High-Order Methods and High Fidelity Simulation of Unsteady Turbulent Fluid Flows

Prof. Antony Jameson

Department of Aeronautics & Astronautics
– *Aerospace Computing Laboratory* –
Stanford University
jameson@baboon.stanford.edu
Assisted by Freddie Witherden

Arlington, VA
August 9 2016

Outline

I. Context

II. Current status of CFD

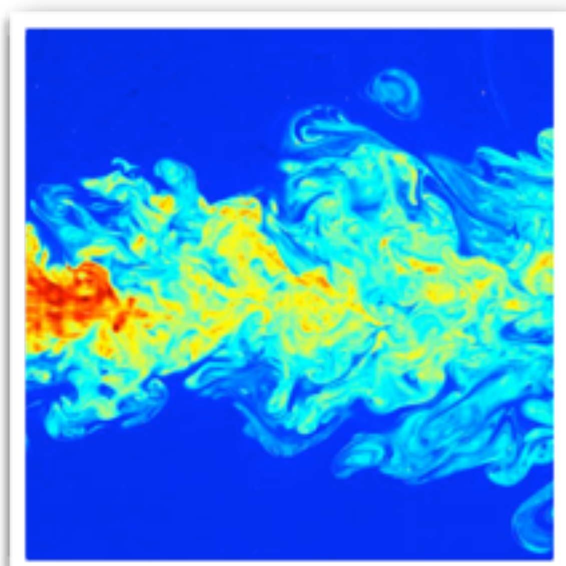
III. Flux Reconstruction

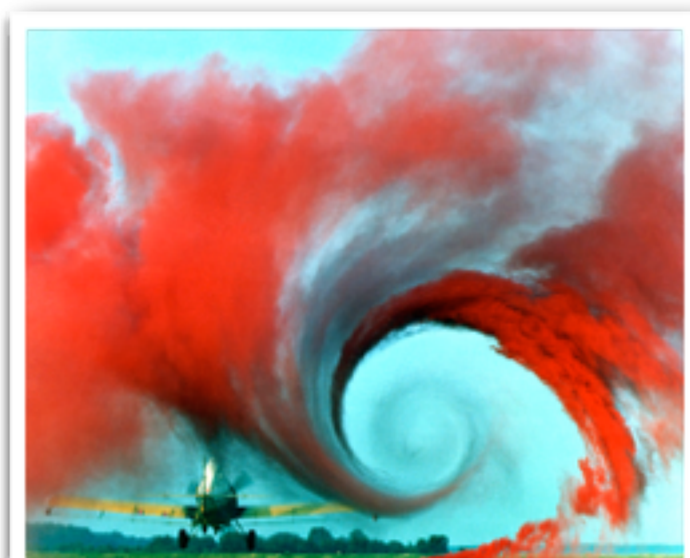
IV. PyFR

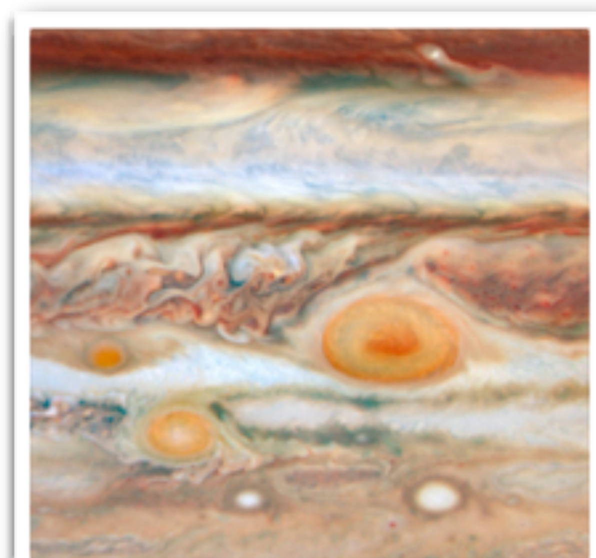
V. LES computations and future work

Context

“When I die and go to Heaven there are two matters on which I hope enlightenment. One is quantum electrodynamics and the other is turbulence. About the former, I am really rather optimistic.”







Outline

I. Context

II. Current status of CFD

III. Flux Reconstruction

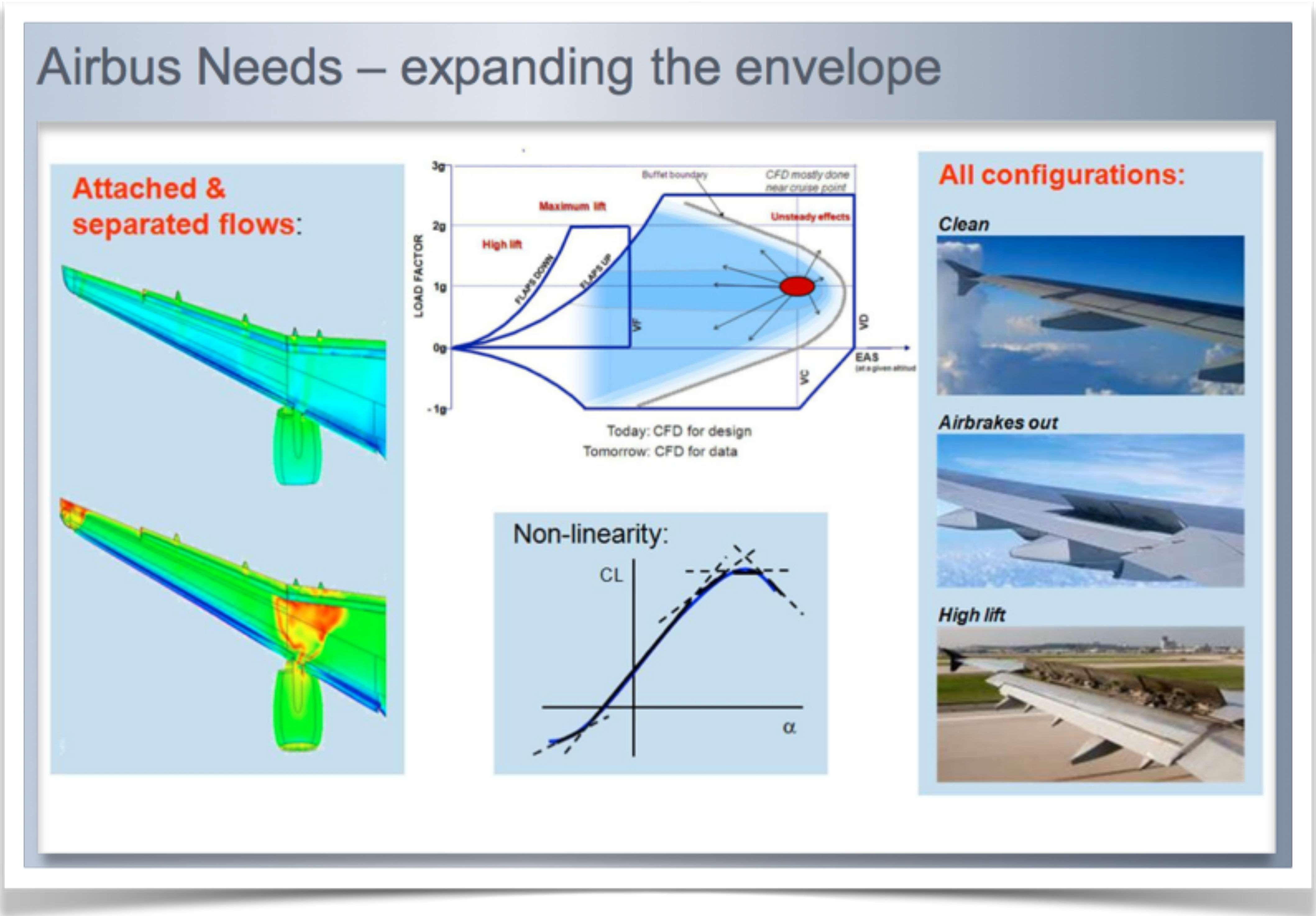
IV. PyFR

V. LES computations and future work

CFD Contributions to B787

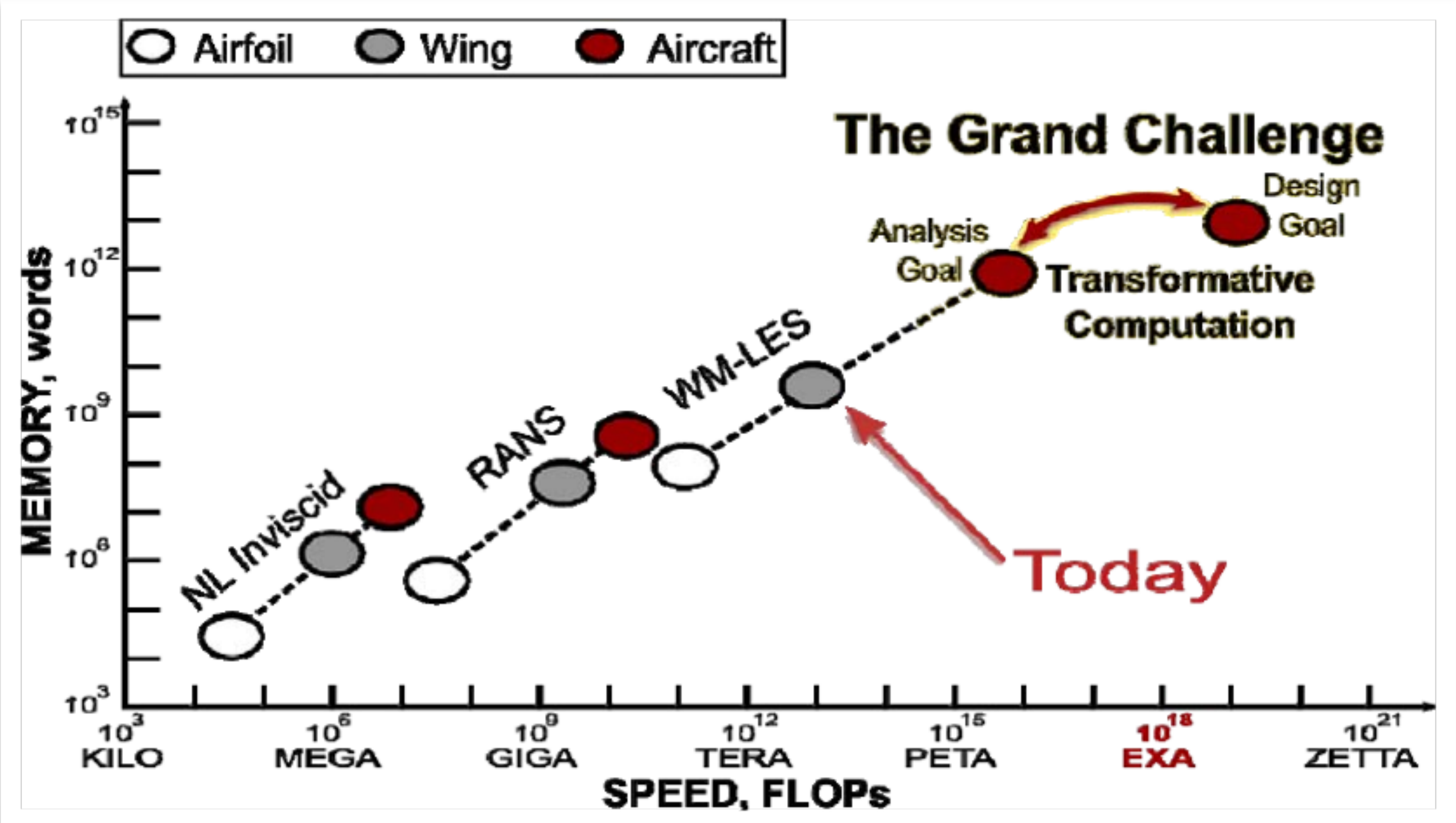
CFD Contribution to A380

The Future of CFD



Murray Cross, Airbus, Technology Product Leader - Future Simulations (2012)

The Future of CFD



Outline

I. Context

II. Current status of CFD

III. Flux Reconstruction

IV. PyFR

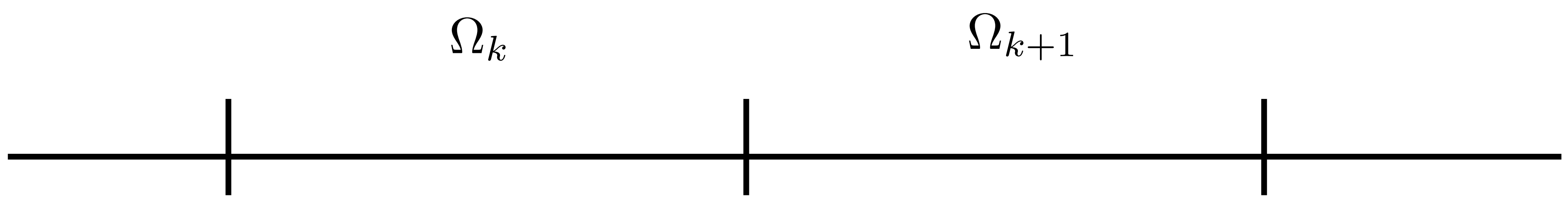
V. LES computations and future work

Outline in 1D

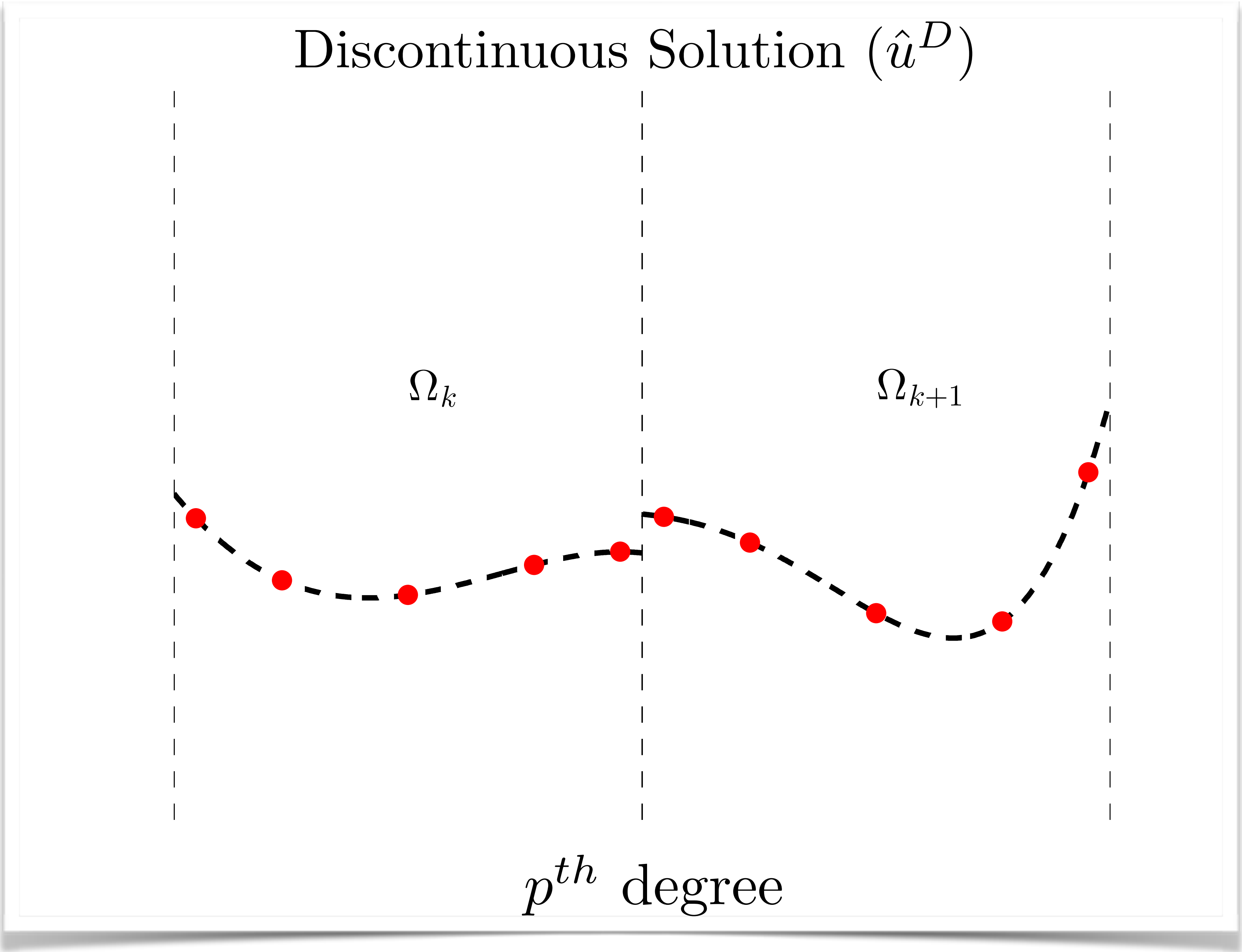
- Consider the 1D Conservation Law

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = 0 \quad \text{where} \quad f = f(u)$$

