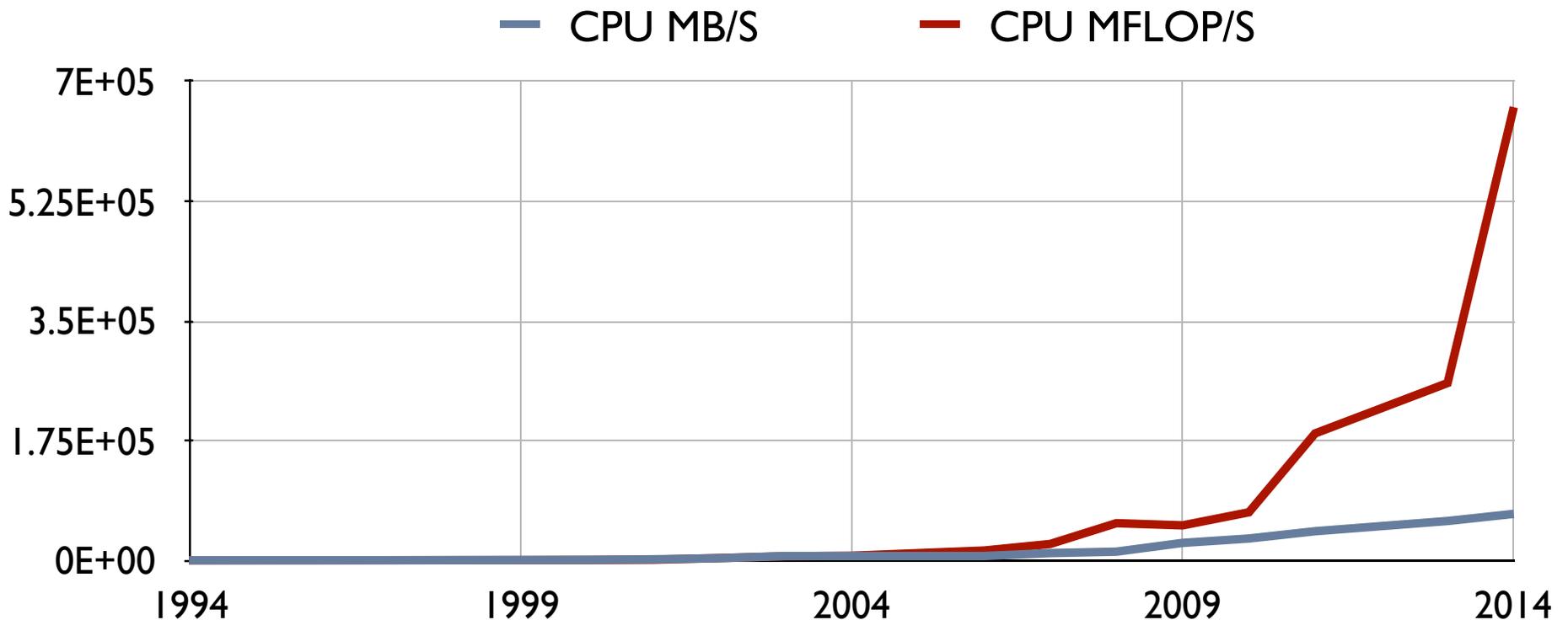


# Hardware



# Hardware

- **FLOPS** increasing faster than **memory bandwidth** [7]



[7] F. D. Witherden et al. PyFR: An Open Source Framework for Solving Advection-Diffusion Type Problems on Streaming Architectures using the Flux Reconstruction Approach. *Computer Physics Communications*. 2014. Data courtesy of Jan Treibig.

# Hardware

- Also **FLOPS** come in parallel ...

# Hardware

- And, different programming languages for different devices ...

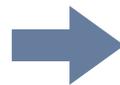
# Hardware

- So a **challenging** environment ...

# Hardware

- But significant **FLOPS** now available if they can be harnessed ...

2.91TFLOPS  
(Double Precision)



# PyFR

Flux Reconstruction  
PyFR  
Modern Hardware



# PyFR

- Features (v1.4.0)

<b>Governing Equations</b>	Compressible Euler Compressible Navier Stokes
<b>Spatial Discretisation</b>	Arbitrary order FR on mixed unstructured grids (tris, quads, hexes, tets, prisms, pyramids)
<b>Temporal Discretisation</b>	Range of explicit Runge-Kutta schemes
<b>Platforms</b>	CPU clusters (C-OpenMP-MPI) Nvidia GPU clusters (CUDA-MPI) AMD GPU clusters (OpenCL-MPI) Xeon Phi Clusters (PyMIC-MPI)
<b>Precision</b>	Single Double
<b>Input</b>	Gmsh, CGNS
<b>Output</b>	Paraview

# PyFR

- Features (v1.4.0)

<b>Governing Equations</b>	Compressible Euler Compressible Navier Stokes
<b>Spatial Discretisation</b>	Arbitrary order FR on mixed unstructured grids (tris, quads, hexes, tets, prisms, pyramids)
<b>Temporal Discretisation</b>	Range of explicit Runge-Kutta schemes
<b>Platforms</b>	CPU clusters (C-OpenMP-MPI) Nvidia GPU clusters (CUDA-MPI) AMD GPU clusters (OpenCL-MPI) Xeon Phi Clusters (PyMIC-MPI)
<b>Precision</b>	Single Double
<b>Input</b>	Gmsh, CGNS
<b>Output</b>	Paraview

# PyFR

- Python Outer Layer (Hardware Independent)

Python Outer Layer  
(Hardware Independent)

- Setup
- Distributed memory parallelism
- Outer 'for' loop and calls to Hardware Specific Kernels

# PyFR

- Need to generate the Hardware Specific Kernels

Python Outer Layer  
(Hardware Independent)

- Setup
- Distributed memory parallelism
- Outer 'for' loop and calls to Hardware Specific Kernels

# PyFR

- Two **types** of kernel are required ...

## Python Outer Layer (Hardware Independent)

- Setup
- Distributed memory parallelism
- Outer 'for' loop and calls to Hardware Specific Kernels

## Matrix Multiply Kernels

- Data interpolation/  
extrapolation  
etc.

## Point-Wise Nonlinear Kernels

- Flux functions,  
Riemann  
solvers etc.

# PyFR

- For matrix multiply kernels it is pretty easy ...

## Python Outer Layer (Hardware Independent)

- Setup
- Distributed memory parallelism
- Outer 'for' loop and calls to Hardware Specific Kernels

## Matrix Multiply Kernels

- Data interpolation/extrapolation etc.

## Point-Wise Nonlinear Kernels

- Flux functions, Riemann solvers etc.

Use DGEMM from  
vendor supplied  
BLAS

# PyFR

- Harder for point-wise nonlinear kernels ...

## Python Outer Layer (Hardware Independent)

- Setup
- Distributed memory parallelism
- Outer 'for' loop and calls to Hardware Specific Kernels

## Matrix Multiply Kernels

- Data interpolation/  
extrapolation  
etc.

## Point-Wise Nonlinear Kernels

- Flux functions,  
Riemann  
solvers etc.

Use DGEMM from  
vendor supplied  
BLAS

Pass Mako  
derived kernel  
templates through  
Mako derived  
templating engine

# PyFR

- These can now be called

## Python Outer Layer (Hardware Independent)

- Setup
- Distributed memory parallelism
- Outer 'for' loop and calls to Hardware Specific Kernels

## Matrix Multiply Kernels

- Data interpolation/extrapolation etc.

## Point-Wise Nonlinear Kernels

- Flux functions, Riemann solvers etc.

## C/OpenMP Hardware Specific Kernels



## CUDA Hardware Specific Kernels



## PyMIC Hardware Specific Kernels



## OpenCL Hardware Specific Kernels

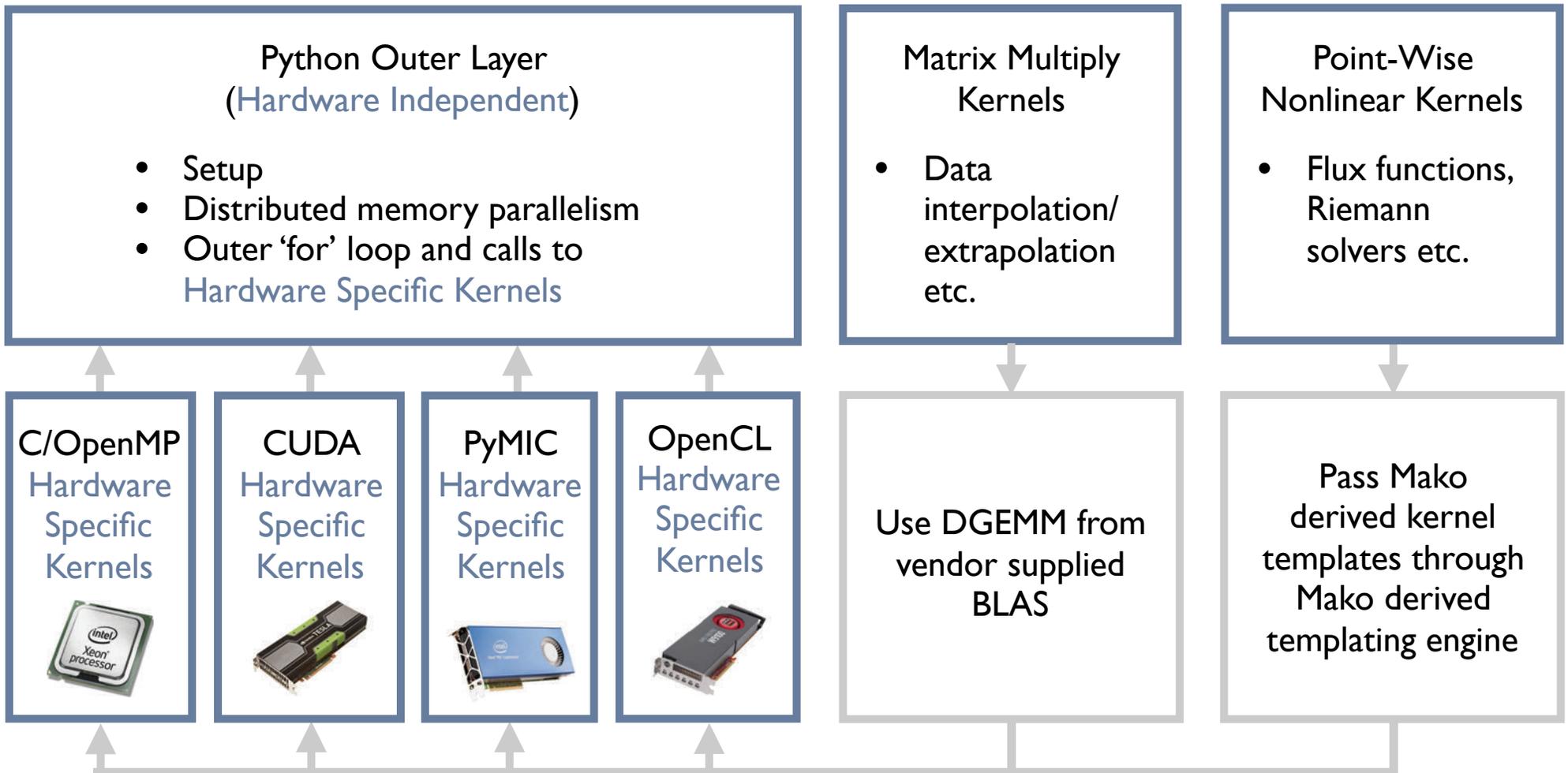


Use DGEMM from  
vendor supplied  
BLAS

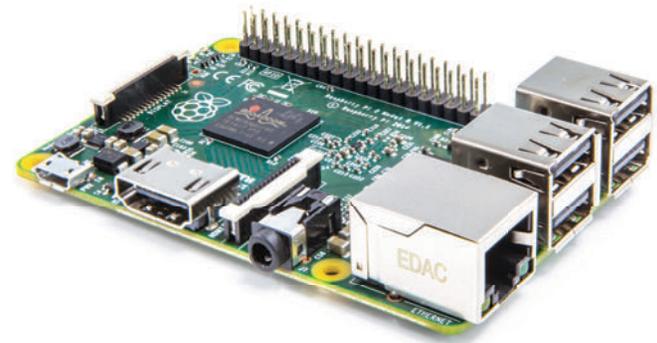
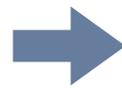
Pass Mako  
derived kernel  
templates through  
Mako derived  
templating engine

# PyFR

- These can now be called

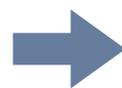


# PyFR



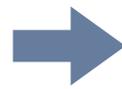
Raspberry Pi

# PyFR



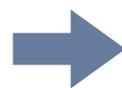
Macbook Pro

# PyFR



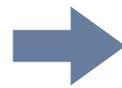
Heterogeneous  
Workstation

# PyFR



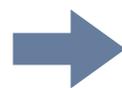
Wilkes (UK)

# PyFR



Piz Daint  
(Switzerland)

# PyFR



Titan (USA)

# PyFR

- ~8.0k lines of code

# PyFR

- Open source '3 Clause New Style BSD License'



# PyFR



- Website: [www.pyfr.org](http://www.pyfr.org)
- Twitter: [@PyFR\\_Solver](https://twitter.com/PyFR_Solver)
- Paper: [Computer Physics Communications \[8\]](#)

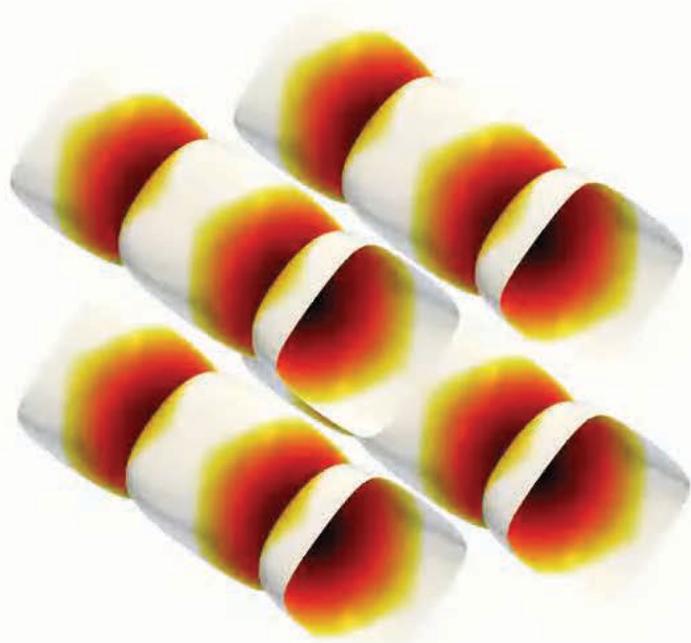
# Results

- 3D Taylor-Green vortex breakdown
- Compare with spectral DNS results of van Rees et al. [9]

[9] W. M. van Rees, A. Leonard, D.I. Pullin, and P. Koumoutsakos. A Comparison of Vortex and Pseudo-Spectral Methods for the Simulation of Periodic Vortical Flows at High Reynolds Numbers. *Journal of Computational Physics*, 2011

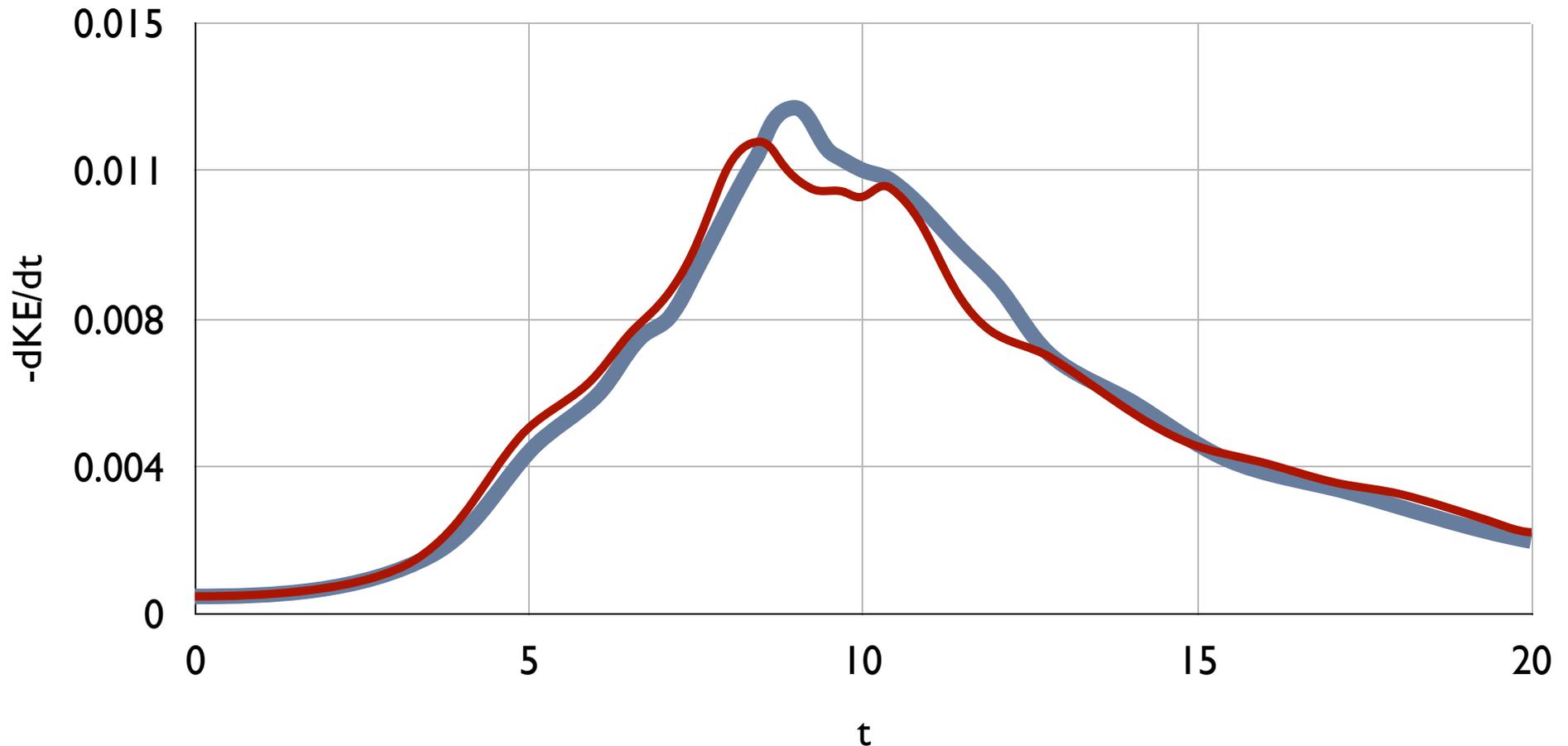
# Results

- A movie ...



