An UM-BRIDGE setup for multi-fidelity surrogate modelling for UQ

Benjamin M. Kent & Lorenzo Tamellini (CNR–IMATI)

kent@imati.cnr.it

UM-BRIDGE Workshop, December 5th, 2024

→ NASA 2DWMH Test Case for Turbulence Modelling

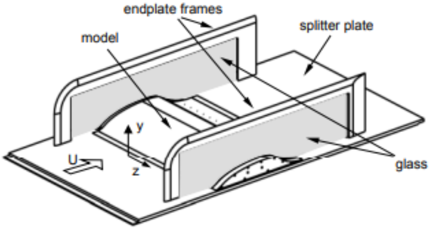
→ Multi–fidelity approximation → Multi-Index Stochastic Collocation(MISC)

→ “Speeding up UQ projects from prototype to HPC”

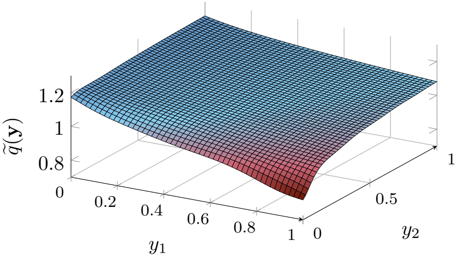
→ Conclusions and challenges

2D NASA Wall-Mounted Hump Separated Flow

parametrised inputs $\vec{y} \in \Gamma \mapsto$ quantity of interest $q(\vec{y}) \in \mathbb{R}$



Greenblatt et al, A Separation Control CFD Validation Test Case Part 1: Baseline & Steady Suction, 2004

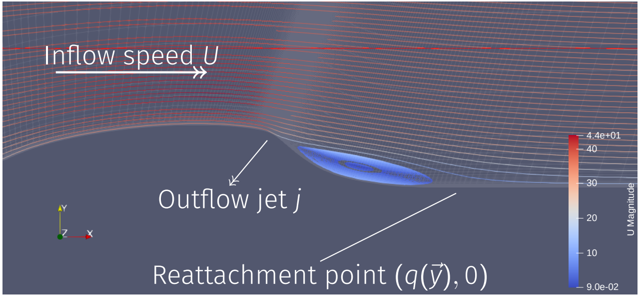


Example Surrogate Model

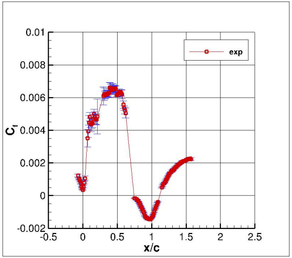
Quantity of Interest

Seek surrogate model for flow reattachment point $q : \Gamma := [0,1]^2 \to \mathbb{R}$

- Outflow jet speed $j(\vec{y}) := 23.4(0.1 + 0.9y_1)$
- Inflow speed $U(\vec{y}) := 34.6(0.1 + 0.9y_2)$



Streamlines for 2DWMH realisation

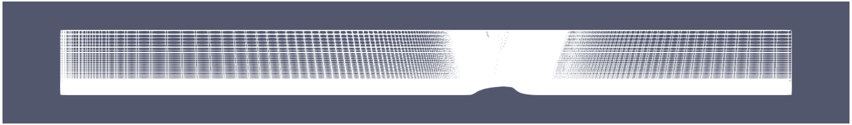


Greenblatt, D. et al, Skin-Friction Measurements on the NASA Hump Model, 2006

2DWMH Solver

Non intrusive: For parameter $\vec{y}$ compute reattachment point $q(\vec{y})$

- Reynolds Averaged Navier Stokes (RANS) with Spalart–Allmaras (SA) turbulence.
- OpenFOAM `simpleFoam` finite volume solver.



