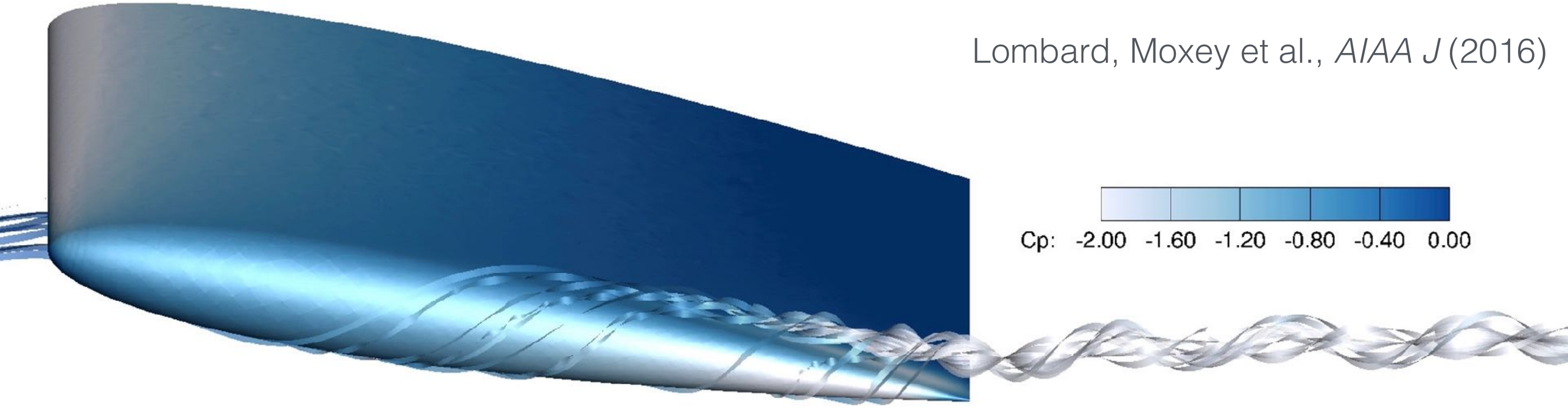
Towards high-fidelity industrial fluid dynamics simulations at high order

David Moxey
College of Engineering, Mathematics & Physical Sciences, University of Exeter

SPPEXA Final Symposium 2019
Dresden, Germany

21st October 2019



Lombard, Moxey et al., *AIAA J* (2016)

Increasing desire for **high-fidelity** CFD simulations in high-end aeronautics applications.

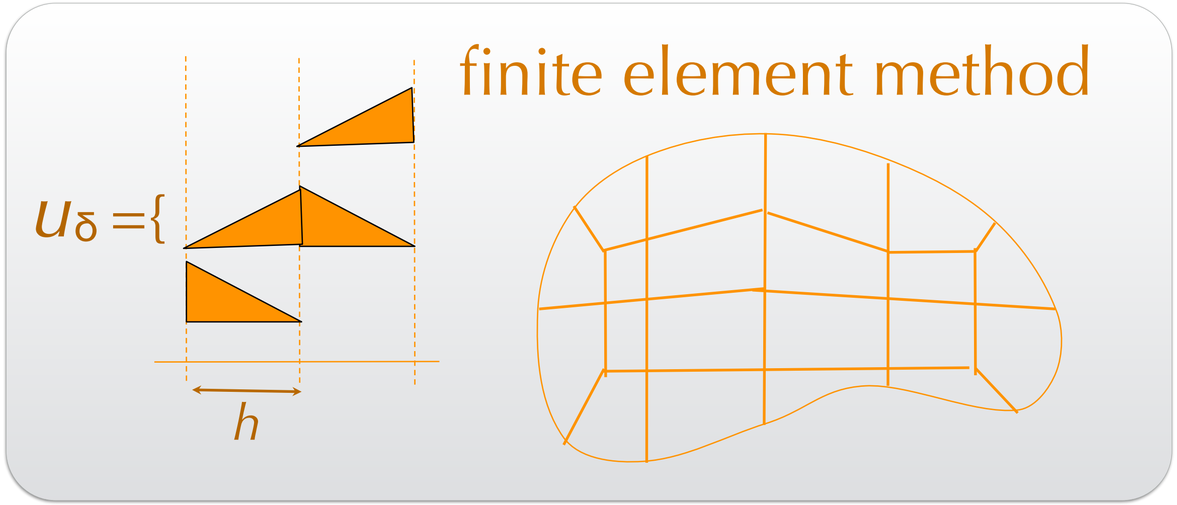
Want to accurately model 'difficult' features:

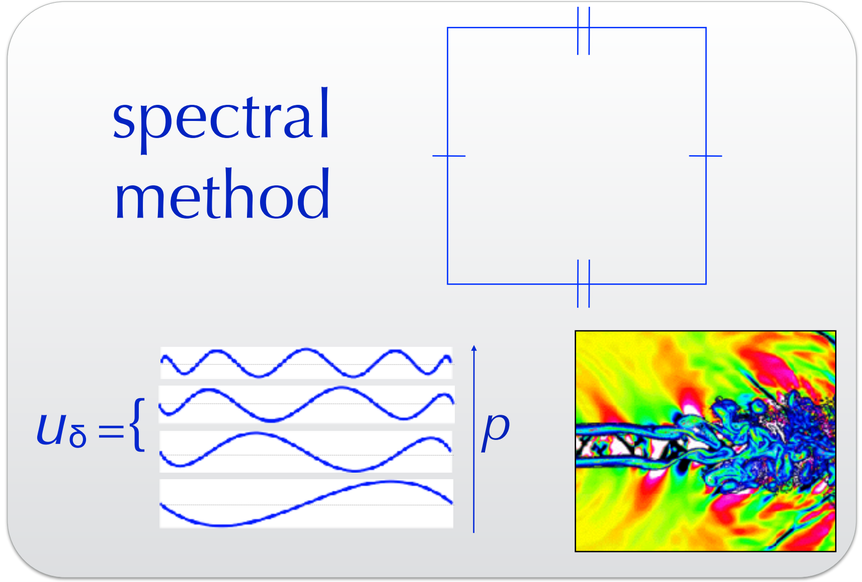
- strongly separated flows, vortex interaction
- feature tracking and prediction

Challenging with current industry-standard tools

Goal: develop methods and techniques for making LES affordable & routine

What are high-order methods?





spatial flexibility (*h*)

+

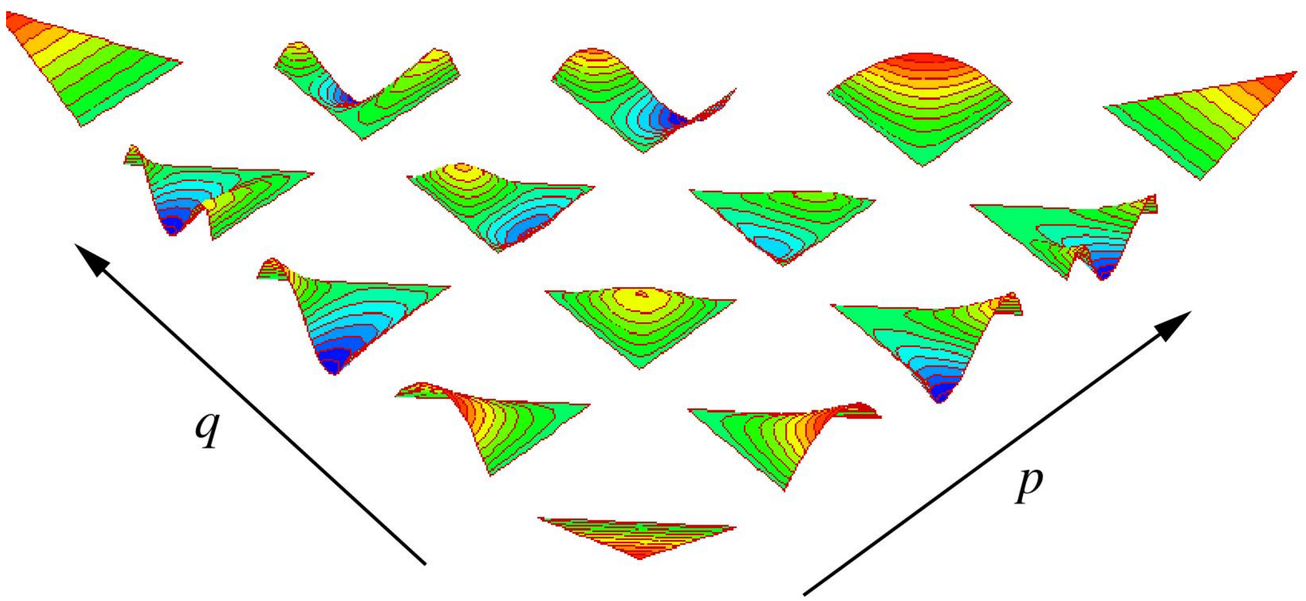
accuracy (*p*)

→

spectral/*hp* element

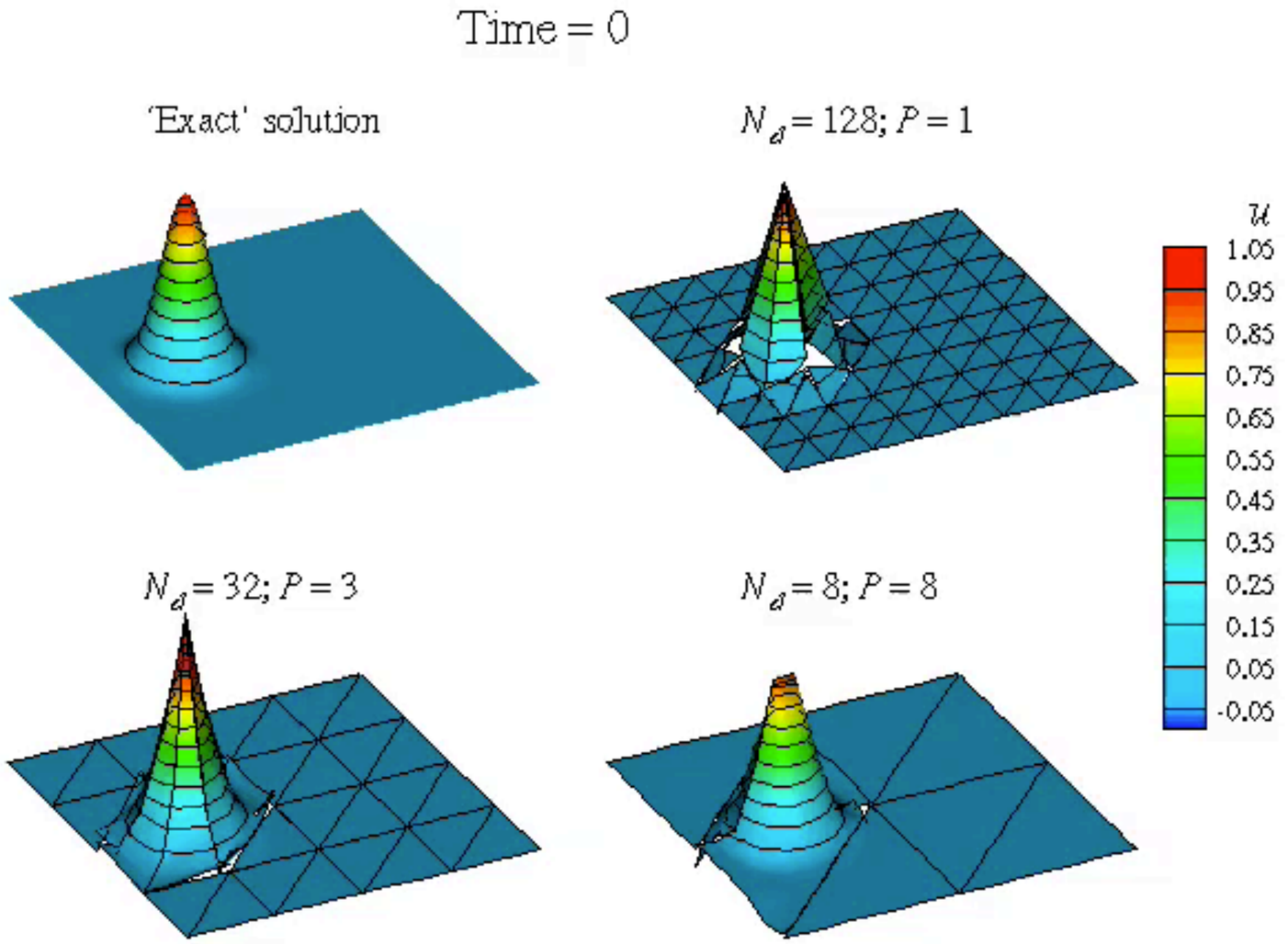
High-order expansions

- Extend traditional FEM by adding higher order polynomials of degree *P* within each element.
- e.g. high-order triangle has (*P*+1)(*P*+2)/2 degrees of freedom at a given order *P*.
- Increased accuracy: comes at additional work done per degree of freedom.