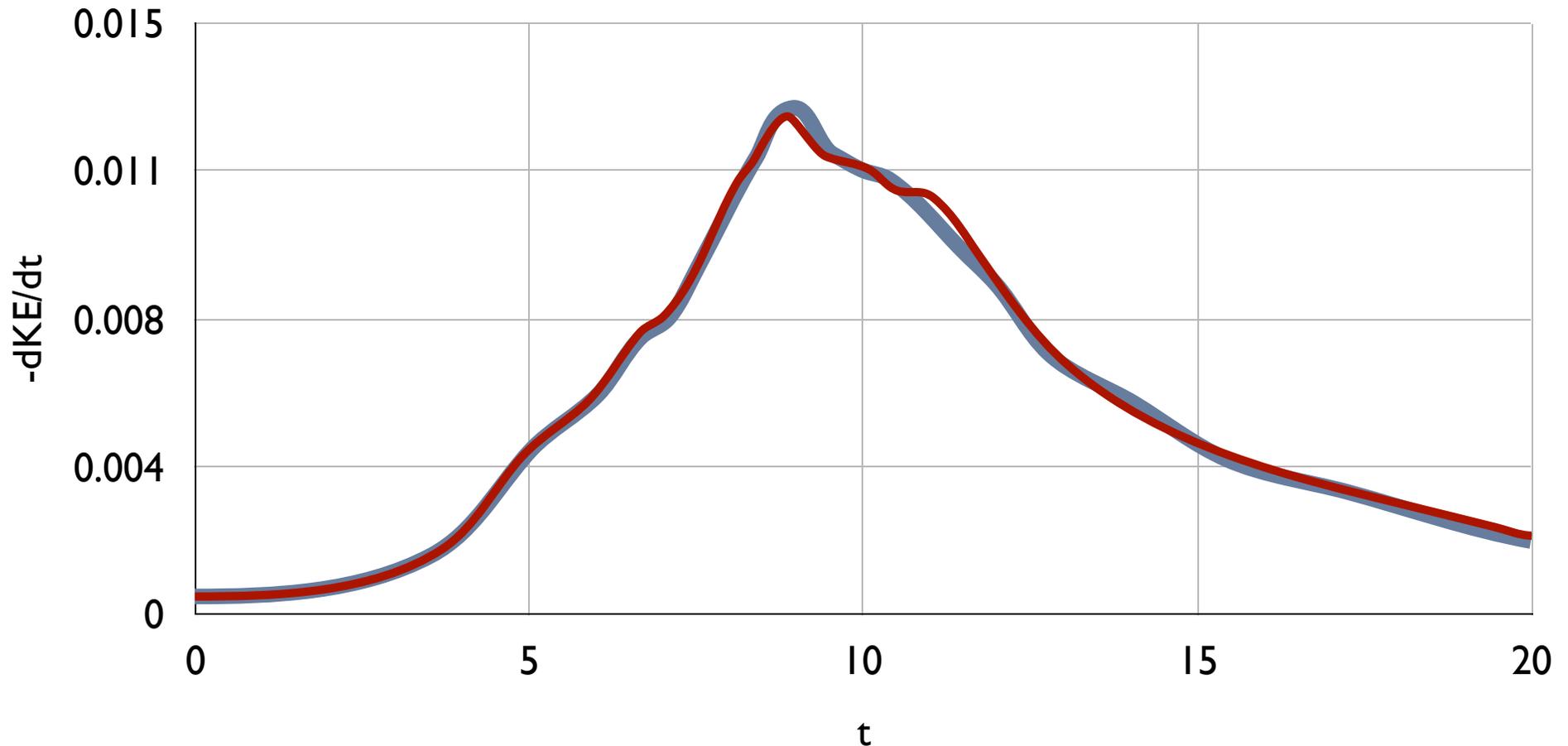
# Results

- van Rees et al. spectral DNS + PyFR (2nd order hex)



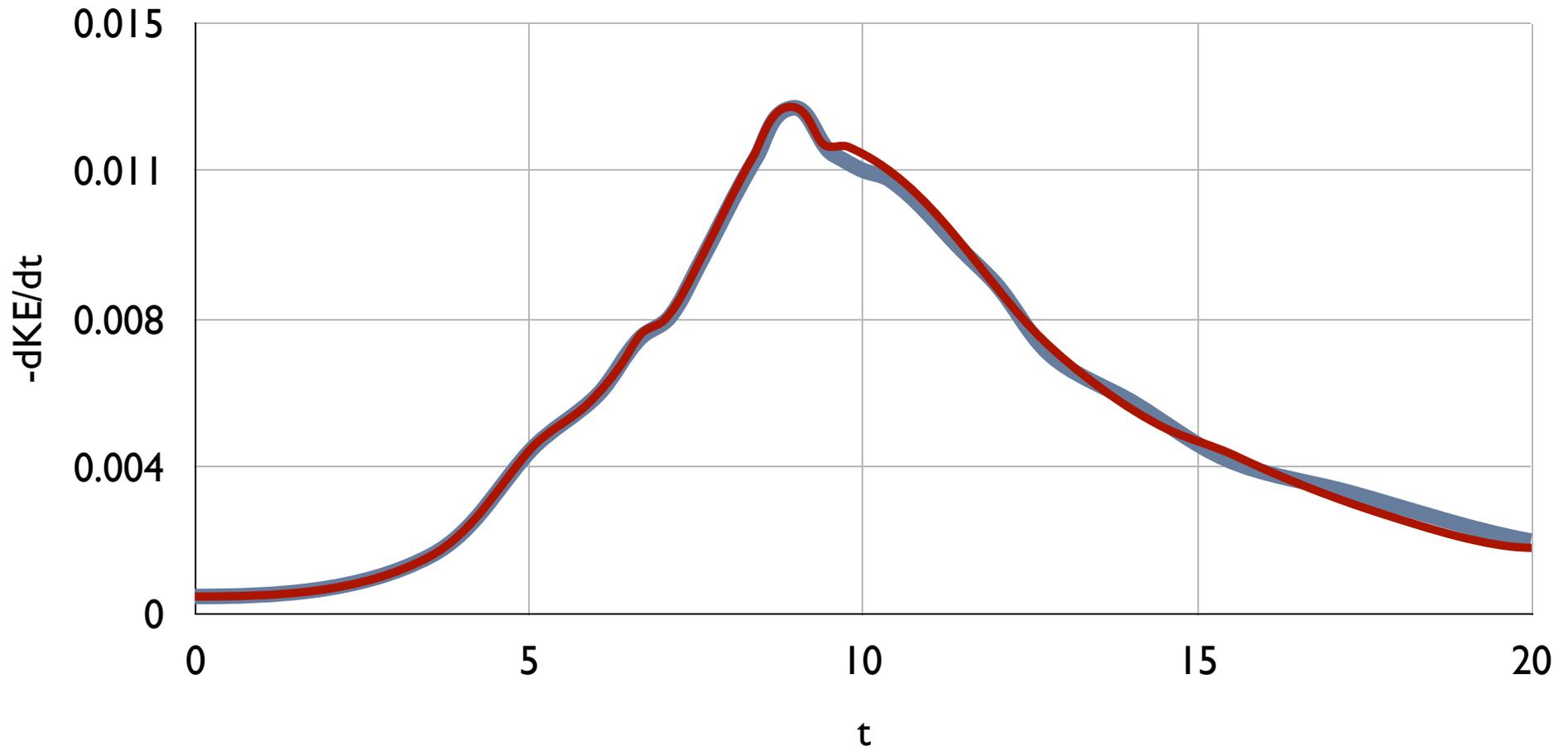
# Results

- van Rees et al. spectral DNS + PyFR (3rd order hex)



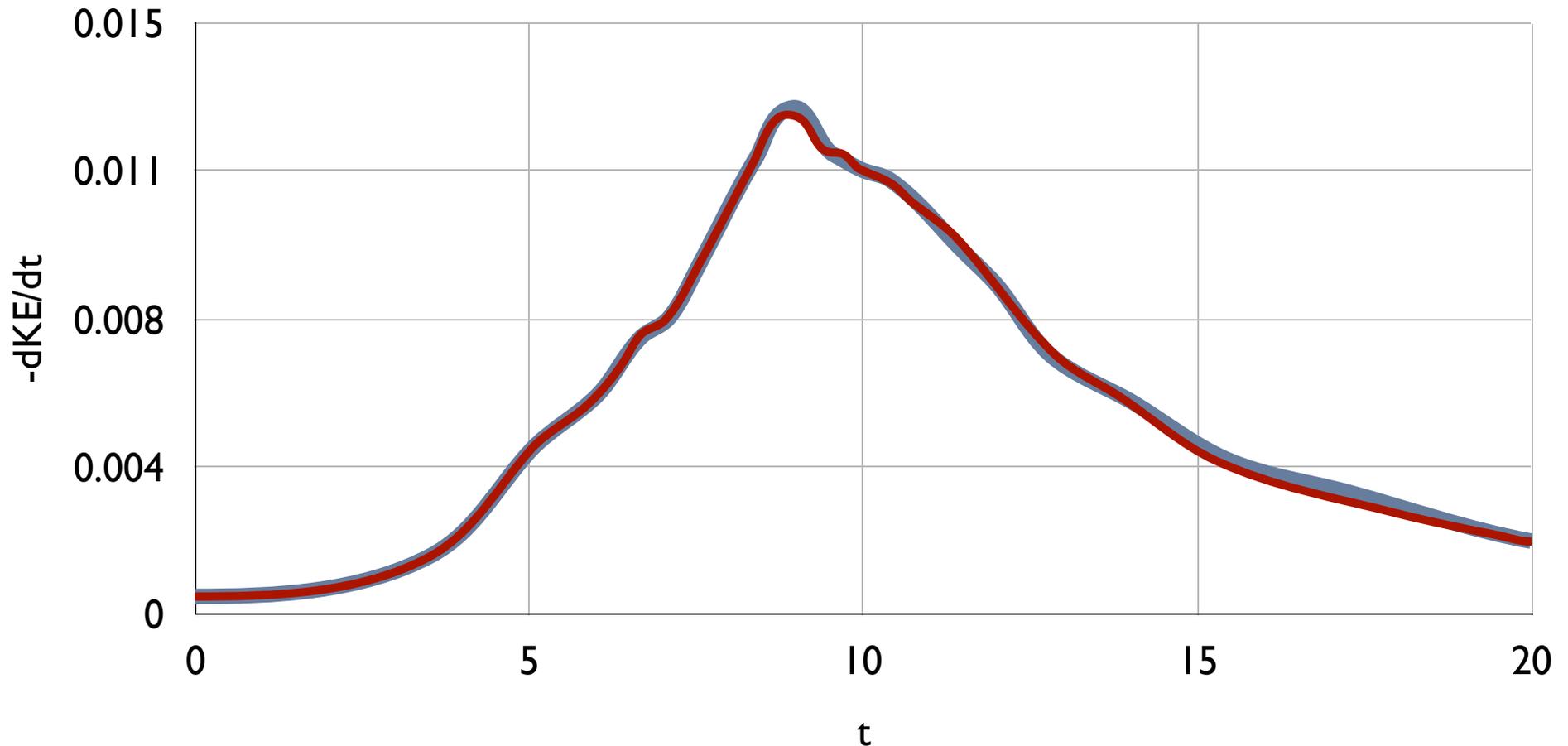
# Results

- van Rees et al. spectral DNS + PyFR (4th order hex)



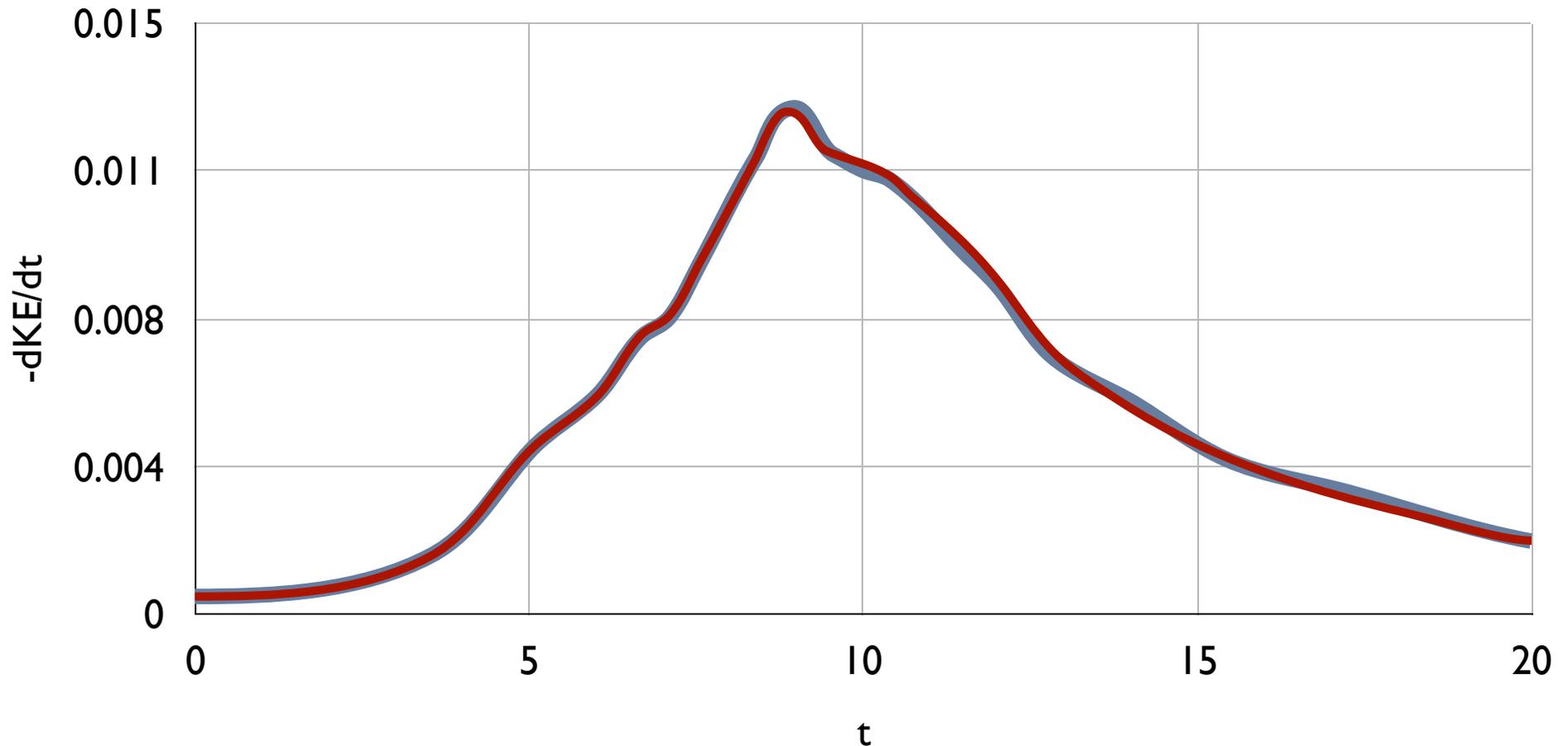
# Results

- van Rees et al. spectral DNS + PyFR (5th order hex)



# Results

- van Rees et al. spectral DNS + PyFR (6th order hex)