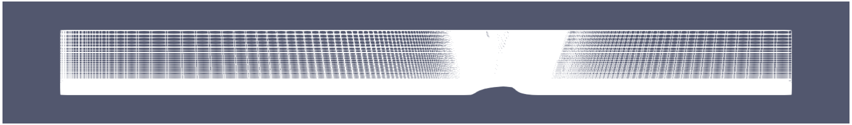
High fidelity: 229282 points, tolerance 10^{-10}

solve time
$\approx 1 \times 10^4$ s
≈ 3 hours

2DWMH Solver

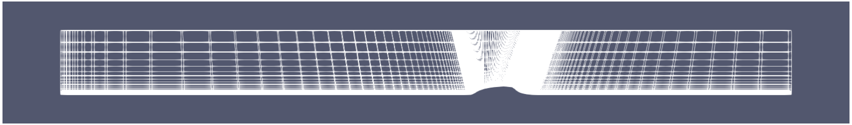
Non intrusive: For parameter $\vec{y}$ compute reattachment point $q(\vec{y})$

- Reynolds Averaged Navier Stokes (RANS) with Spalart–Allmaras (SA) turbulence.
- OpenFOAM `simpleFoam` finite volume solver.



High fidelity: 229282 points, tolerance 10^{-10}

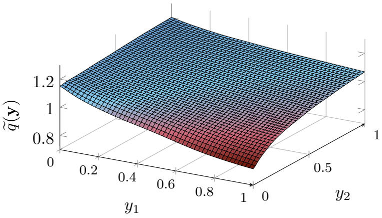
solve time
$\approx 1 \times 10^4$ s
≈ 3 hours



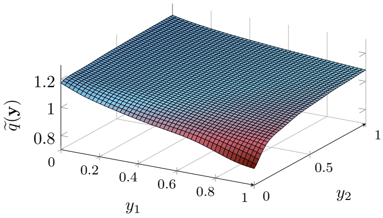
Low fidelity: 35788 points, tolerance 10^{-4}

solve time
$\approx 8 \times 10^1$ s
≈ 1 minute

Surrogate Models — Sparse Grid Interpolation

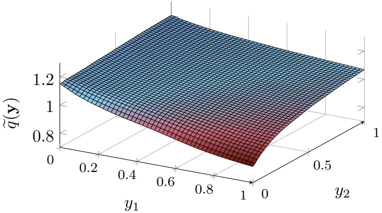


Low fidelity surrogate (scaled cost $\approx 10^1$)

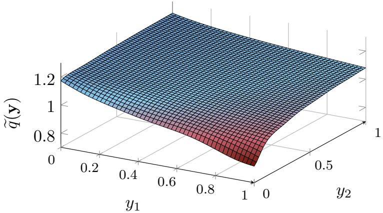


High fidelity surrogate (scaled cost $\approx 10^3$)

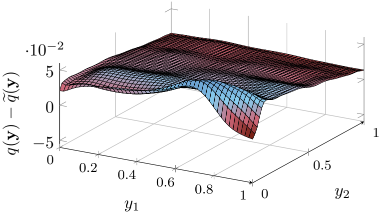
Surrogate Models — Sparse Grid Interpolation



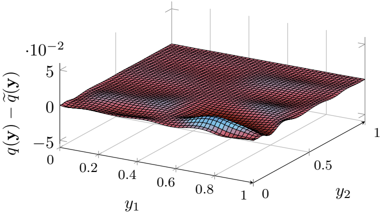
Low fidelity surrogate (scaled cost $\approx 10^1$)



High fidelity surrogate (scaled cost $\approx 10^3$)