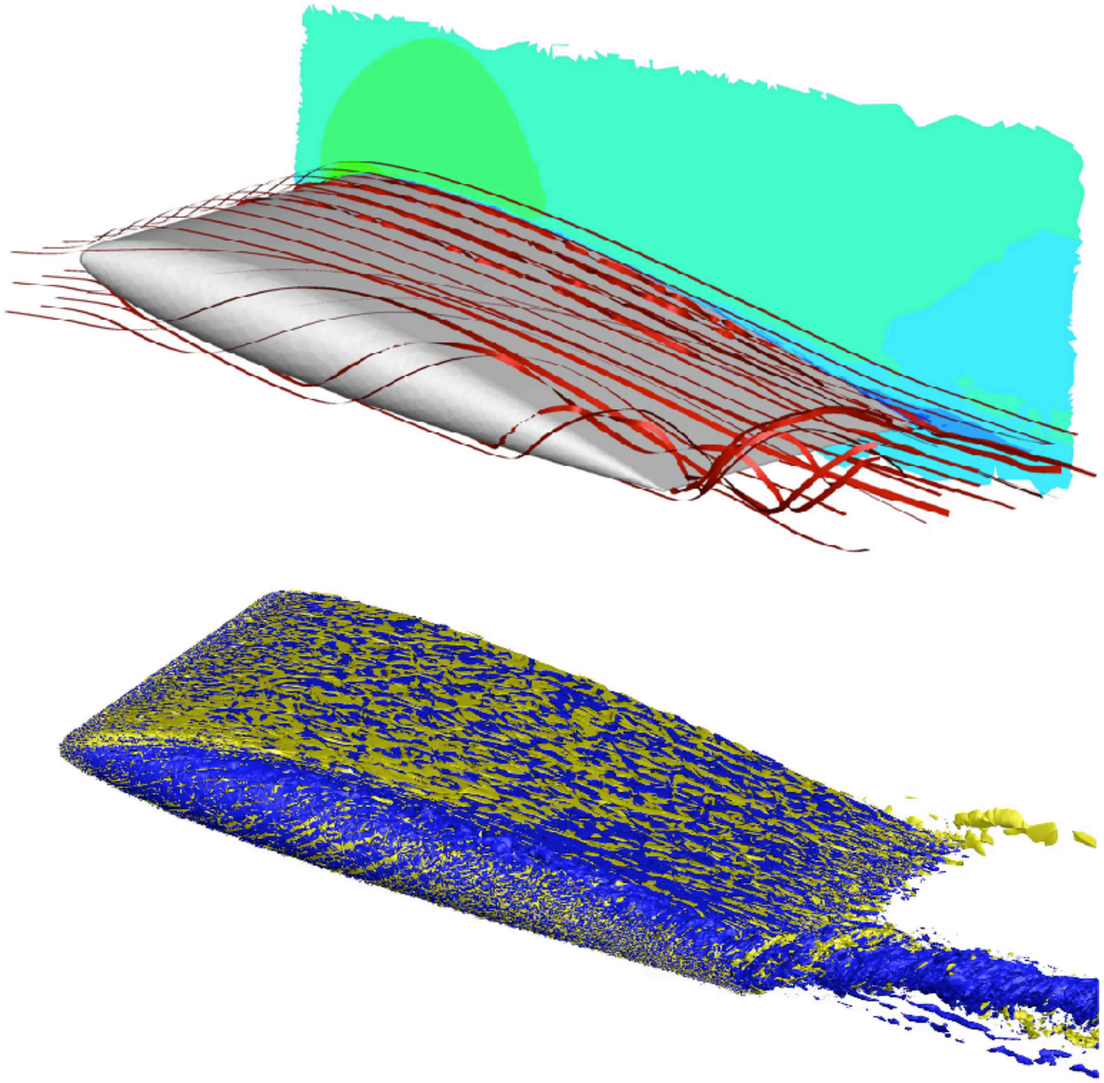


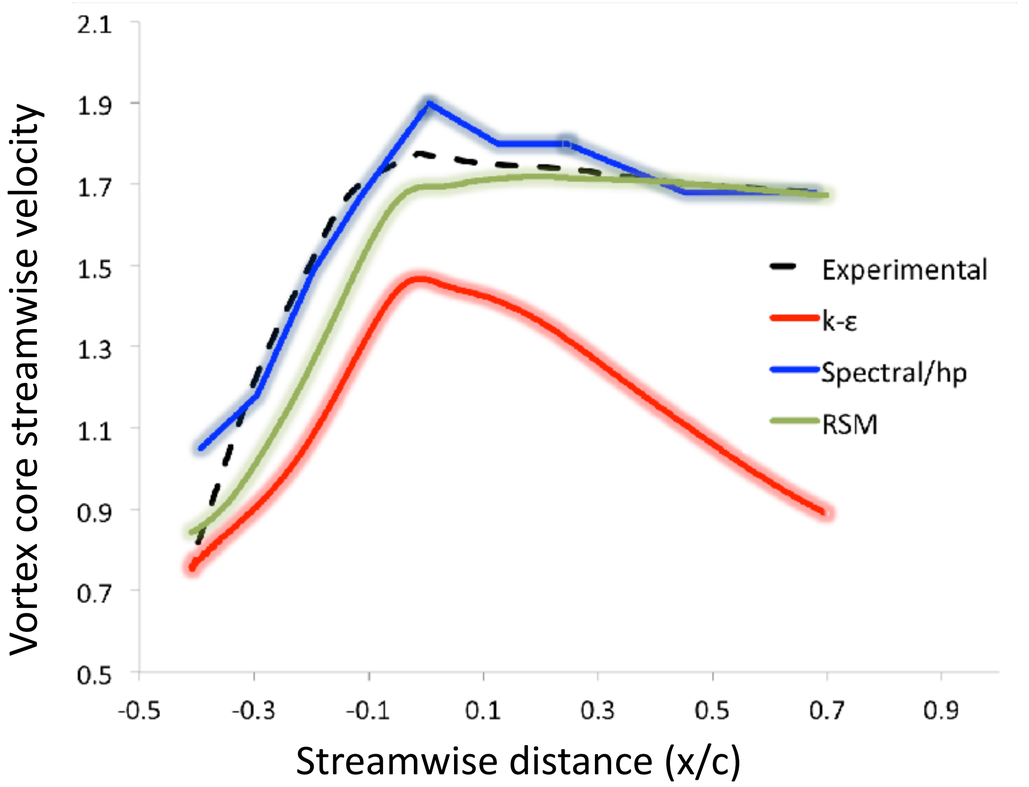
spectral/*hp* element basis functions

Why use a high-order method for CFD?



NACA 0012 example





Lombard, Moxey, Hoessler, Dhandapani, Taylor and Sherwin
AIAA Journal (2016)

Challenges

High order methods can provide a lot of benefits for these problems:

- ✓ very high accuracy over time/space;
- ✓ excellent for tracking transient structures;
- ✓ tuneable arithmetic intensity: overcome flops/byte barrier;
- ✓ geometric flexible.

but there are also some challenges…

- ➤ implementation difficulty;
- ➤ extra work per degree of freedom means computational efficiency is important;
- ➤ other numerical challenges, e.g. scalable preconditioning;
- ➤ mesh generation.

Nektar++

spectral/hp element framework

- Nektar++ is an open-source MIT-licensed framework for high-order methods.
- Arbitrary order curvilinear meshes to support complex geometries in a wide range of application areas, including incompressible/compressible fluids.
- Wide range of discretisation choices: CG/DG/HDG, Fourier, modal/nodal expansions, 1/2/3D, embedded manifolds.
- Parallel MPI support, scalable to many thousands of cores.
- C++11, extensive testing, CI & distributed source control, etc

Development team

Mike Kirby

THE UNIVERSITY OF UTAH

Spencer Sherwin

Imperial College London

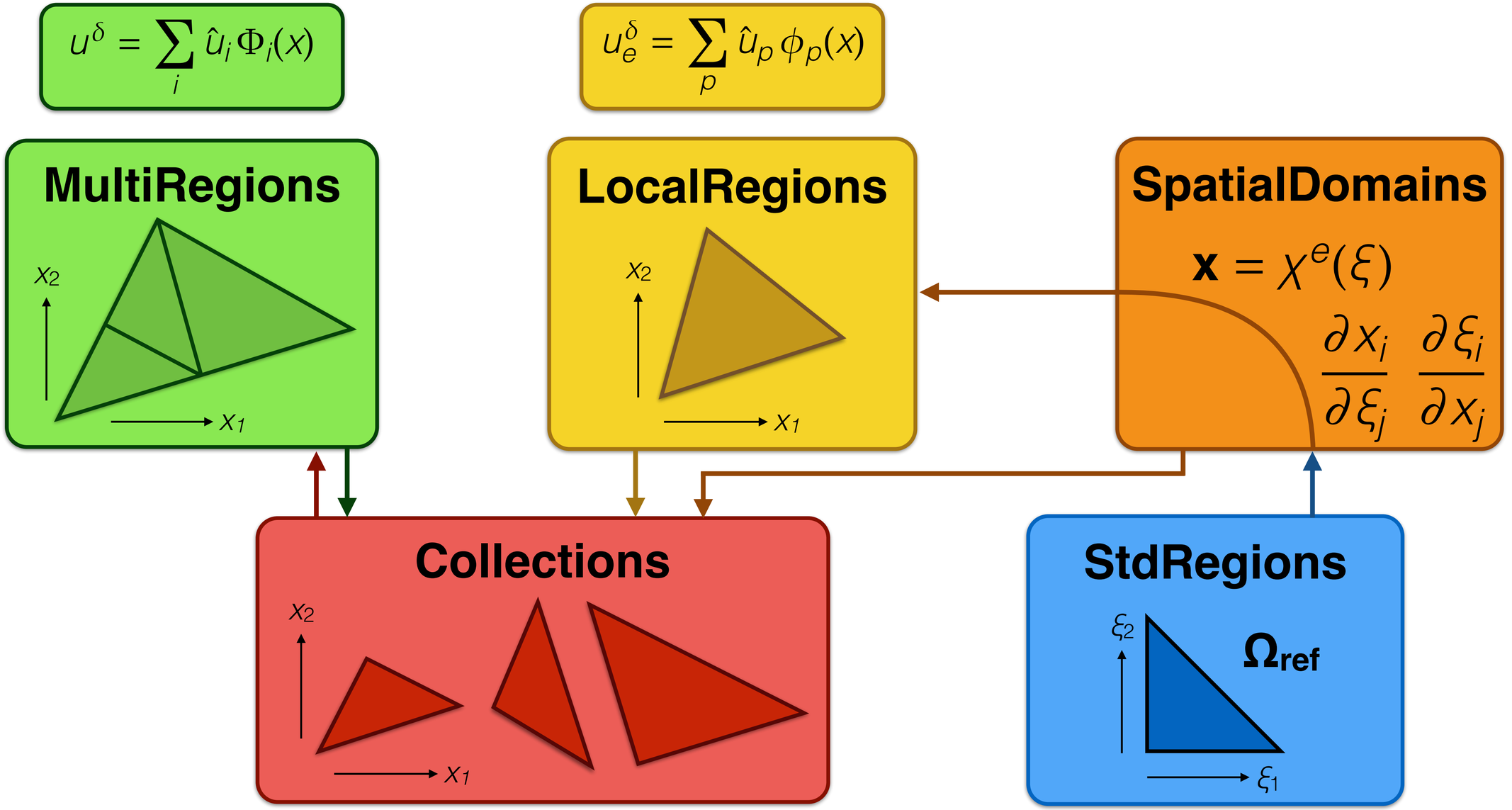
Chris Cantwell

David Moxey

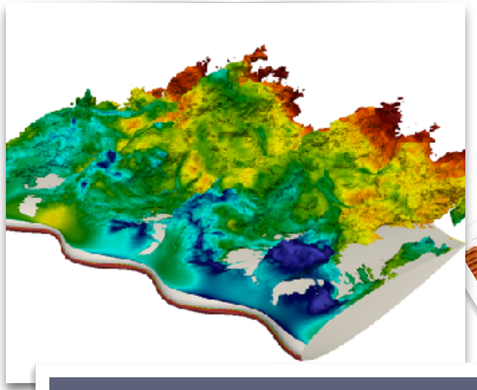
UNIVERSITY OF EXETER

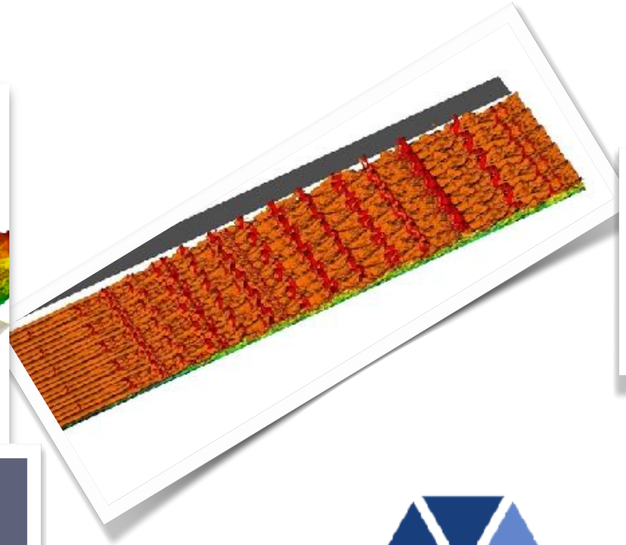
- **Project coordinators:** Joaquim Peiró, Gianmarco Mengaldo
- **Senior developers:** Kilian Lackhove, Douglas Serson, Giacomo Castiglioni