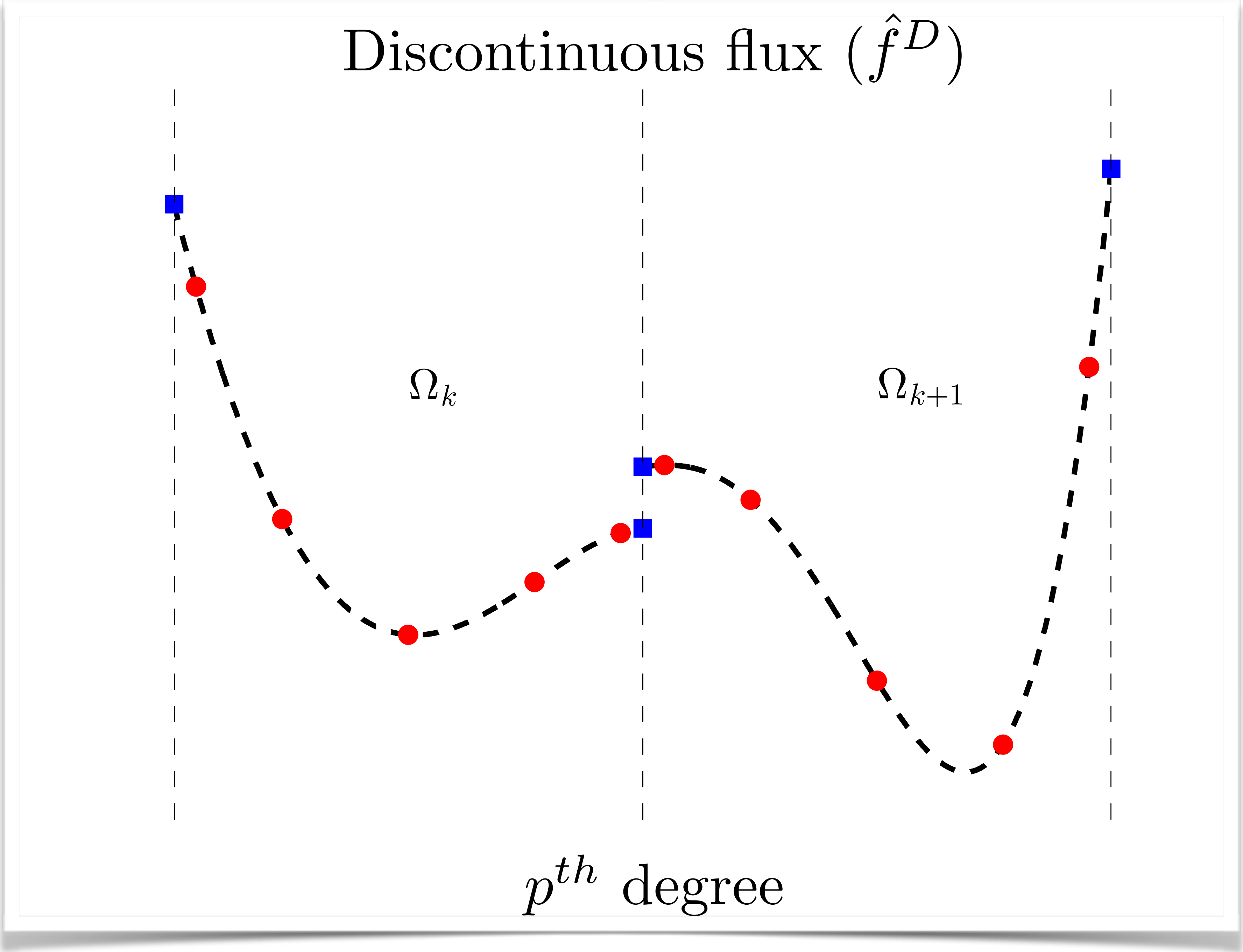
- Discretize the domain into elements



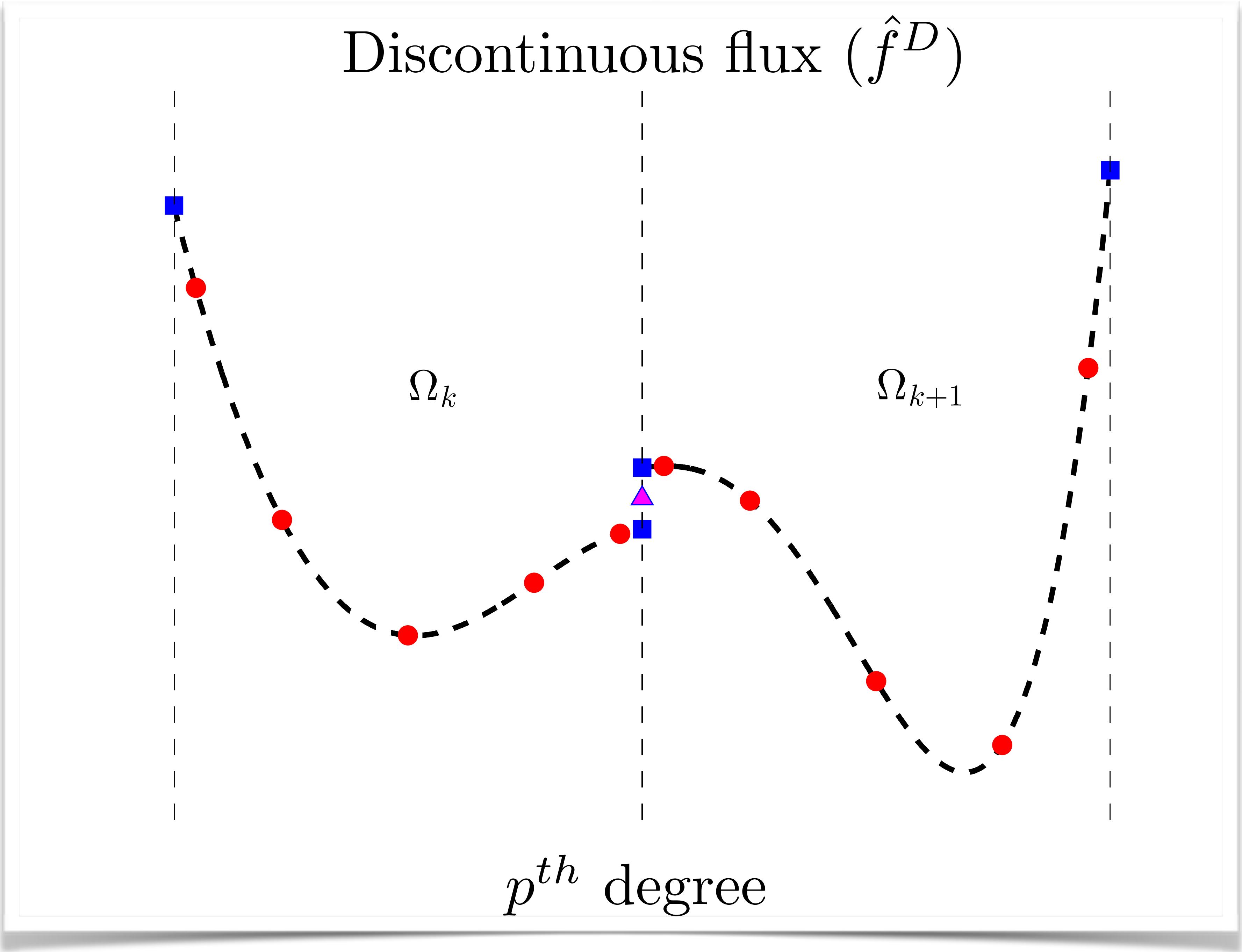
Outline in 1D



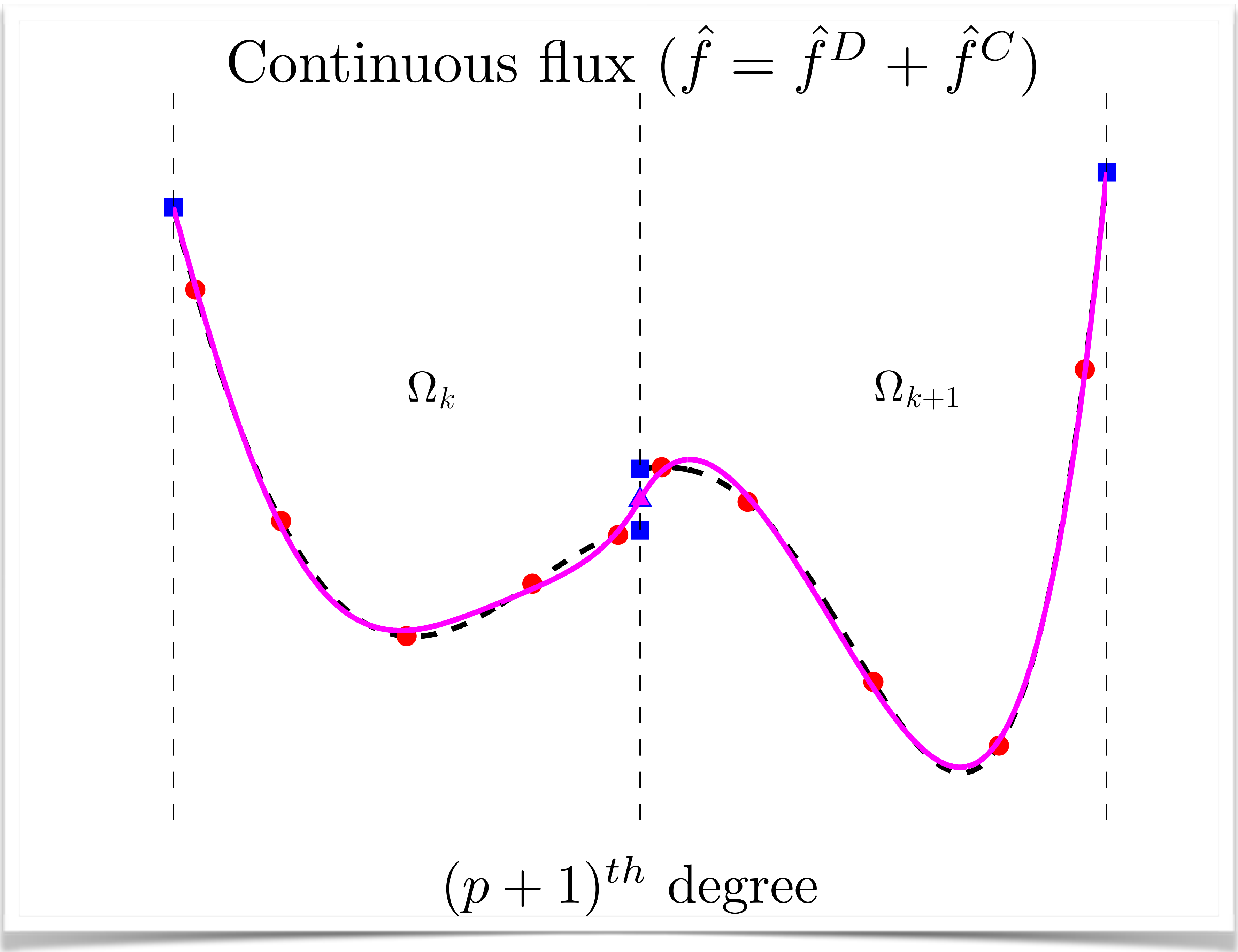
Outline in 1D



Outline in 1D



Outline in 1D



Outline in 1D

- Calculate the derivative of the flux and time-advance to get $\hat{u}^D$ at next time-step

$$\frac{d\hat{u}_k^D}{dt} = -D^h \hat{f_k}^D - D\hat{f_k}^C$$

- For second order PDEs (diffusive fluxes), split into first-order PDEs and perform similar procedure for each

Linear Energy Stability

- **There exists a family of Flux Reconstruction schemes that are guaranteed to be linearly stable [Vincent et al., J. Sci. Comput, 2011]**
  - Parameterized with a constant *c* which changes the scheme
  - Recover NDG, SD, plus other previously-found energy-stable FR schemes

$$g_R = \frac{1}{2}\left[L_p + \frac{\eta_p L_{p-1} + L_{p+1}}{1+\eta_p}\right] \quad \eta_p = \frac{c(2p+1)(a_p p!)^2}{2} \quad a_p = \frac{(2p)!}{2^p (p!)^2}$$

- **Accomplished showing a broken Sobolev type norm of the solution**

$$||u^\delta||_{p,2} = \left[\sum_{n=0}^{N-1}\int_{x_n}^{x_{n+1}} (u_n^{\delta D})^2 + \frac{c}{2}(J_n)^{2p}\left(\frac{\partial^p u_n^{\delta D}}{\partial x^p}\right)^2 \mathrm{d}x\right]^{1/2}$$

is guaranteed to be non-increasing.

Linear Energy Stability

- **Proof has been extended to simplex elements where the correction function is in a Raviart-Thomas space and advection diffusion problems.**
  - Castonguay et al. J. Sci. Comput., 51(1):224–256, 2012.
  - Williams et al. Journal of Computational Physics, 2013.
- **However, the proof did not extend to tensor product quadrilateral elements except for $c = 0$ which recovers nodal DG.**
  - A. Jameson, AIAA paper 2011-3226, Hawaii, 2011.
- **Proof has now been achieved by Abhishek Sheshadri by further augmenting the norm.**

$$\|u^D\|^2_{W^{2p,2}_\delta} = \sum_{k=1}^{N} \left(\int_{\Omega_k} \left[(u_k^D)^2 + \frac{c}{2}\left(\left(\frac{\partial^p u_k^D}{\partial \xi^p} \right)^2 + \left(\frac{\partial^p u_k^D}{\partial \eta^p} \right)^2 \right) + \frac{c^2}{4} \left(\frac{\partial^{2p} u_k^D}{\partial \xi^p \partial \eta^p} \right)^2 \right] d\Omega_k \right)$$

Shock Capturing

Method	Advantages	Disadvantages
Limiting	• Eliminates oscillations • Robust	• Smeared over elements • Expensive
Artificial Viscosity	• Sub-cell shock capturing • Smoothly varying viscosity	• High-order derivatives • Time-step restrictions • Too many parameters
Filtering	• Sub-cell shock capturing • Very Inexpensive	• Varying dissipation not easy • Needs a good sensor

Shock Capturing

Method	Advantages	Disadvantages
Limiting	• Eliminates oscillations • Robust	• Smeared over elements • Expensive
Artificial Viscosity	• Sub-cell shock capturing • Smoothly varying viscosity	• High-order derivatives • Time-step restrictions • Too many parameters
Filtering	• Sub-cell shock capturing • Very Inexpensive	• Varying dissipation not easy • Needs a good sensor

For explicit FR on GPUs filtering is attractive…but requires a good sensor.

Shock Capturing: Our Approach

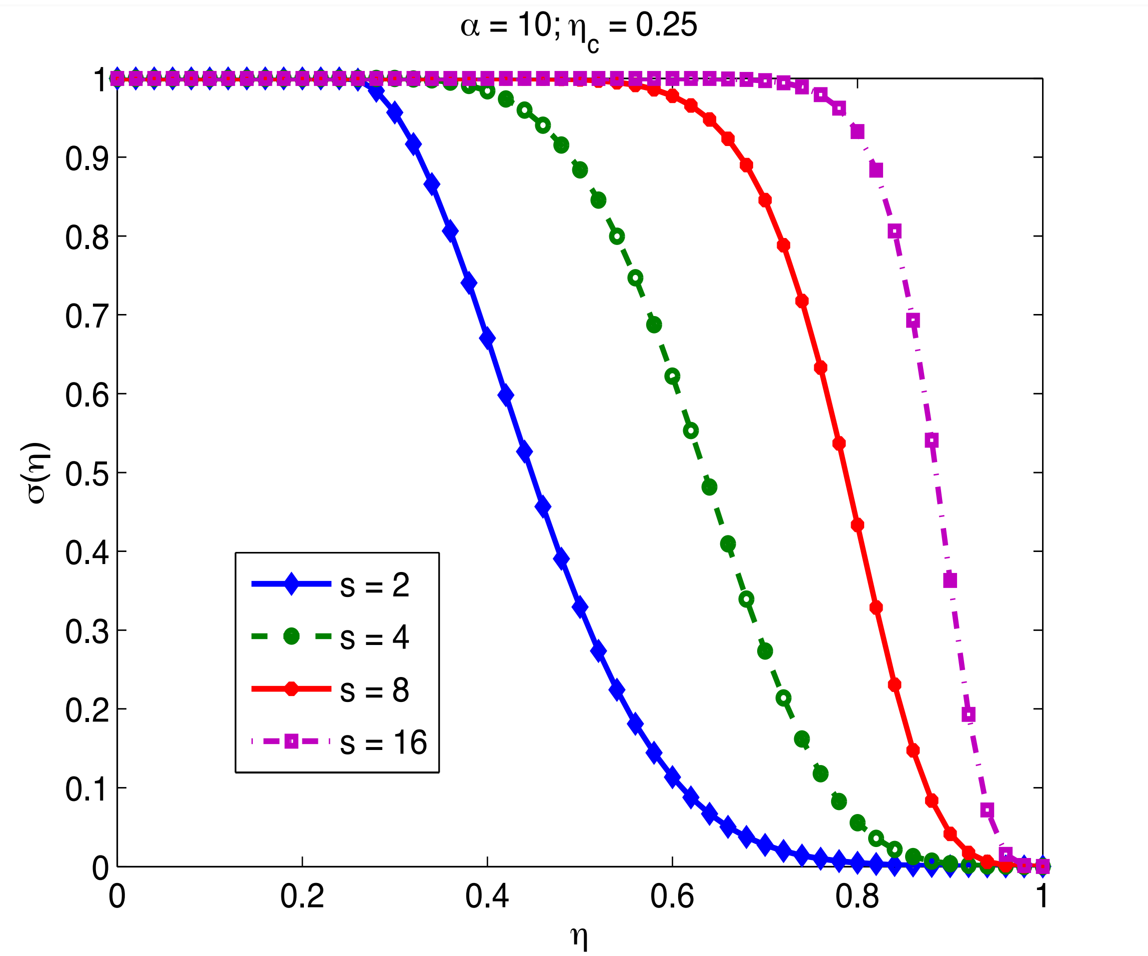
Two-step approach

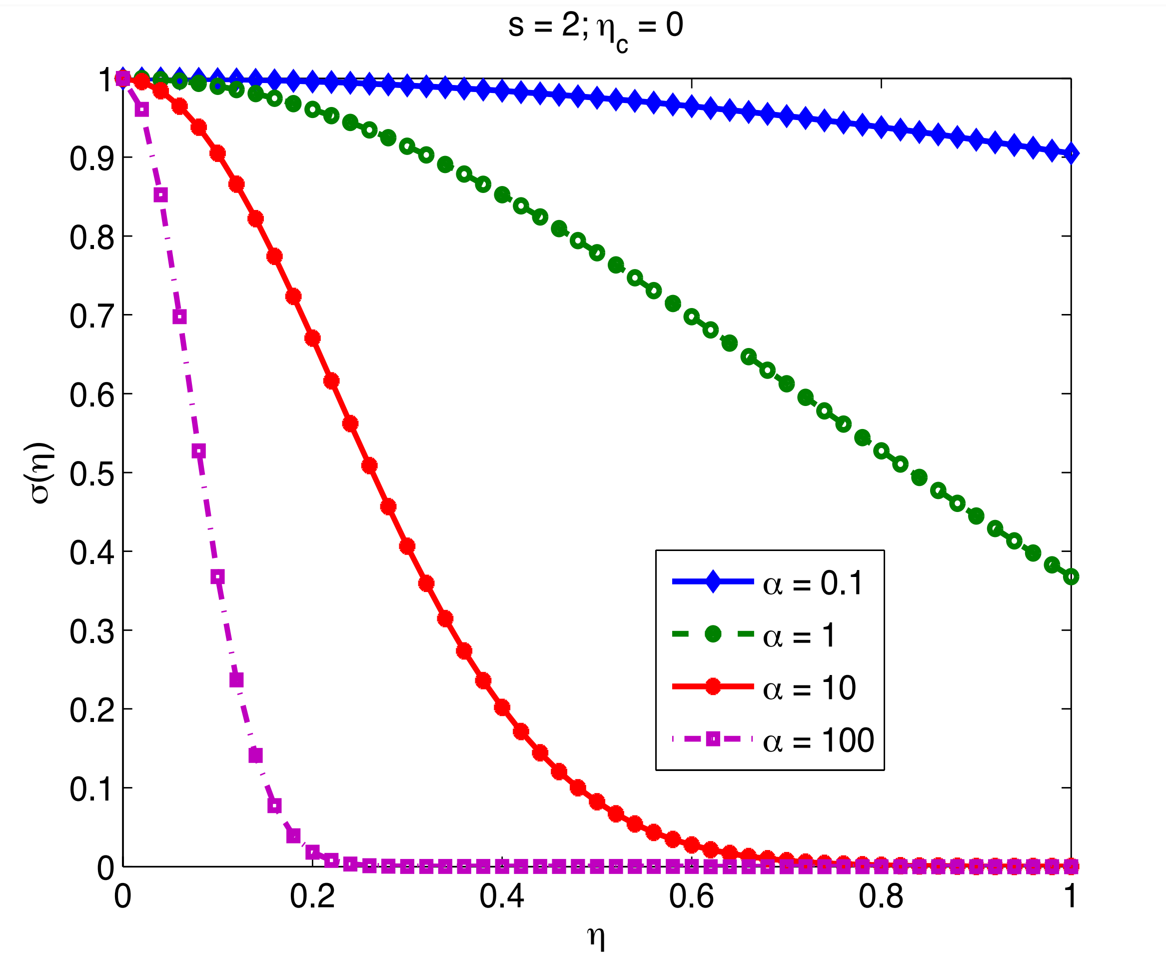
Detect Shocks → Distinguish between shocks and vortices/boundary-layers

Filter locally → Strong filter in shocked elements

Minimize parameter fine-tuning

Shock Capturing: Exponential Modal Filtering





$$\sigma(\eta) = \begin{cases} 1, & 0 \le \eta \le \eta_c = \frac{N_c}{P} \\ \exp\left(-\alpha\left(\frac{\eta-\eta_c}{1-\eta_c}\right)^s\right), & \eta_c \le \eta \le 1 \end{cases}$$

Shock Capturing: Current Sensors

- **Physics based**
  - Specific to problem or type of discontinuity
  - Need derivatives: expensive
  - Hard to extend to unstructured grids
- **Smoothness based**
  - Used successfully in low-order schemes
  - Persson and Peraire — high order unstructured methods

Shock Capturing: Concentration Method

- Used for image/MRI edge detection
- Works directly on Fourier spectral information

Shock Capturing: Concentration Method

- Suppose we have the spectral projection of a function *f*

$$S_N(f) = -\sum_{k=-N}^{N} \hat{f}_k e^{ikx}$$

- If *f* has a discontinuity at a point then

$$\hat{f}_k = [f](c)\frac{e^{-ikc}}{2\pi ik} + \mathcal{O}(\frac{1}{k^2})$$

Shock Capturing: Concentration Method

- There exist special kernels such that

$$K_\epsilon * S_N(f) = [f](x) + \mathcal{O}(\epsilon)$$

- Kernel action is of the form

$$K_N^\sigma * S_N(f) = i\pi \sum_{k=-N}^{N} sgn(k)\sigma(\frac{|k|}{N})\hat{f}_k e^{ikx}$$

- where the σ coefficients are *concentration factors*.

Shock Capturing: Concentration Method

We have extended this to work with Jacobi polynomials

Shock Capturing: Concentration Method

- Jacobi polynomials are the eigenfunctions of the Sturm-Liouville problem

$$((1-x^2)\omega(x)P_k'(x))' = -\lambda_k \omega(x) P_k(x) \qquad -1 \leq x \leq 1$$

with weight $\omega(x) = (1-x^2)^\alpha$

- Polynomial modes also show a decreased decay rate:

$$\hat{f}_k = \frac{1}{\lambda_k}[f](x)(1-x^2)\omega(c)P_k'(x) + \mathcal{O}(\frac{1}{\lambda_k^2})$$

where $\lambda_k = k(k + 2\alpha + 1)$

Shock Capturing: Concentration Method

- Concentration property for Jacobi polynomials

$$\left|\frac{\pi\sqrt{1-x^2}}{N}S_N(f)'(x)-[f](x)\right| \leq \frac{\mathrm{Const}}{(1-x^2)^{\alpha/2+1/4}}\cdot\frac{logN}{N}$$

- Legendre polynomials are special cases Jacobi polynomials with α = 0.