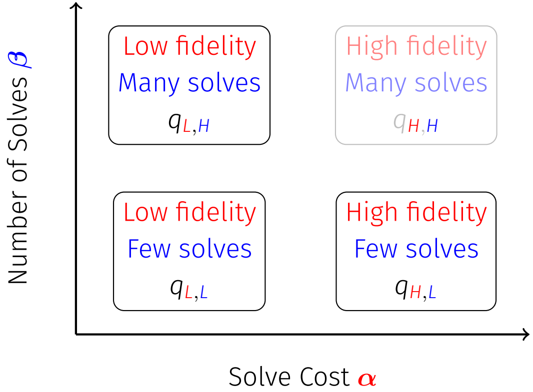


Low fidelity error surface



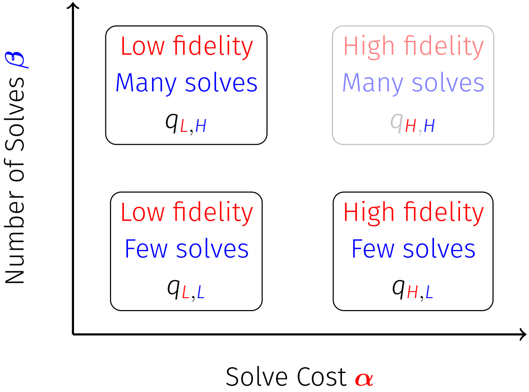
High fidelity error surface

Multi-Fidelity Approximation



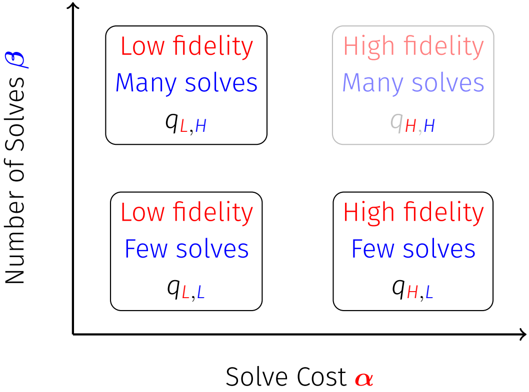
$$q \approx q_{H,H}$$

Multi-Fidelity Approximation



$$
\begin{aligned}
q &\approx q_{H,H} \\
&\approx q_{L,L} + (q_{H,L} - q_{L,L}) \\
&\quad + (q_{L,H} - q_{L,L})
\end{aligned}
$$

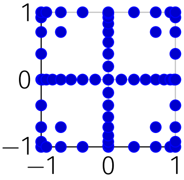
Multi-Fidelity Approximation



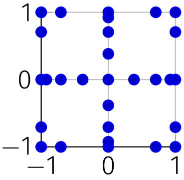
$$
\begin{aligned}
q &\approx q_{H,H} \\
&\approx q_{L,L} + (q_{H,L} - q_{L,L}) \\
&\qquad + (q_{L,H} - q_{L,L}) \\
&\approx q_{H,L} + q_{L,H} - q_{L,L}
\end{aligned}
$$

Multi Fidelity Sparse Grid Surrogates

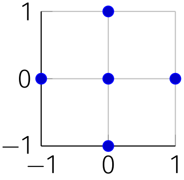
Sparse Grids



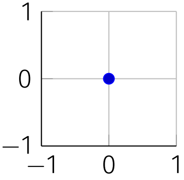
Fidelity $\alpha = 1$



Fidelity $\alpha = 2$



Fidelity $\alpha = 3$



Fidelity $\alpha = 4$

Multi-Index Stochastic Collocation

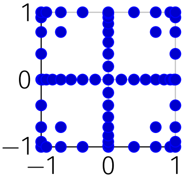
$$\tilde{q}(\vec{y}) := \sum_{[\boldsymbol{\alpha},\boldsymbol{\beta}]\in I} c_{[\boldsymbol{\alpha},\boldsymbol{\beta}]}\mathcal{I}^{\boldsymbol{\beta}}[q^{\boldsymbol{\alpha}}]$$

where

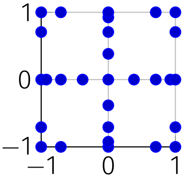
- $\boldsymbol{\alpha}$ controls model fidelity,
- $\boldsymbol{\beta}$ controls parametric accuracy.

Multi Fidelity Sparse Grid Surrogates

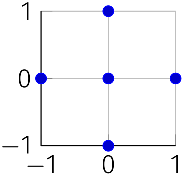
Sparse Grids



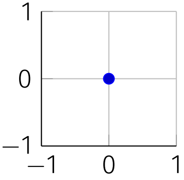
Fidelity $\alpha = 1$



Fidelity $\alpha = 2$



Fidelity $\alpha = 3$



Fidelity $\alpha = 4$

Multi-Index Stochastic Collocation

$$\widetilde{q}(\vec{y}) := \sum_{[\boldsymbol{\alpha},\boldsymbol{\beta}]\in I} c_{[\boldsymbol{\alpha},\boldsymbol{\beta}]} \mathcal{I}^{\boldsymbol{\beta}}[q^{\boldsymbol{\alpha}}]$$