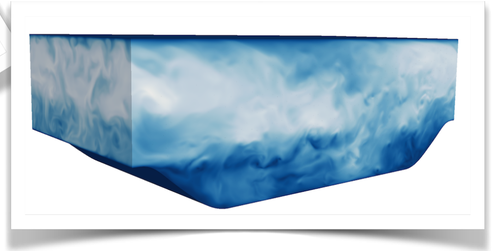
Framework design

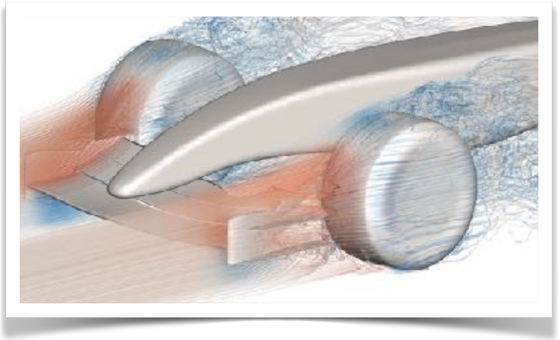


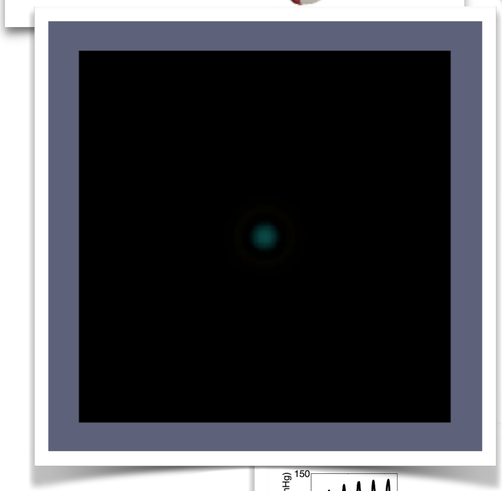
Some application areas



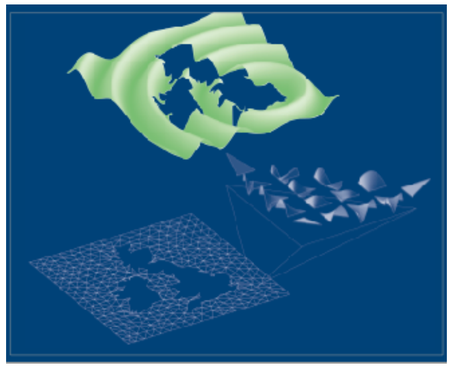


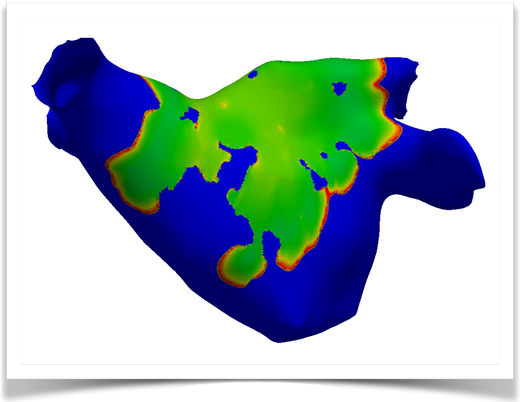


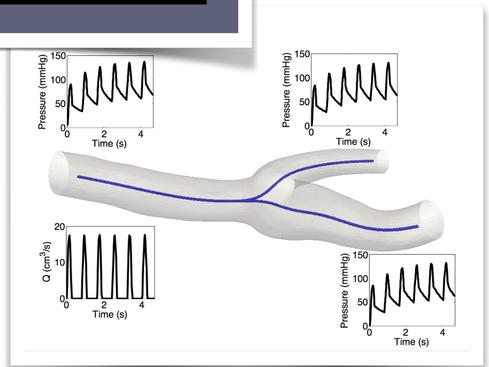


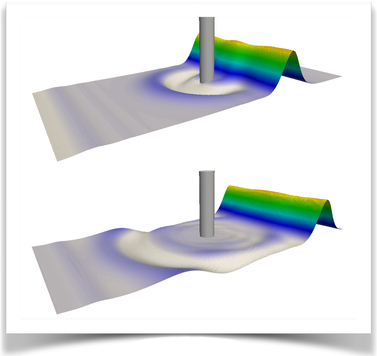


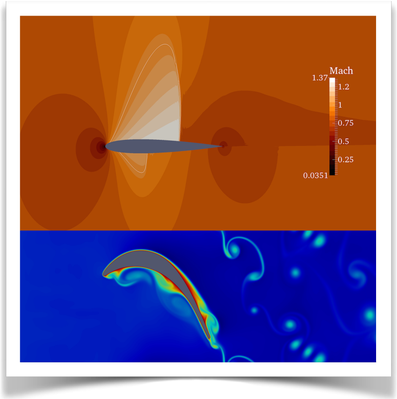
www.nektar.info

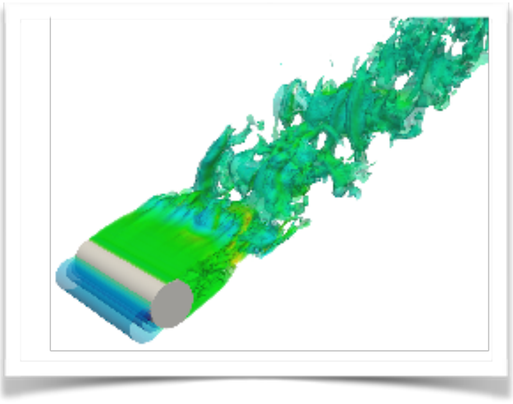






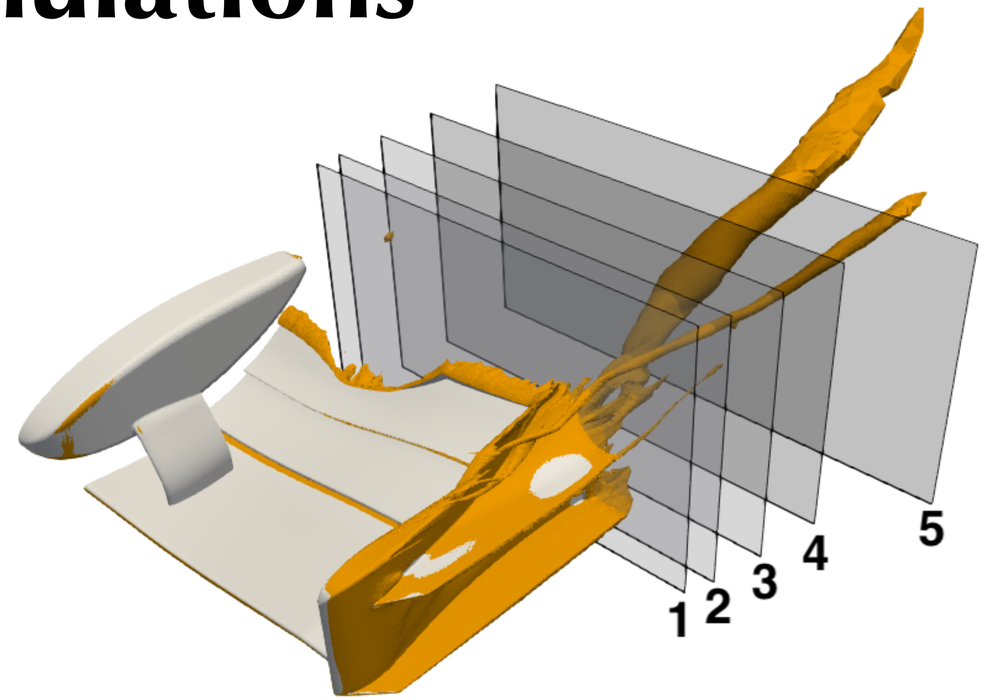


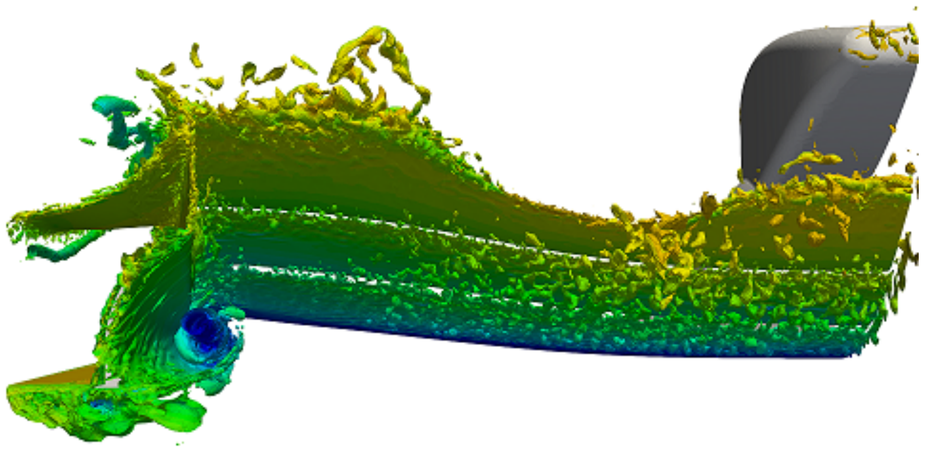




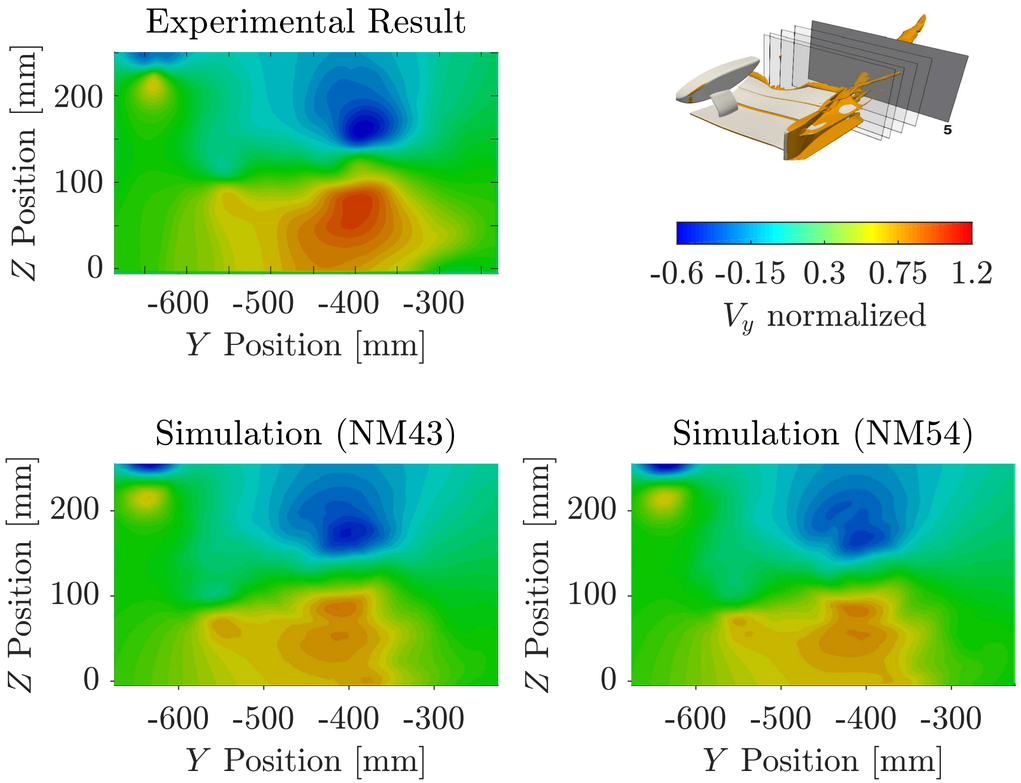
Recent F1 simulations

- F1 simulations highlight complex vortex interaction: ideal candidates for LES.
- Front wing simulations with experimental PIV datasets as new proposed benchmark case.
- Analysis found in Buscariolo et al, arXiv 1909.06701.
- Dataset/geometry doi: 10.14469/hpc/6049