Shock Capturing: Concentration Method

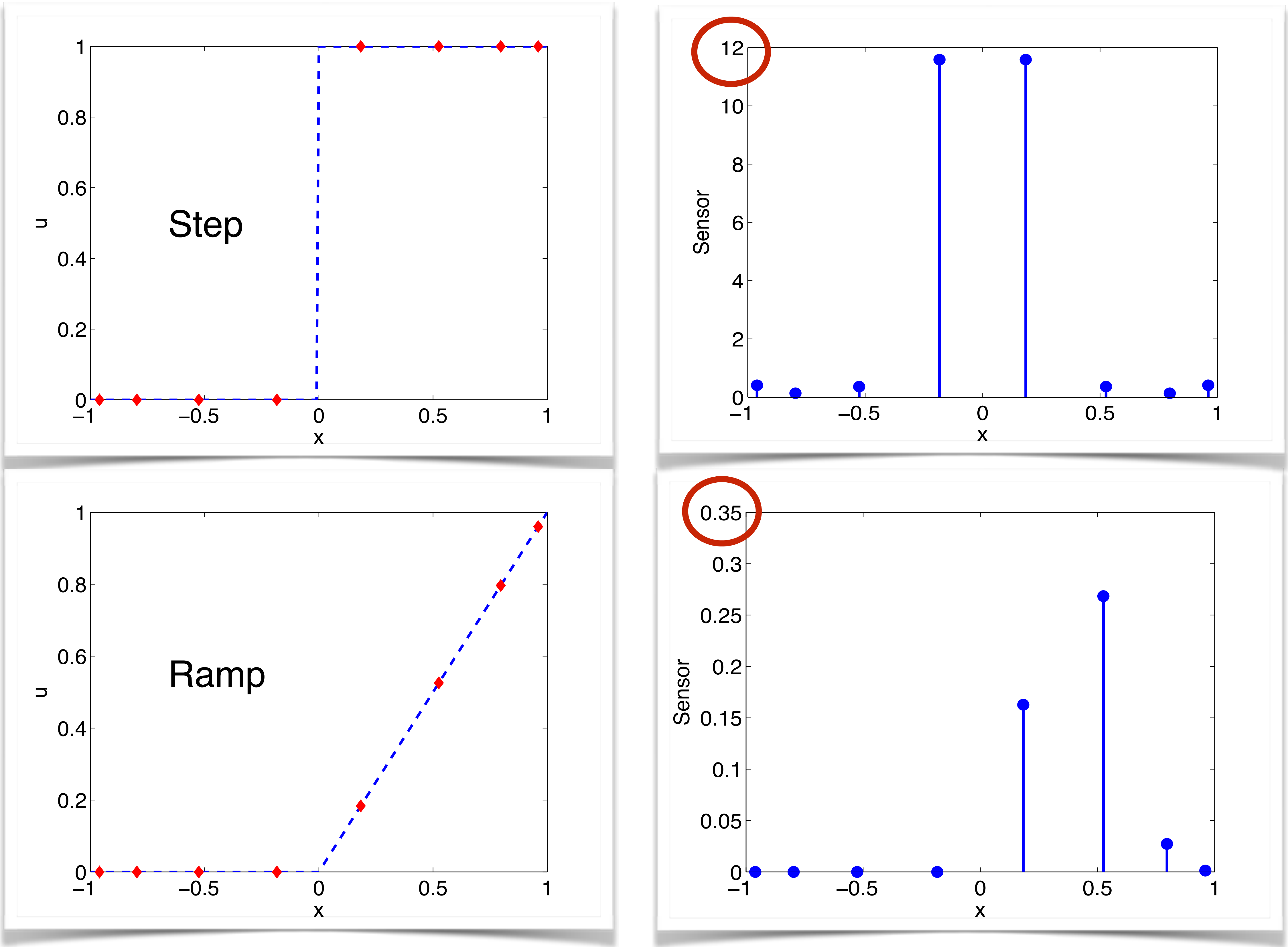
- Concentration factors are of the form $\sigma(\xi) = \xi\mu(\xi)$. Two forms have been investigated

$$\mu(\xi) = r\xi^{r-1}$$

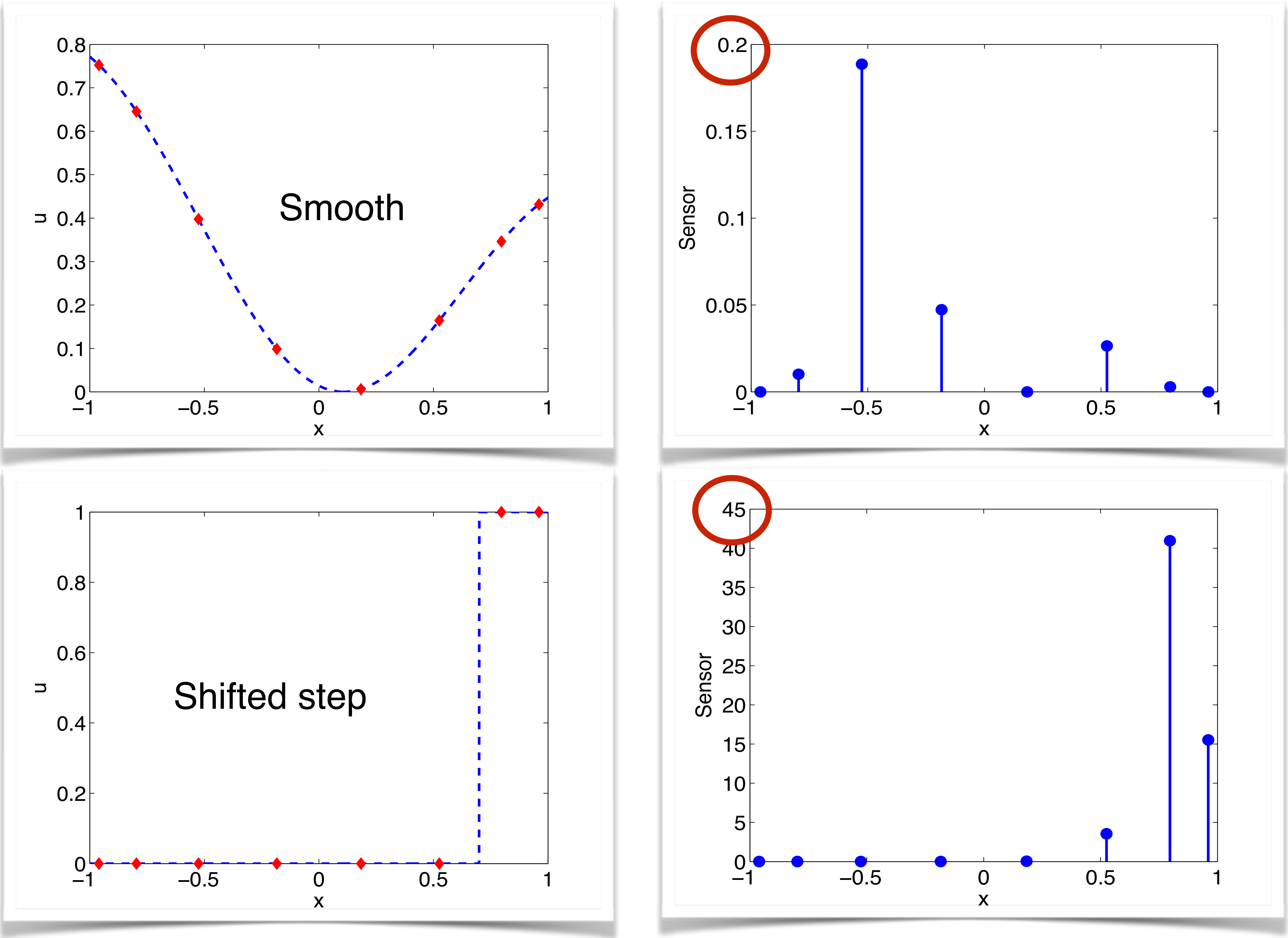
$$\mu(\xi) = Ce^{\frac{1}{\alpha\xi(\xi-1)}}$$

Exponential factors have been found to work better in in practice

Shock Capturing: Our Sensor



Shock Capturing: Our Sensor



Shock Capturing: Implementation

1. At start-up compute the Concentration matrix **C**
2. Choose a quantity normalize the elemental solution to [0,1]
3. Evaluate the kernel by multiplying through by **C**
4. If any point in the element has a value greater than *threshold*, mark element for filtering
   - Take *threshold* to be mid point between *step* and *ramp*.
5. Apply a modal filter to these elements.

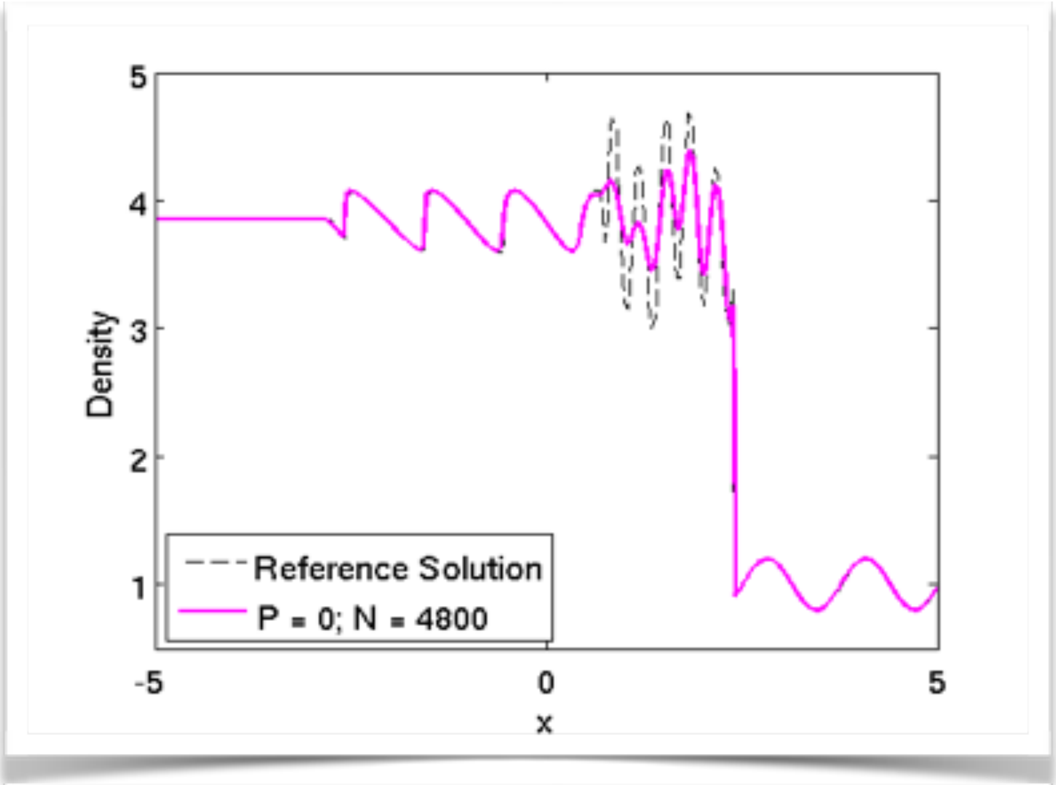
Shock Capturing: Shock Entropy Interaction

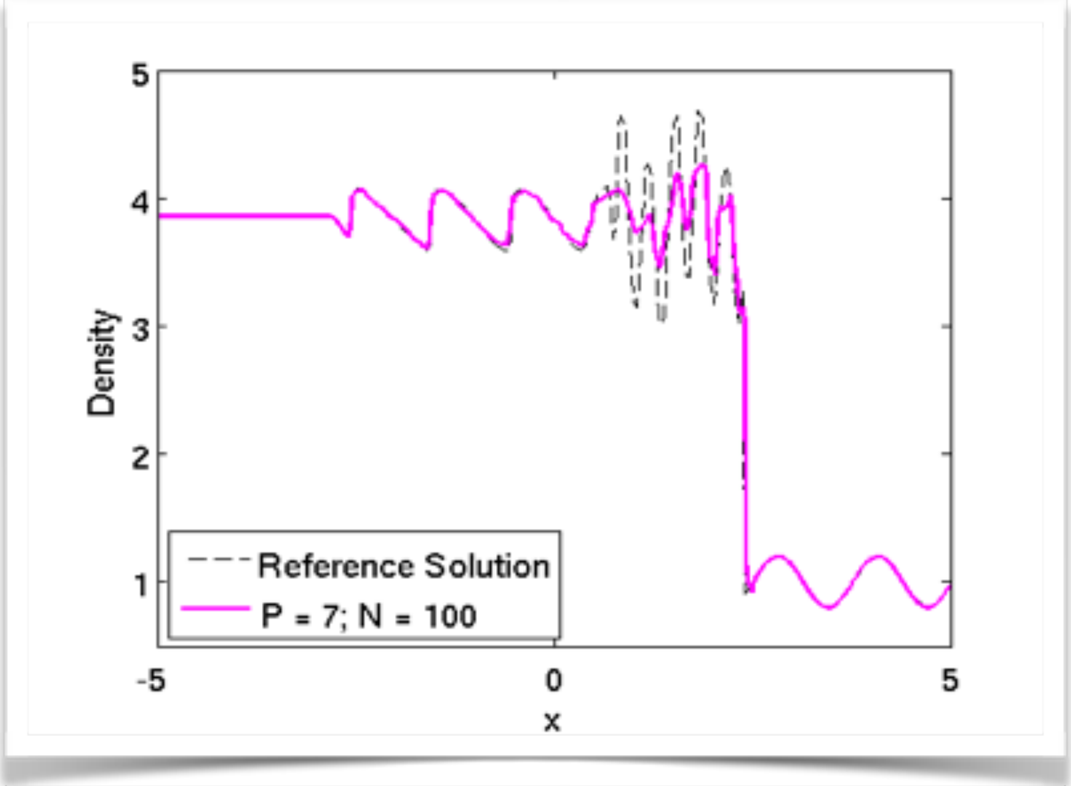
$$\begin{array}{llllll} \rho = 3.857143; & u = 2.629369; & p = 10.33333 & \text{for} & x \leq 4 \\ \rho = 1 + \epsilon \sin 5x; & u = 0; & p = 1 & \text{for} & x > 4 \end{array}$$

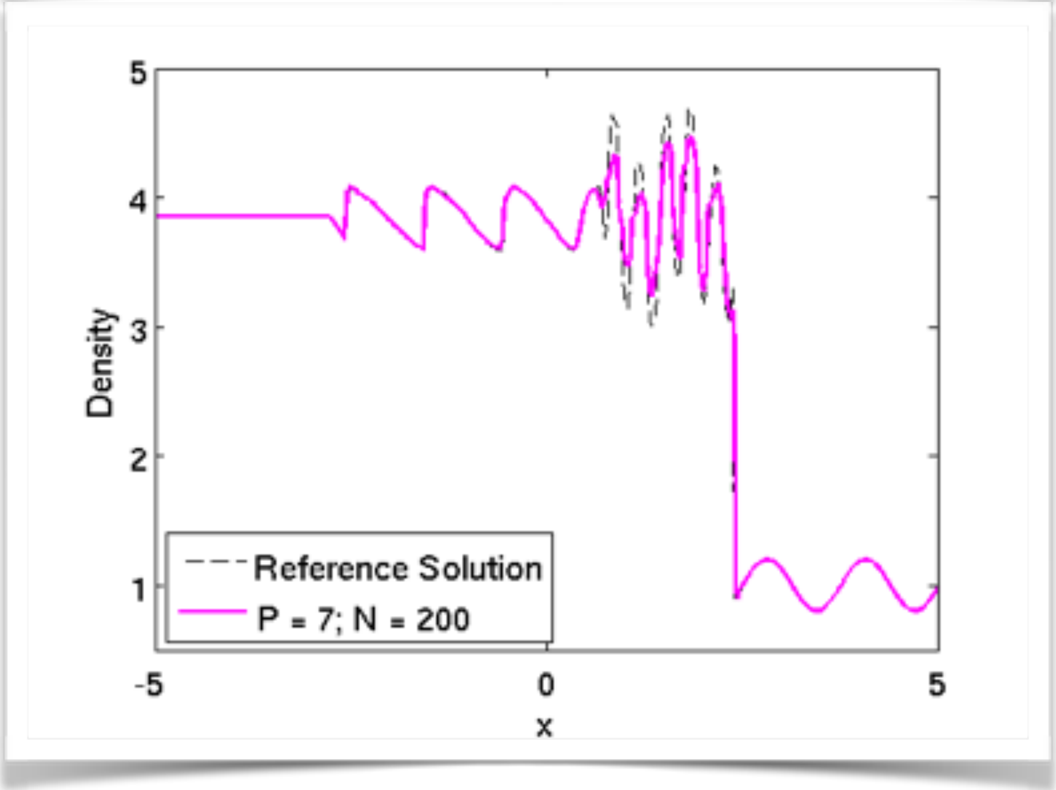
- Mach 3 shock wave moving into a stationary fluid with density perturbations.
- Interactions generates oscillations and small amplitude shocks giving rise to a fine structure.

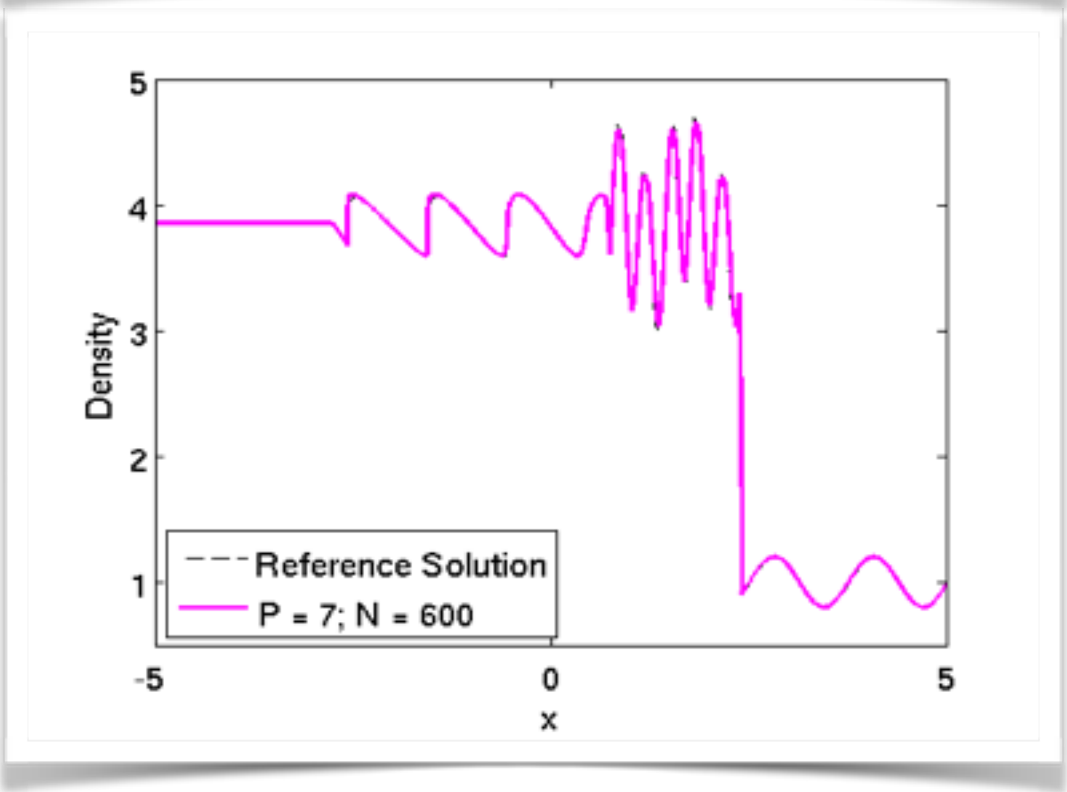
Filter Order	Filter Strength	Final Time
2	1	0.038s