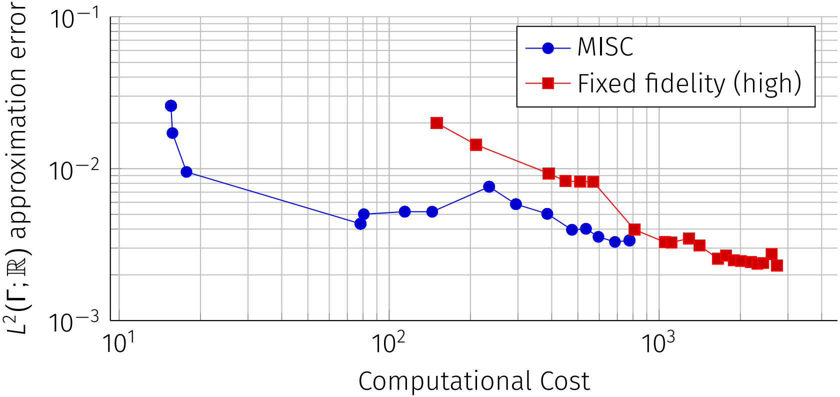
where

- $\boldsymbol{\alpha}$ controls model fidelity,
- $\boldsymbol{\beta}$ controls parametric accuracy.

→ adaptive algorithm to select $[\boldsymbol{\alpha}, \boldsymbol{\beta}]$ see e.g. C. Piazzola, L. Tamellini, et al. **Comparing multi-index stochastic collocation...** In: Engineering with Computers (Feb. 2022).

Surrogate $L^2(\Gamma; \mathbb{R})$ approximation error

Consider reattachment point $q(\vec{y})$ in 2DWMH test problem



Further details

- C. Piazzola and L. Tamellini. "Algorithm 1040: The Sparse Grids Matlab Kit - a Matlab implementation of sparse grids for high-dimensional function approximation and uncertainty quantification". In: ACM Transactions on Mathematical Software (2024). DOI: 10.1145/3630023

  $\Longrightarrow$ single fidelity: `sites.google.com/view/sparse-grids-kit`

Further details

- C. Piazzola and L. Tamellini. "Algorithm 1040: The Sparse Grids Matlab Kit - a Matlab implementation of sparse grids for high-dimensional function approximation and uncertainty quantification". In: ACM Transactions on Mathematical Software (2024). DOI: 10.1145/3630023

  $\Longrightarrow$ single fidelity: `sites.google.com/view/sparse-grids-kit`
- C. Piazzola, L. Tamellini, et al. "Comparing multi-index stochastic collocation and multi-fidelity stochastic radial basis functions for forward uncertainty quantification of ship resistance". In: Engineering with Computers (Feb. 2022). DOI: 10.1007/s00366-021-01588-0
- **Work in progress:** Release of MISC MATLAB code

Further details

- C. Piazzola and L. Tamellini. "Algorithm 1040: The Sparse Grids Matlab Kit - a Matlab implementation of sparse grids for high-dimensional function approximation and uncertainty quantification". In: ACM Transactions on Mathematical Software (2024). DOI: 10.1145/3630023

  $\Longrightarrow$ single fidelity: `sites.google.com/view/sparse-grids-kit`
- C. Piazzola, L. Tamellini, et al. "Comparing multi-index stochastic collocation and multi-fidelity stochastic radial basis functions for forward uncertainty quantification of ship resistance". In: Engineering with Computers (Feb. 2022). DOI: 10.1007/s00366-021-01588-0
- **Work in progress:** Release of MISC MATLAB code
- **Work in progress:** BMK, L. Tamellini, M. Giacomini, and A. Huerta. **Multi-index stochastic collocation approximation of NASA 2DWMH test case.**

“Speeding up UQ projects from prototype to HPC”