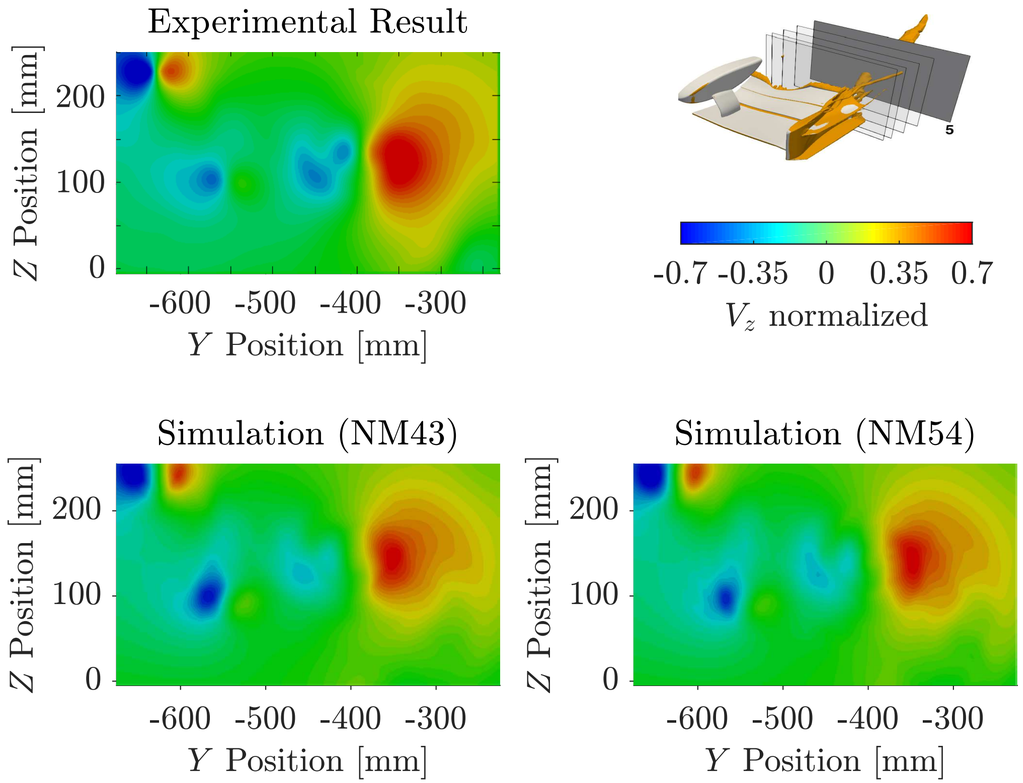




Comparison with experiment

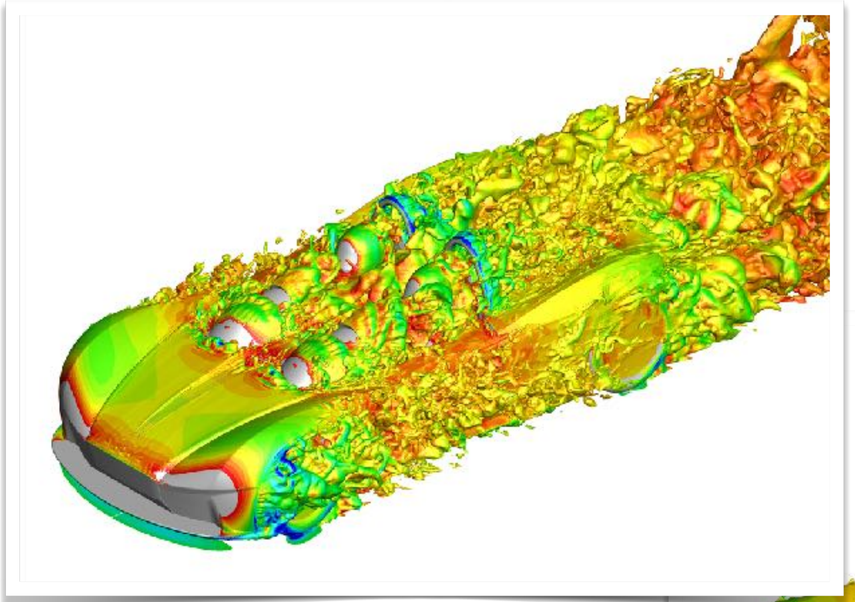


u component

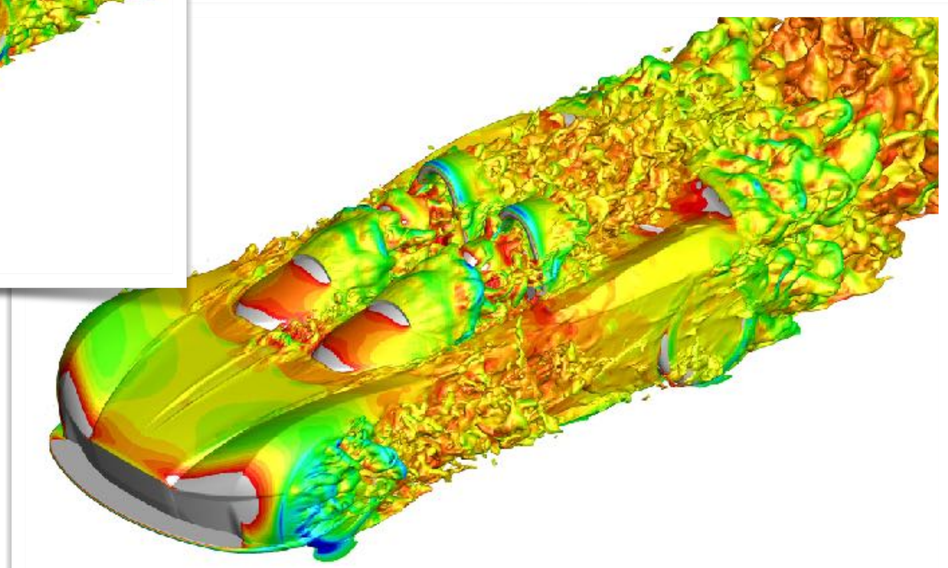


v component

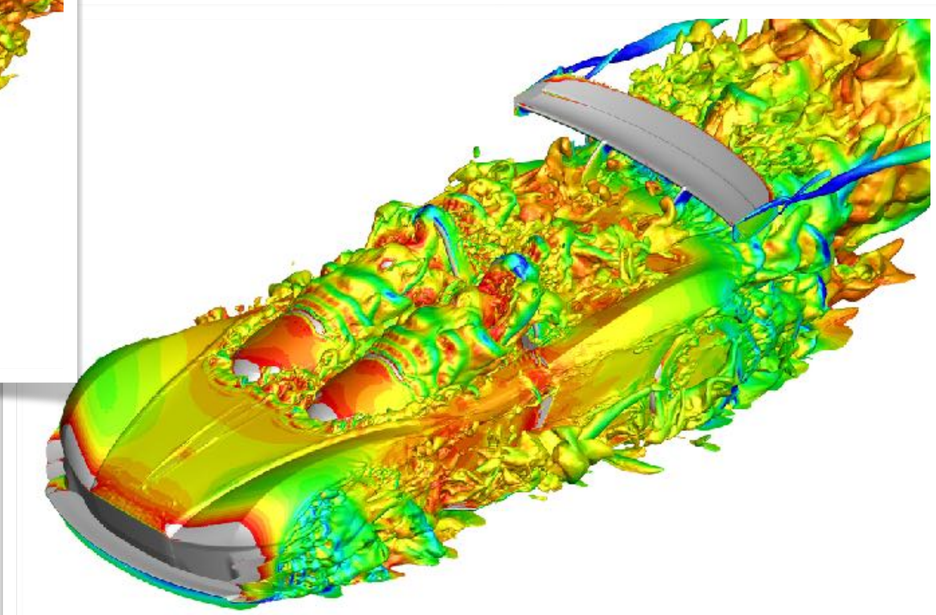
Elemental road race car



5th order
Re = 1m



Design 2: +33% Downforce



Design 3: +270% Downforce

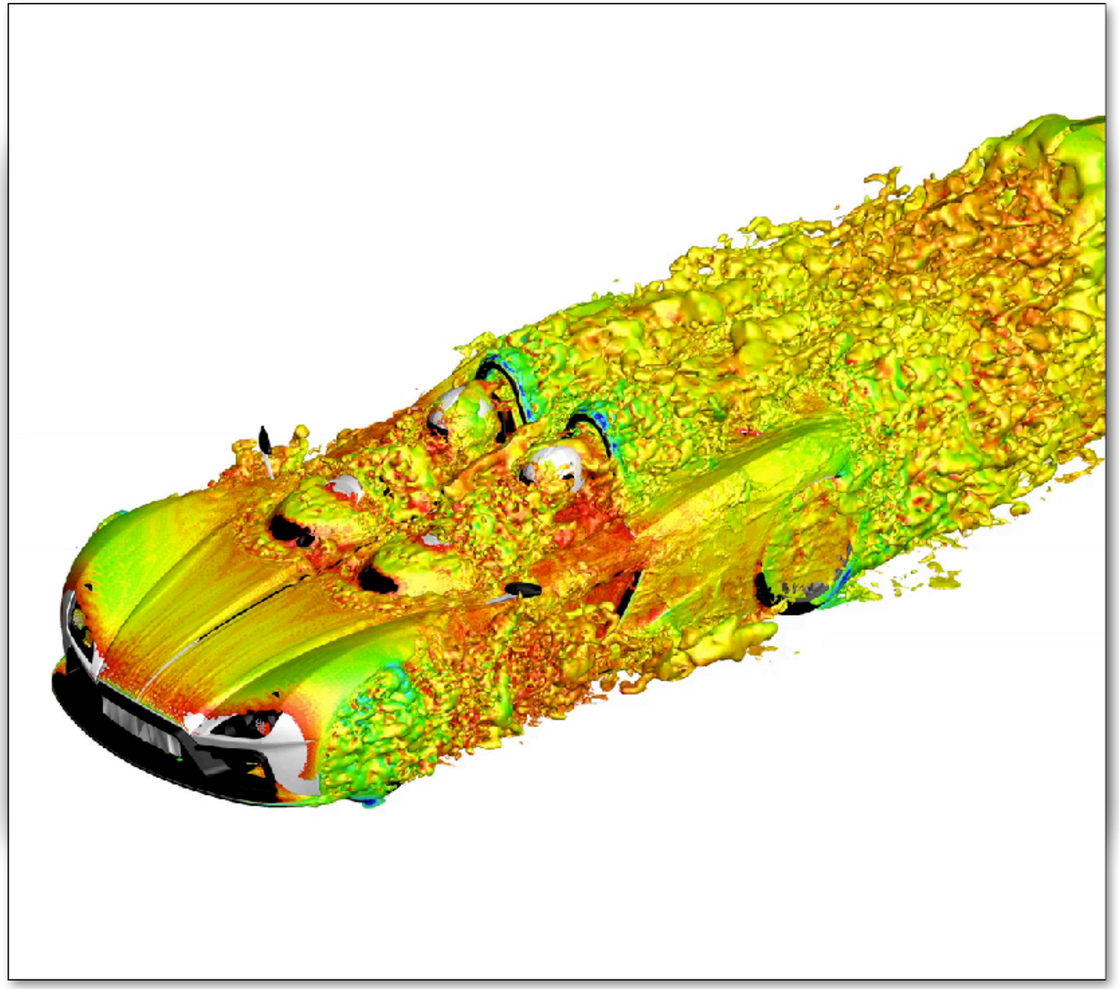
Moxey, Turner, Jassim, Taylor, Peiro & Sherwin

High-performance methods

High-order methods potentially map well onto the exascale hardware landscape where memory bandwidth is significant bottleneck:

- Use of **matrix-free** formulations of operators to reduce reliance on slow memory bandwidth performance: higher arithmetic intensity.
- Use of **sum-factorisation** to reduce evaluation complexity, which relies on the use of tensor-product basis.
- Can effectively use element structure to exploit **SIMD instructions** to achieve close to theoretical peak performance.