Shock Capturing: Shock Entropy Interaction

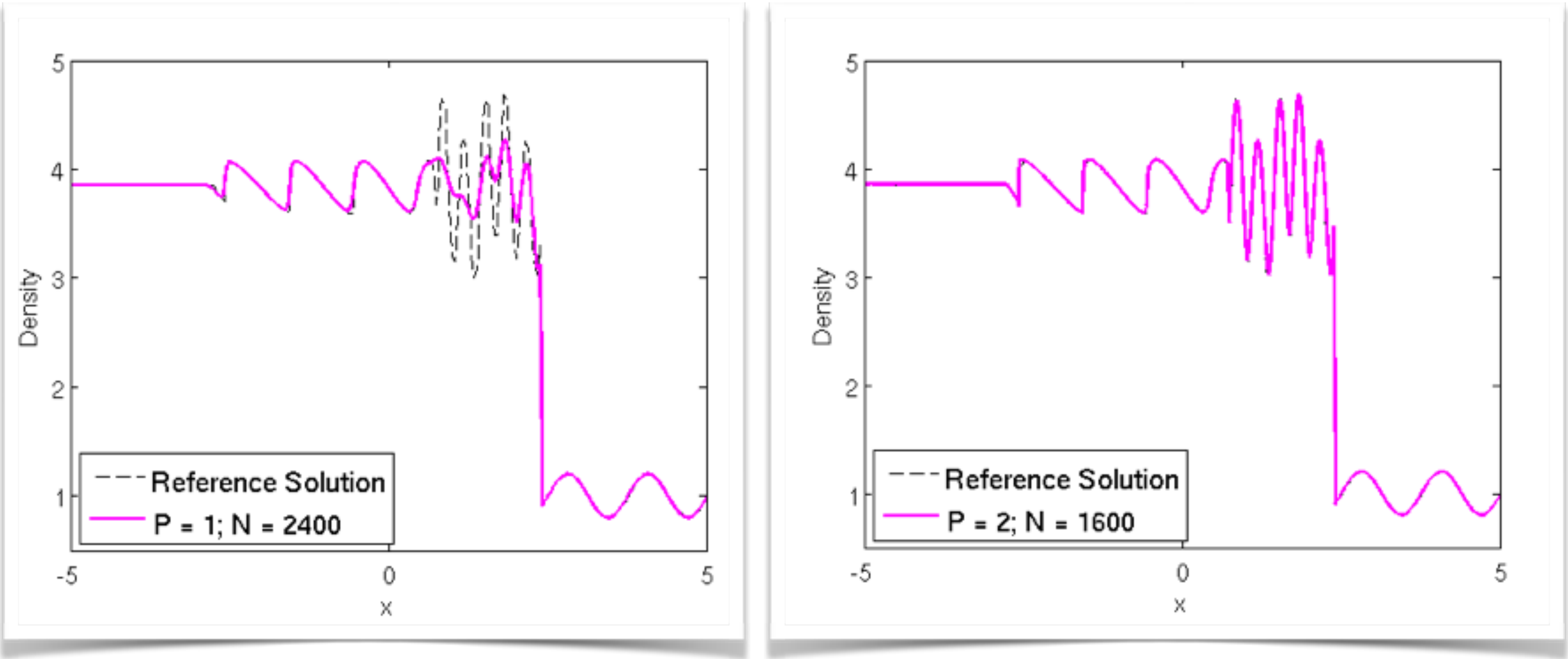




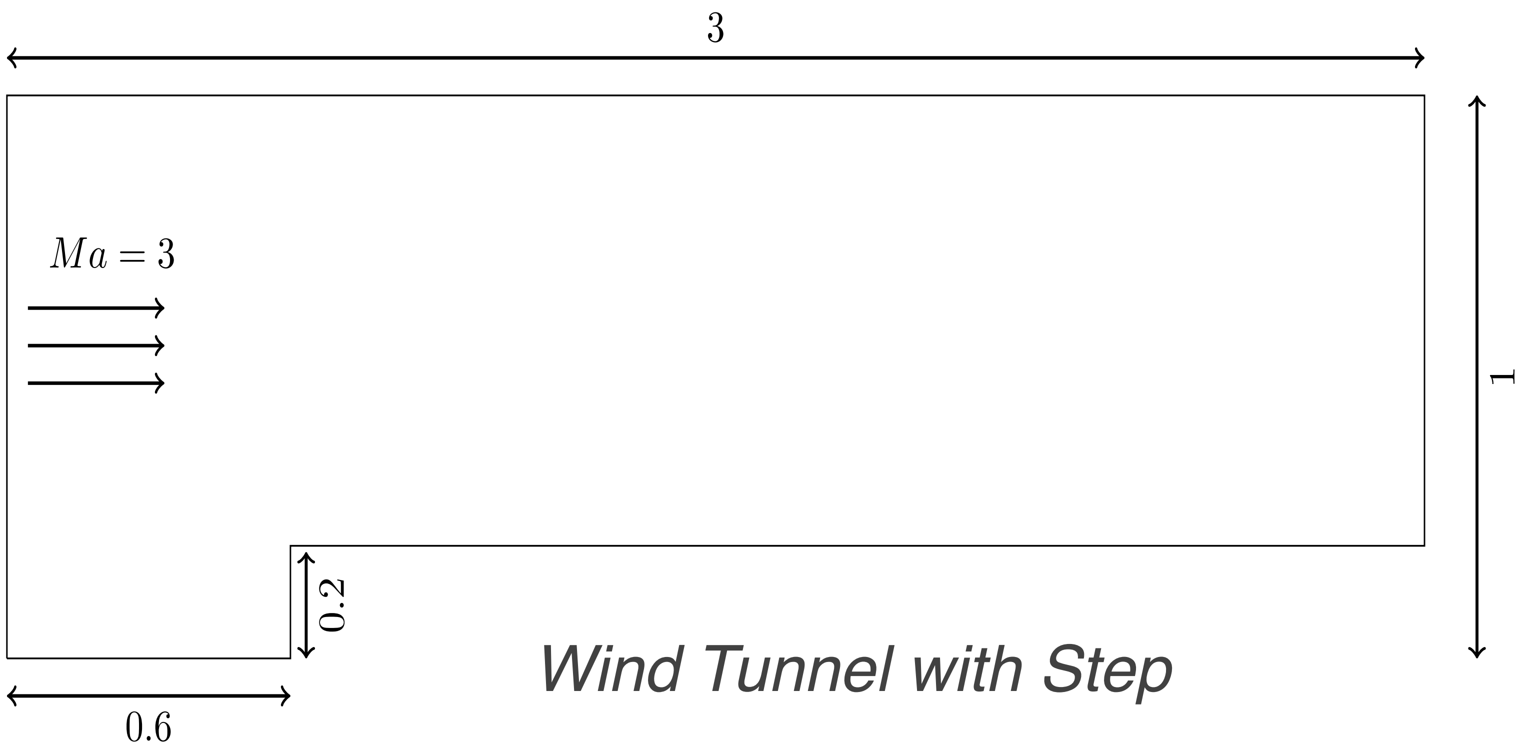




Shock Capturing: Shock Entropy Interaction



Shock Capturing: Flow Over a Step

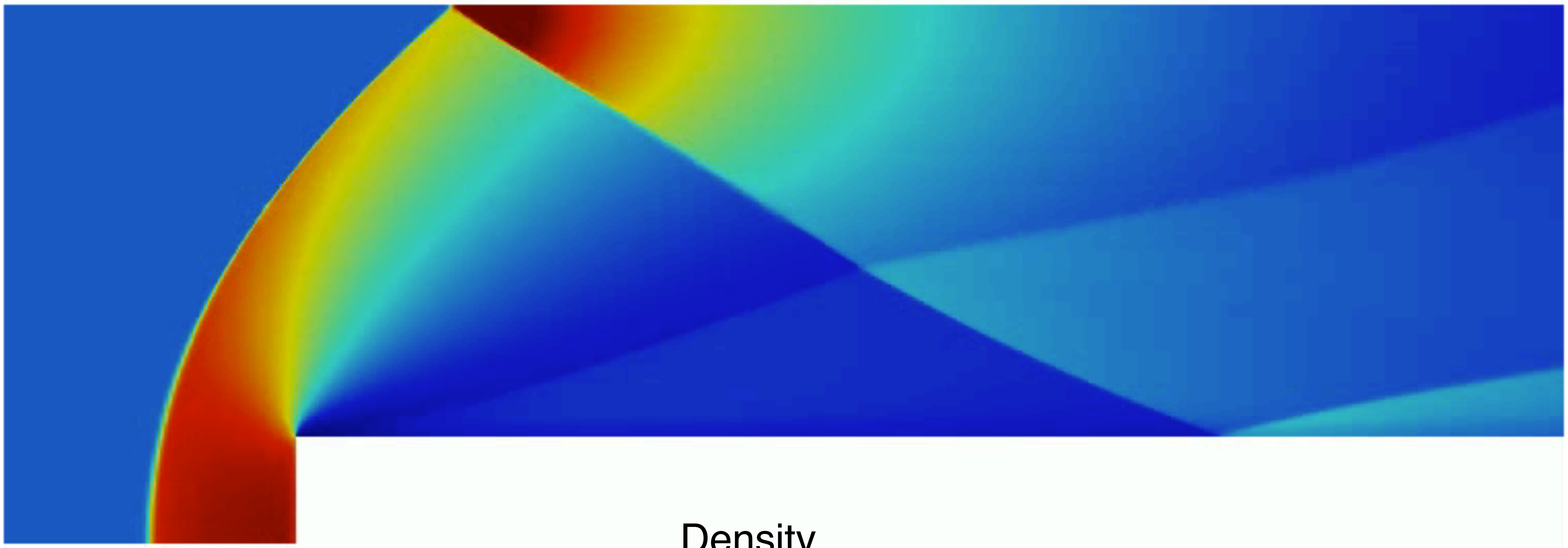


Wind Tunnel with Step

- Euler Equations
- Structured Quad Mesh
- Sensor at ramp
- Positivity Limiter

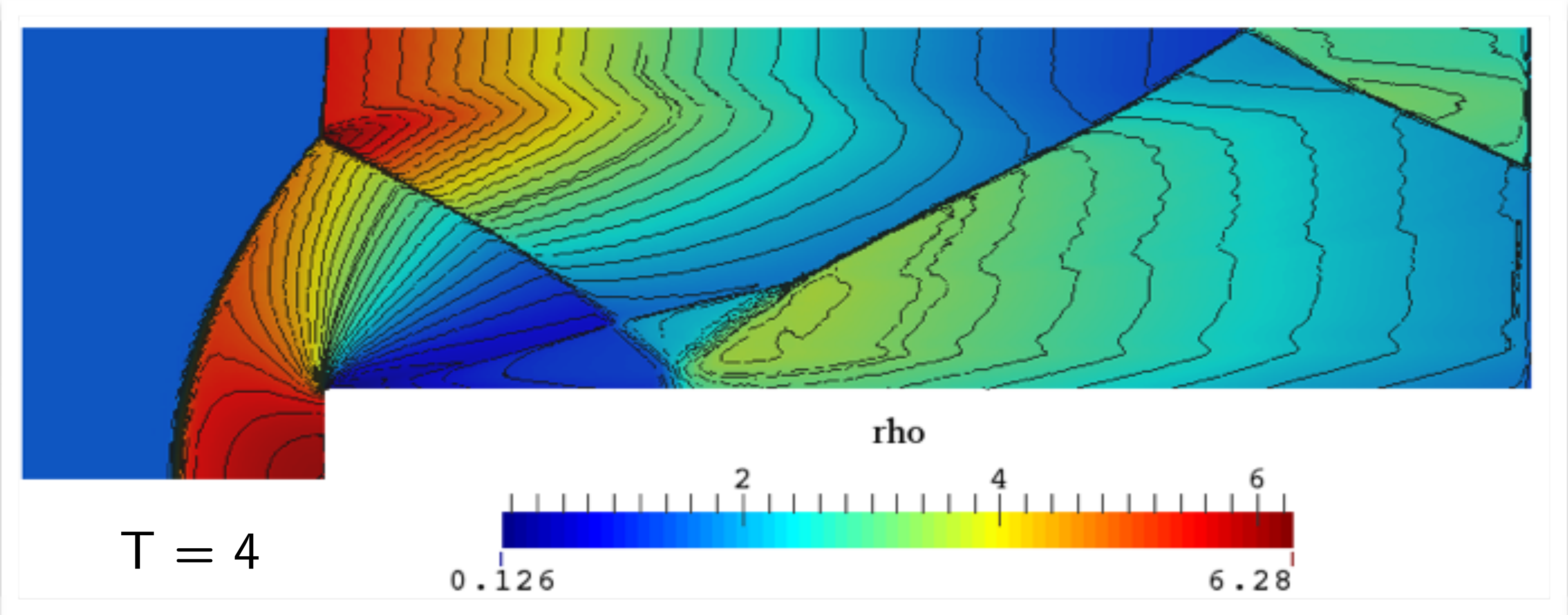
Mach	Flow Angle	Num Elem	Order	Filter Order	Filter Strength
3.0	0°	63,004	3	2	5

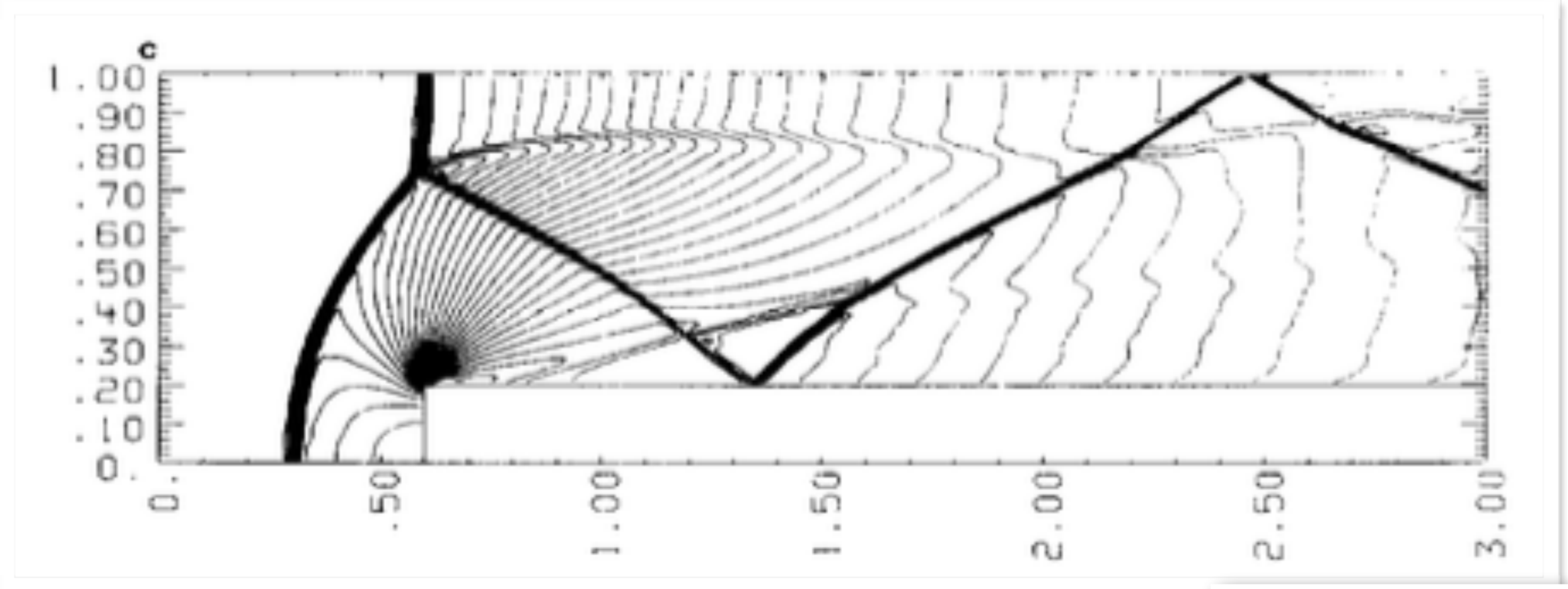
Shock Capturing: Flow Over a Step



Density

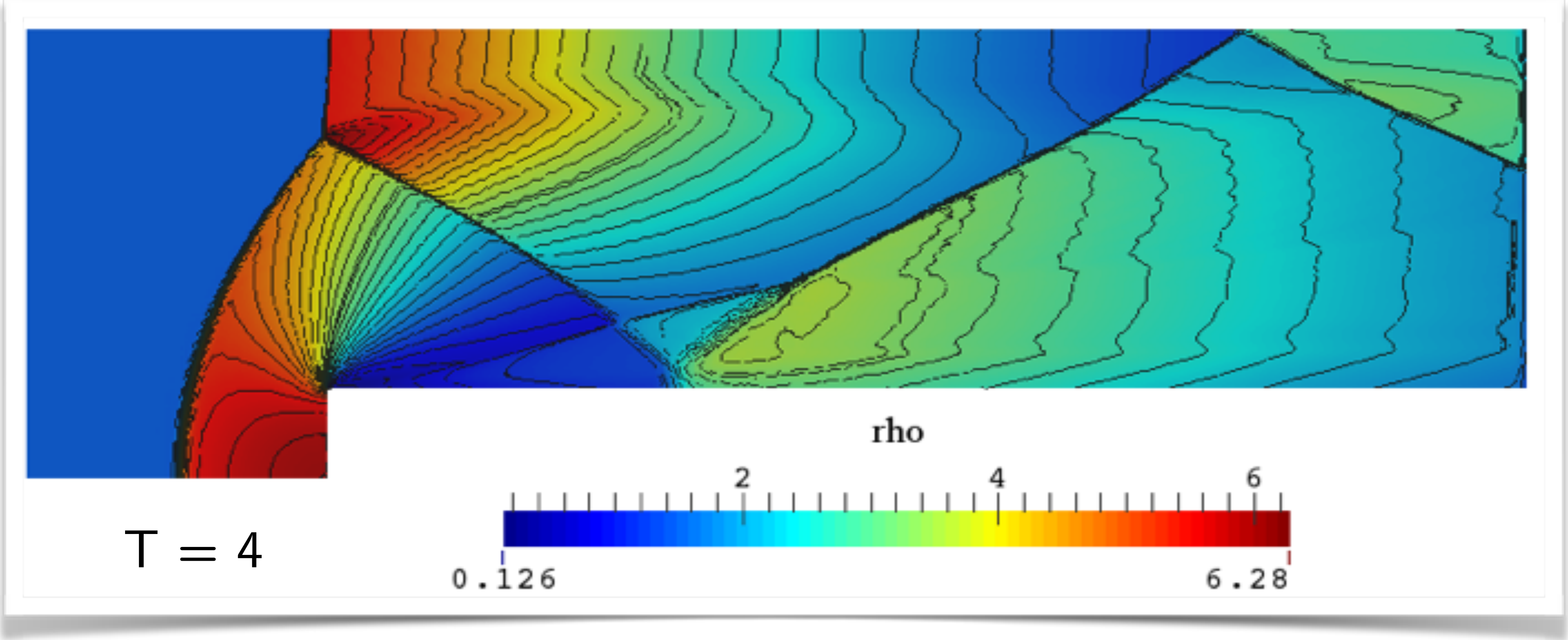
Shock Capturing: Flow Over a Step

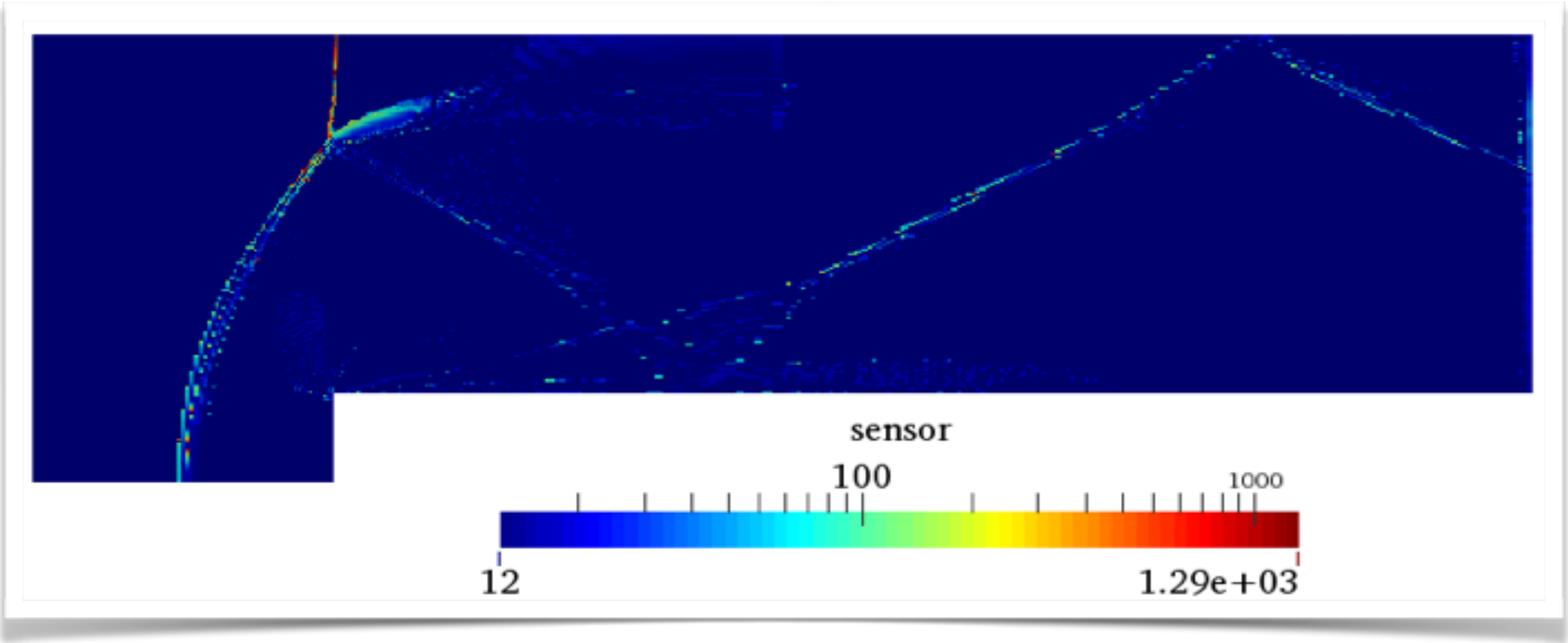




Reference: Woodward and Colella

Shock Capturing: Flow Over a Step





Convergence Acceleration

Recent work has focused on convergence acceleration on GPUs.

Convergence Acceleration: BDF1

- Fully discrete equation

$$\frac{\Delta \boldsymbol{u}_{\text{ele}}}{\Delta t} = \frac{(\boldsymbol{u}_{\text{ele}}^{n+1} - \boldsymbol{u}_{\text{ele}}^{n})}{\Delta t} = \boldsymbol{R}(\boldsymbol{u}_{\text{ele}}^{n+1}, \boldsymbol{u}_{\text{eleN}}^{n+1})$$

- Linearize to obtain global linear system

$$\left(\frac{I}{\Delta t} + \boxed{\frac{\partial \boldsymbol{R}_{\text{ele}}^{n}}{\partial \boldsymbol{u}_{\text{ele}}}}\right) \Delta \boldsymbol{u}_{\text{ele}} - \sum_{\text{eleN}} \boxed{\frac{\partial \boldsymbol{R}_{\text{ele}}^{n}}{\partial \boldsymbol{u}_{\text{eleN}}}} \Delta \boldsymbol{u}_{\text{eleN}} = \boldsymbol{R}(\boldsymbol{u}_{\text{ele}}^{n}, \boldsymbol{u}_{\text{eleN}}^{n})$$

Element local Jacobian

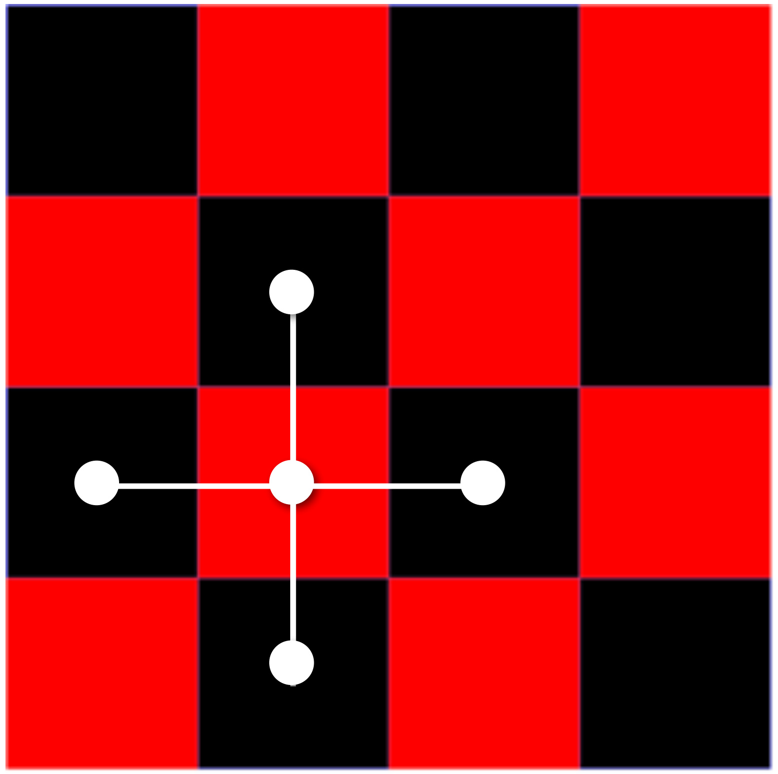
Element neighbor Jacobian

Convergence Acceleration: BDF1

- Solve using multicolored Gauss-Seidel.
- For example with red/black coloring:

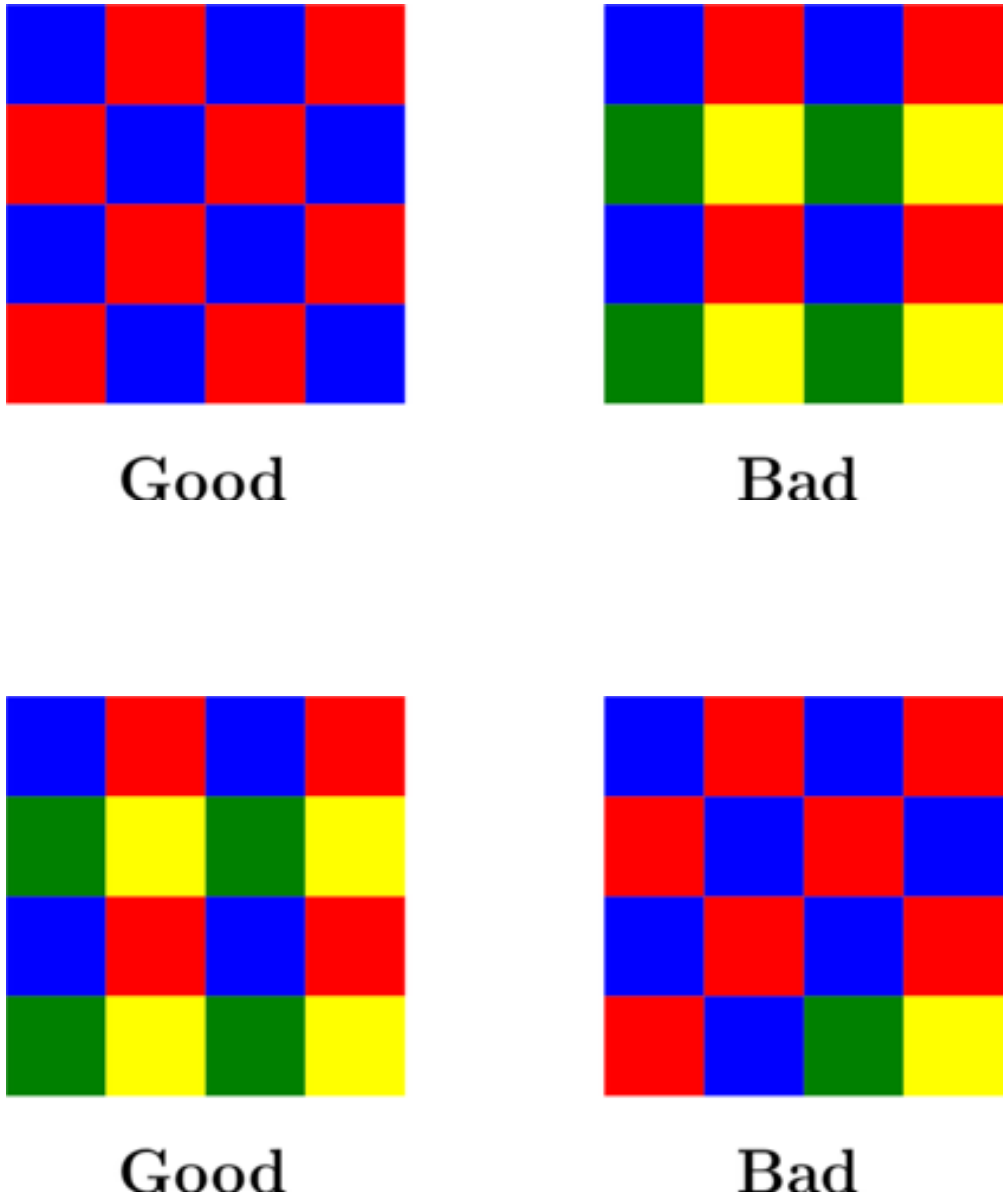
$$\begin{pmatrix} D_R & C_B \\ C_R & D_B \end{pmatrix} \begin{pmatrix} x_R \\ x_B \end{pmatrix} = \begin{pmatrix} b_R \\ b_B \end{pmatrix}$$

$$x_R^{n+1} = D_R^{-1} \left(b_R - C_B x_B^n \right)$$
$$x_B^{n+1} = D_B^{-1} \left(b_B - C_R x_R^n \right)$$

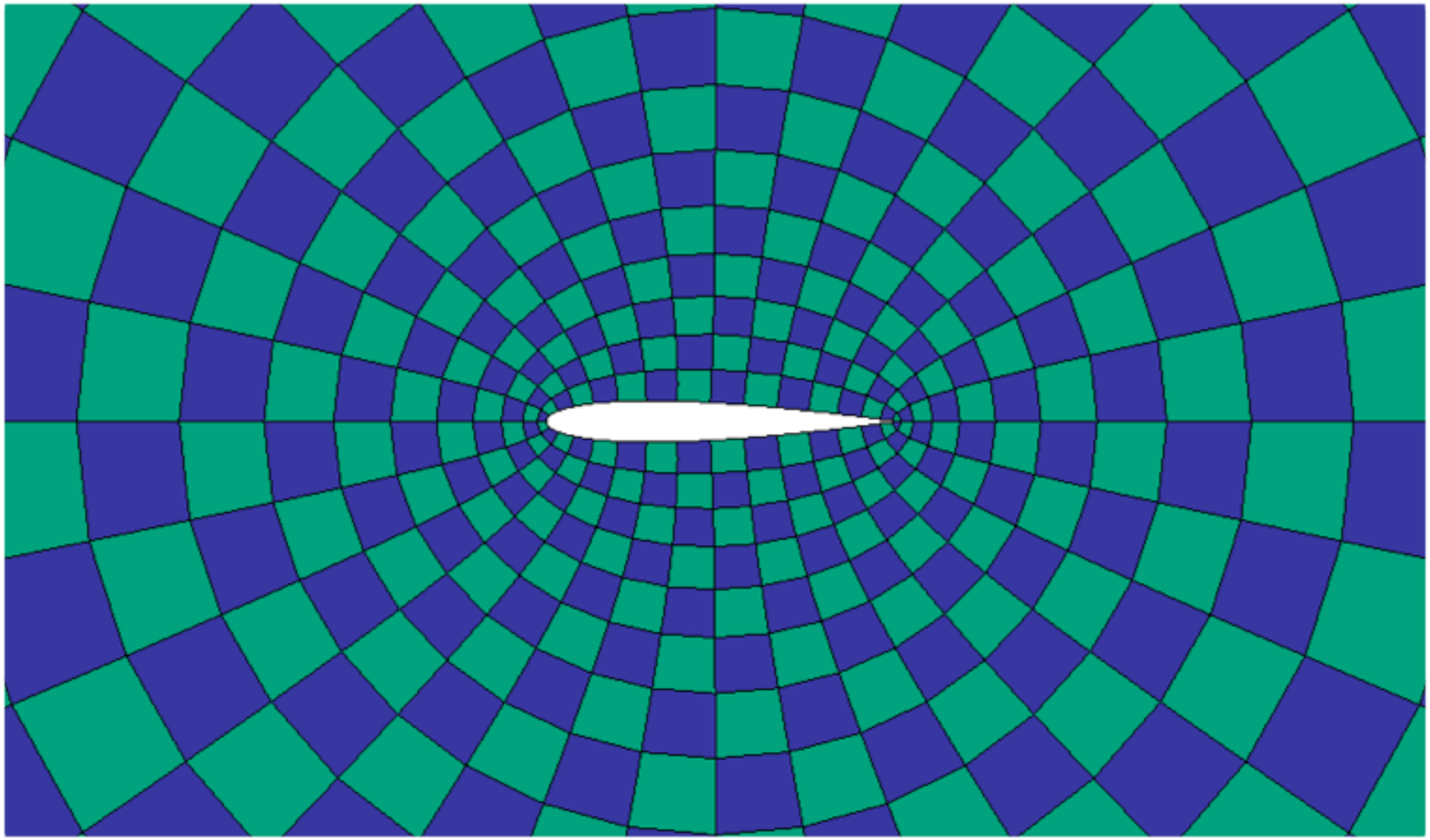


Convergence Acceleration: Mesh Coloring

- **Requirements**
  - Minimise number of colours
  - Distribute work evenly

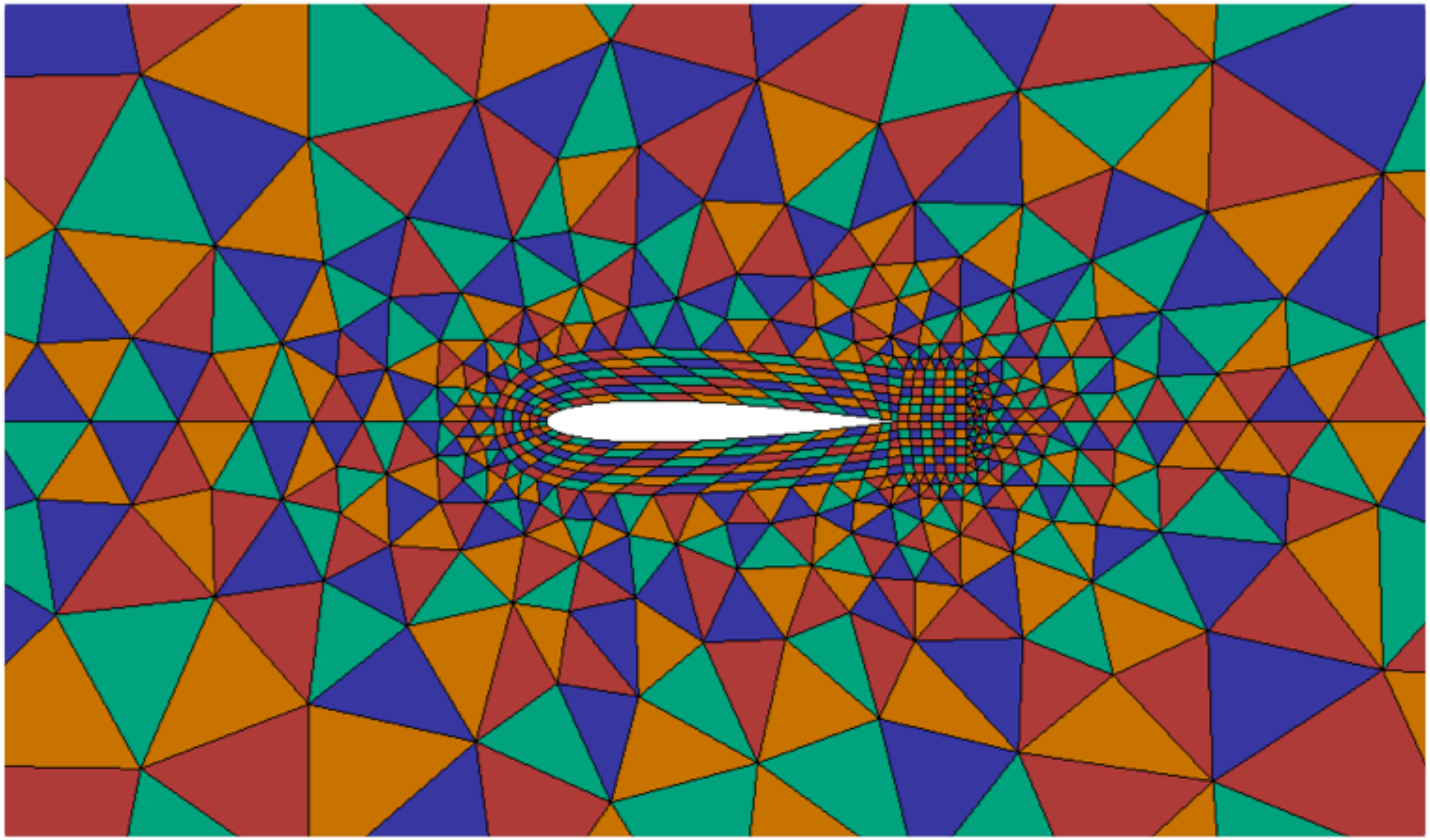


Convergence Acceleration: Mesh Coloring



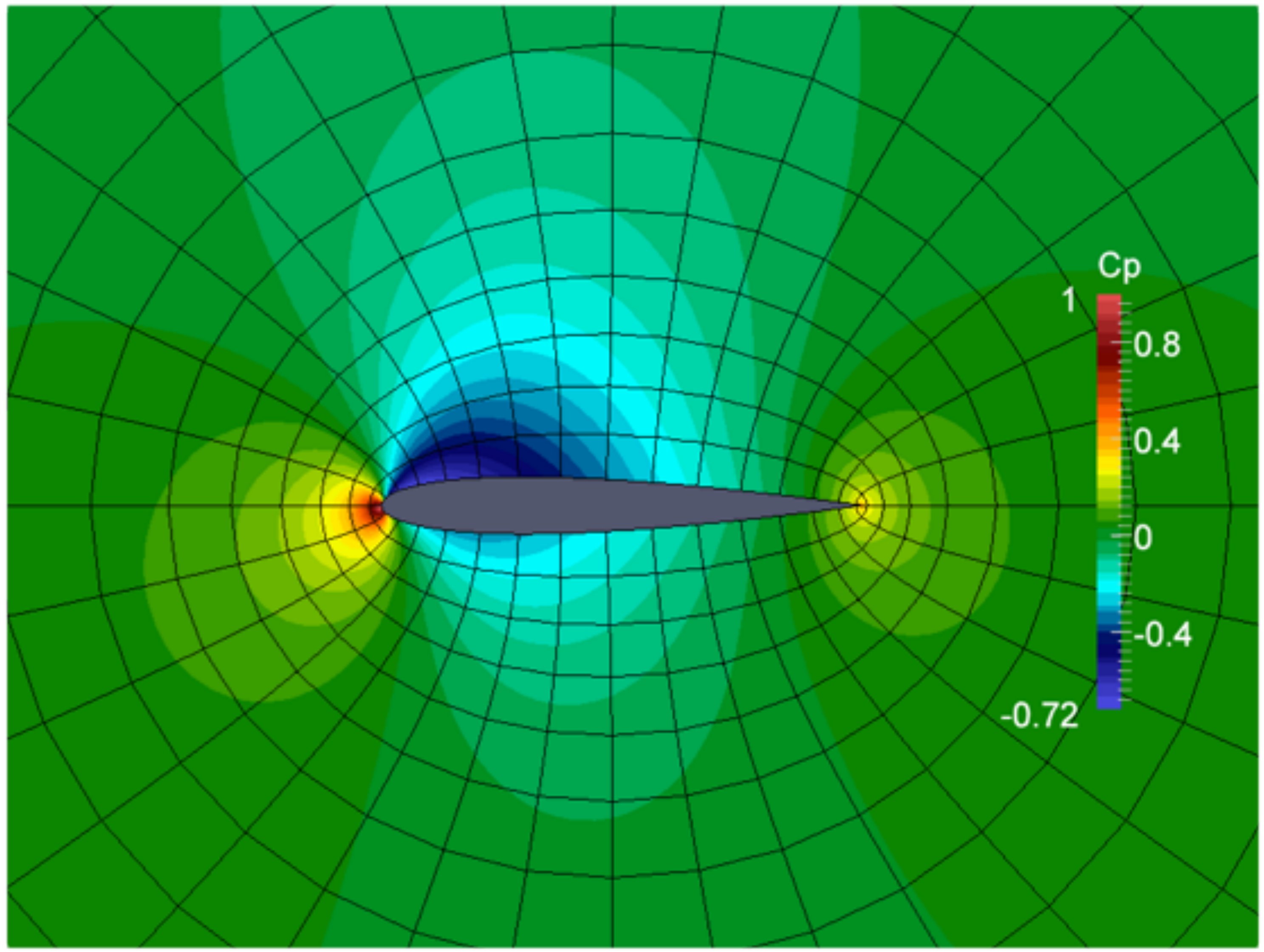
Structured NACA 0012

Convergence Acceleration: Mesh Coloring



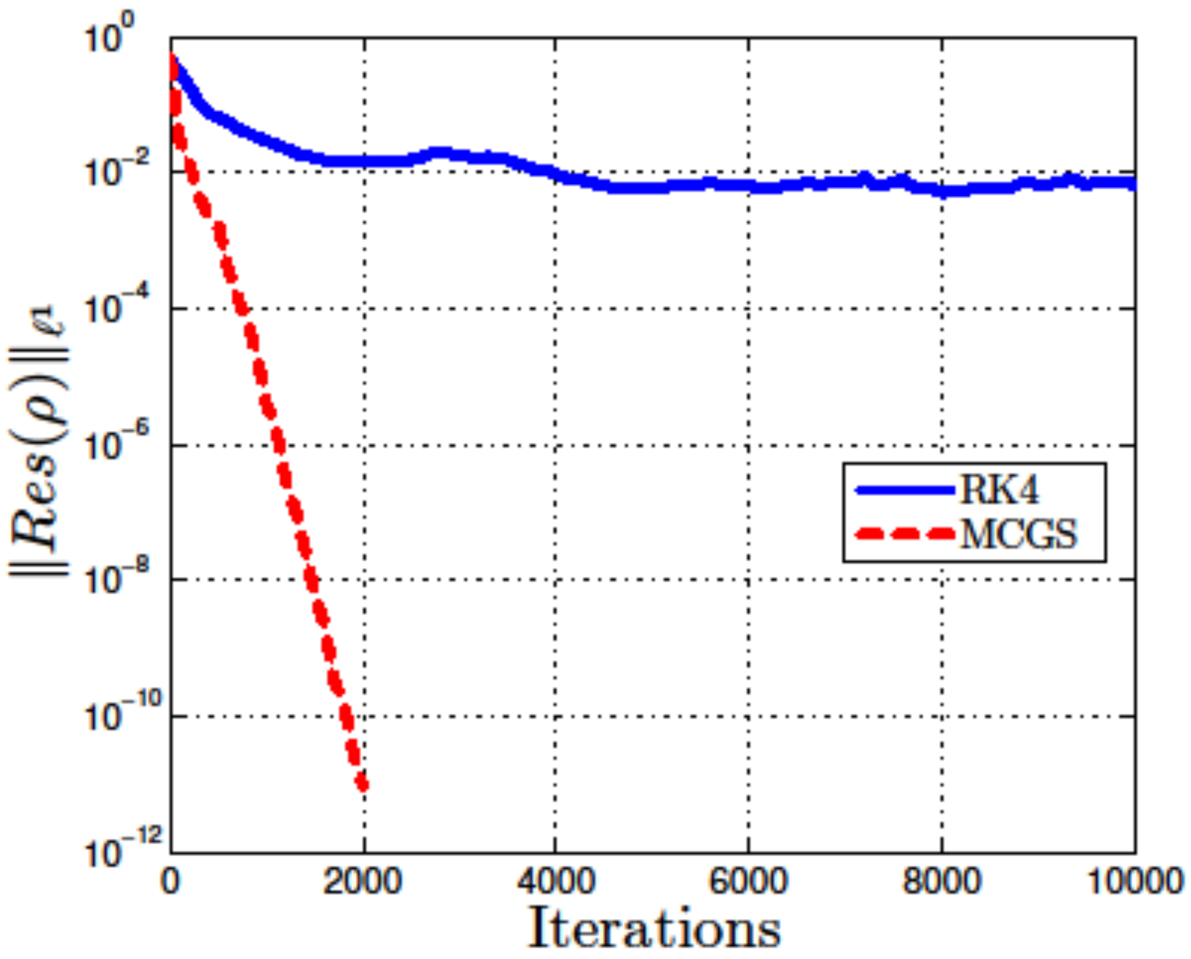
Unstructured NACA 0012

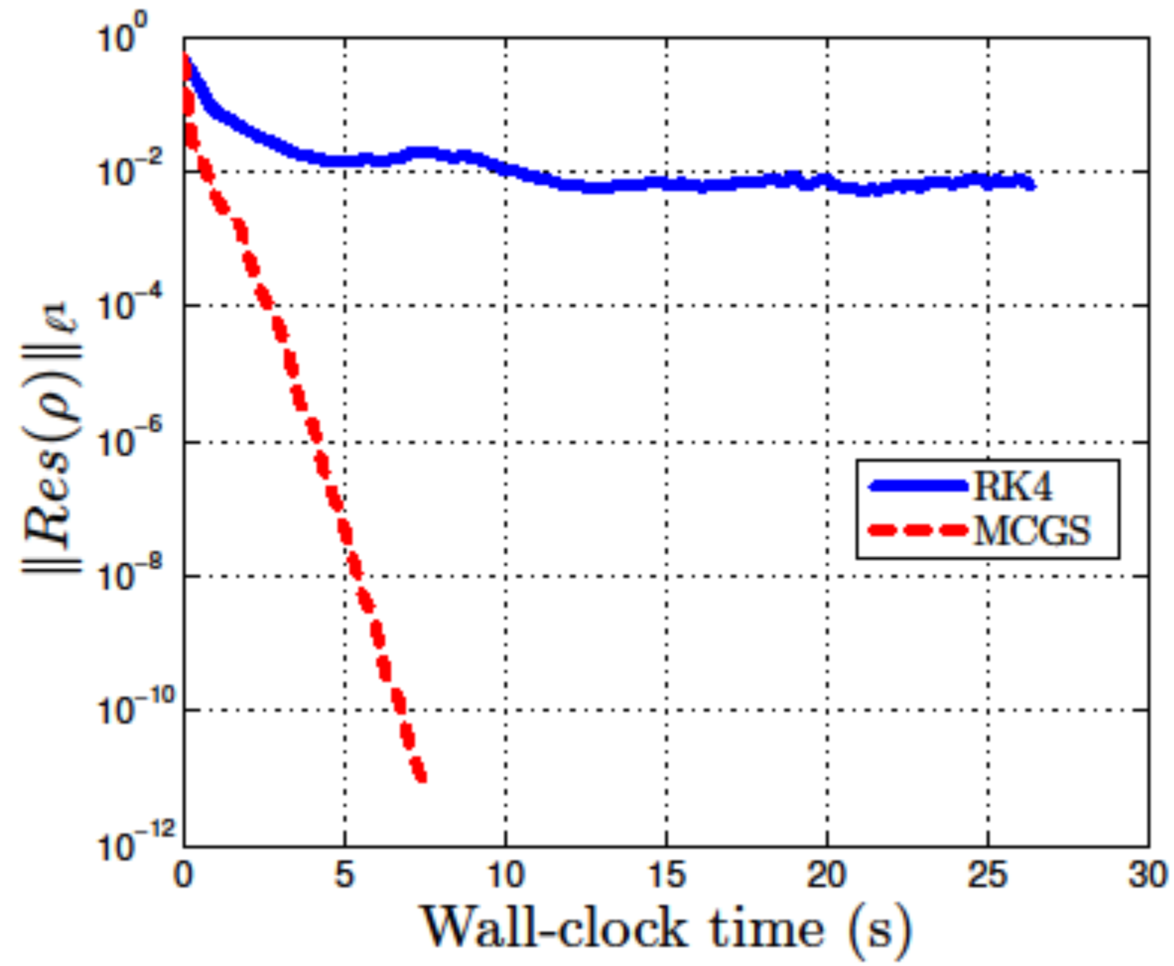
Convergence Acceleration: NACA 0012



Euler eq, NACA 0012, 32 by 32 grid, P = 4, Ma = 0.5, α = 1.25°

Convergence Acceleration: NACA 0012





Rapid improvement compared with explicit RK4.