Progession of model complexity

Development and exposition e.g. analytic functions

$$u(\vec{y}) := \exp(-c_1^2(y_1 - 0.5)^2)\exp(-c_2^2(y_2 - 0.5)^2)$$

Progession of model complexity

Development and exposition e.g. analytic functions

$$u(\vec{y}) := \exp(-c_1^2(y_1 - 0.5)^2)\exp(-c_2^2(y_2 - 0.5)^2)$$

"Practical" examples e.g. elliptic / parabolic PDEs

Progession of model complexity

Development and exposition e.g. analytic functions

$$u(\vec{y}) := \exp(-c_1^2(y_1 - 0.5)^2)\exp(-c_2^2(y_2 - 0.5)^2)$$

"Practical" examples e.g. elliptic / parabolic PDEs

PETSc

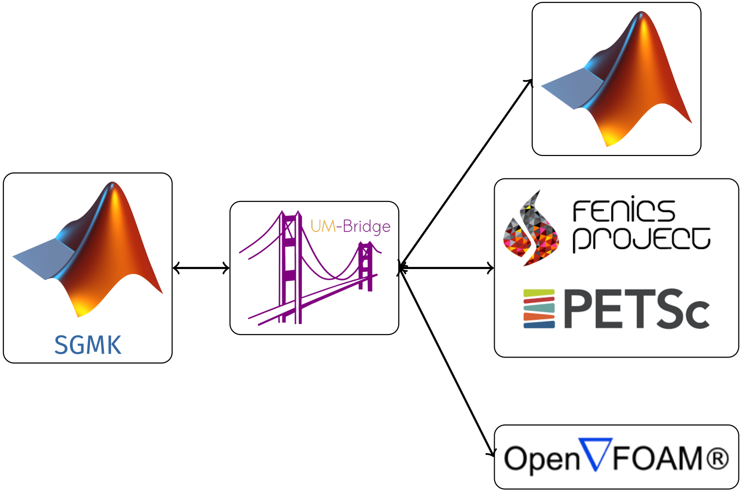
FEniCS project

"Real world" examples e.g. turbulent flow over aerofoils

OpenFOAM®

Abstraction via UM-BRIDGE

- Common interface for evaluating $u^{\alpha}(z)$ for sample points $z \in Z^{\beta}$.
- Hybrid workflow allowing local development and external solvers.
- Support for parallel evaluations.



Common Interface

Analytic function

```
u = @(y,alpha) genz(y) + noise(y)*10^(-2*alpha(1));
```

Common Interface

Analytic function

```
u = @(y,alpha) genz(y) + noise(y)*10^(-2*alpha(1));
```

Parabolic PDE

```
modelname = 'forwardparabolic'; uri = 'http://0.0.0.0:4242';
model = HTTPModel(uri,modelname, 'webwrite');
u = @(y,alpha) model.evaluate(y,struct('Fidelity',alpha));
```

Common Interface

Analytic function

```
u = @(y,alpha) genz(y) + noise(y)*10^(-2*alpha(1));
```

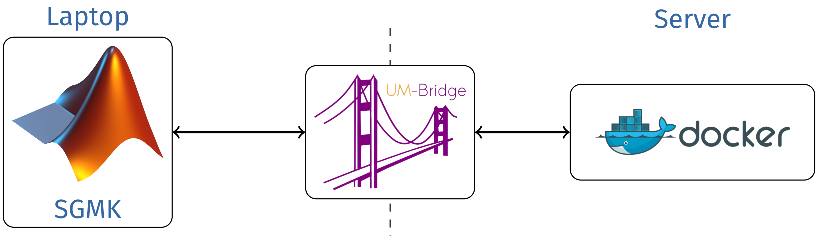
Parabolic PDE

```
modelname = 'forwardparabolic'; uri = 'http://0.0.0.0:4242';
model = HTTPModel(uri,modelname, 'webwrite');
u = @(y,alpha) model.evaluate(y,struct('Fidelity',alpha));
```

RANS Solver with SA turbulence model

```
modelname = 'forwardopenfoam'; uri = 'http://0.0.0.0:4242';
model = HTTPModel(uri,modelname, 'webwrite');
u = @(y,alpha) model.evaluate(y,struct('Fidelity',alpha));
```