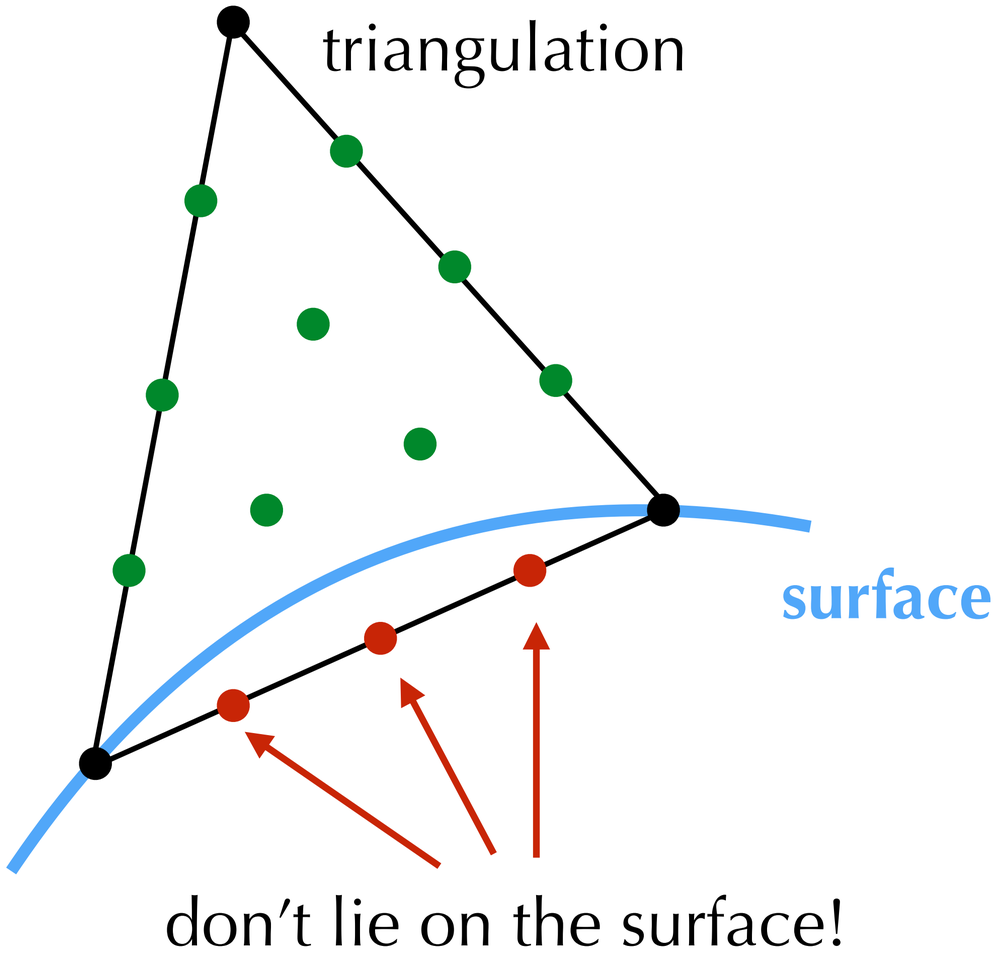
Unstructured simulations



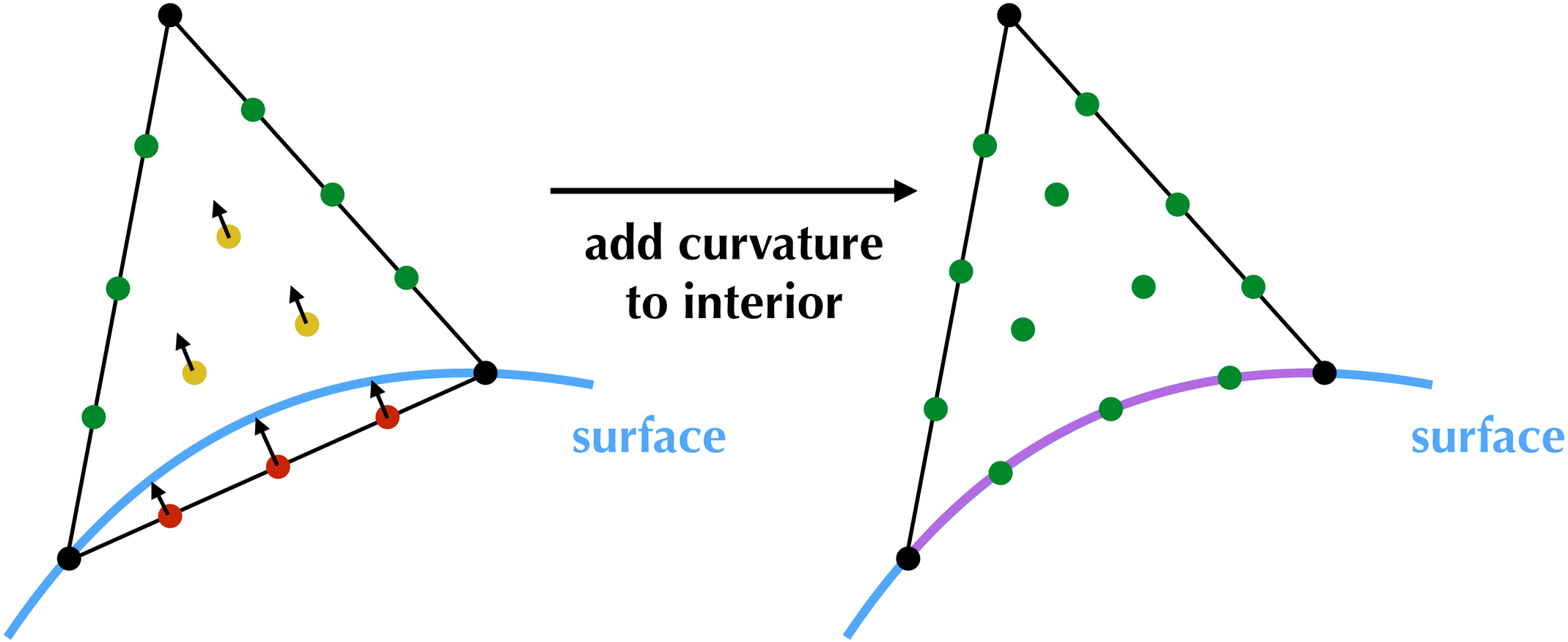
- Common knowledge: hex/quad elements yield best performance.
- However complex geometries presently require meshes of ‘unstructured’ elements: tets, prisms, etc.
- Potentially use tensor-product basis on unstructured elements, enable matrix-free operators + sum factorisation.

High-order mesh generation

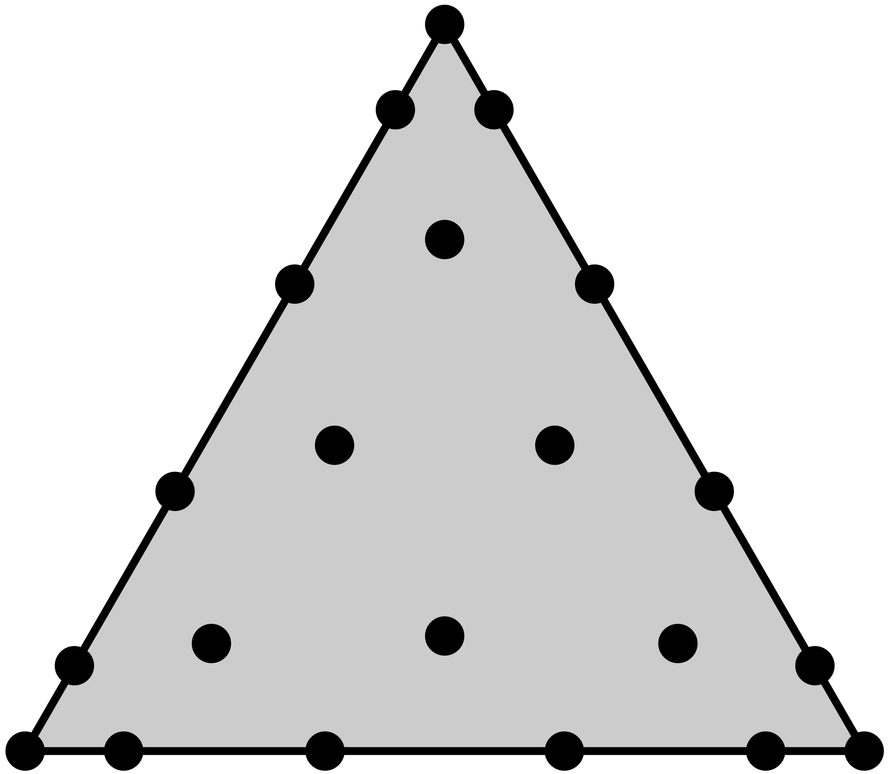
- At high orders we must **curve the elements** to fit the geometry.
- This significantly complicates the mesh generation process.
- Additionally has performance implications: Jacobian now spatially-variable, additional storage required.



High-order mesh generation



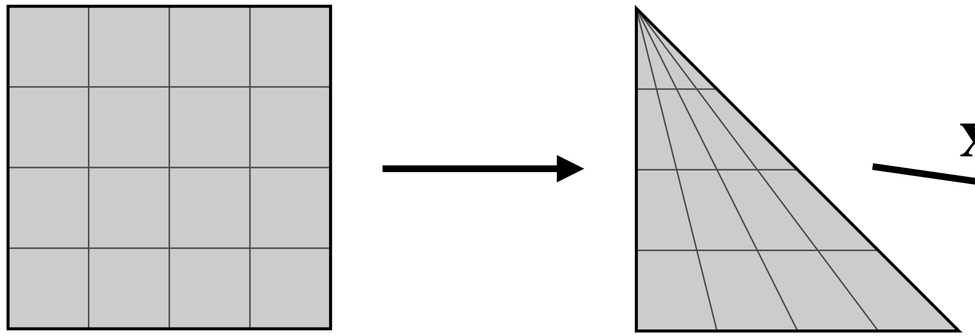
Unstructured elements



P5 triangle, Fekete points

- Unstructured elements generally make use of Lagrange basis functions.
- Combine this with a suitable set of cubature points: either collocated or not.
- However this loses tensor-products structure: i.e. no sum factorisation possible.

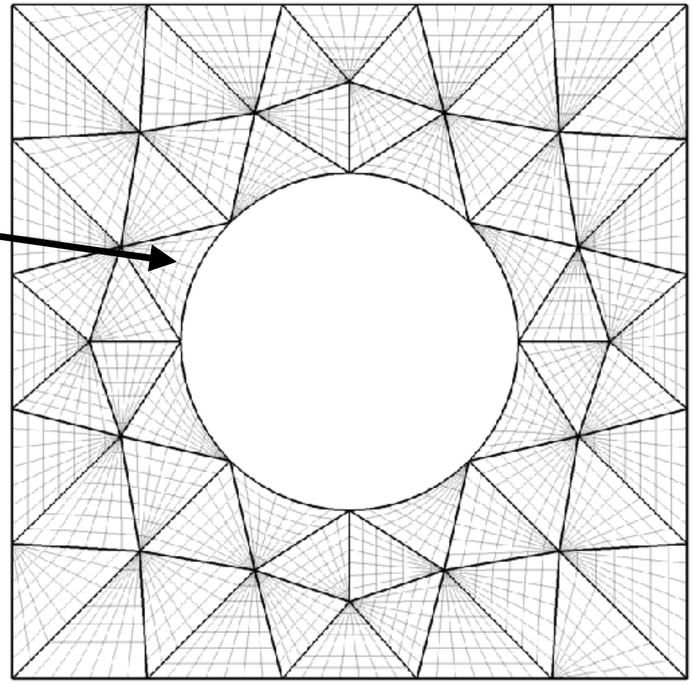
Spectral/*hp* element formulation



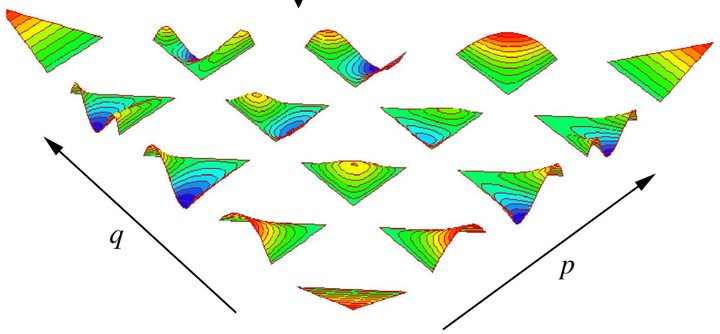
collapsed coordinates
$(\eta_1, \eta_2) \in [-1,1]^2$

reference element
$(\xi_1, \xi_2) \in \Omega_{st}$

$\mathbf{x} = \chi^e(\xi)$



(C⁰) tensor product basis
$\phi_p^a(\eta_1)\,\phi_{pq}^b(\eta_2)$



"Defining" features of spectral/hp method

Generally *not* collocated

$$u(\xi_{1i}, \xi_{2j}) = \sum_{n=0}^{P^2} \hat{u}_n \phi_n(\xi) = \sum_{p=0}^{P} \sum_{q=0}^{Q} \hat{u}_{pq} \phi_p(\xi_{1i}) \phi_q(\xi_{2j})$$