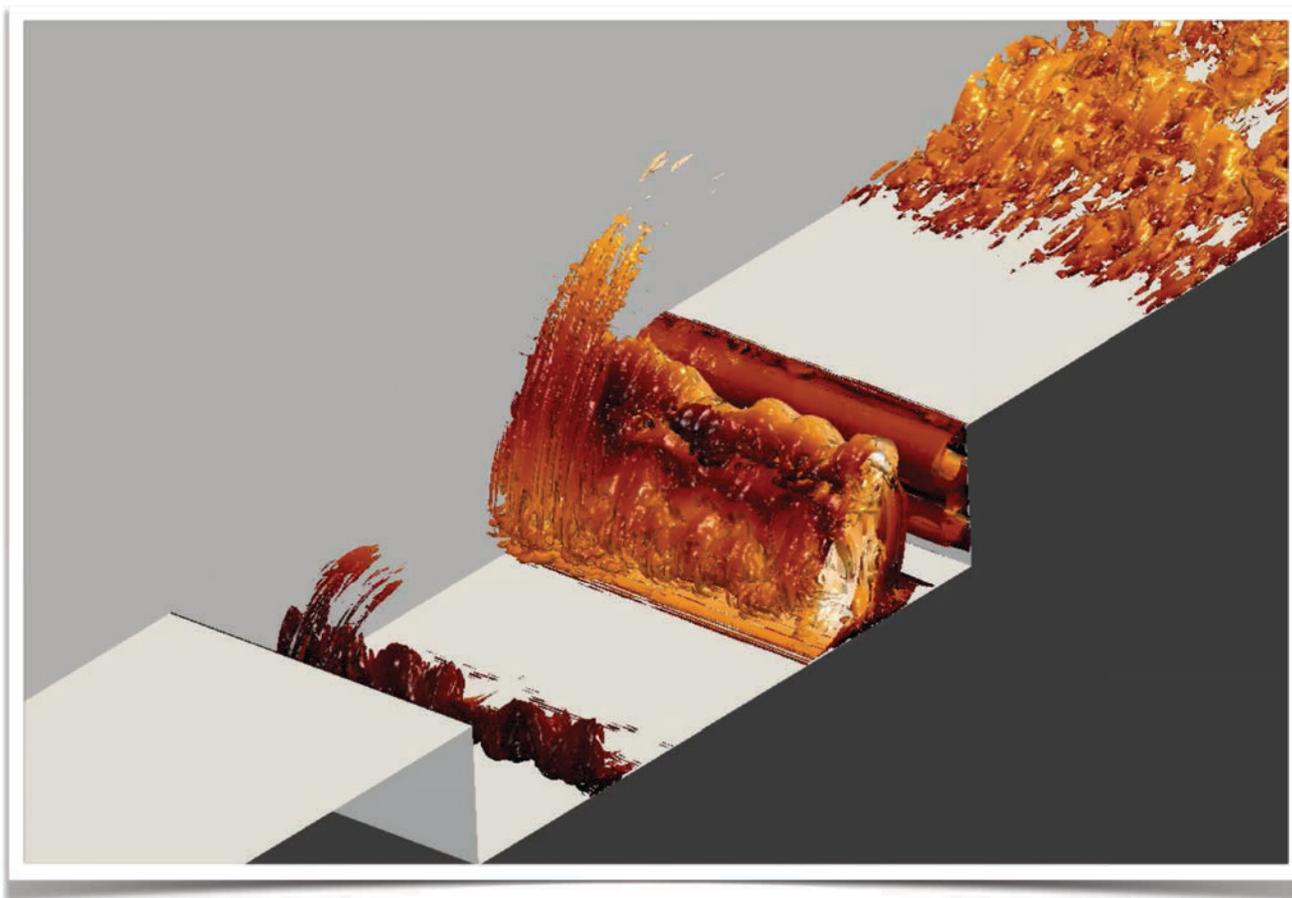


# Results

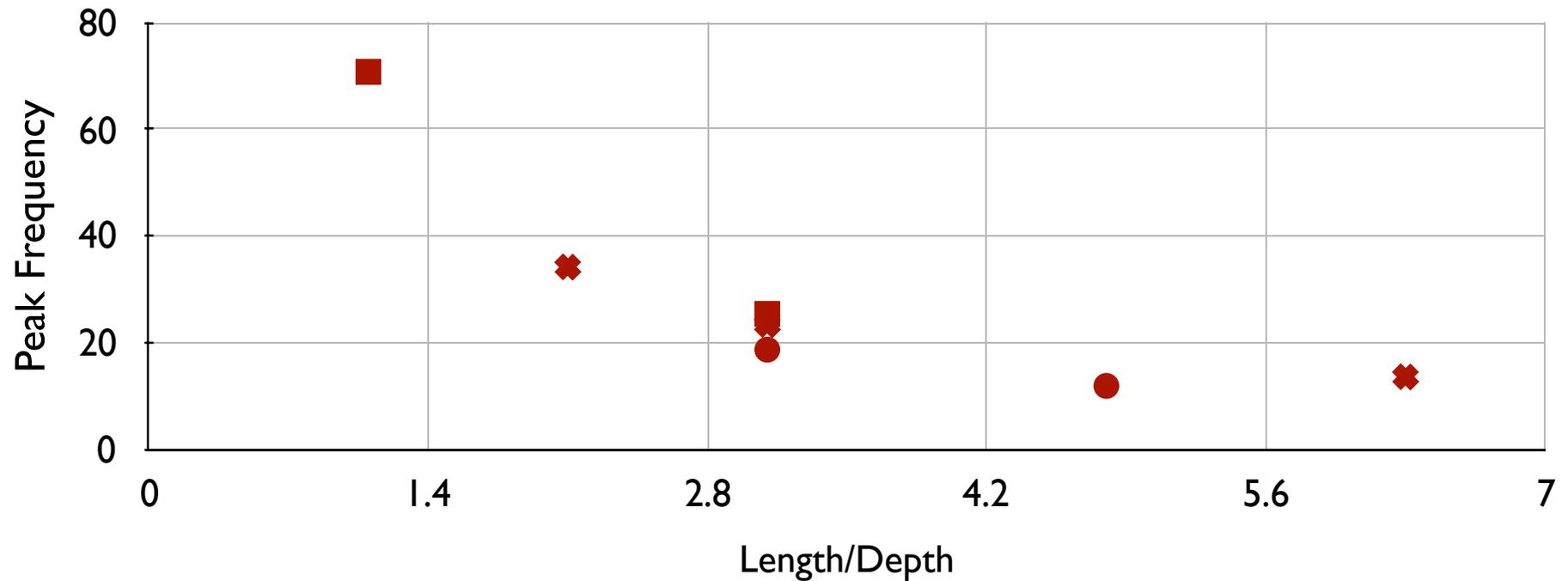
- Flow over a cavity
- $Re \sim 30,000$
- $Ma \sim 0.5-0.82$

# Results

- A movie ...



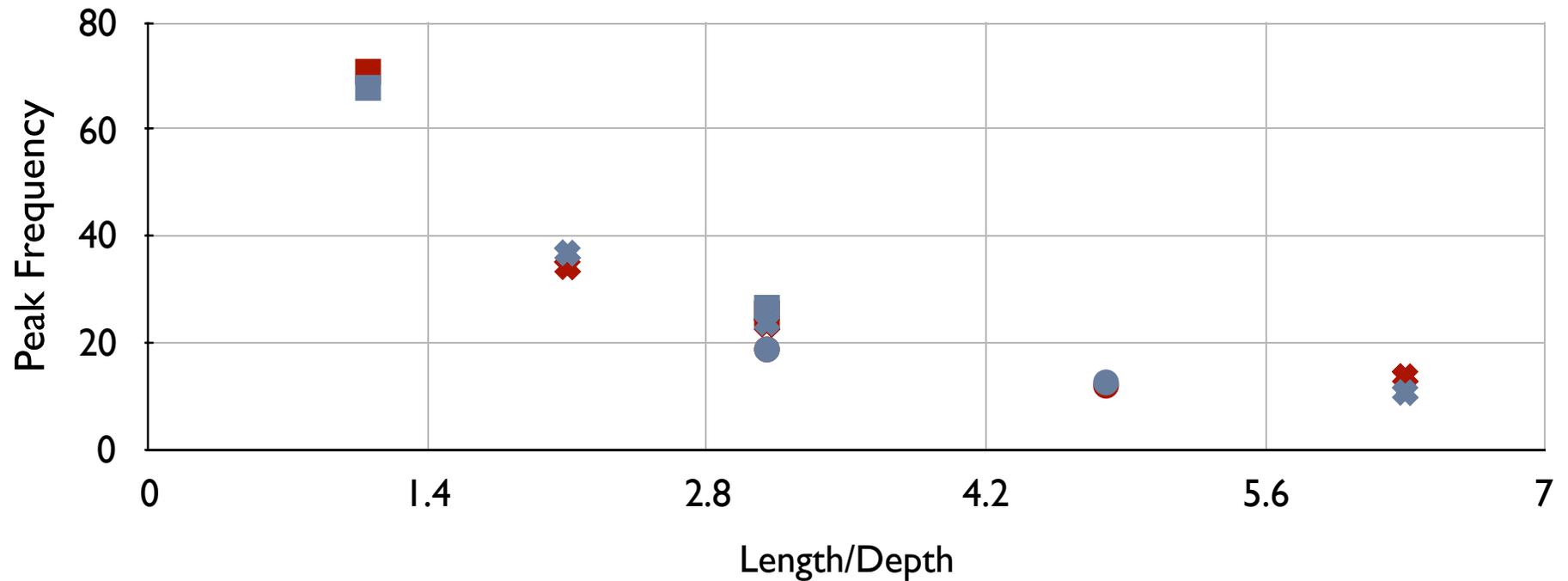
# Results



- Experiment Ma 0.5 [10]
- ✕ Experiment Ma 0.7 [10]
- Experiment Ma 0.82 [10]

[10] K. Krishnamurty. Acoustic radiation from two-dimensional rectangular cutouts in aerodynamic surfaces. National Advisory Committee for Aeronautics (NACA) TN3487. 1955

# Results



- PyFR Ma 0.5
- × PyFR Ma 0.7
- PyFR Ma 0.82
- Experiment Ma 0.5 [10]
- × Experiment Ma 0.7 [10]
- Experiment Ma 0.82 [10]

[10] K. Krishnamurty. Acoustic radiation from two-dimensional rectangular cutouts in aerodynamic surfaces. National Advisory Committee for Aeronautics (NACA) TN3487. 1955

# Results

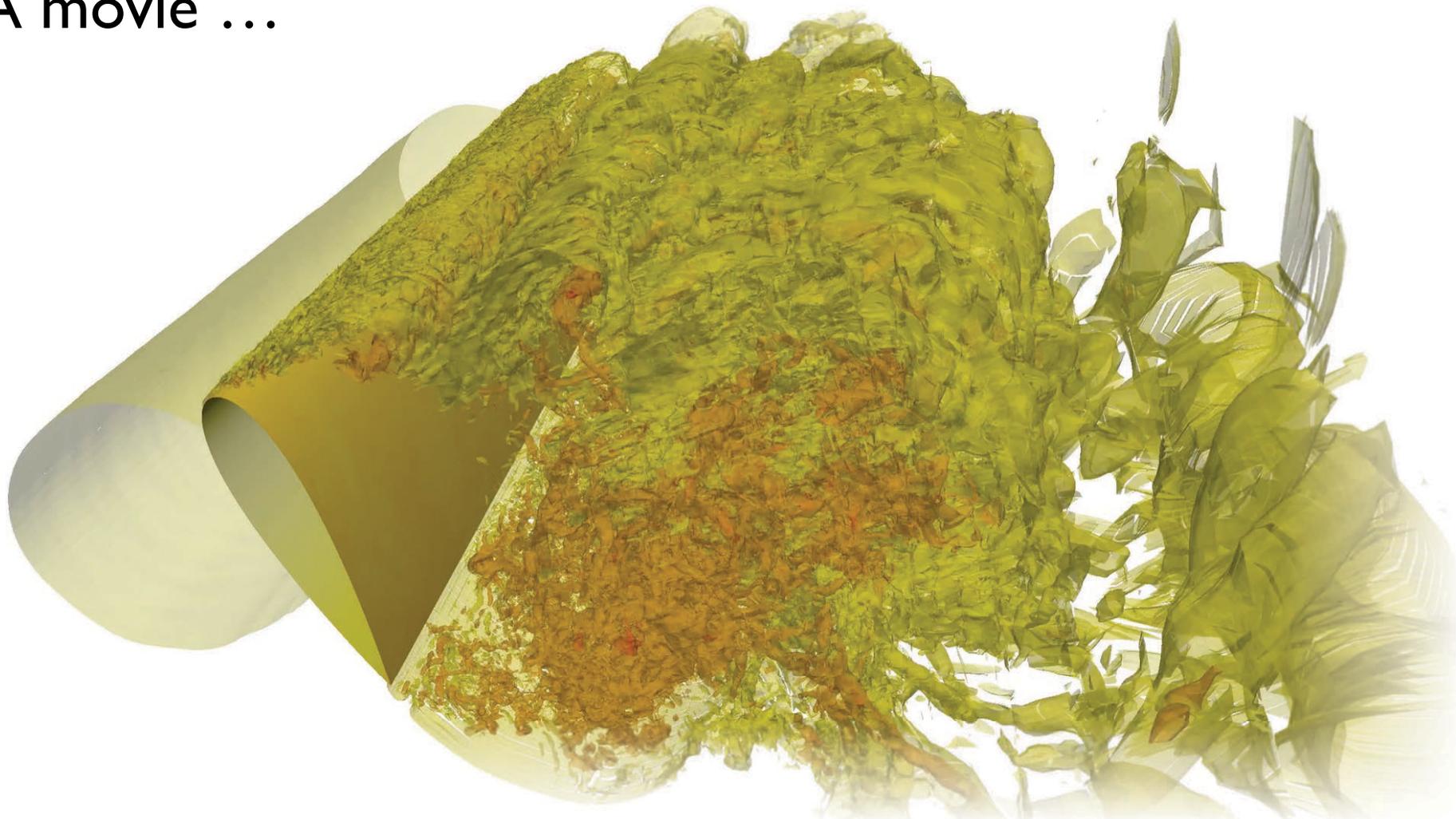
- Flow over a **NACA 0021** at 60 degree AoA
- $Re = 270,000$
- $Ma = 0.1$
- Compare with Swalwell and DESider [11][12]

[11] K. Swalwell. The Effect of Turbulence on Stall of Horizontal Axis Wind Turbines. PhD Thesis. 2005.

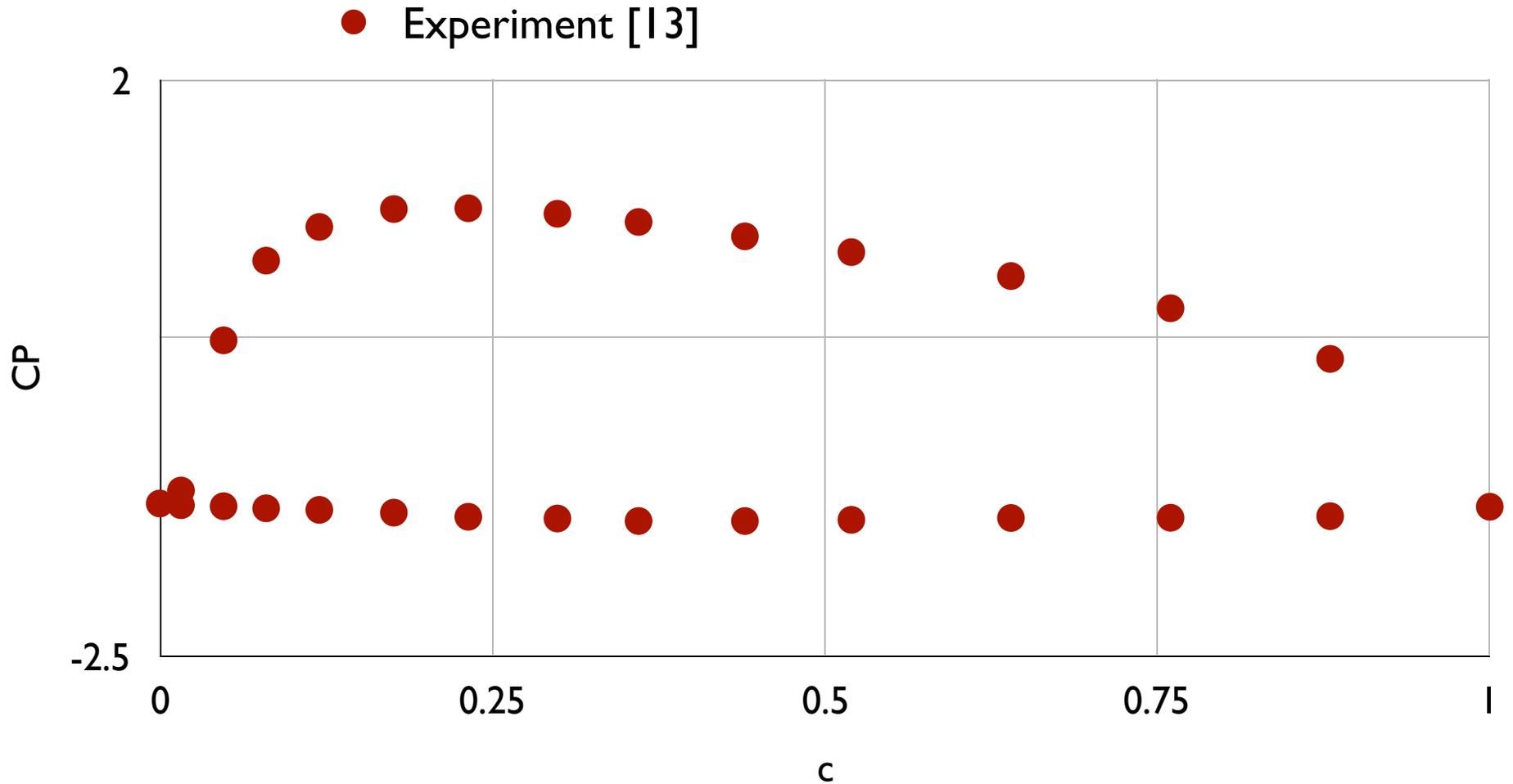
[12] W. Haase, M. Braza, A. Revell. DESider A European Effort on Hybrid RANS-LES Modelling. 2009.