Outline

I. Context

II. Current status of CFD

III. Flux Reconstruction

IV. PyFR

V. LES computations and future work

PyFR

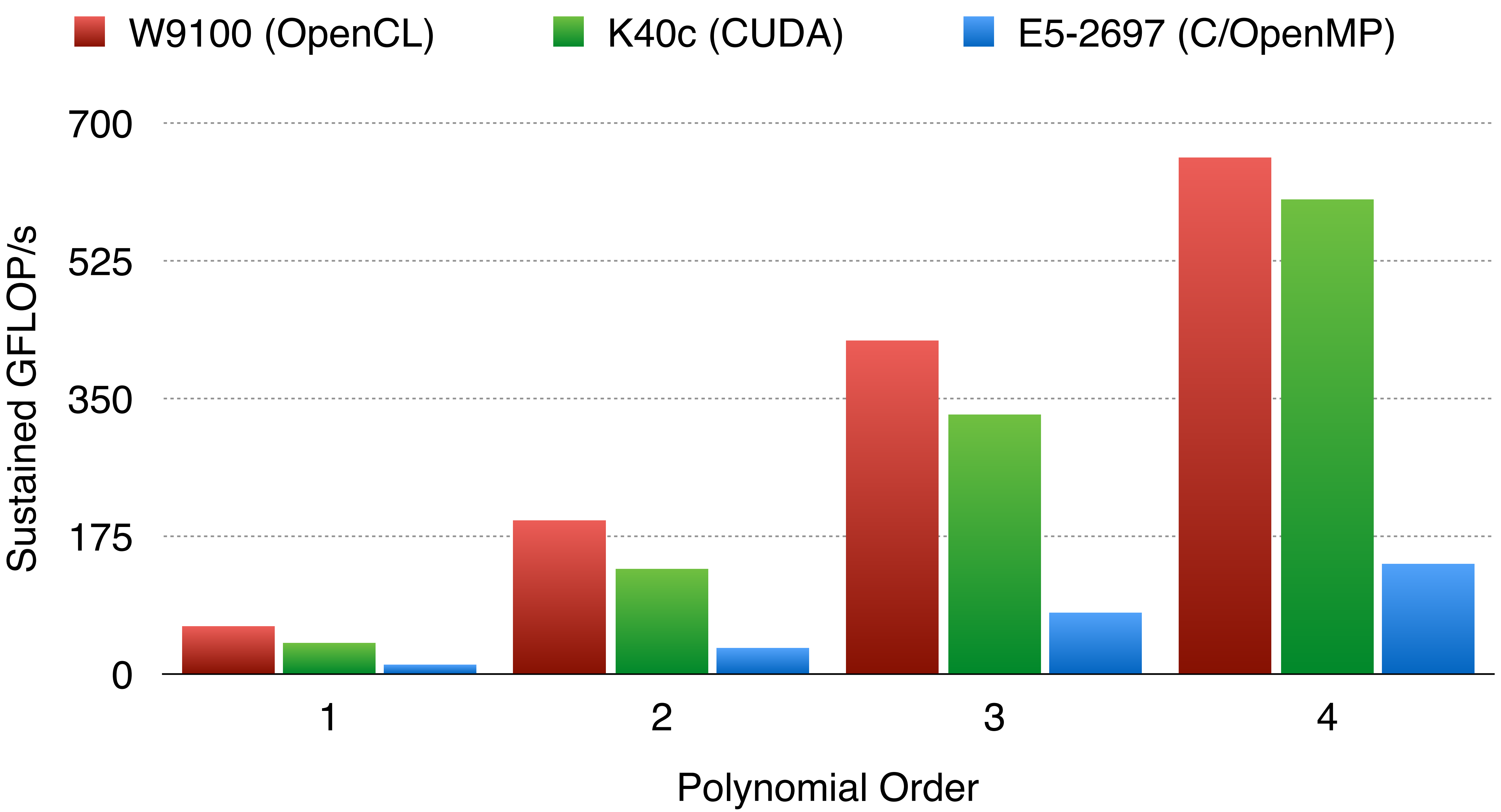
- Open source implementation of FR for modern hardware.
- Started at Imperial College London
  - PI: Peter Vincent.
  - Lead developer: Freddie Witherden
  - Many other contributors!

PyFR

Governing Equations	Compressible Euler/Navier-Stokes (Incompressible Euler/Navier-Stokes)
Spatial Discretisation	Arbitrary order FR on mixed unstructured grids
Temporal Discretisation	Range of explicit Runge-Kutta schemes
Backends	CPUs, NVIDIA GPUs, AMD GPUs, (Intel MIC).
Precision	Single, Double
Input	Gmsh, (CGNS)
Output	VTK, (In situ)

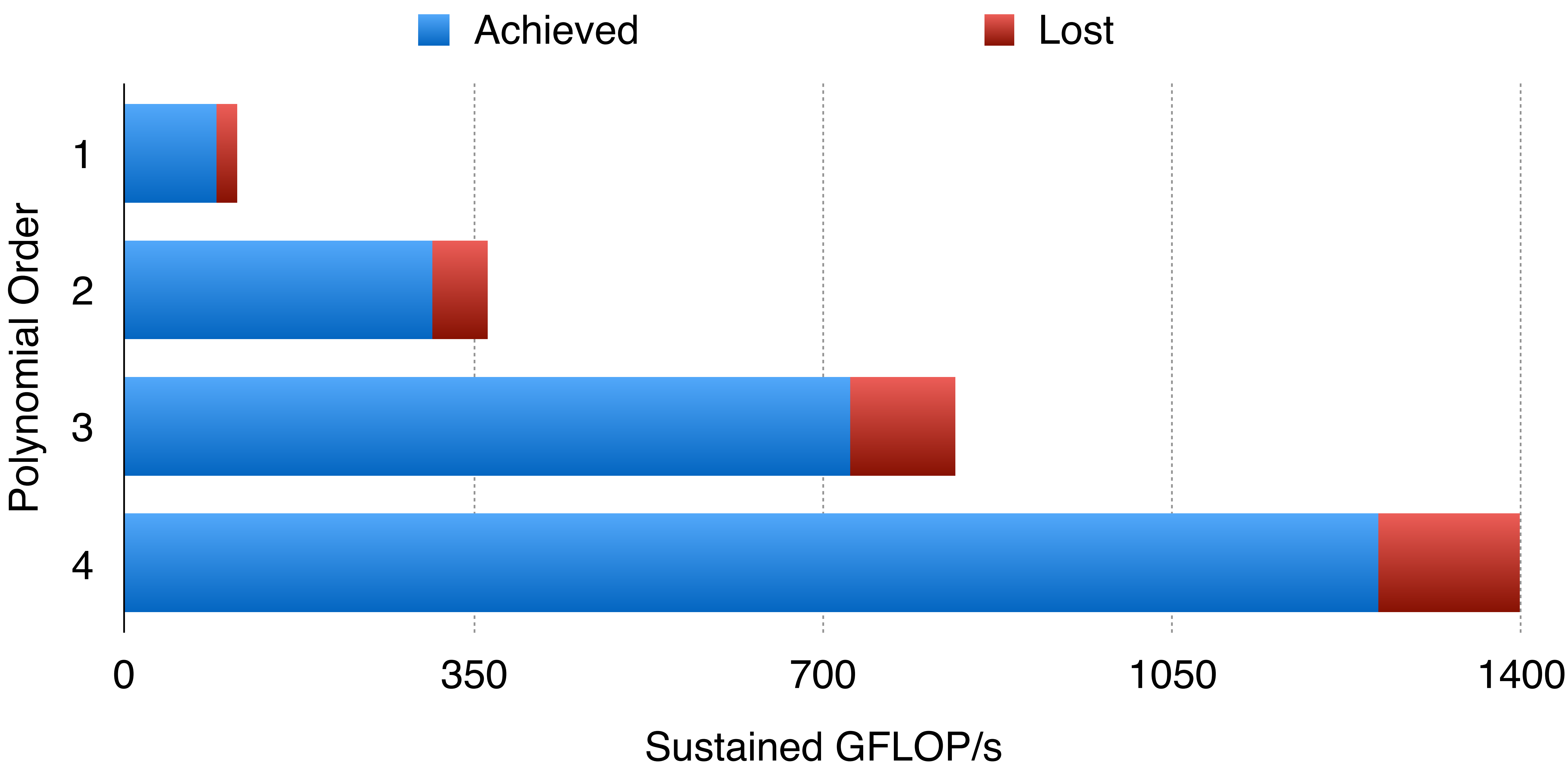
PyFR

- Single node performance on a mixed prism/tet grid.



PyFR

- Multi node heterogeneous performance on the same grid.

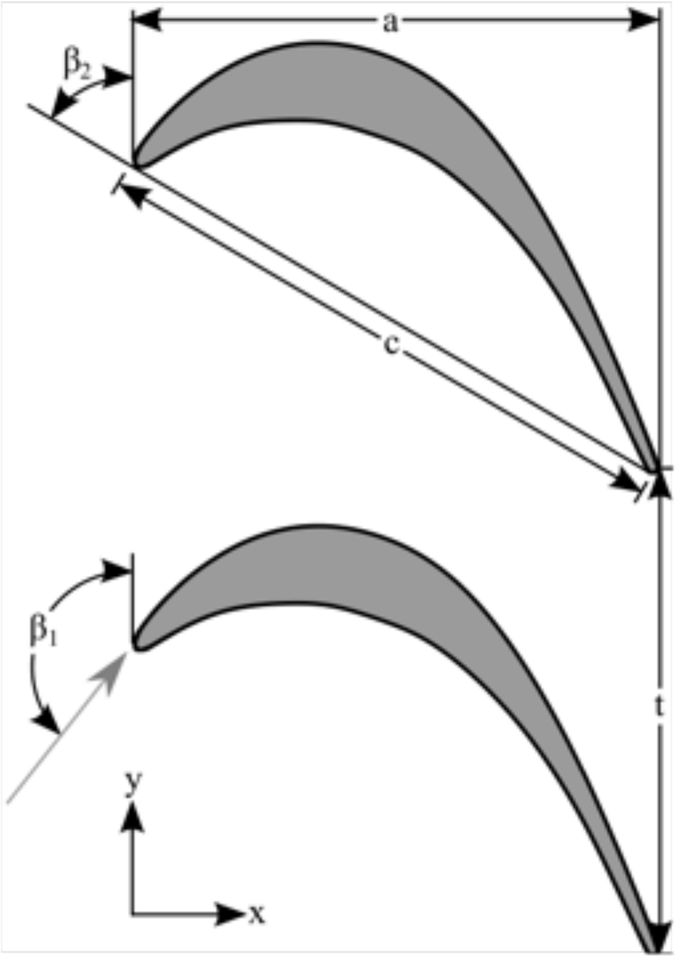


PyFR

- Scaling evaluated on the *Titan* cluster at *ORNL*.
- Most powerful GPU cluster with 18,000 NVIDIA K20X GPUs.
- Test case is a turbine blade with fourth order solution polynomials.