quadrature points

modal coefficients

Uses tensor products of 1D basis functions, even for non-tensor product shapes

$$u(\xi_{1i}, \xi_{2j}, \xi_{3j}) = \sum_{p=0}^{P} \sum_{q=0}^{Q-p} \sum_{r=0}^{R-p-q} \hat{u}_{pqr} \phi_p^a(\xi_{1i}) \phi_{pq}^b(\xi_{2j}) \phi_{pqr}^c(\xi_{3k})$$

basis function indexing harder

Sum-factorisation

Key to performance at high polynomial orders: complexity $\mathrm{O}(P^{2\mathrm{d}})$ to $\mathrm{O}(P^{\mathrm{d}+1})$

$$\sum_{p=0}^{P}\sum_{q=0}^{Q}\hat{u}_{pq}\phi_p(\xi_{1i})\phi_q(\xi_{2j}) = \sum_{p=0}^{P}\phi_p(\xi_{1i})\left[\sum_{q=0}^{Q}\hat{u}_{pq}\phi_q(\xi_{2j})\right]$$

store this

This works in essentially the same way for more complex indexing:

$$\sum_{p=0}^{P}\sum_{q=0}^{Q-p}\hat{u}_{pq}\phi_p^a(\xi_{1i})\phi_{pq}^b(\xi_{2j}) = \sum_{p=0}^{P}\phi_p^a(\xi_{1i})\left[\sum_{q=0}^{Q-p}\hat{u}_{pq}\phi_{pq}^b(\xi_{2j})\right]$$

store this

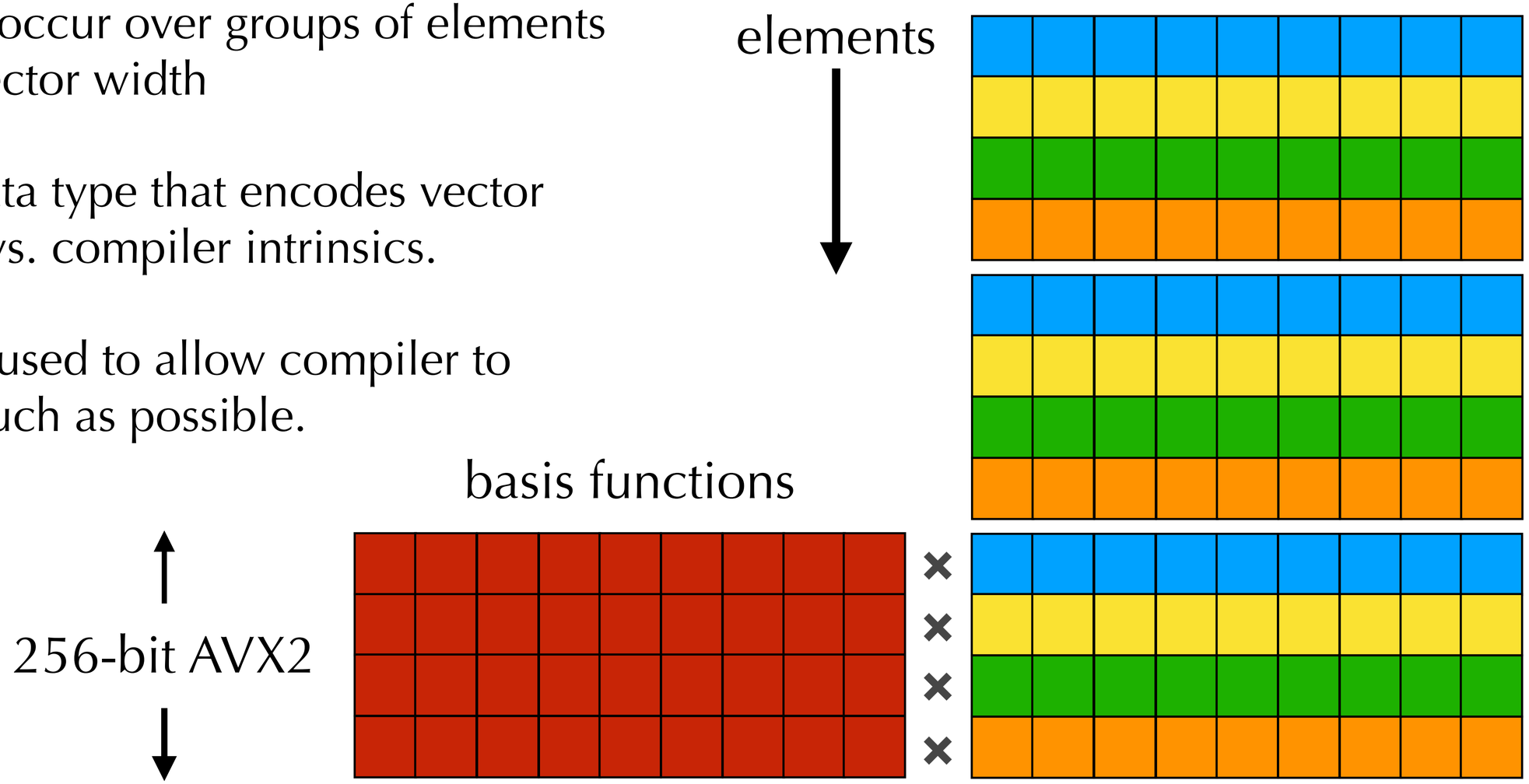
Key questions

- Under spectral/*hp* approach, sum-factorisation matrix-free operators are certainly possible for any element type. Some questions:
  - How much performance do we lose relative to hex/quad?
  - How should SIMD be used?
- Developed a benchmarking utility for Helmholtz operator to test viability of this approach, to be used later in *Collections* library.

$$\nabla^2 u - \lambda u = f(x) \quad \rightarrow \quad (\mathbf{L} + \lambda \mathbf{M})\hat{\mathbf{u}} = \hat{\mathbf{f}}$$

Data layout

- Operations occur over groups of elements of size of vector width
- Use C++ data type that encodes vector operations vs. compiler intrinsics.
- Templating used to allow compiler to unroll as much as possible.



Tests

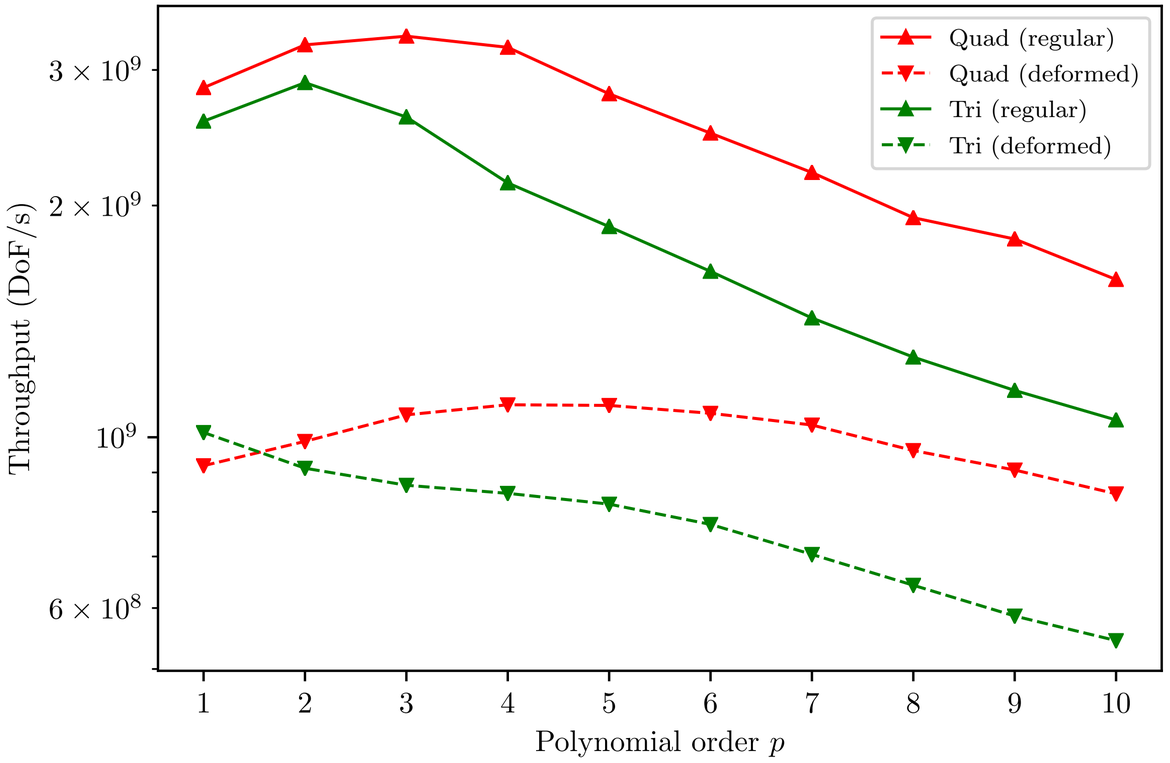
- Benchmarking of Helmholtz operator performed on two architectures with varying SIMD widths.

	Broadwell (AVX2)	Skylake (AVX512)
Model	E5-2697 v4	Xeon Gold 6130
Clock speed(s)	2.3 / 2.0 GHz standard / AVX2	2.1 / 1.7 / 1.3 GHz standard / AVX / AVX512
Cores / sockets	18 / 2	16 / 2
Max node GFLOP/s	1152	870 (AVX2) 1331 (AVX512)

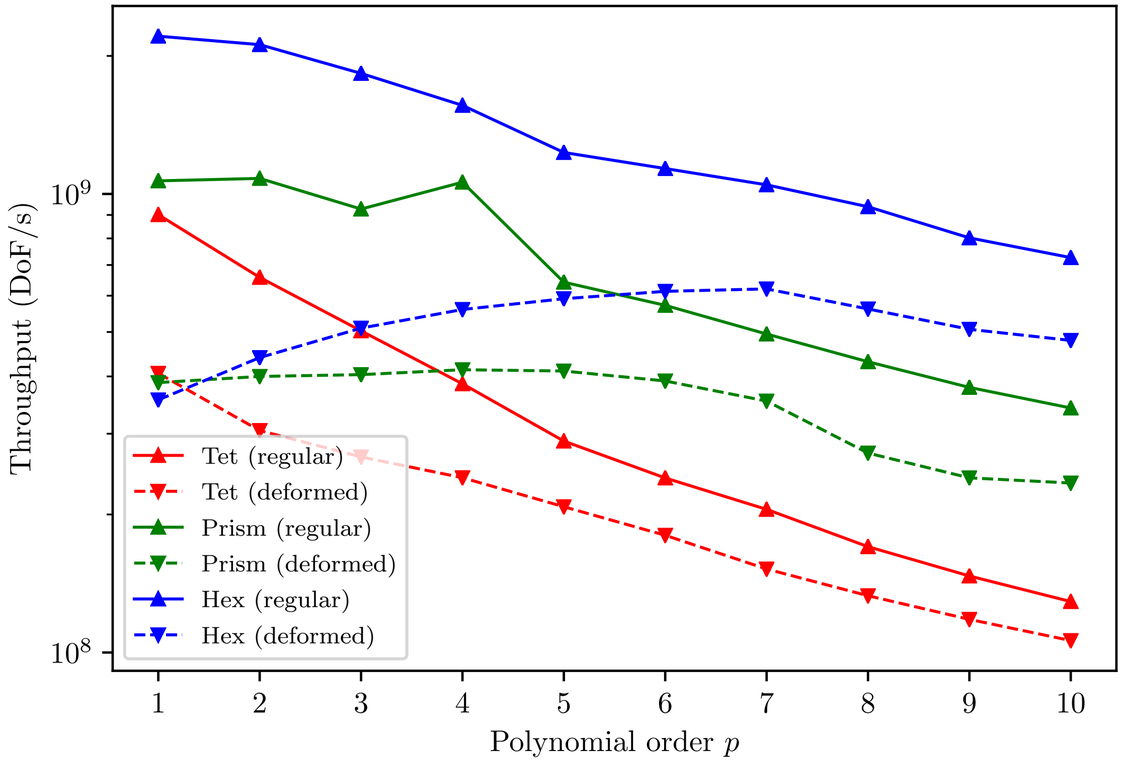
Assessing performance

- Various techniques used to assess kernel performance:
  - **Throughput**: number of local DoF/s processed, for a mesh whose sizes exceeds available cache.
  - GFLOP/s gives some indication of capabilities, provided we are not memory-bound.
  - Better is **roofline analysis**: where do we sit in terms of memory bandwidth to arithmetic intensity?
- Note all results for local elemental operation evaluation only: C^0 work in progress.