# Results

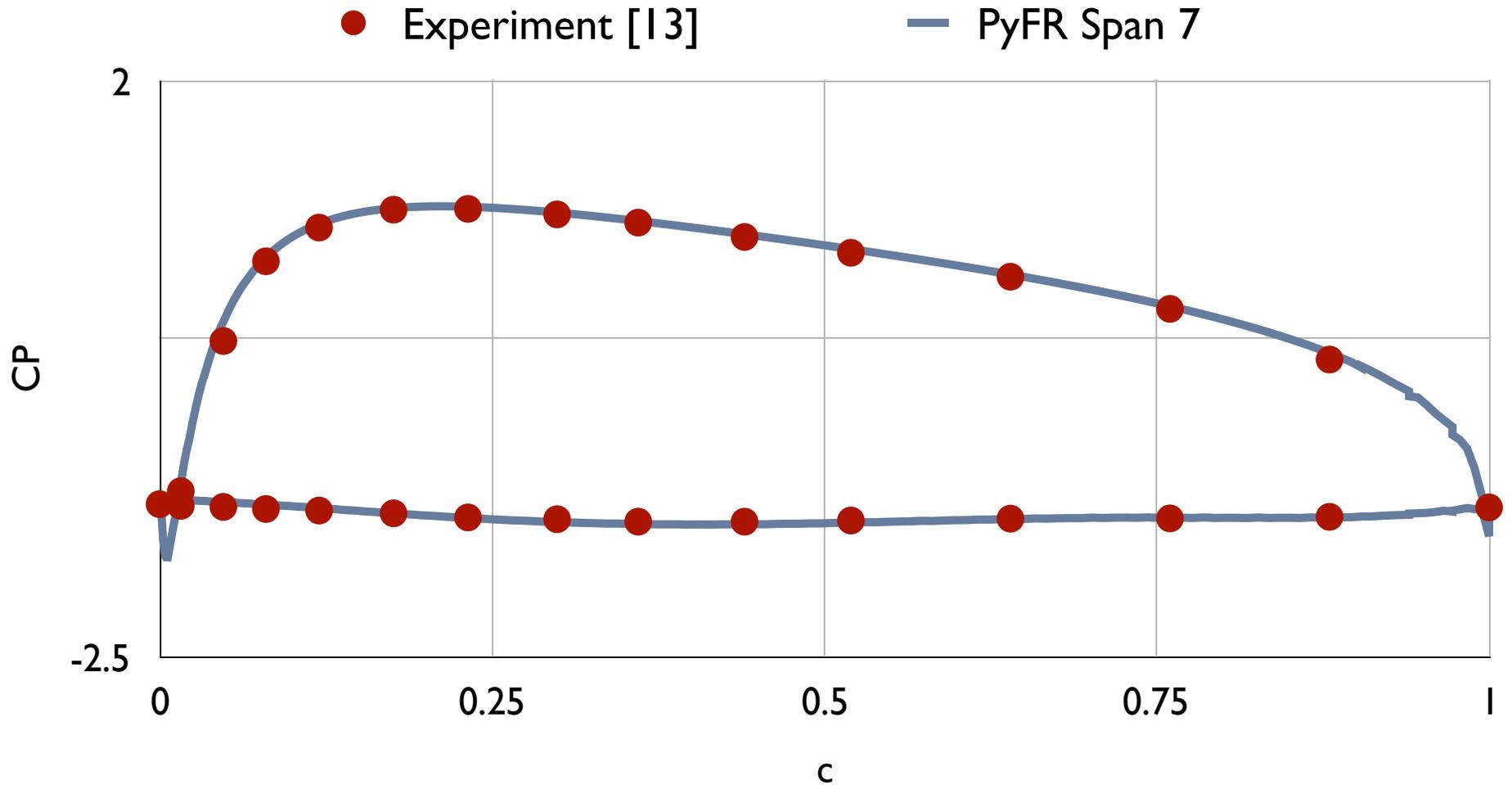
- A movie ...



# Results

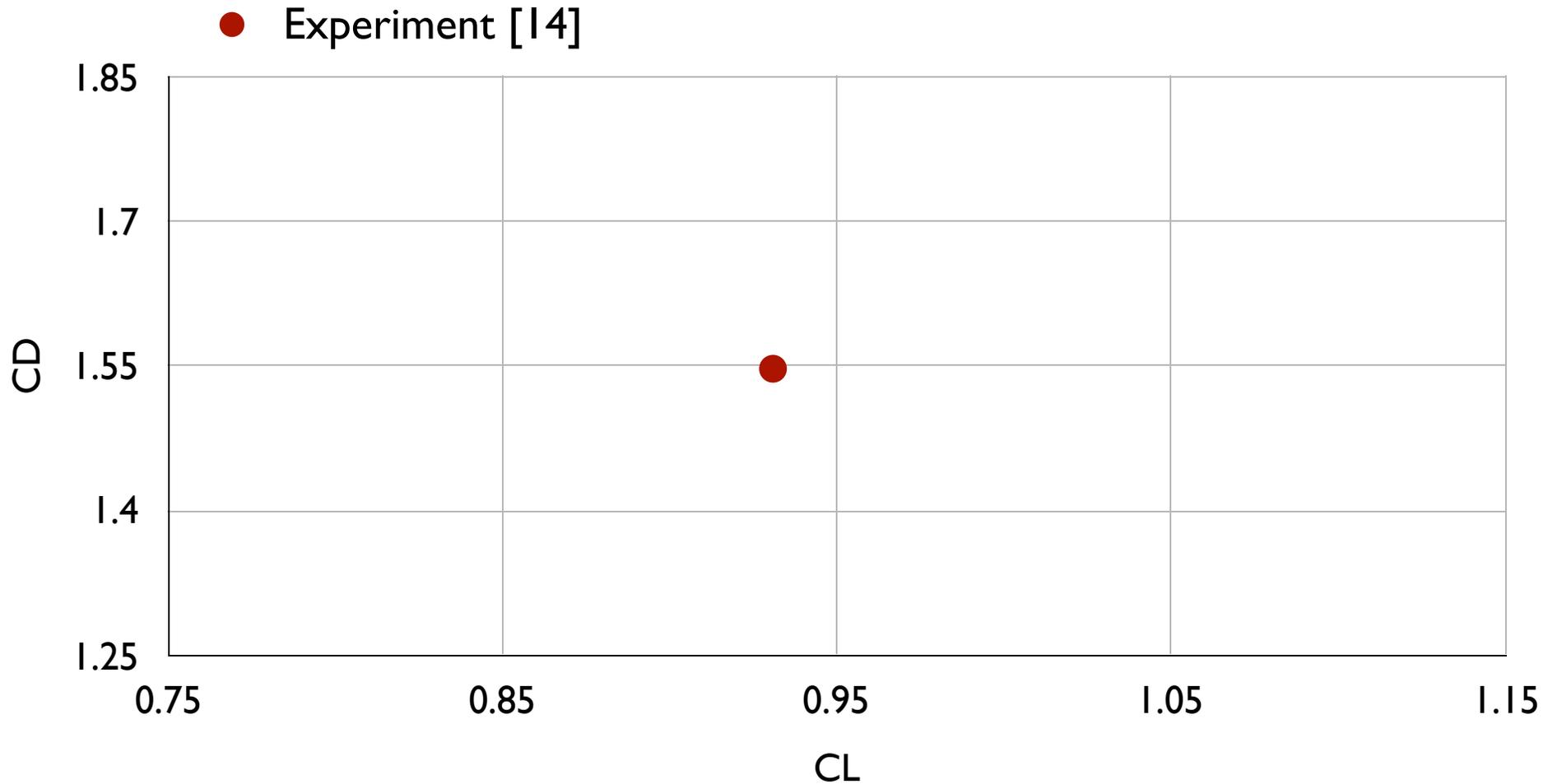


# Results



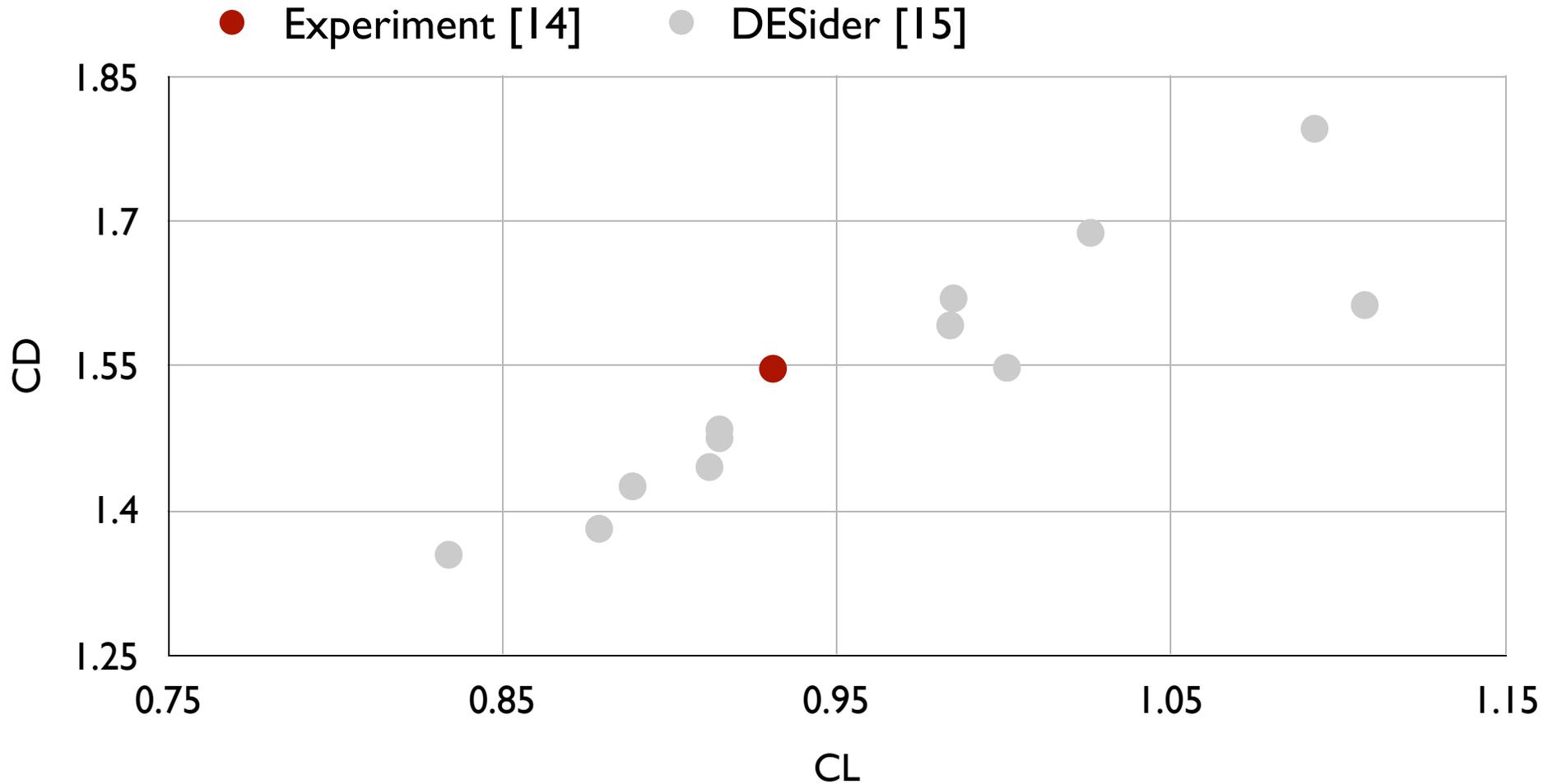
[13] K. Swalwell. The Effect of Turbulence on Stall of Horizontal Axis Wind Turbines. PhD Thesis. 2005.

# Results



- [14] K. Swalwell. The Effect of Turbulence on Stall of Horizontal Axis Wind Turbines. PhD Thesis. 2005.  
[15] W. Haase et al. DESider A European Effort on Hybrid RANS-LES Modelling. 2009.

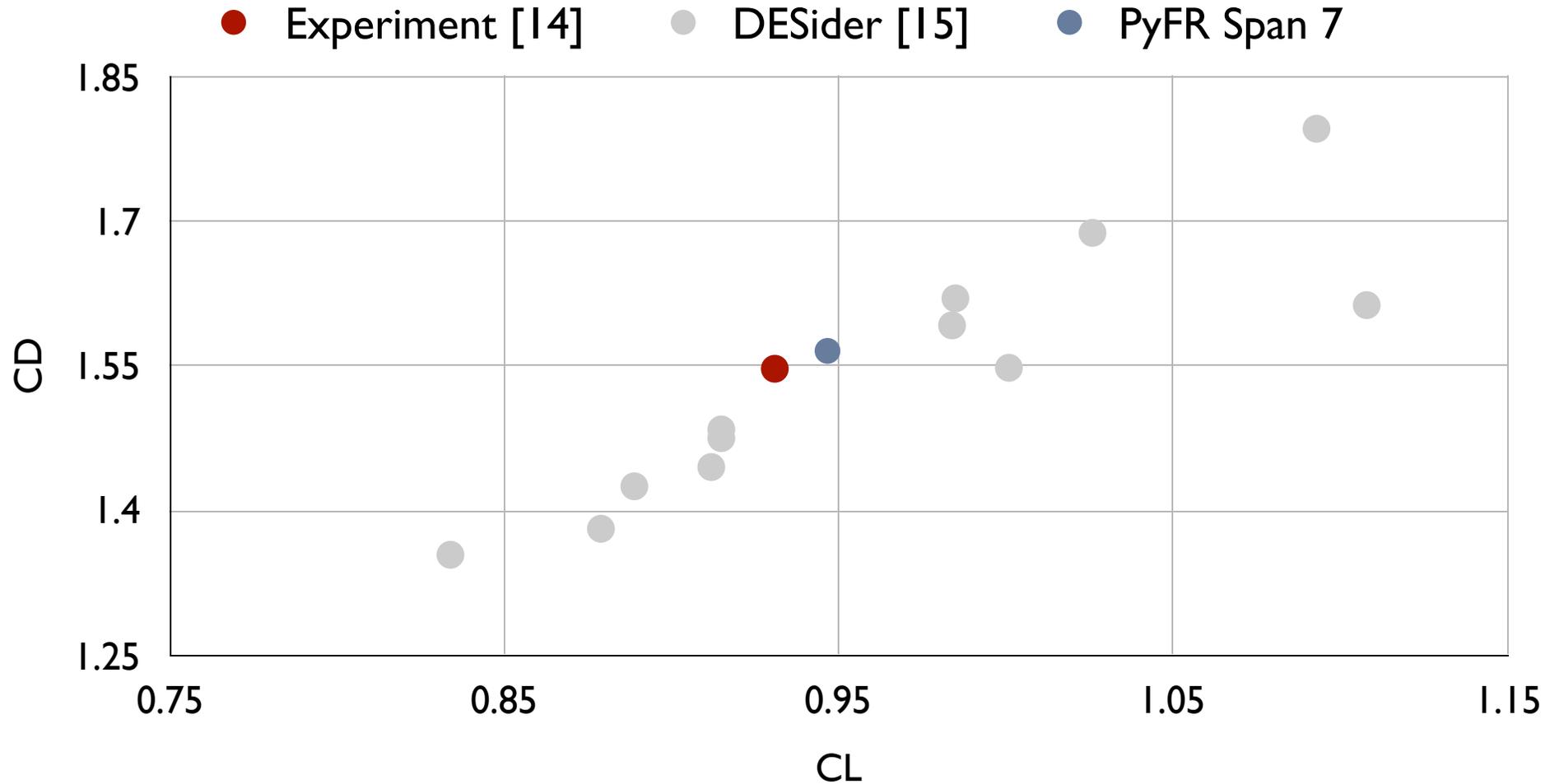
# Results



[14] K. Swalwell. The Effect of Turbulence on Stall of Horizontal Axis Wind Turbines. PhD Thesis. 2005.

[15] W. Haase et al. DESider A European Effort on Hybrid RANS-LES Modelling. 2009.