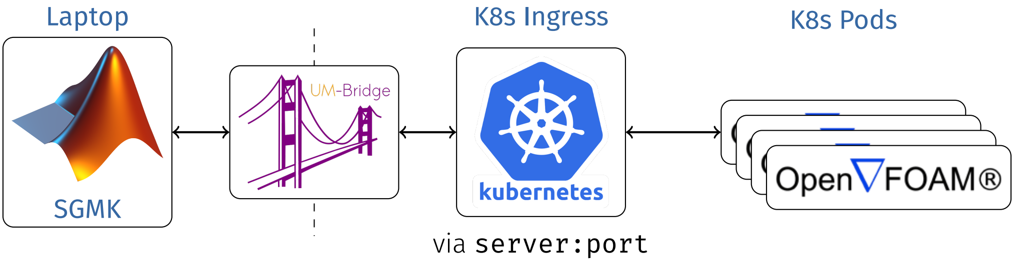
External Solver Workflow



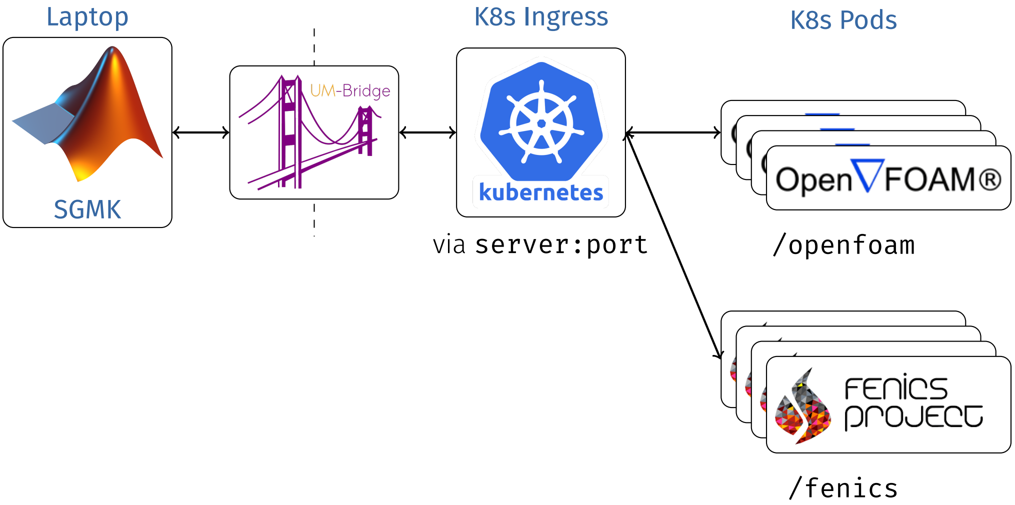
Sketch of Dockerfile

```
FROM ubuntu:latest ... # Set up environment
RUN apt install -y openfoam2406-dev ... # Install OpenFOAM
EXPOSE 4242
CMD python3 umbridge_server.py # Run python server
```