Throughput (AVX2, Broadwell)

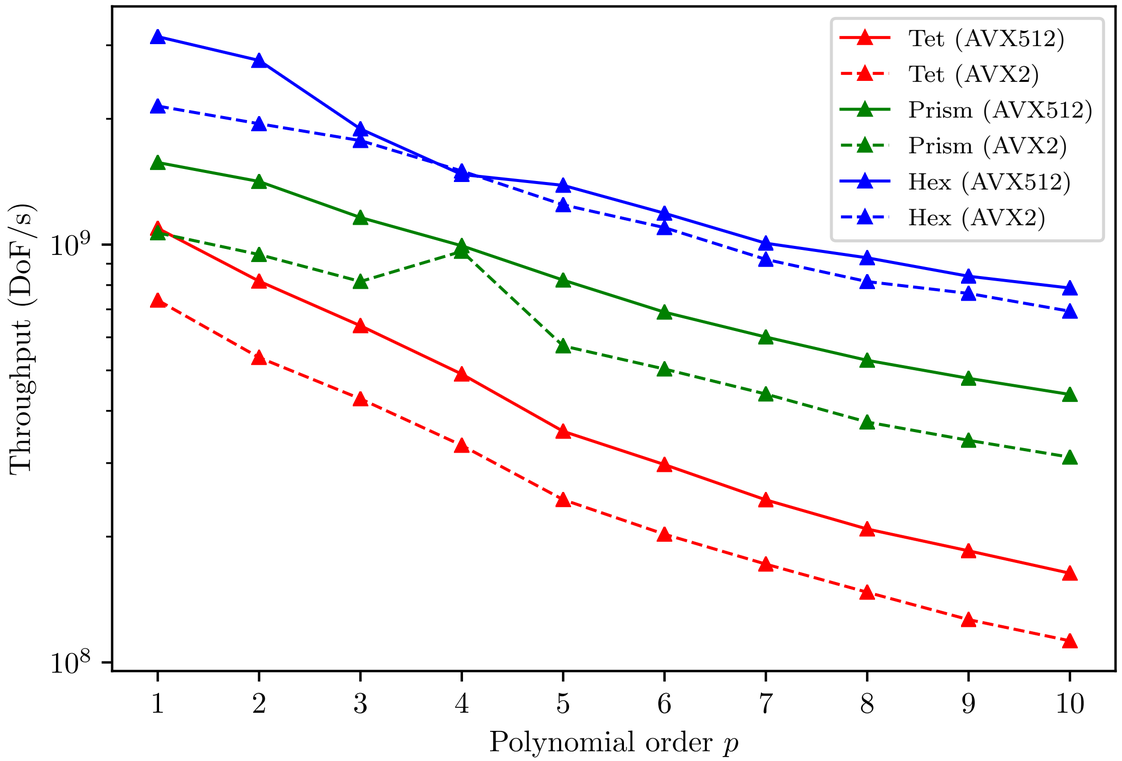


2D: Quads, triangles

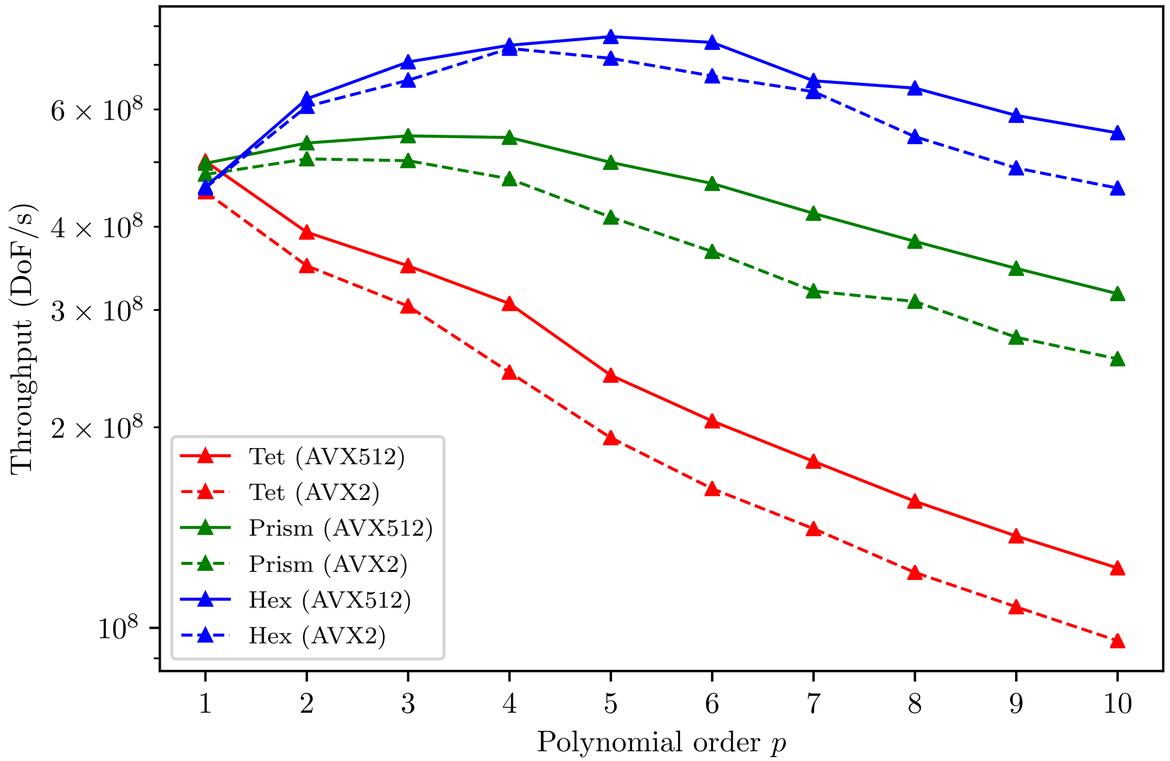


3D: Hexahedra, prisms, tetrahedra

Throughput (AVX512/AVX2, Skylake)