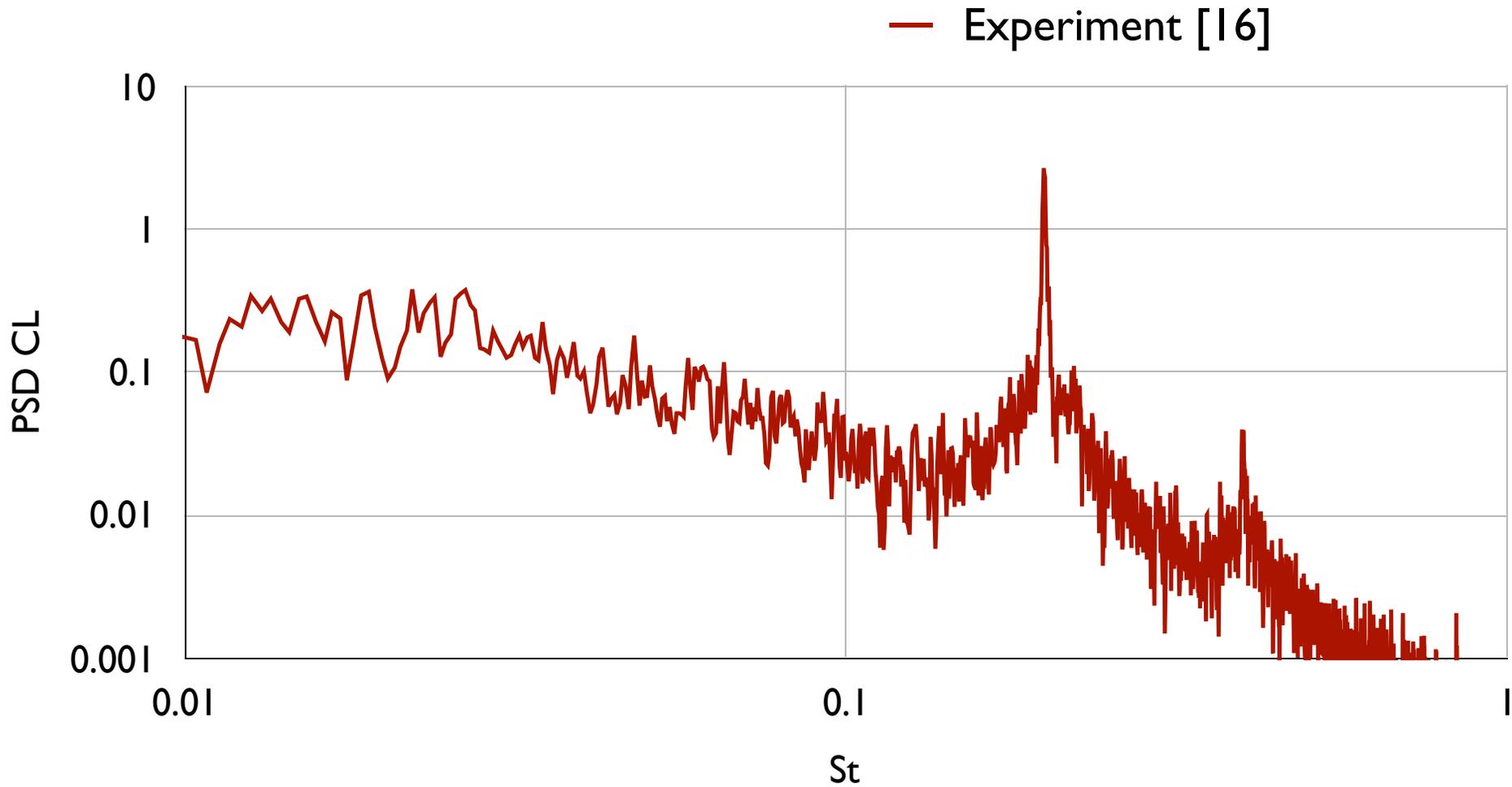
# Results



[14] K. Swalwell. The Effect of Turbulence on Stall of Horizontal Axis Wind Turbines. PhD Thesis. 2005.

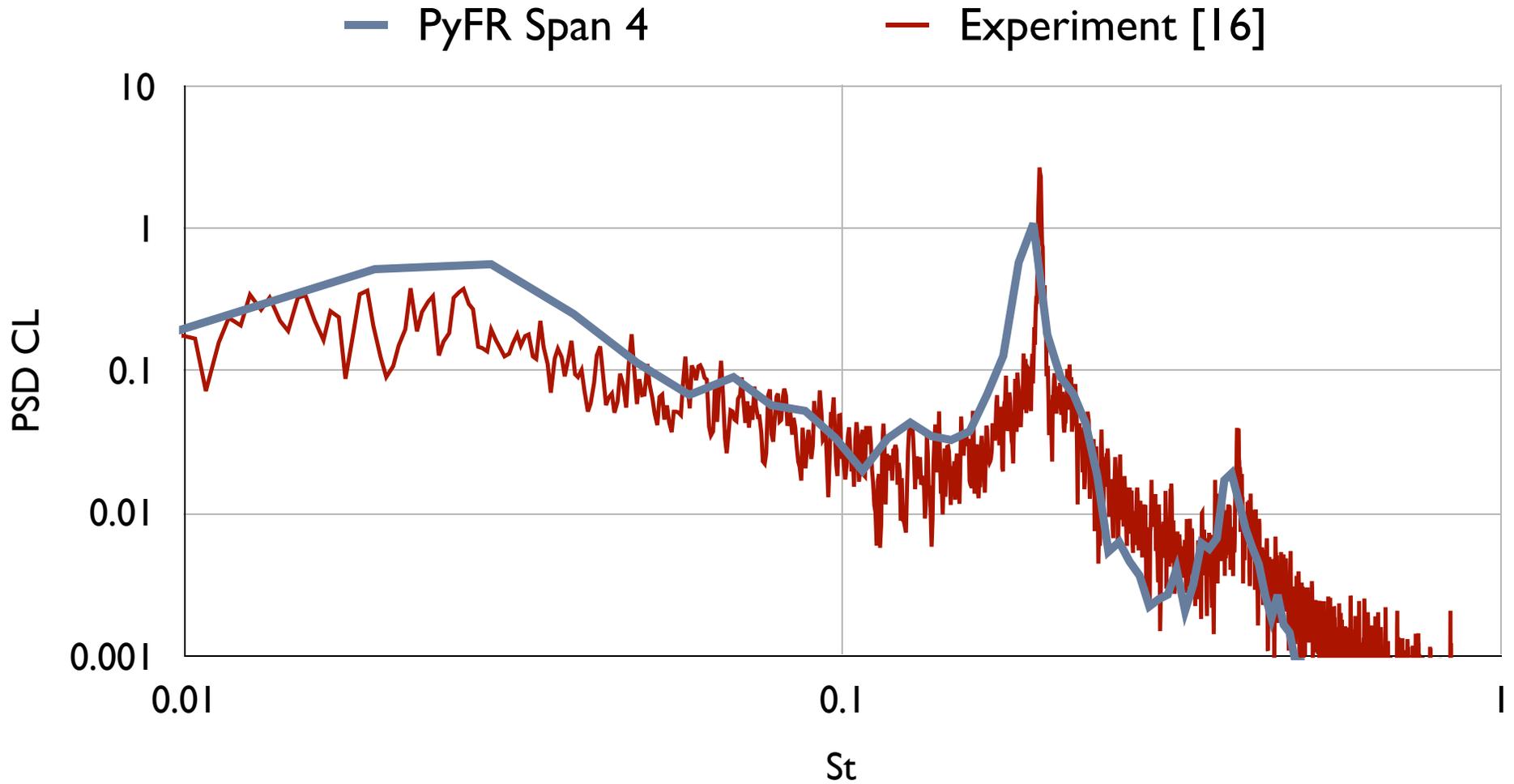
[15] W. Haase et al. DESider A European Effort on Hybrid RANS-LES Modelling. 2009.

# Results



[16] K. Swalwell. The Effect of Turbulence on Stall of Horizontal Axis Wind Turbines. PhD Thesis. 2005.

# Results



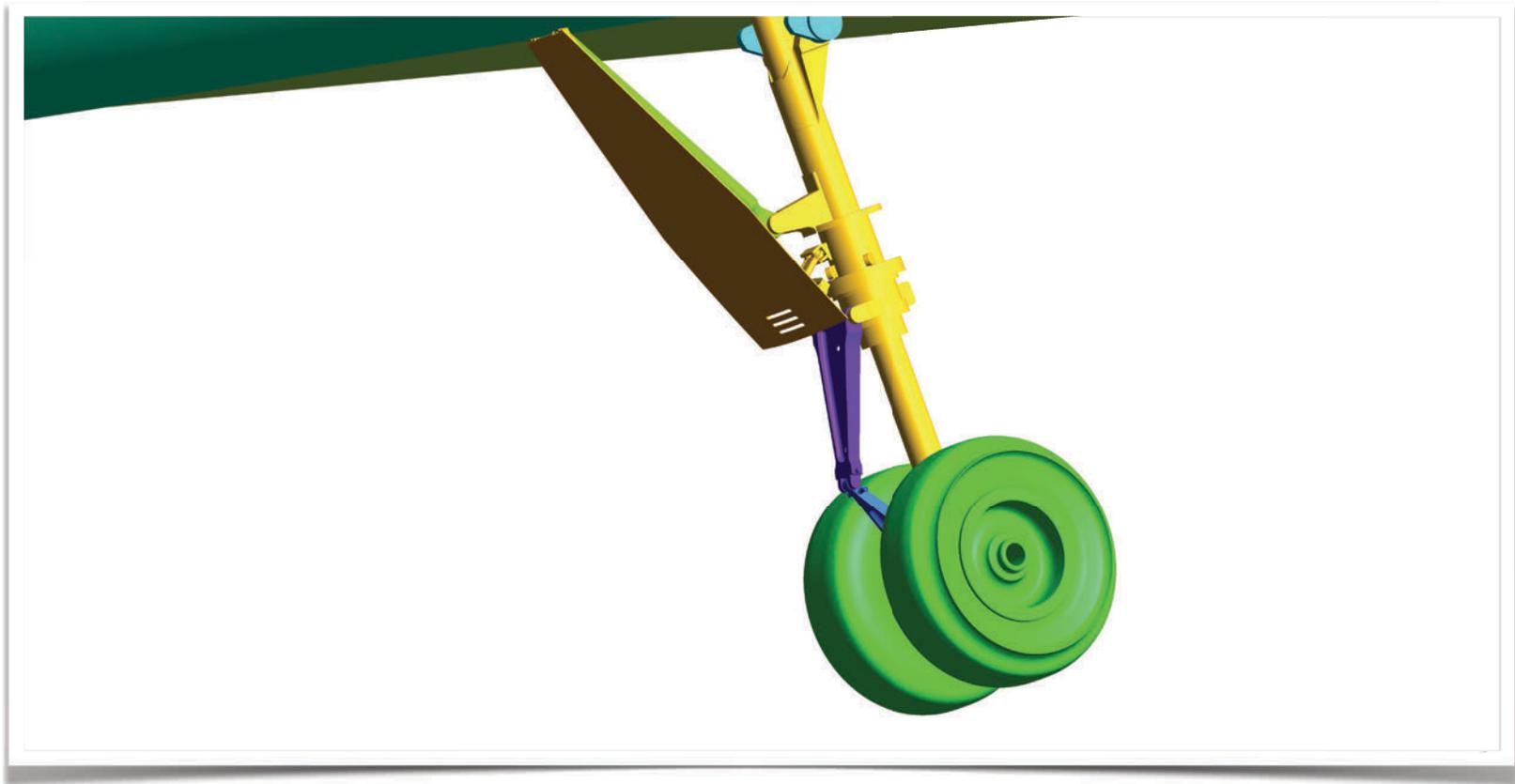
[16] K. Swalwell. The Effect of Turbulence on Stall of Horizontal Axis Wind Turbines. PhD Thesis. 2005.

# Results

- Landing gear

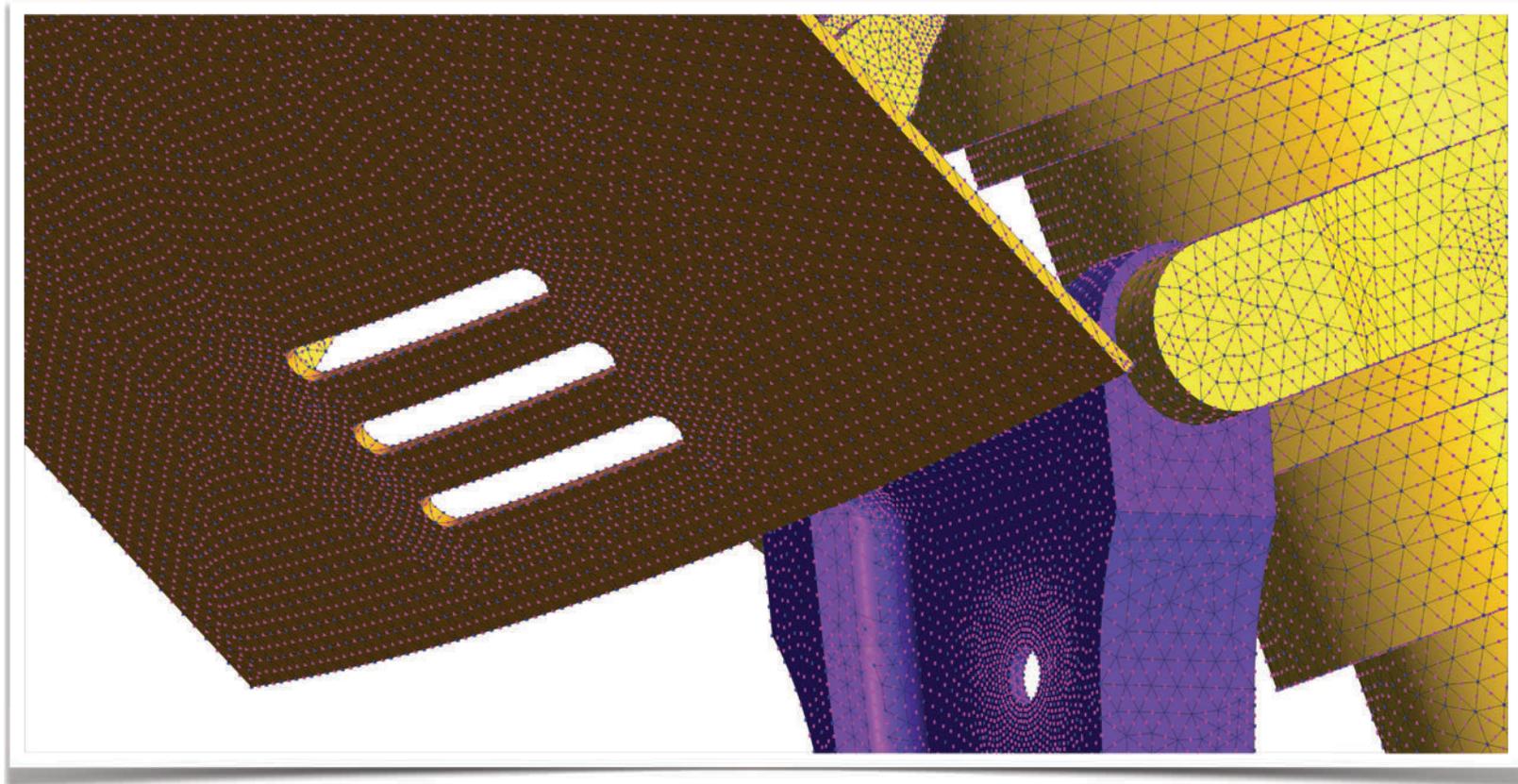
## Results

- BANC workshop **PDCC-NLG** test case (courtesy of Mehdi Khorrami at NASA Langley)



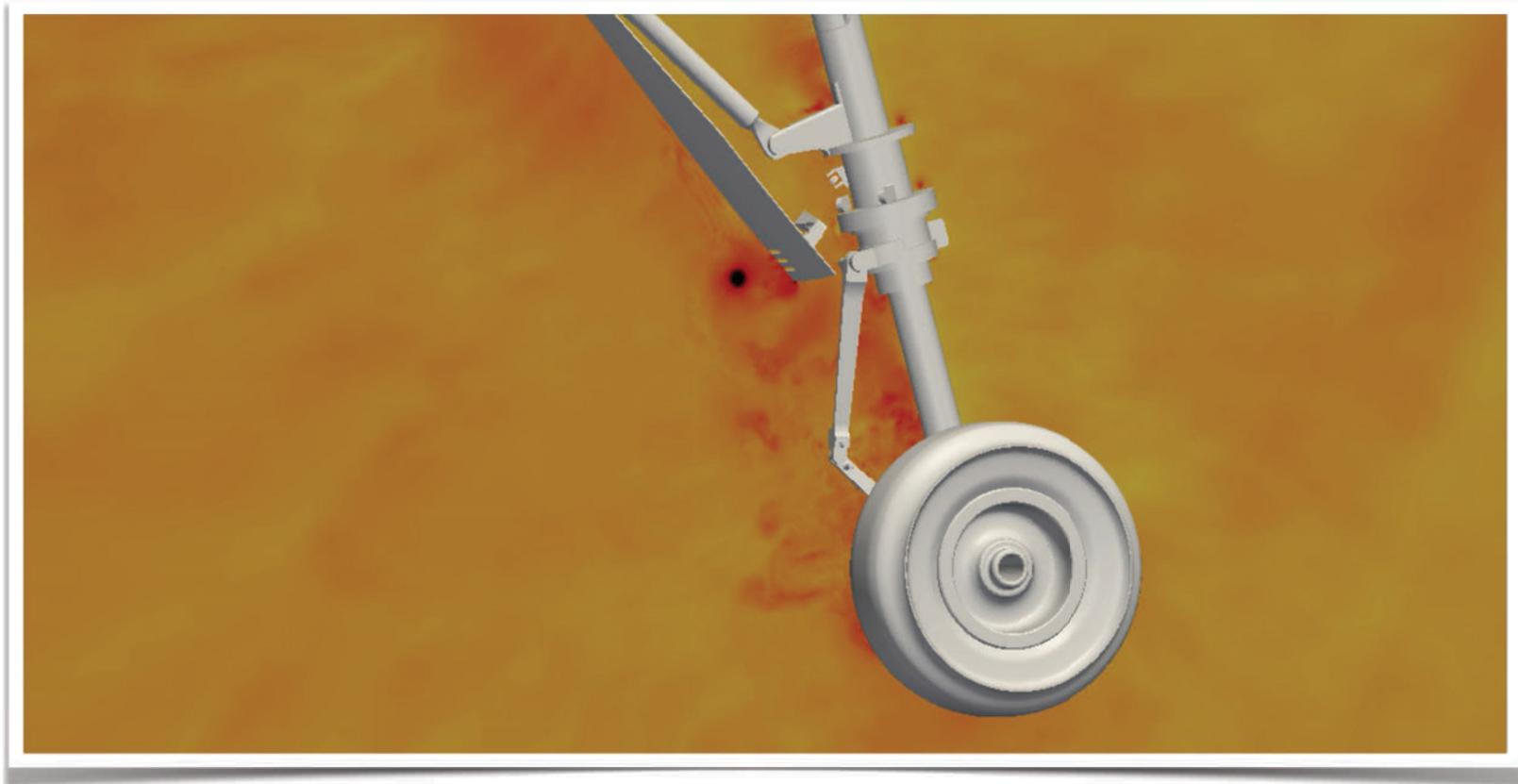
# Results

- **Quadratically curved unstructured tetrahedral mesh** (courtesy of Steve Karman at Pointwise)