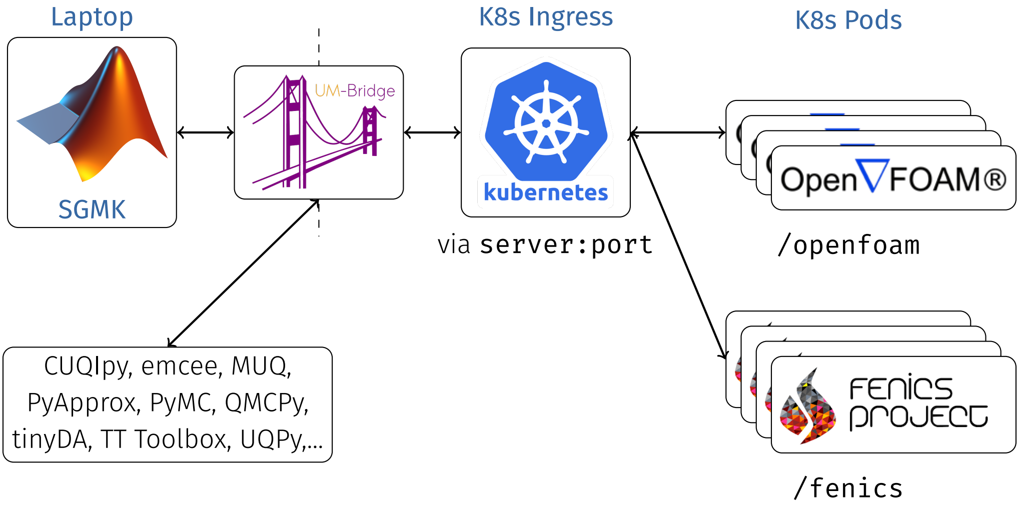
Parallel Computing — Kubernetes



Parallel Computing — Kubernetes

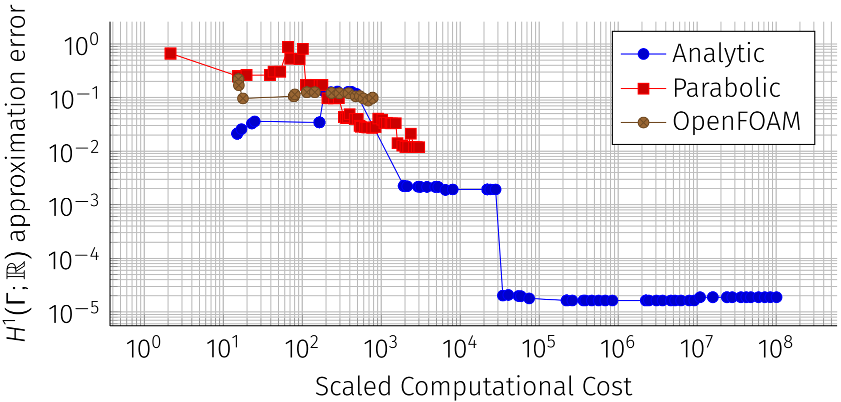


Parallel Computing — Kubernetes



MISC Results — $H^1(\Gamma;\mathbb{R})$ Approximation Error

Supports the development of a novel variant of MISC algorithm.



Advantages (and some challenges)

- Local development of algorithm (and models via CI/GITHUB ACTIONS with Dockerhub).
- Common interfaces allow for simple shared algorithm codebase.
- “Speeding up UQ projects from prototype to HPC” — can run many models (for many different projects) using shared resources.

Advantages (and some challenges)

- Local development of algorithm (and models via CI/GITHUB ACTIONS with Dockerhub).
- Common interfaces allow for simple shared algorithm codebase.
- "Speeding up UQ projects from prototype to HPC" — can run many models (for many different projects) using shared resources.

Challenges?

- Repeated calls to expensive solver — save output to a `PersistentVolume`.
  - Full field data $N = n_{realisations} n_{iter} n_{spatial} n_{fields} = 10^2 \cdot 10^3 \cdot 10^6 \cdot 10^1 = 10^{12} \approx 1TB$
  - Post processed subset $N = n_{realisations} n_{spatial} n_{fields} = 10^2 \cdot 10^3 \cdot 10^1 = 10^6 \approx 1MB$
  - QoI $N = n_{realisations} = 10^2 \approx 100 bytes$

Advantages (and some challenges)

- Local development of algorithm (and models via CI/GITHUB ACTIONS with Dockerhub).
- Common interfaces allow for simple shared algorithm codebase.
- "Speeding up UQ projects from prototype to HPC" — can run many models (for many different projects) using shared resources.

Challenges?

- Repeated calls to expensive solver — save output to a `PersistentVolume`.
  - Full field data $N = n_{realisations} n_{iter} n_{spatial} n_{fields} = 10^2 \cdot 10^3 \cdot 10^6 \cdot 10^1 = 10^{12} \approx 1TB$
  - Post processed subset $N = n_{realisations} n_{spatial} n_{fields} = 10^2 \cdot 10^3 \cdot 10^1 = 10^6 \approx 1MB$
  - QoI $N = n_{realisations} = 10^2 \approx 100 bytes$
- Other HPC architecture? e.g. we are yet to experiment with Slurm (it may also be straightforward!)

Thank you!

Benjamin M. Kent `kent@imati.cnr.it`
Lorenzo Tamellini `tamellini@imati.cnr.it`
CNR–IMATI, Pavia