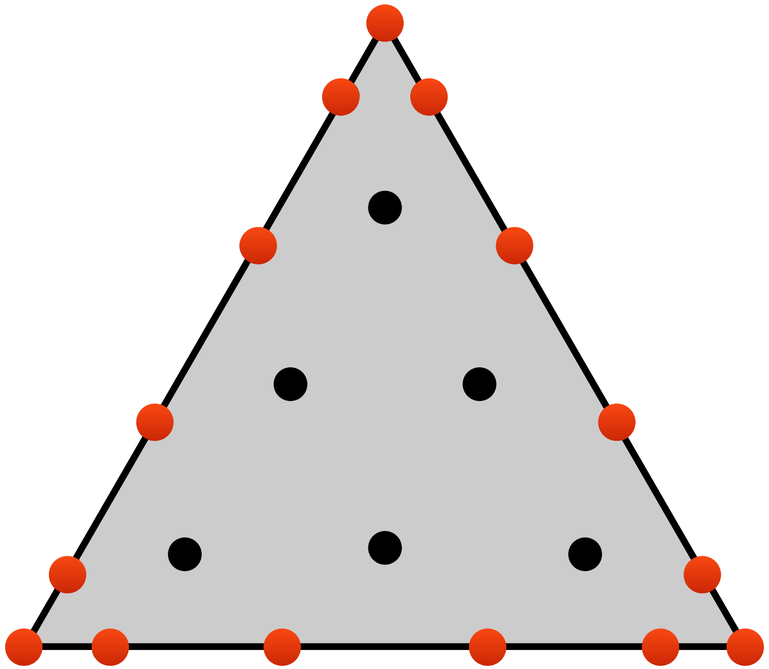


3D: ‘Regular’ elements

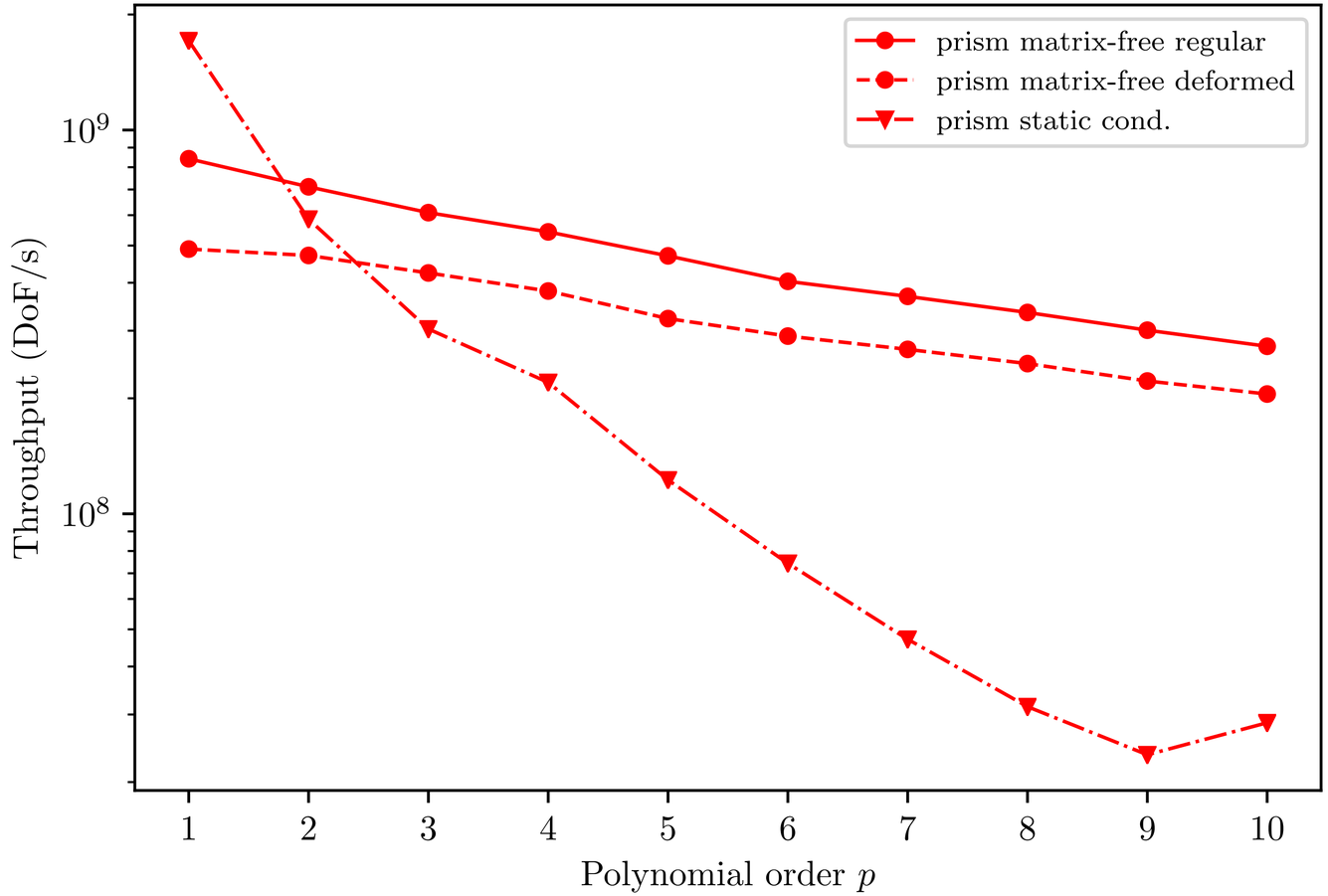


3D: ‘Deformed’ elements

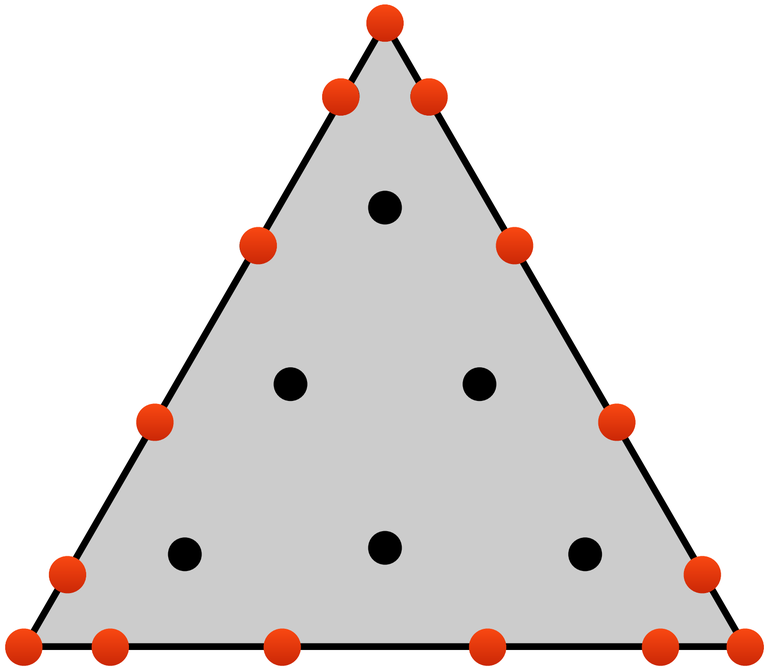
Throughput against static condensation



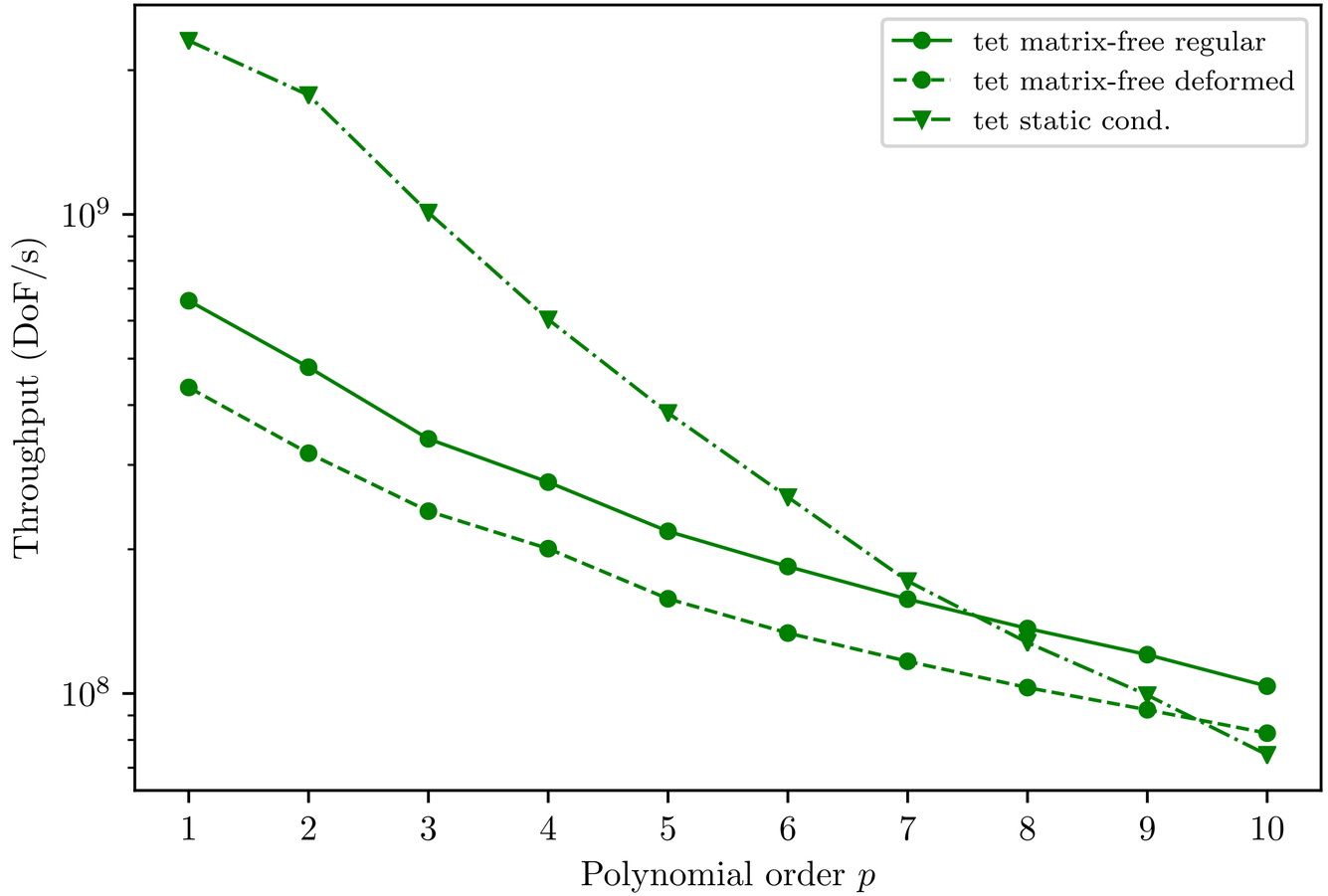
AVX512, Xeon Gold 5120



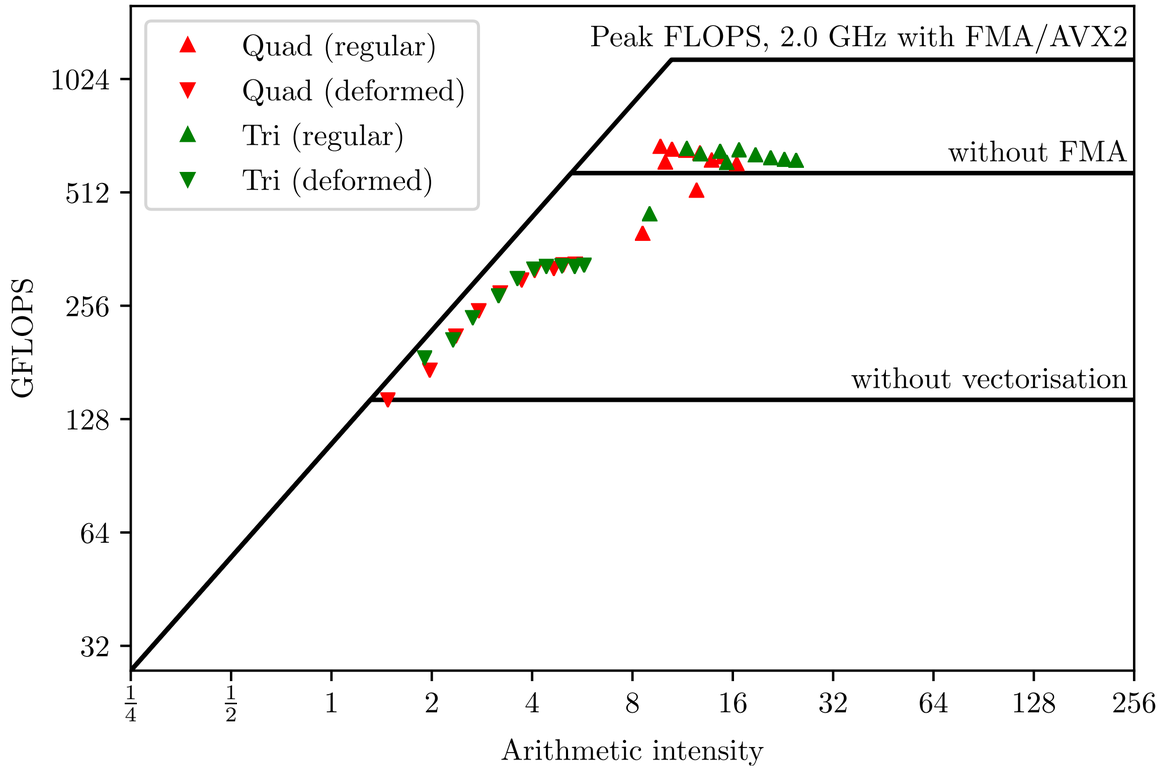
Throughput against static condensation



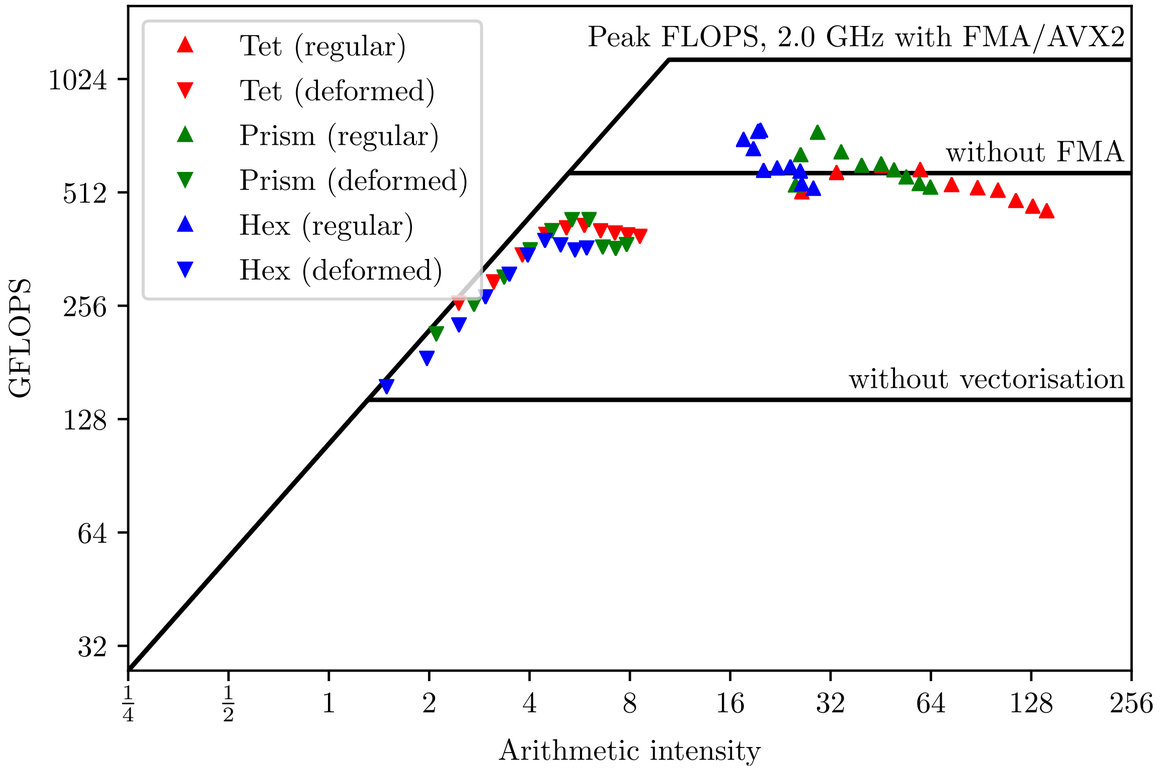
AVX512, Xeon Gold 5120



Roofline results



2D: Quads, triangles



3D: Hexahedra, prisms, tetrahedra

Some clear trends

- This behaves pretty much as you might anticipate:
  - For regular elements, clear hierarchy of element type/dimension, where throughput is lost as dimension/complexity of indexing increases.
  - Regular elements outperform deformed elements due to increased memory bandwidth.
  - Relative performance gap between deformed elements decreases at moderate polynomial orders.

Summary

- Efficient matrix-free implementations of key finite element operators are certainly achievable on modern architectures for ‘unstructured’ elements.
- Inevitable drop in performance from quads/hexahedra: complexity of indexing, additional cache pressure, etc.
- However relative performance of e.g. hex/prism and quad/tri is actually pretty good, particularly for deformed elements; important for e.g. boundary layer problems with large proportion of BL prisms.
- Future directions: revisit elemental basis formulations.

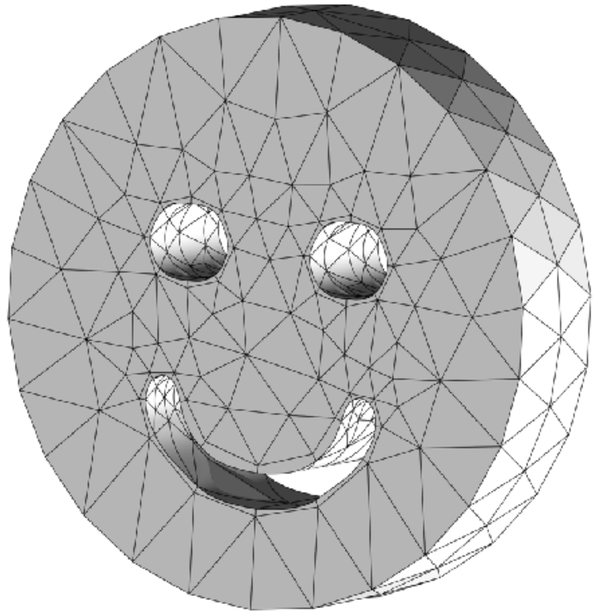
Thanks for listening!

https://davidmoxey.uk/

d.moxey@exeter.ac.uk

www.nektar.info

https://prism.ac.uk/



EPSRC
Pioneering research and skills

Nektar++: enhancing the capability and application of high-fidelity spectral/*hp* element methods

David Moxey[1], Chris D. Cantwell[2], Yan Bao[3], Andrea Cassinelli[2], Giacomo Castiglioni[2], Sehun Chun[4], Emilia Juda[2], Ehsan Kazemi[4], Kilian Lackhove[6], Julian Marcon[2], Gianmarco Mengaldo[7], Douglas Serson[2], Michael Turner[2], Hui Xu[5,2], Joaquim Peiró[2], Robert M. Kirby[8], Spencer J. Sherwin[2]