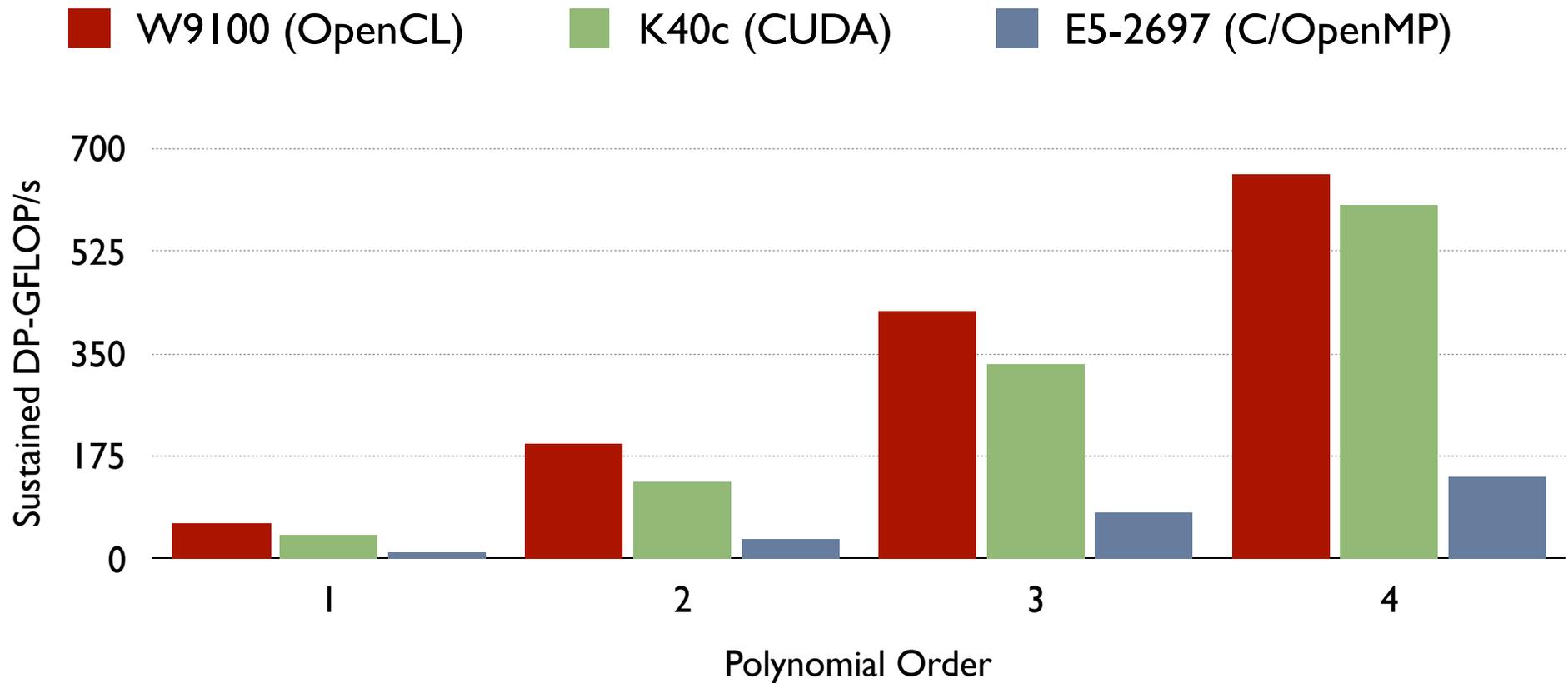
# Results

- Preliminary **second-order accurate** flow solution obtained using PyFR



# Performance

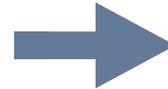
- **Single-node** on prism/tetrahedral mesh [17]



[17] F.D.Witherden et al. Heterogeneous Computing on Mixed Unstructured Grids with PyFR. Computers and Fluids. 2015.

# Performance

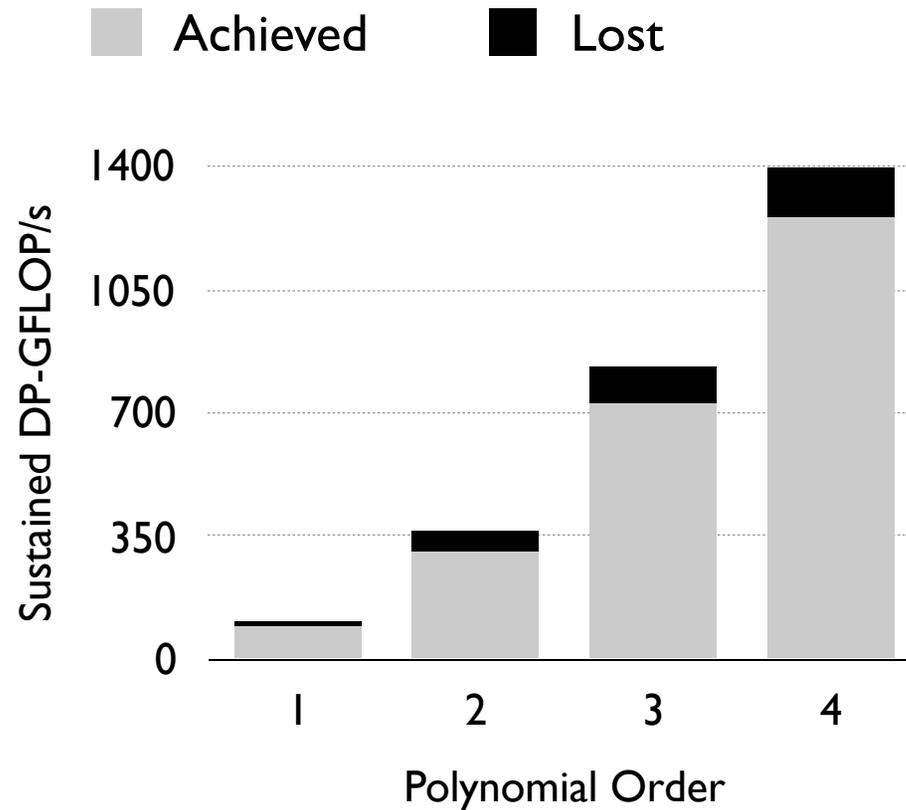
- Performance on heterogeneous systems



# Performance

- Multi-node heterogeneous on prism/tetrahedral mesh

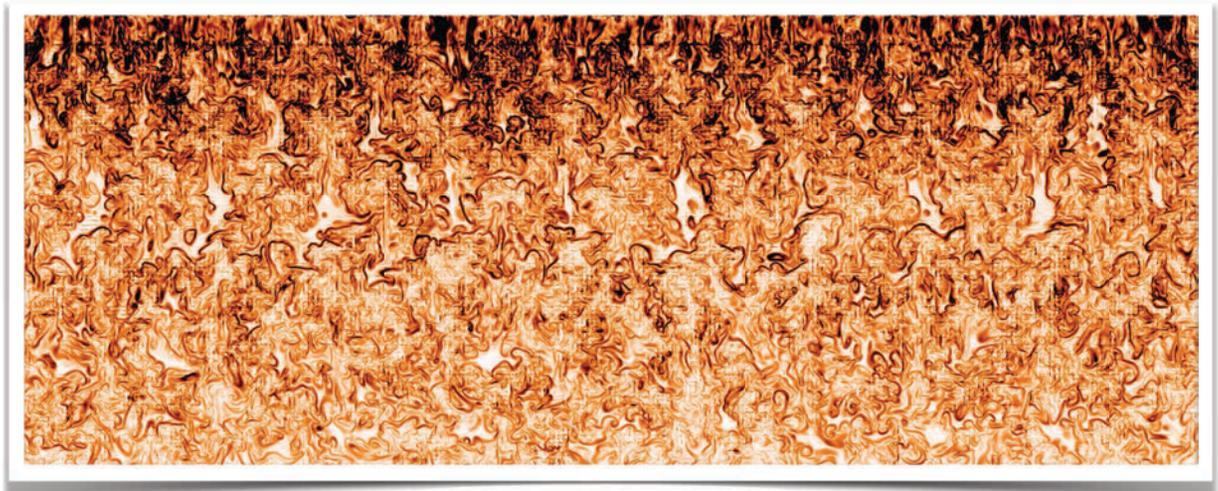
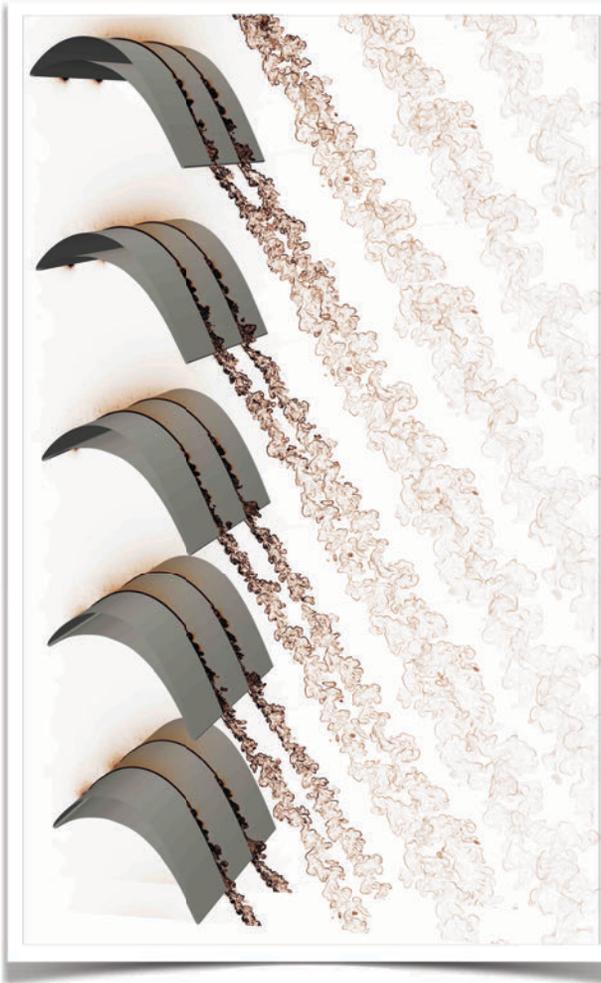
[17]



[17] F. D. Witherden et al. Heterogeneous Computing on Mixed Unstructured Grids with PyFR. Computers and Fluids. 2015.

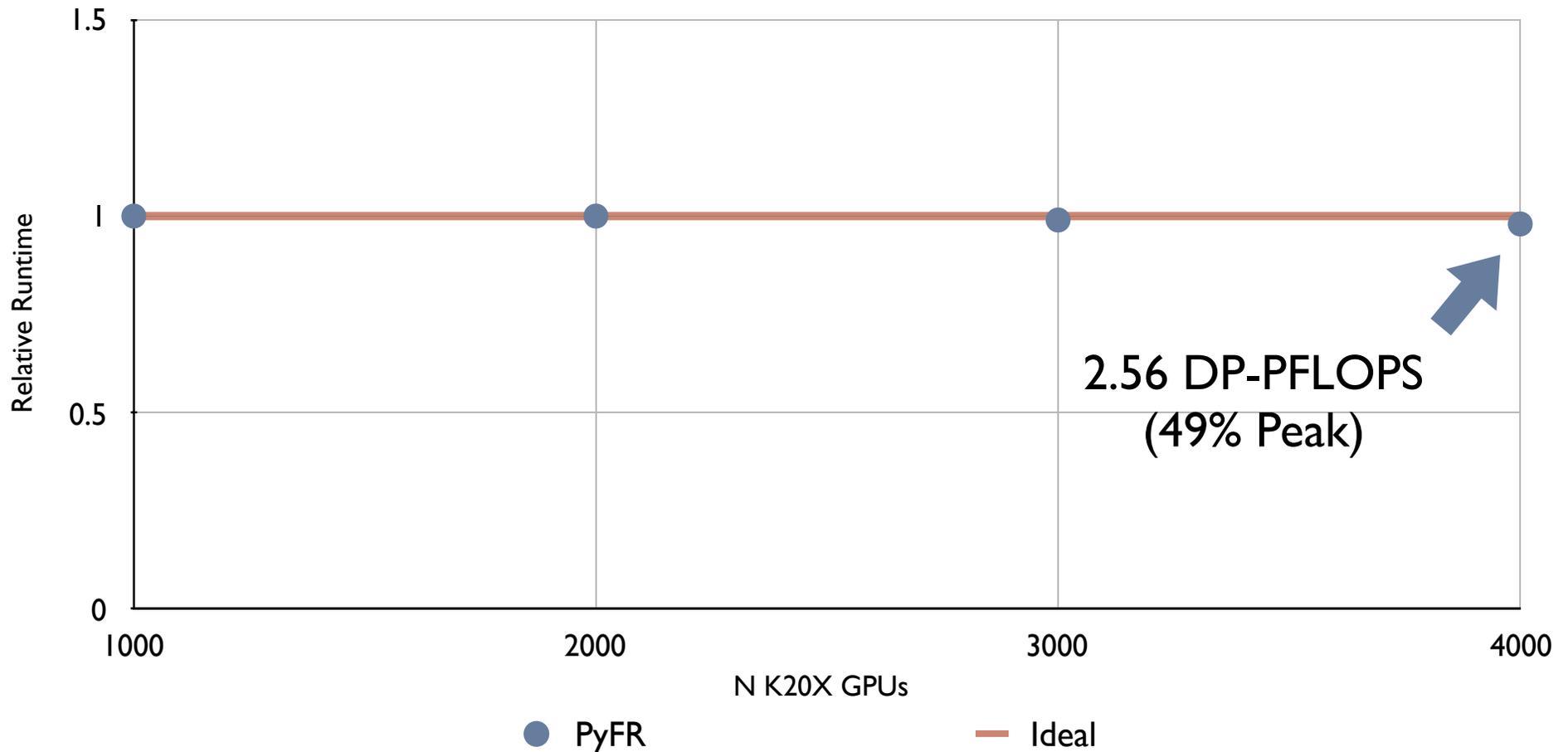
# Performance

- Full-scale T106D LPT cascade test case



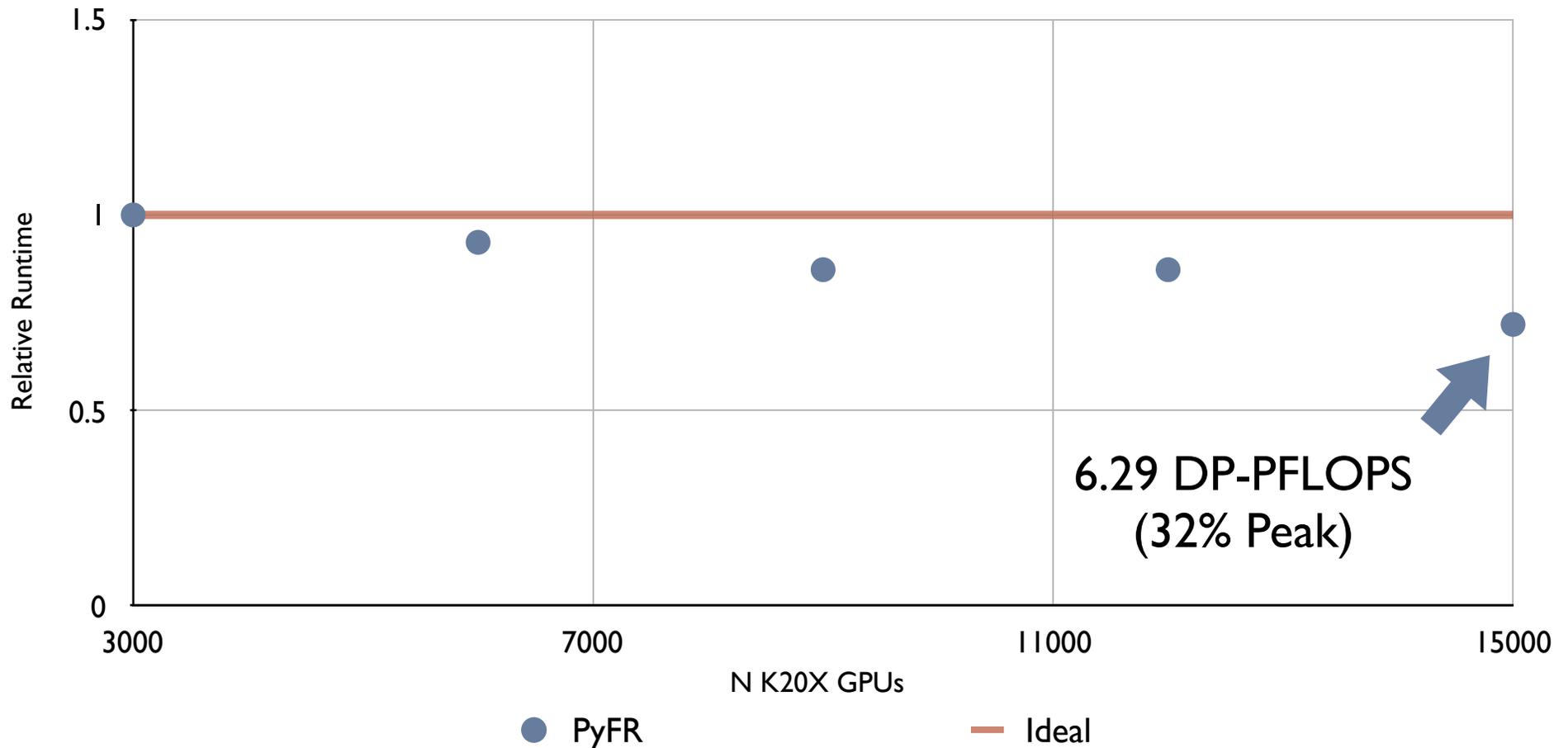
# Performance

- Weak scaling on Piz Daint ( $p = 4$ )