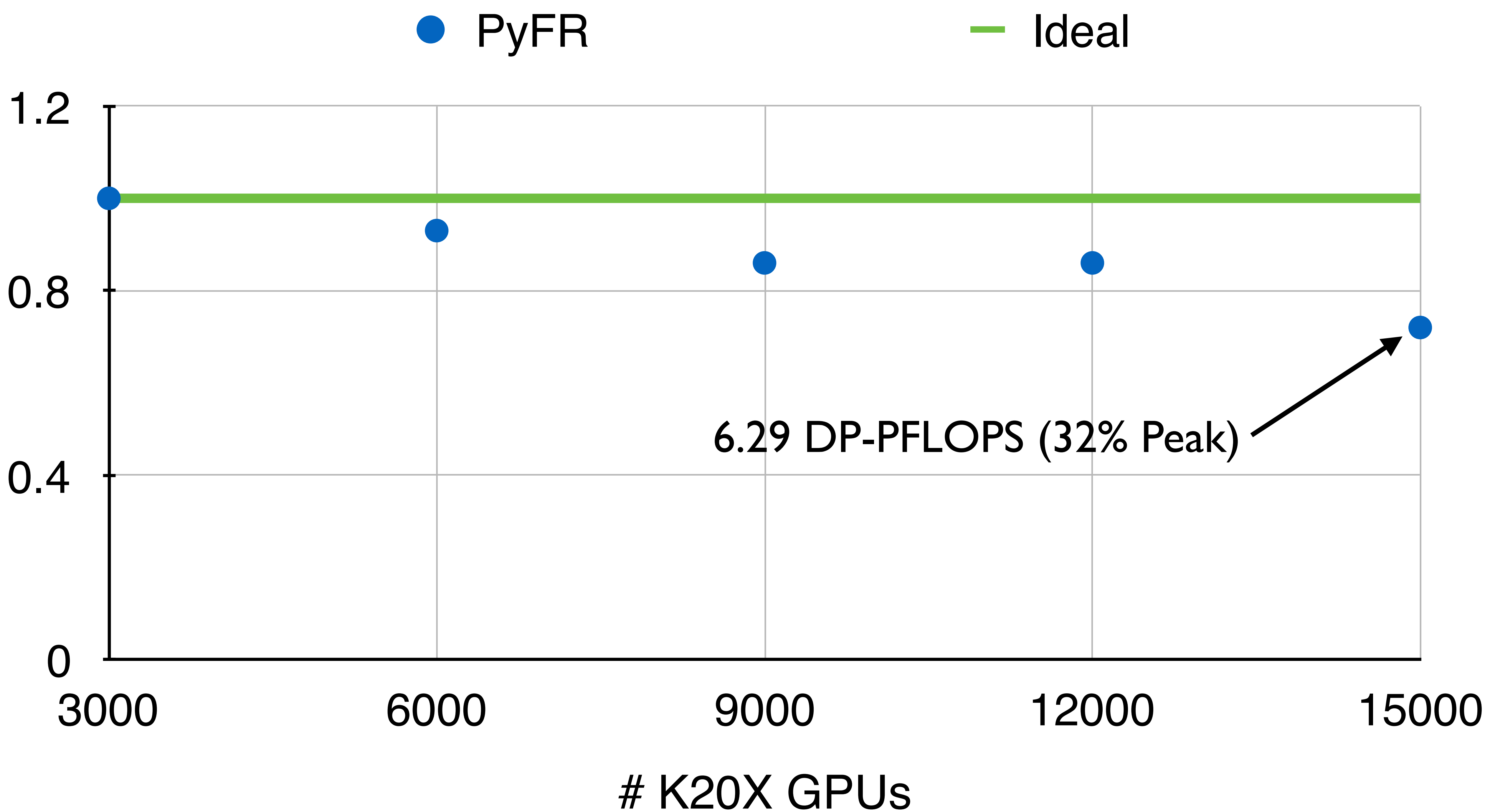
PyFR

T106d Cascade



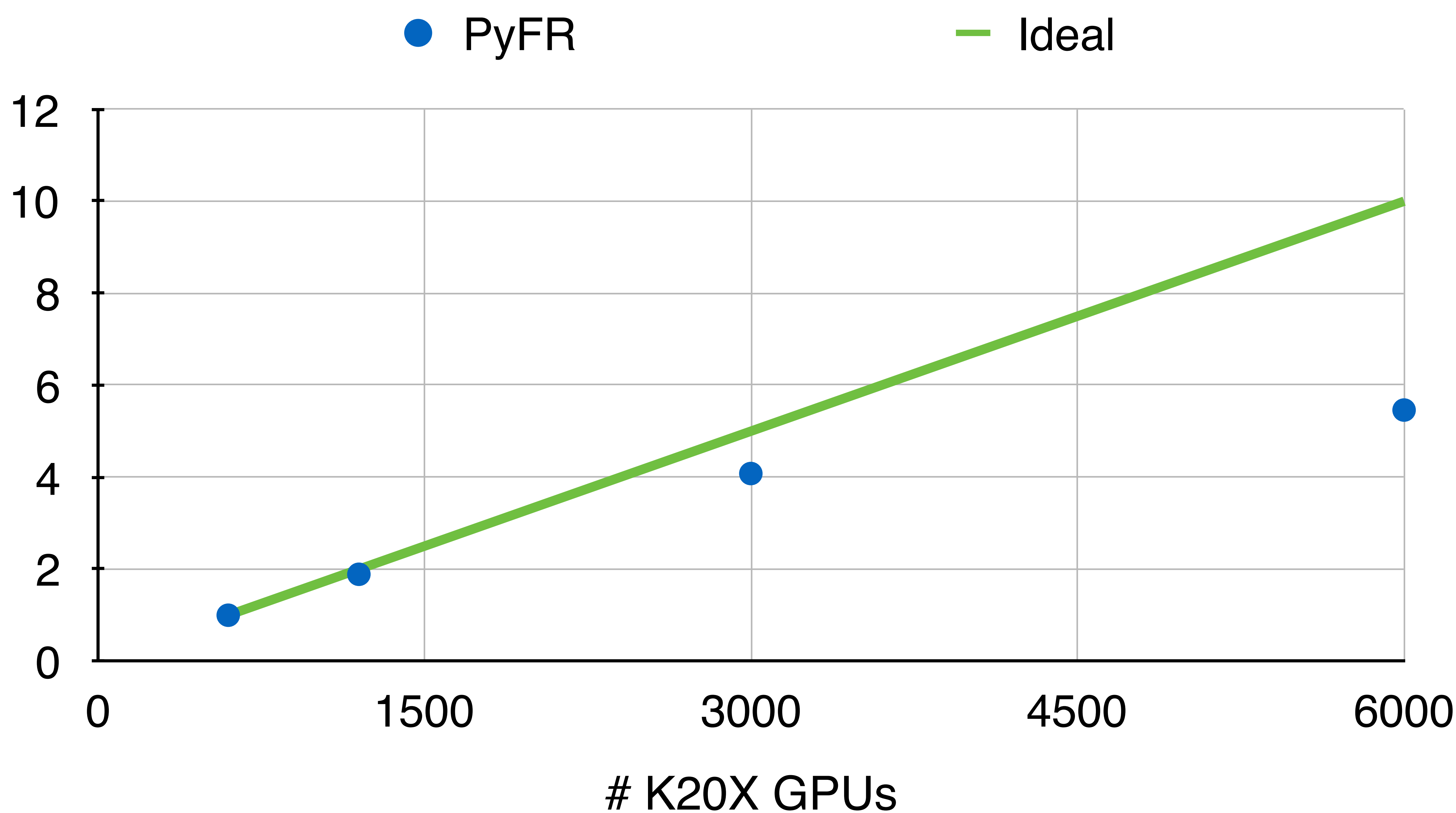
PyFR

- Weak scaling



PyFR

- Strong scaling



Outline

I. Context

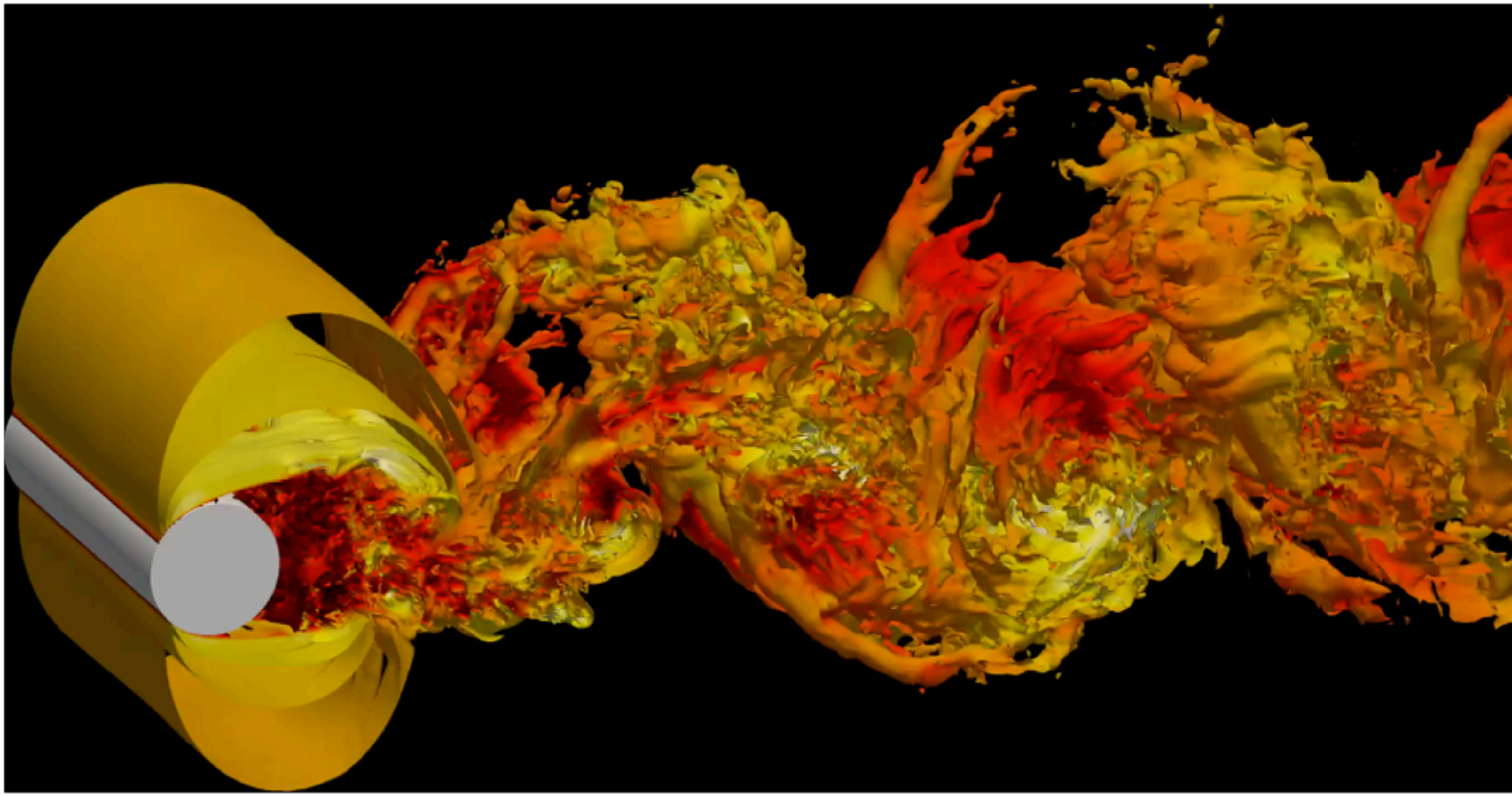
II. Current status of CFD

III. Flux Reconstruction

IV. PyFR

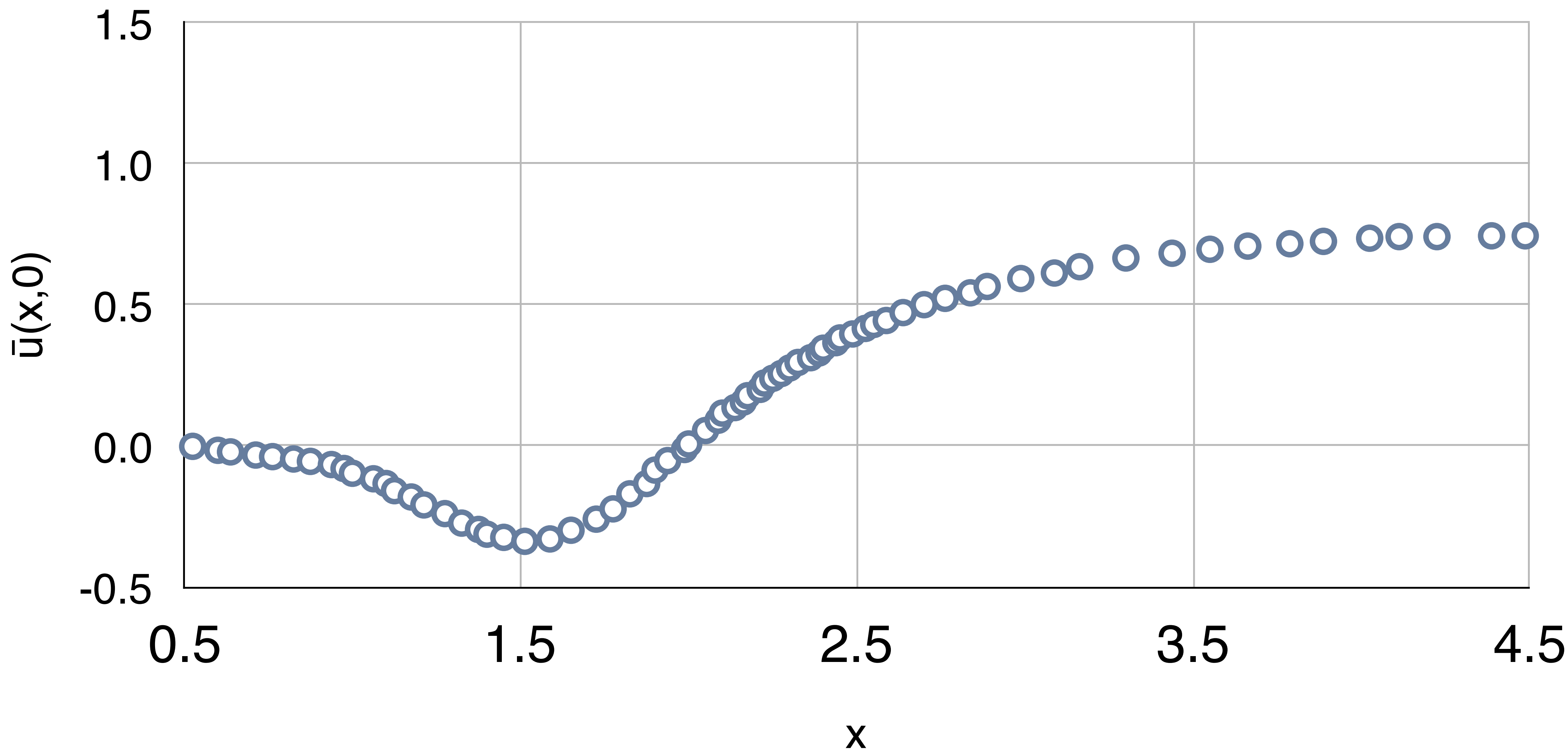
V. LES computations and future work

Flow past a Circular Cylinder: $Re_D = 3600$



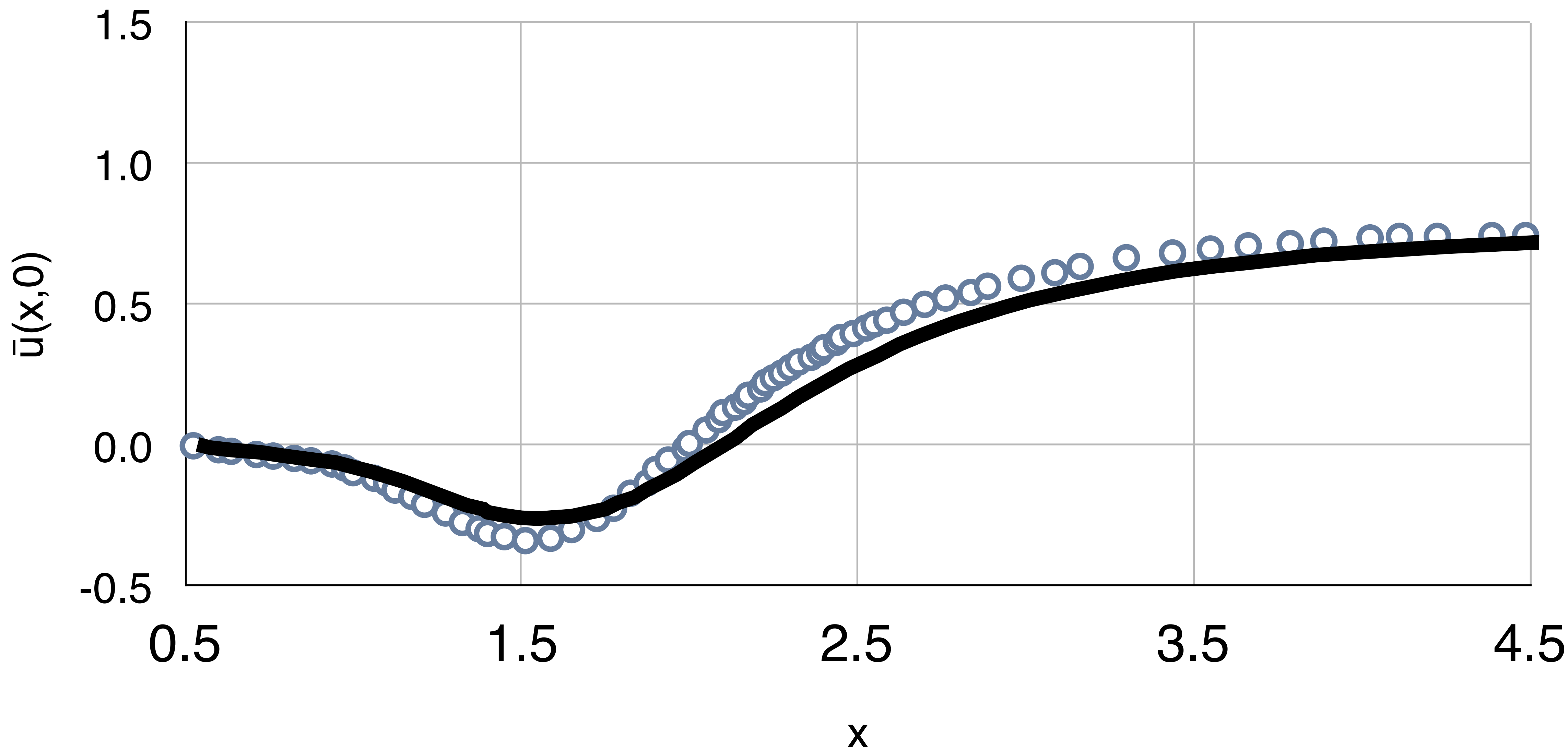
Flow past a Circular Cylinder: Re_D = 3600

- Parnaudeau et al. experiment.



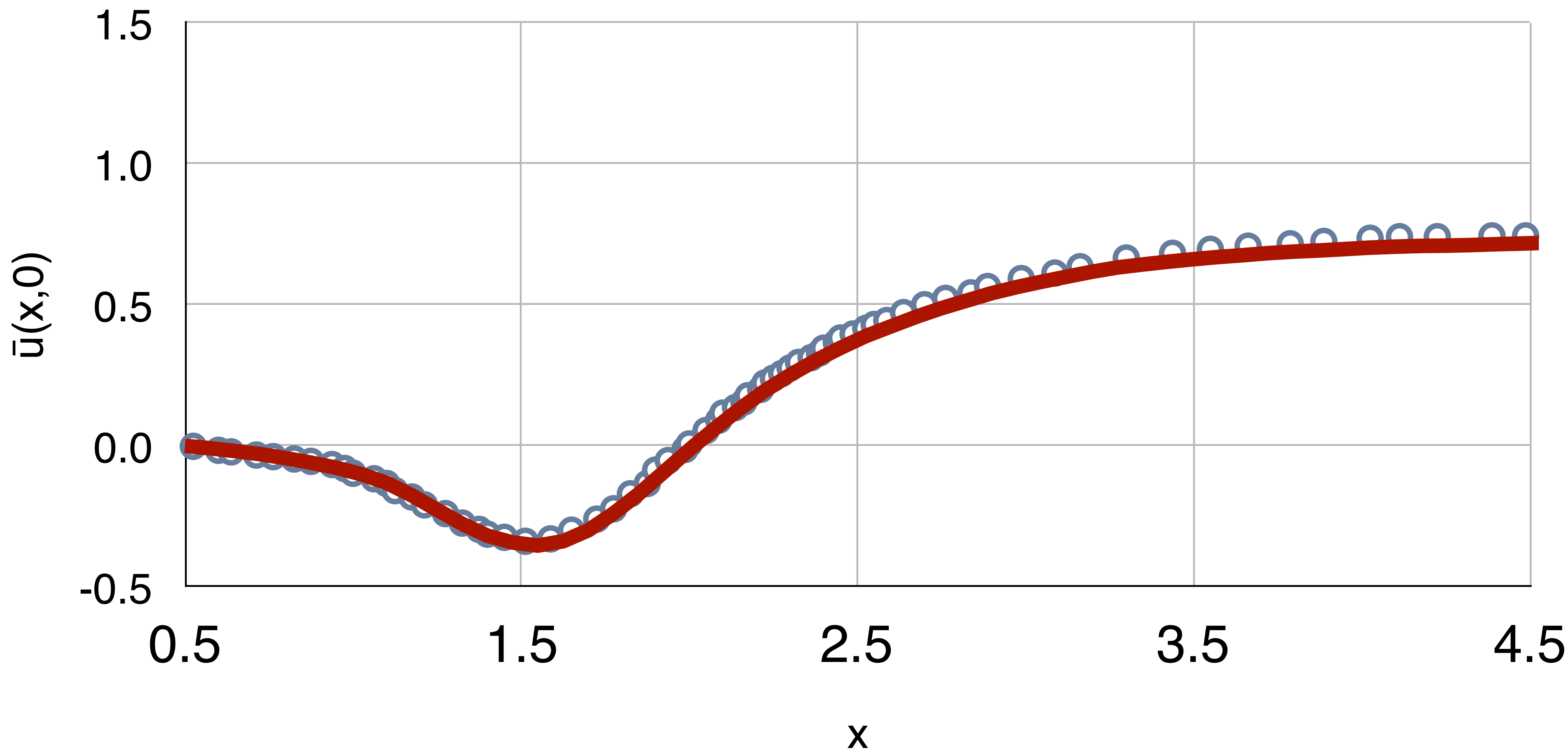
Flow past a Circular Cylinder: Re_D = 3600

- Parnaudeau et al. experiment + Parnaudeau et al. LES.



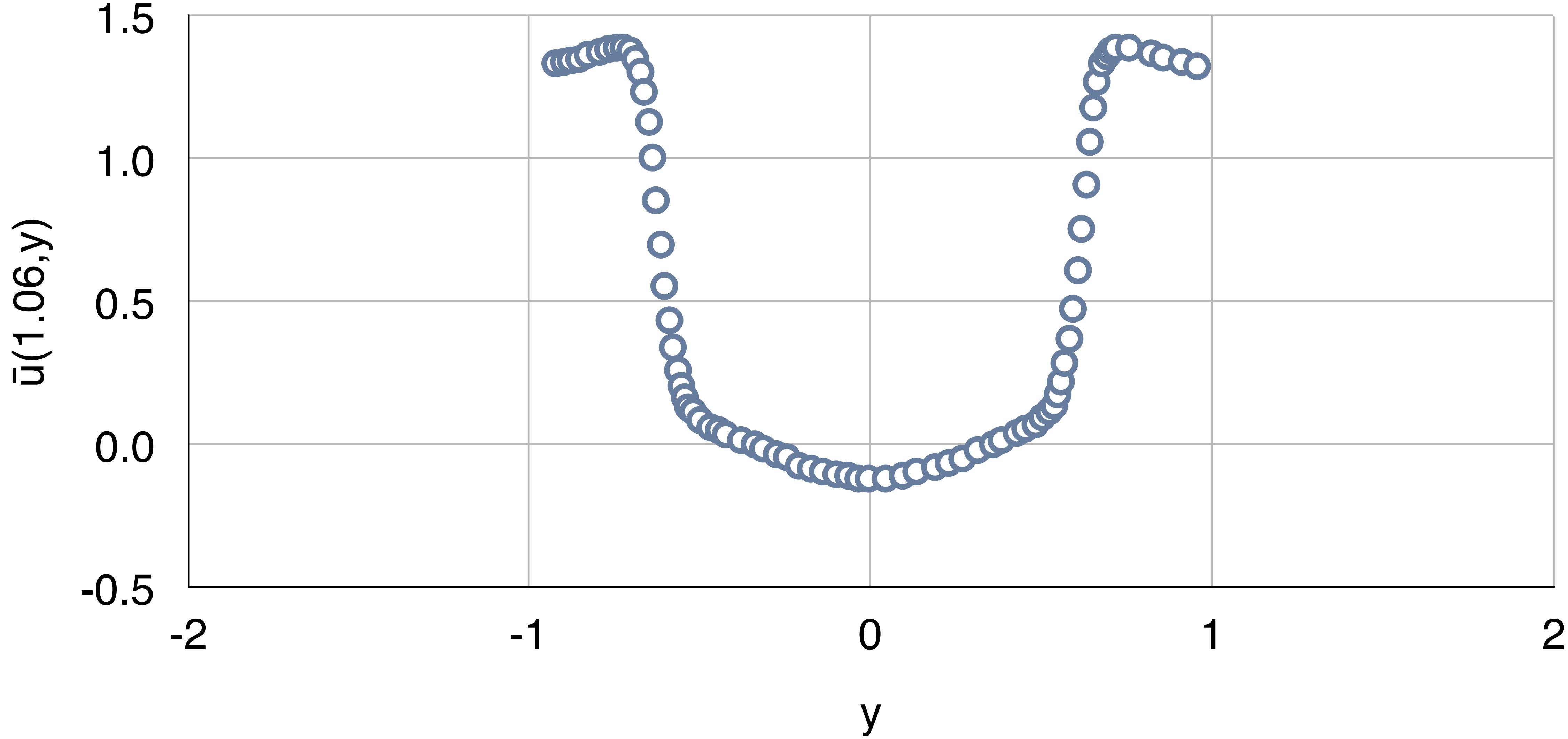
Flow past a Circular Cylinder: Re_D = 3600

- Parnaudeau et al. experiment + PyFR (5th order hex) ILES.



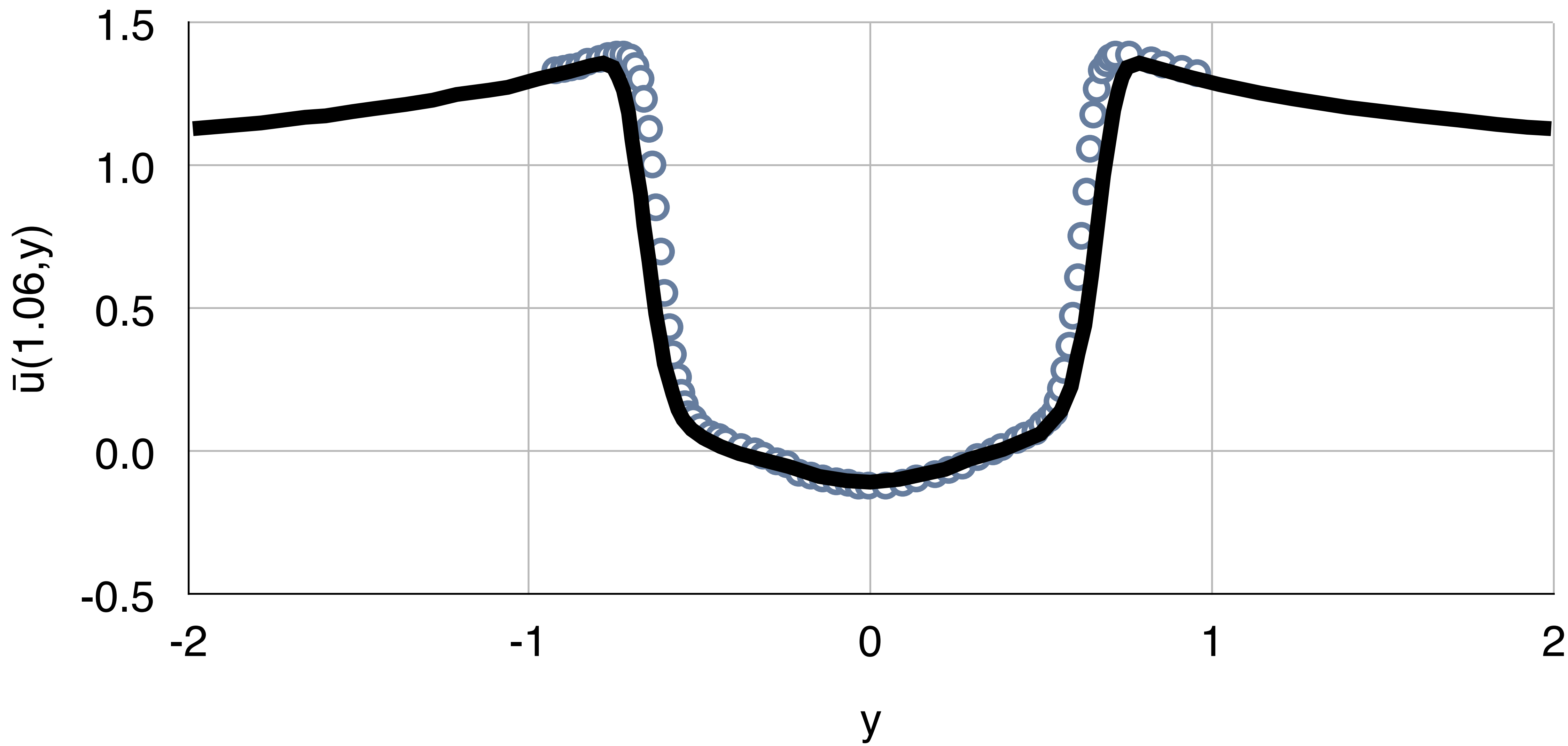
Flow past a Circular Cylinder: $Re_D = 3600$

- Parnaudeau et al. experiment.