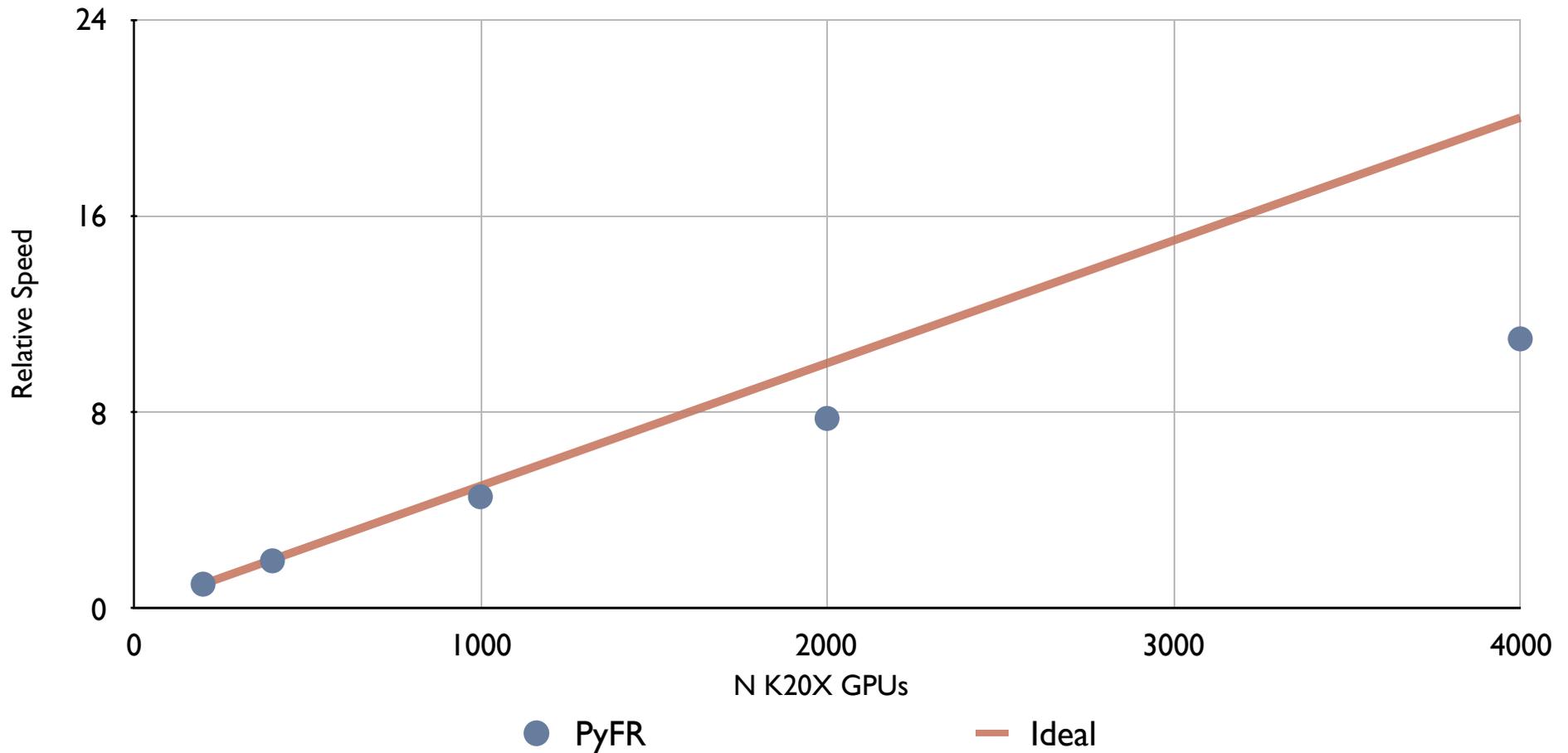
# Performance

- Weak scaling on Piz Daint ( $p = 4$ )



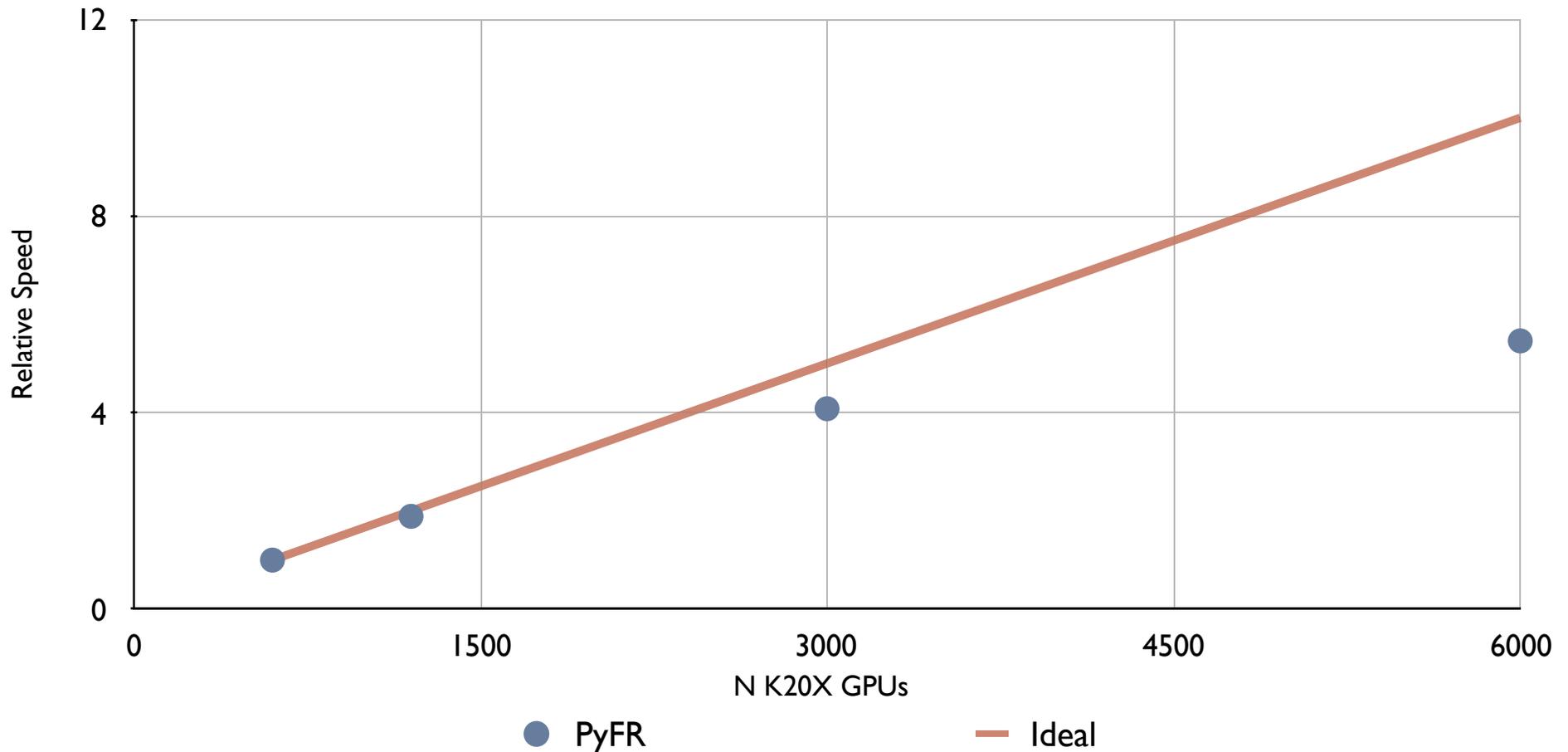
# Performance

- Strong scaling on Piz Daint ( $p = 4$ )



# Performance

- Strong scaling on Piz Daint ( $p = 4$ )



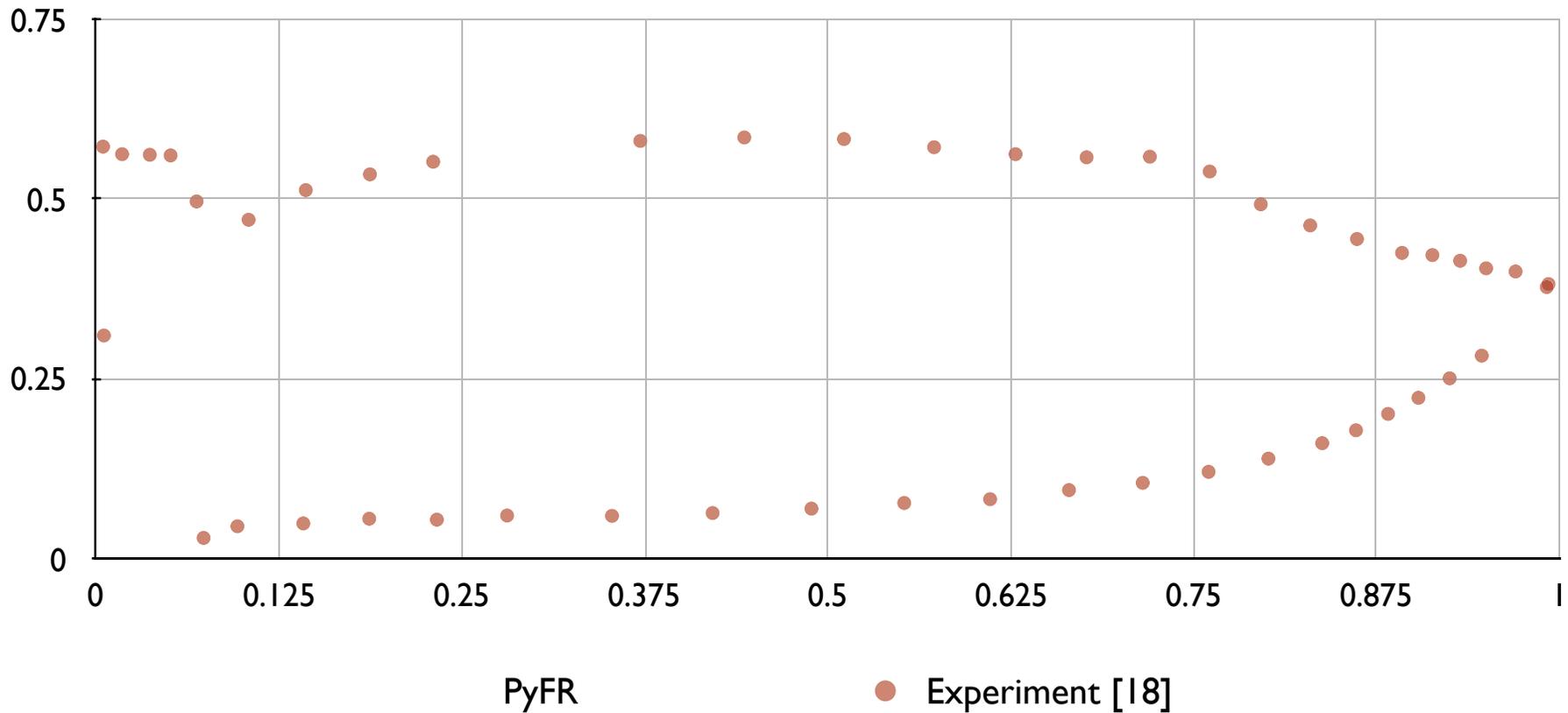
# Performance

- Peak sustained performance run ( $p = 5$ )

18,000 GPUs  
195 billion DOFs  
10.6 DP-PFLOP/s  
45% peak

# Performance

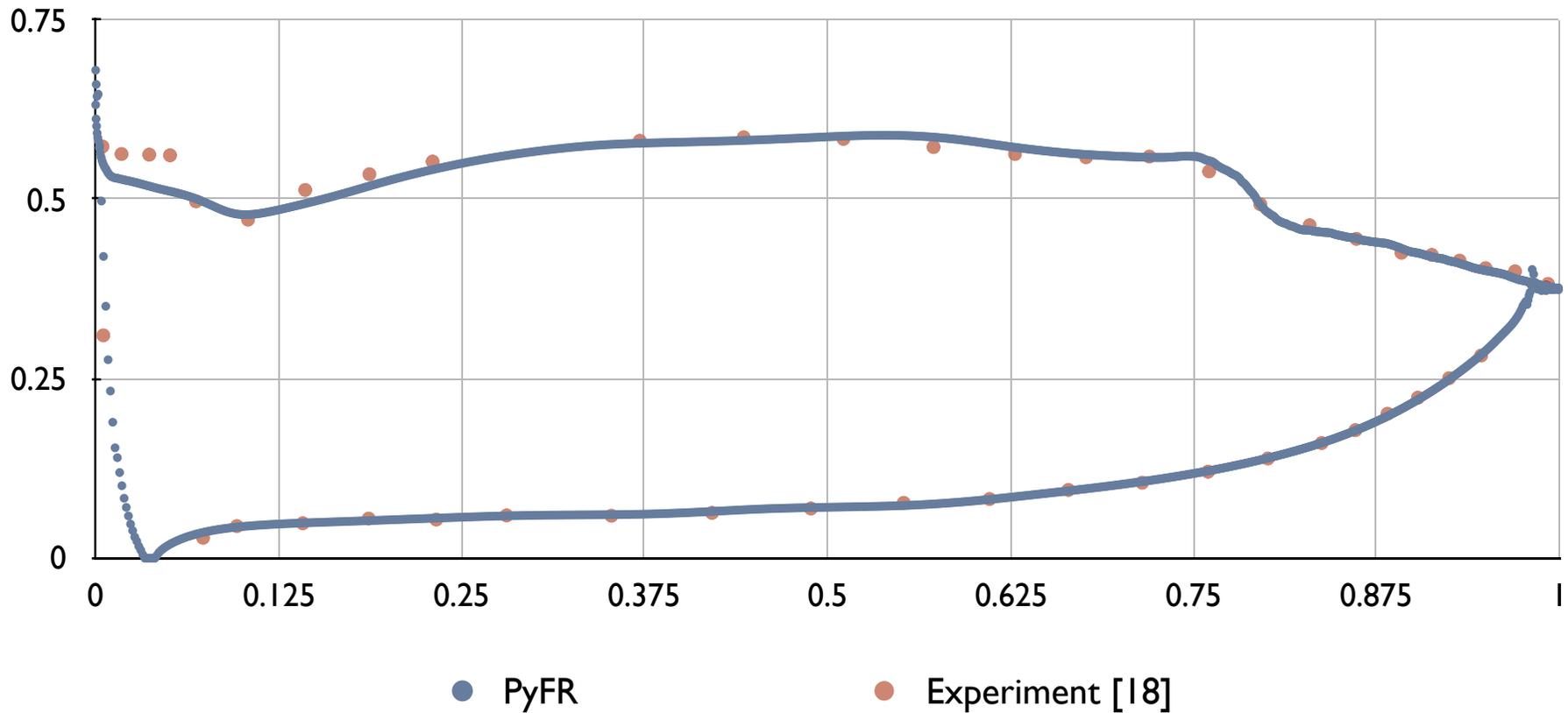
- Physics run ( $p = 4$ )



[18] P. Stadtmüller et al. Experimental and Numerical Investigation of Wake-Induced Transition on a Highly Loaded LP Turbine at Low Reynolds Numbers. ASME TurboExpo. 2000.

# Performance

- Physics run ( $p = 4$ )



[18] P. Stadtmüller et al. Experimental and Numerical Investigation of Wake-Induced Transition on a Highly Loaded LP Turbine at Low Reynolds Numbers. ASME TurboExpo. 2000.

# Visualisation

- Interactive **visualisation/processing** of large datasets is a significant challenge!

# Visualisation

Remote Cluster



# Visualisation

Remote Cluster



Local Workstation



# Visualisation

## Remote Cluster



Solution in Remote GPU Memory  
Copy to Remote CPU Memory  
IO to Remote Magnetic Disk  
IO from Remote Magnetic Disk



## Local Workstation



Render in Local GPU Memory  
Analyse in Local CPU Memory  
IO from Local Magnetic Disk  
IO to Local Magnetic Disk



Network

# Visualisation

Remote Cluster



Solution in Remote GPU Memory  
Analyse in Remote GPU Memory  
Render in Remote GPU Memory

Local Workstation



Network

# Visualisation

Remote Cluster



Local Workstation



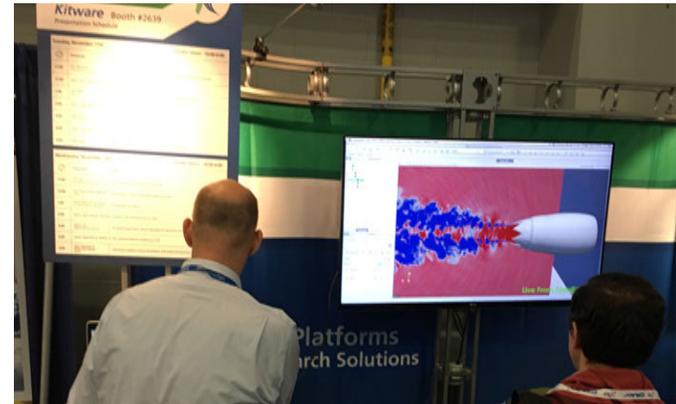
Network

# Visualisation

Titan at ORNL



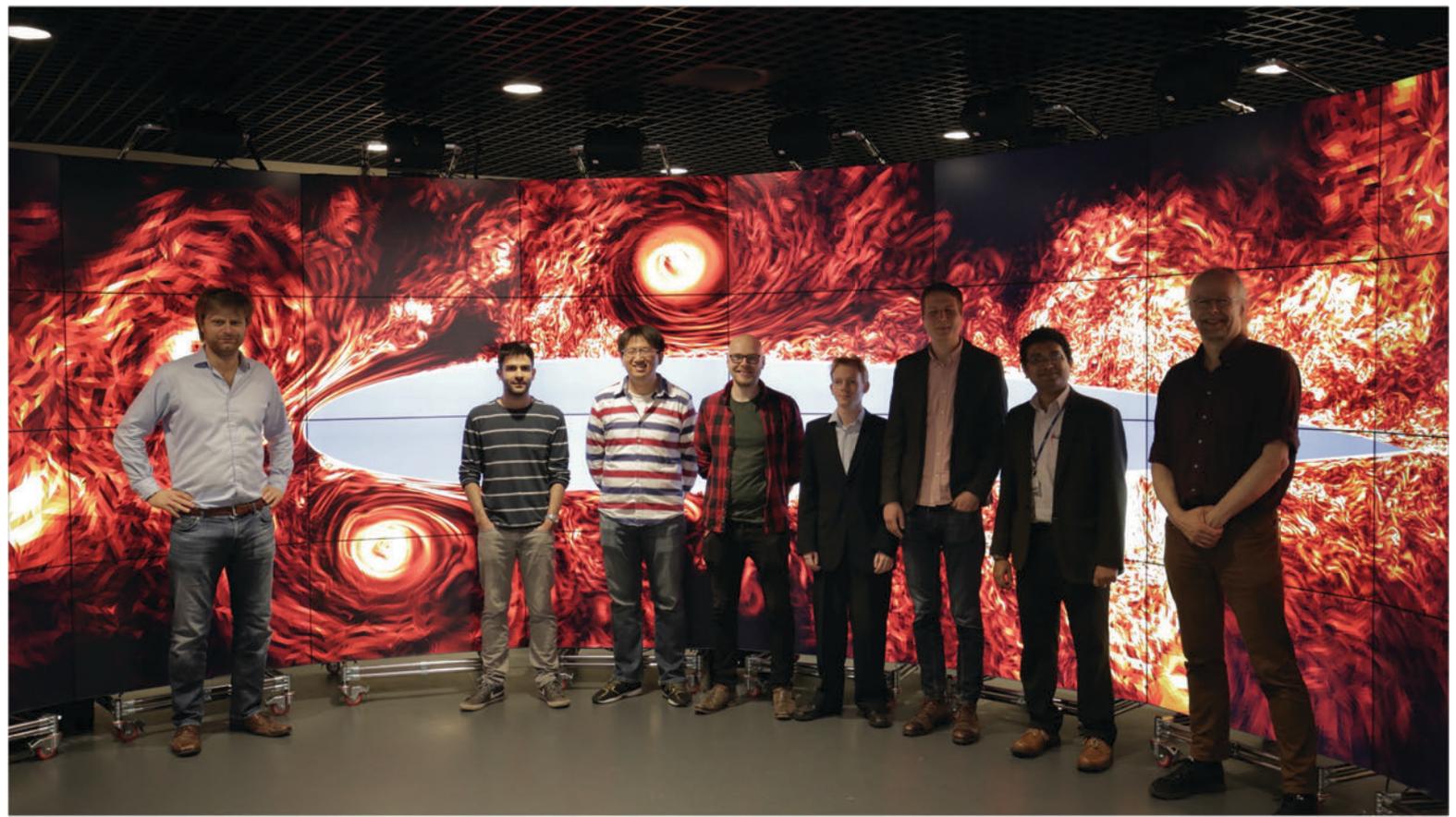
SCI 5 Show Floor



Network

# Visualisation

- Fun with Europe's largest vis-wall!



# Translation



# Translation



# Translation



**AIRBUS**

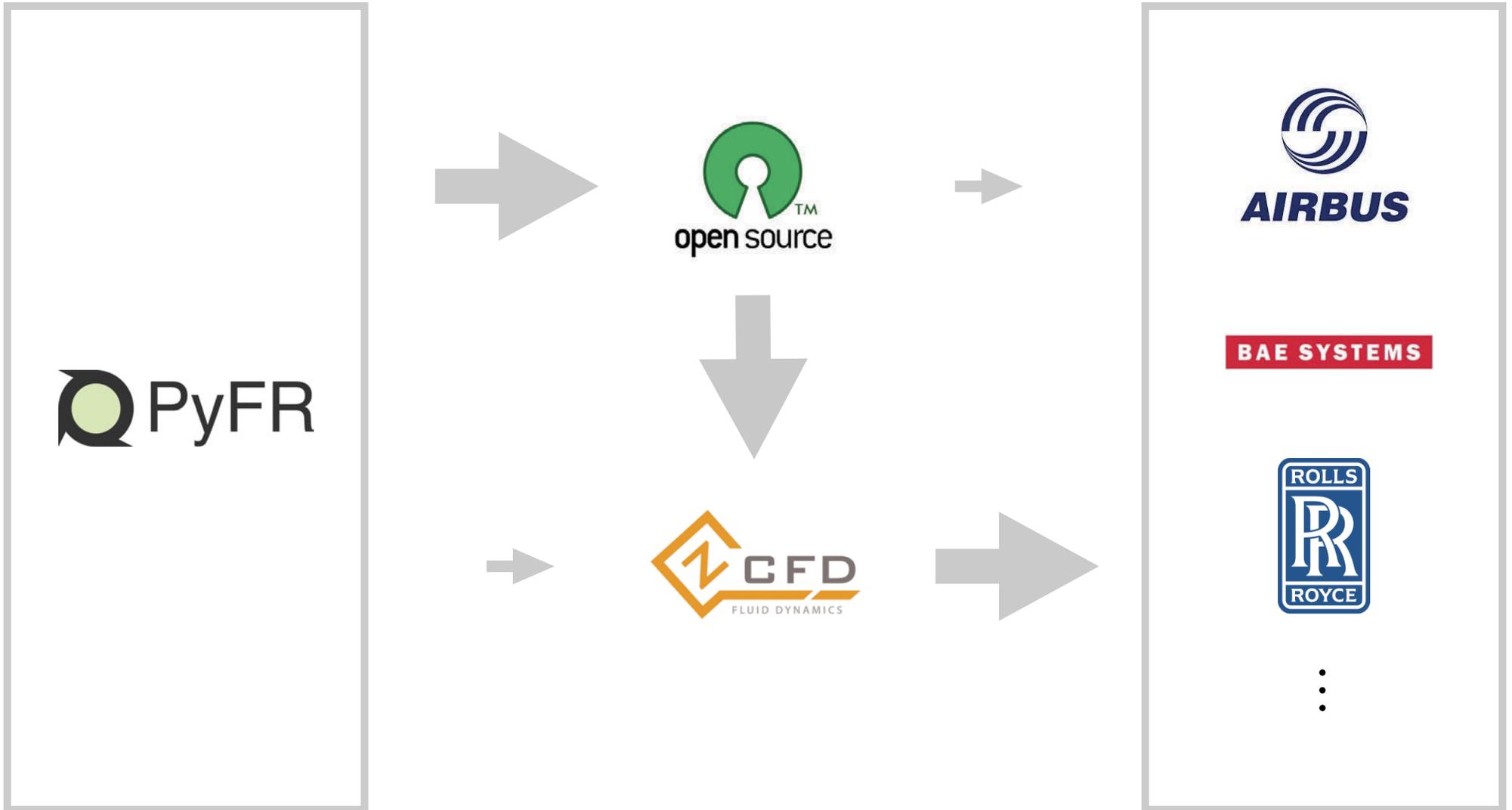
**BAE SYSTEMS**



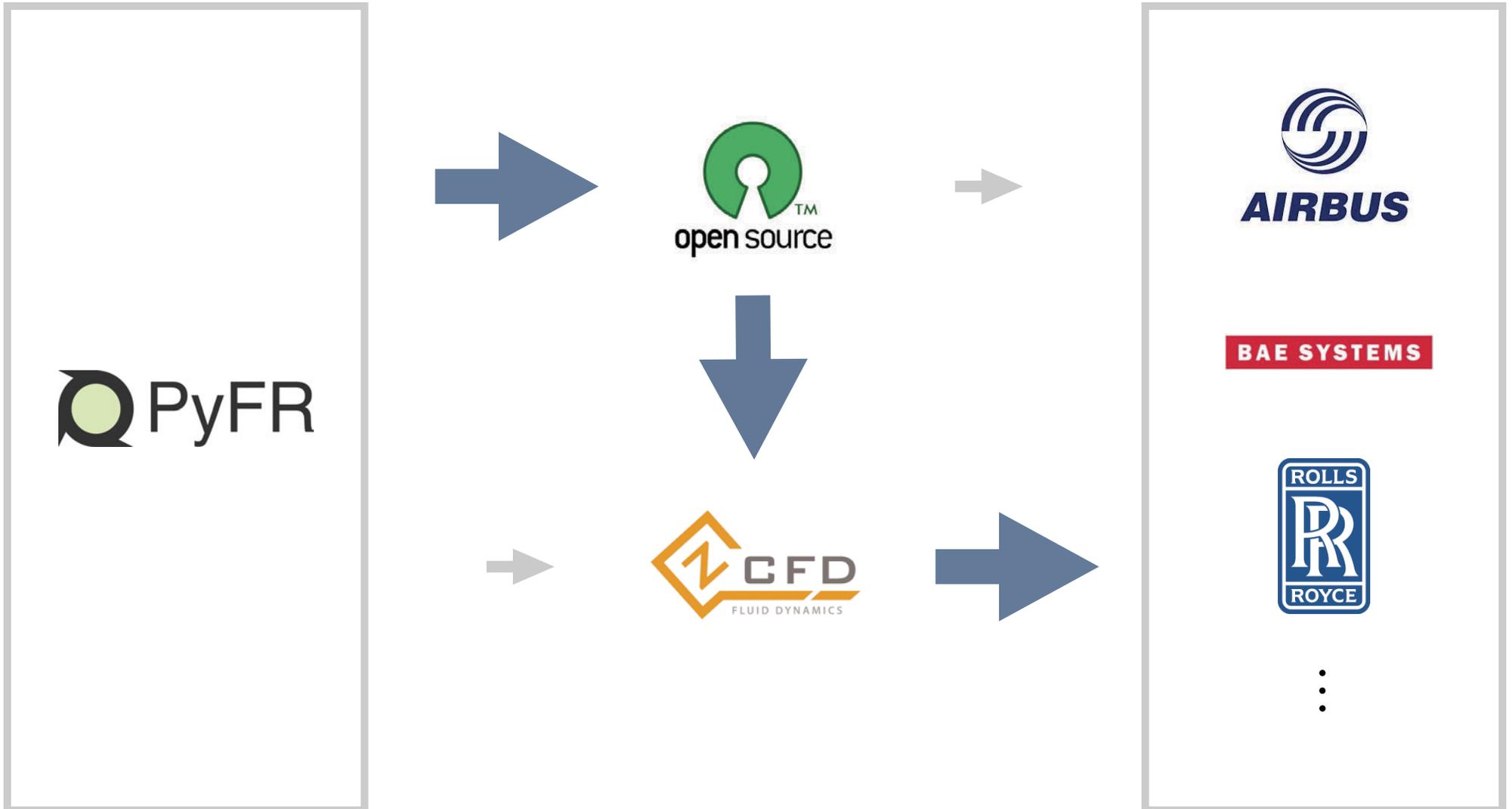
# Translation



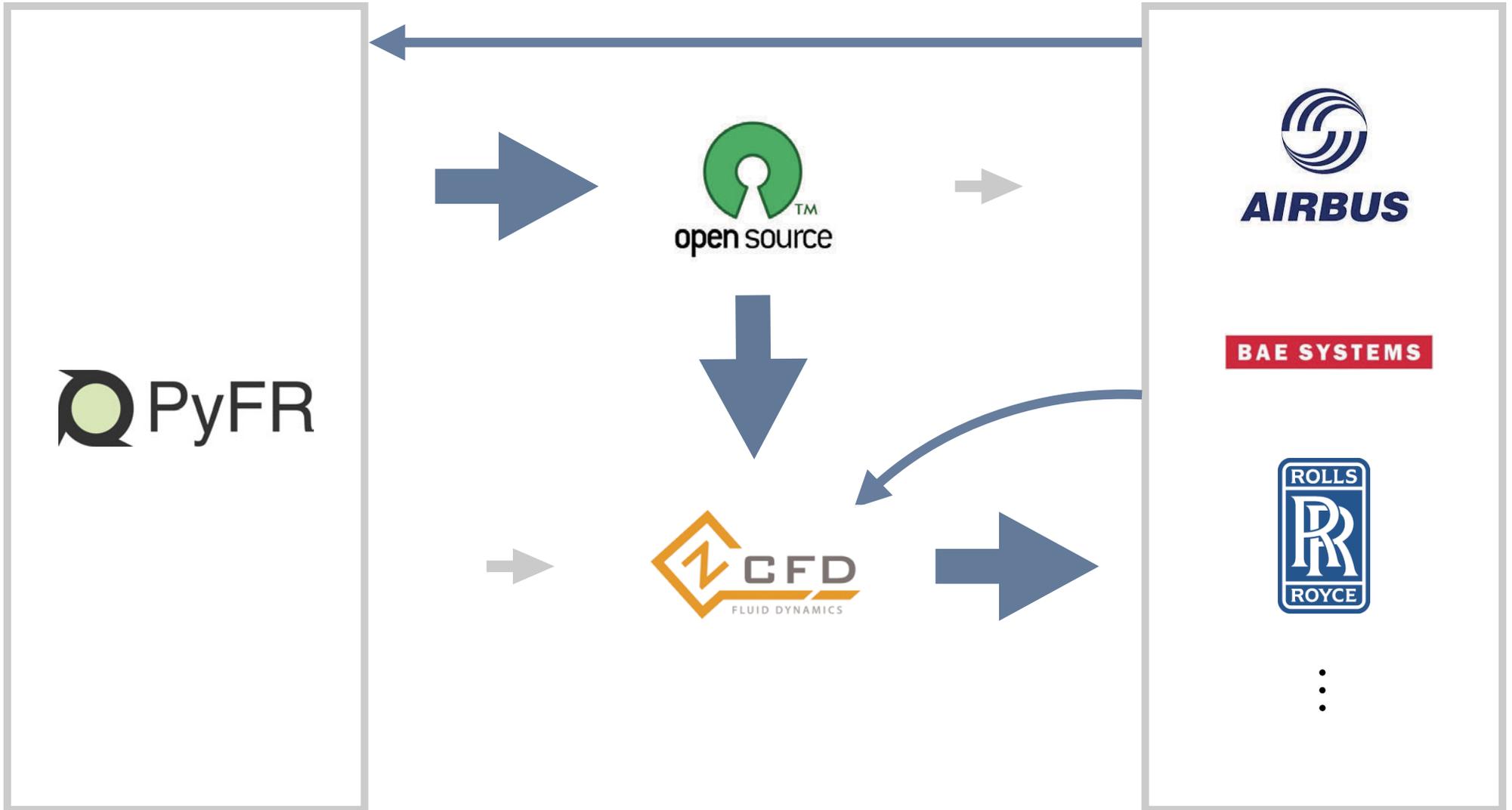
# Translation



# Translation



# Translation



# Team



Brian Vermeire



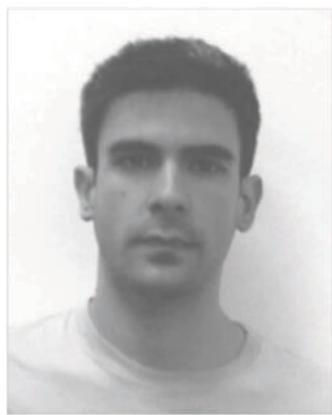
Jin Seok Park



Lorenza Grechy



Arvind Iyer



George Ntemos



Francesco Iori



Antony Farrington



Niki Loppi

# Funding

**EPSRC**

Pioneering research  
and skills

**Innovate UK**

Technology Strategy Board

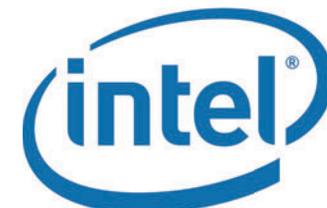


**EUROPEAN COMMISSION**



**AIRBUS**

**BAE SYSTEMS**



# Computers

- Emerald (CFI - UK)
- Wilkes (Cambridge University - UK)
- Piz Daint (CSCS - Switzerland)
- Titan (Oak Ridge National Laboratory - USA)