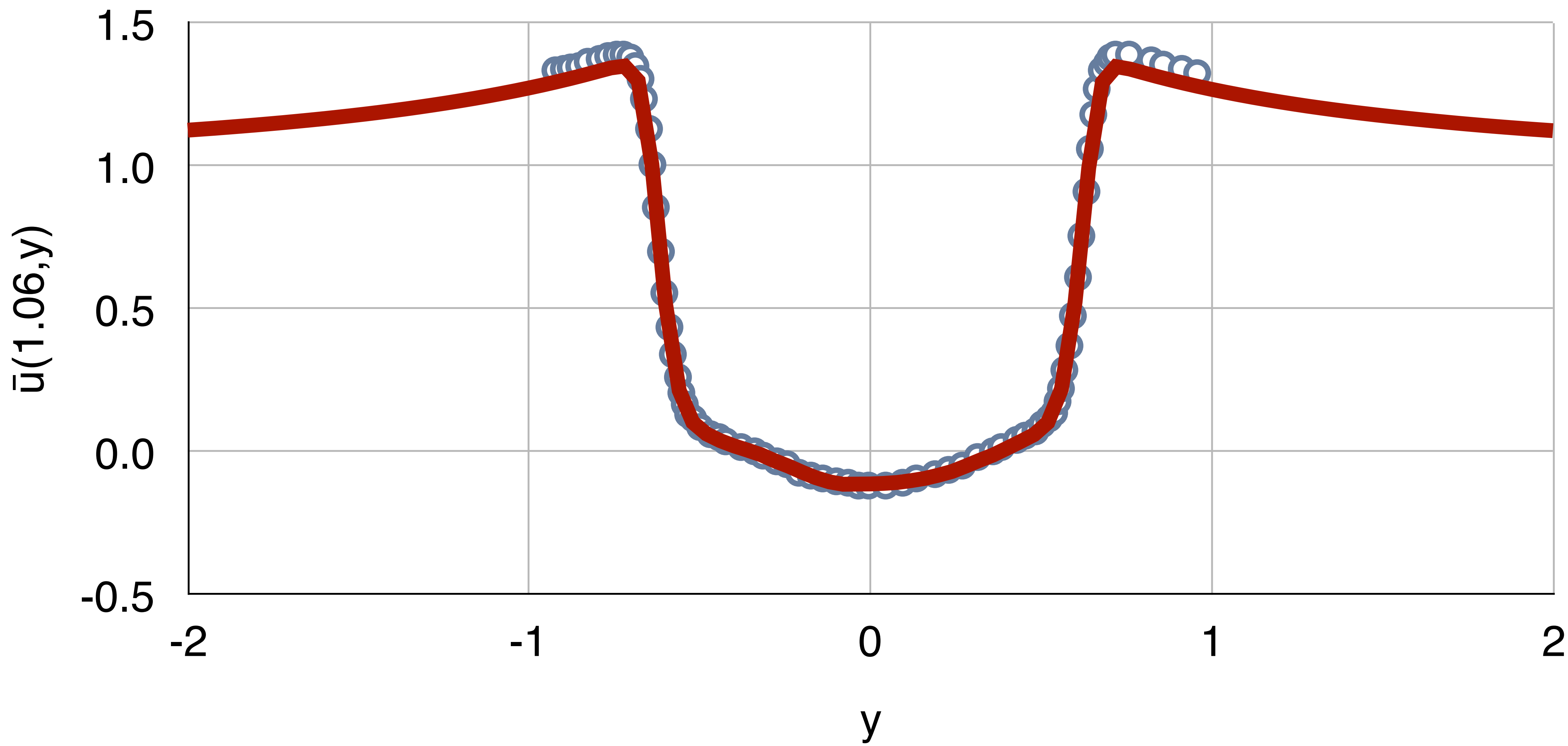
Flow past a Circular Cylinder: $Re_D = 3600$

- Parnaudeau et al. experiment + Parnaudeau et al. LES.



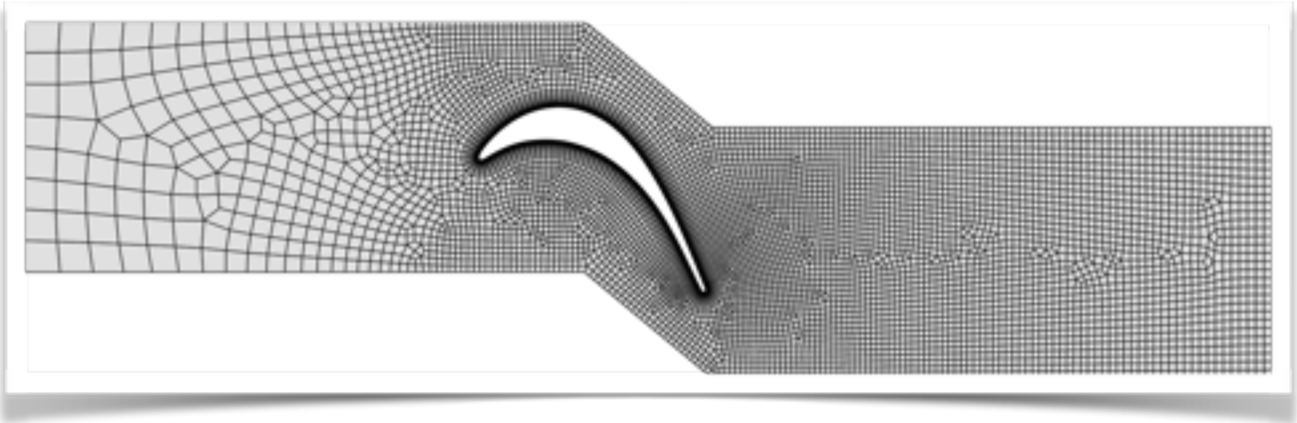
Flow past a Circular Cylinder: $Re_D = 3600$

- Parnaudeau et al. experiment + PyFR (5th order hex) ILES.

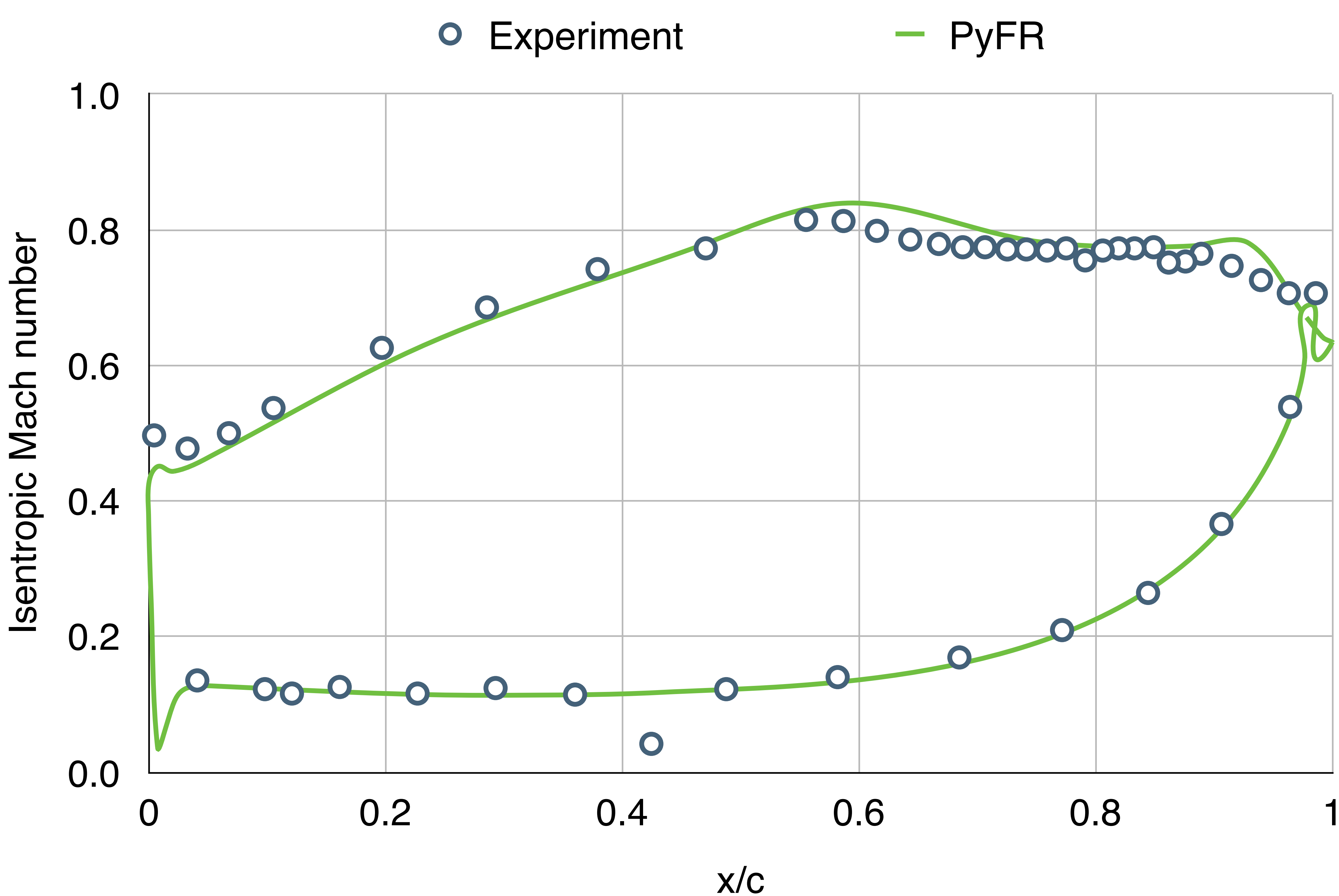


T106c Cascade

- Have also performed ILES of a T106c cascade at Re = 80,000 and Ma = 0.65.
- Domain is meshed with 60,000 hexahedra and run with p = 2.
- Compare with experimental data of Michálek et al.



T106c Cascade



Exact SFS Model

- Consider the inverse Helmholtz differential operator

$$\widetilde{\mathbf{u}} = \mathcal{G}[\mathbf{u}] = (1 - \alpha^2 \nabla^2)^{-1} \mathbf{u}.$$

- which has an inverse

$$\mathcal{Q} = \mathcal{G}^{-1} = \left(1 - \alpha^2 \nabla^2\right).$$

- We can use this to derive an *exact* expression for the resolved SFS stress tensor

$$\begin{aligned}\mathcal{G}[\mathbf{u}\mathbf{u}^T] &= \mathcal{G}[(\widetilde{\mathbf{u}} - \alpha^2 \nabla^2 \widetilde{\mathbf{u}})(\widetilde{\mathbf{u}} - \alpha^2 \nabla^2 \widetilde{\mathbf{u}})^T] \\ &= \mathcal{G}[\widetilde{\mathbf{u}}\, \widetilde{\mathbf{u}}^T - \alpha^2 (\widetilde{\mathbf{u}}\, \nabla^2 \widetilde{\mathbf{u}}^T + \widetilde{\mathbf{u}}^T\, \nabla^2 \widetilde{\mathbf{u}}) + \alpha^4 \nabla^2 \widetilde{\mathbf{u}}\, \nabla^2 \widetilde{\mathbf{u}}^T].\end{aligned}$$

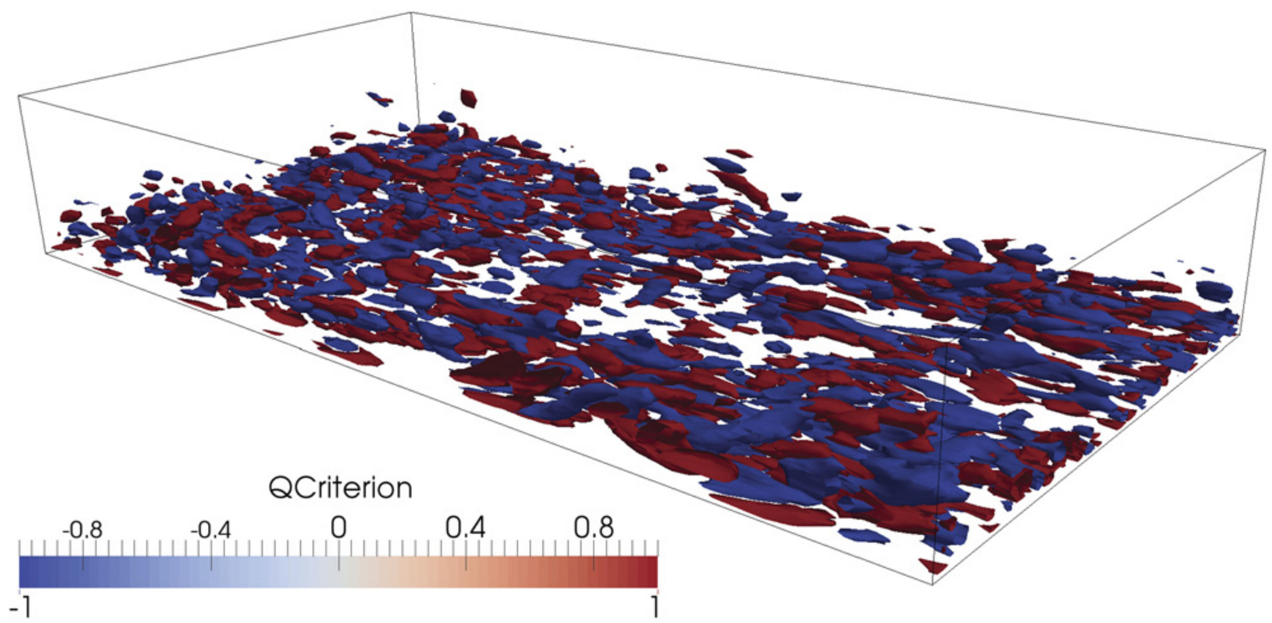
$$\tau_{SFS} = \mathcal{G}[2\alpha^2 \nabla \widetilde{\mathbf{u}}\, \nabla \widetilde{\mathbf{u}}^T + \alpha^4 \nabla^2 \widetilde{\mathbf{u}} \nabla^2 \widetilde{\mathbf{u}}^T].$$

$$\mathcal{Q}\tau_{SFS} = (1 - \alpha^2 \nabla^2)\tau_{SFS} = 2\alpha^2 \nabla \widetilde{\mathbf{u}}\, \nabla \widetilde{\mathbf{u}}^T + \alpha^4 \nabla^2 \widetilde{\mathbf{u}}\, \nabla^2 \widetilde{\mathbf{u}}^T,$$

Exact SFS Model: Channel Flow at $Re_\tau = 180$

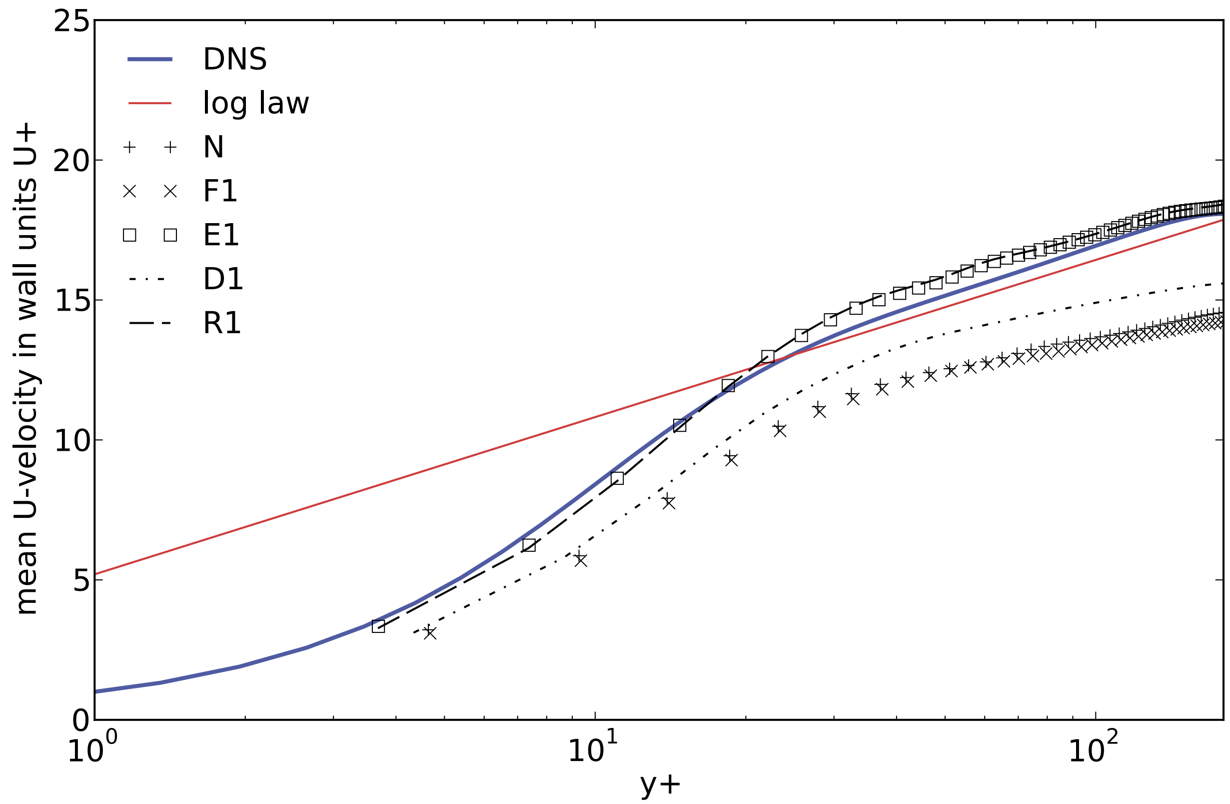
- Test case is turbulent channel flow at $Re_\tau = 180$.
- Compare with:
  - DNS
  - Implicit filtering, no SGS
  - Explicit filtering, no SFS
  - Implicit filtering, dynamic Smagorinsky SGS
  - Explicit filtering, dynamic Smagorinsky SFS
  - Explicit filtering, rational LES SFS

Right: Contours of Q criterion



Exact SFS Model

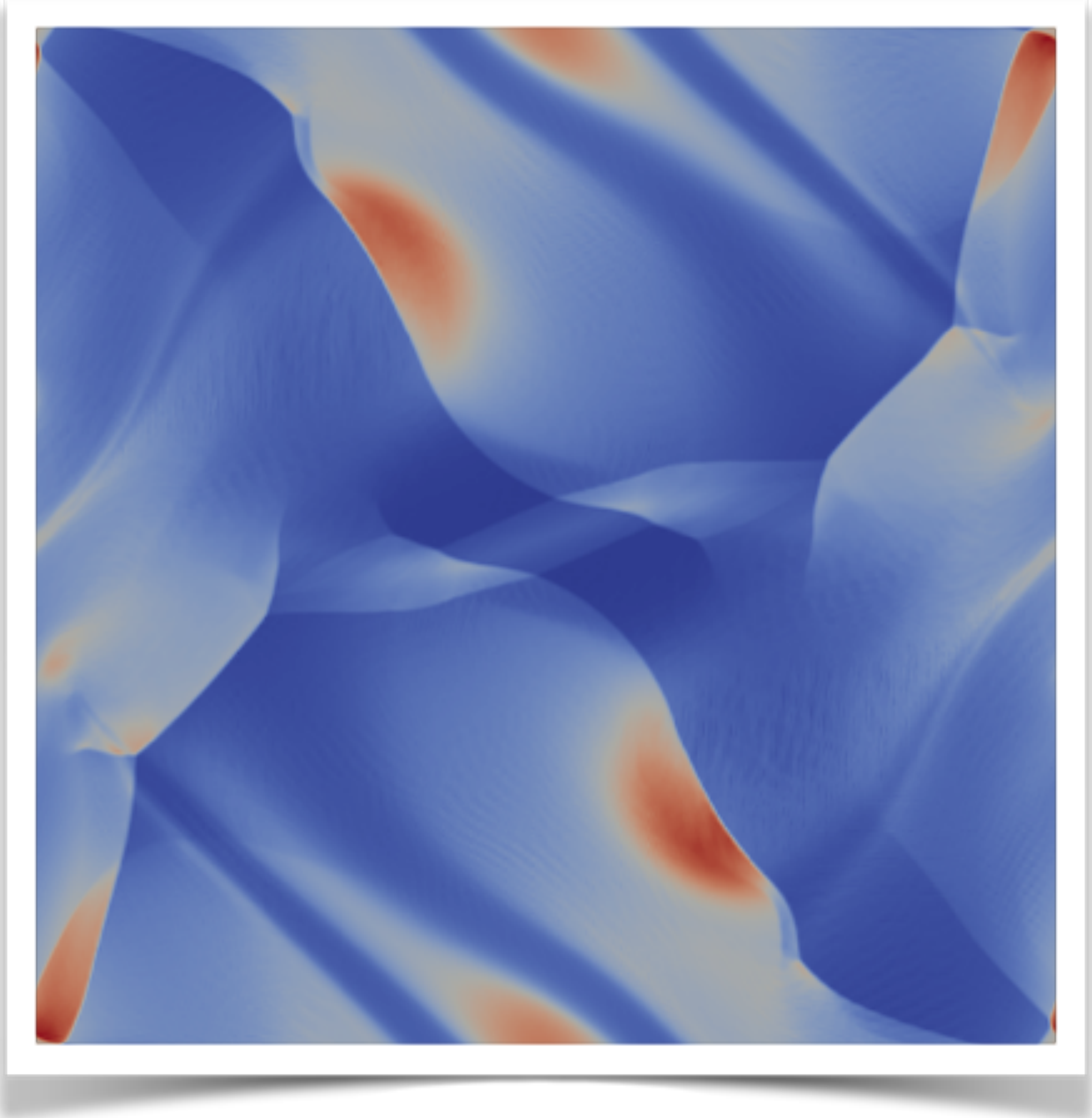
Model	Re_δ	Re_τ	U_B/U_τ	U_C/U_B	$C_f \times 10^3$
DNS	**3440**	**180**	**15.63**	**1.16**	**8.18**
no model, unfiltered	4126.7	232.1	17.78	1.132	6.33
no model, filtered	4144.6	234.1	17.70	1.129	6.38
dynamic, unfiltered	4329.3	216.8	19.97	1.126	5.02
dynamic, filtered	4315.6	216.2	19.96	1.123	5.02
rational LES	2987.0	184.1	16.22	1.155	7.59
exact SFS	2975.7	184.5	16.12	1.157	7.68



Exact SFS shows strong agreement with DNS compared with other models

Ideal MHD

- Also working towards an FR based ideal MHD solver.
- Uses *Powell's method*.
- Right: snapshot of pressure for a 2D Orszag-Tang vortex test-case.



Acknowledgement

The research is a combined effort by

- *Postdocs*: Peter Vincent, Guido Lodato, Jonathan Bull, Freddie Witherden
- *Ph.D. students*: Patrice Castonguay, Yves Allaneau, Kui Ou, David Williams, Manuel López, Kartikey Asthana, Abhishek Sheshadri, Jacob Crabill, Joshua Romero, Jerry Watkins

It has been made possible by the support of

- the *Air Force Office of Scientific Research* under grants FA9550-10-1-0418 and FA9550-14-1-0186 monitored by Jean-Luc Cambier
- the *National Science Foundation* under grants 0708071 and 0915006 monitored by Dr. Leland Jameson
- Stanford Graduate Fellowship

Questions & Anwers

Thank you for listening