AI, Machine Learning & Visualisation at UKAEA

5th IFERC Workshop on GPU Fusion Applications

20 June 2024

S.Pamela, R.Akers, N.Amorisco, N.Bhatia, D.Brennand, J.Buchanan, N.Carey, E.Crovini, O.El-Zobaidi, S.Etches, V.Gopakumar, S.Jackson, E.Lewis, E.Ozturk, K.Pentland, C.Siddle, L.Zanisi, J.Brandstetter, M.Hoelzl, G.Huijsmans
and many others…

This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014-2018 and 2019-2020 under grant agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

Outline

Foreword: Only a small fraction of all activities (and co-authors)

- The various teams

- Overview of main activities

- Neural-Parareal

- Foundation Models

Outline

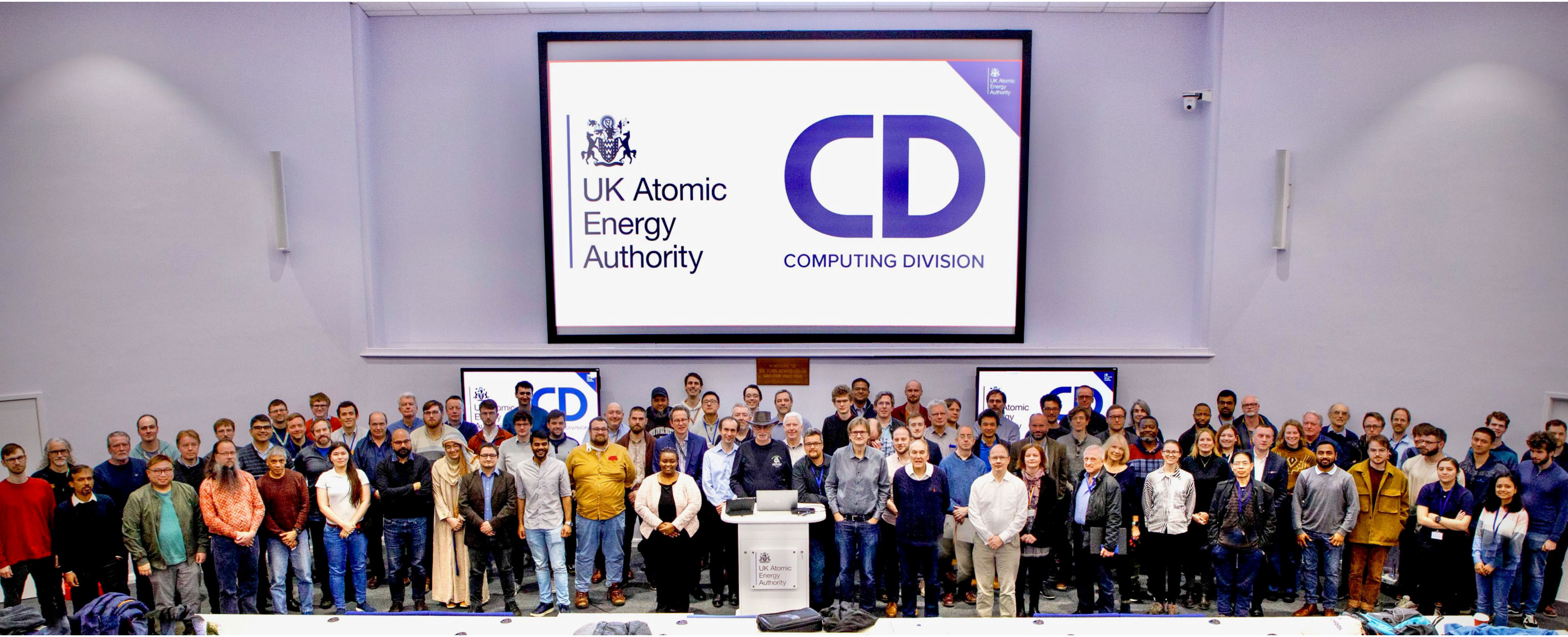
Foreword: Only a small fraction of all activities (and co-authors)

- **The various teams**
- Overview of main activities
- Neural-Parareal
- Foundation Models

The AI-ML-Viz Team

The AI-ML-Viz Team

The UKAEA Computing Division



The UK Fusion Computing Lab

STEP: Spherical Tokamak For Energy Production

STEP: Deliver energy to the grid by 2040


iter
china eu india japan korea russia usa
TOKAMAK
&PLANT SYSTEMS
L4
L3
L2
L1
B1
B2
JUNE2014
DESIGNED BY LAURIS HONORE

Tokamak Design is an AI & Exascale Challenge

Cannot build 20 demonstration power plants
=> *Exascale and AI* is needed to design & optimise STEP and future fusion power plants

- **complex engineering**
- **in-silico design optimization**
- **model-based predictions with large uncertainty**

Outline

Foreword: Only a small fraction of all activities (and co-authors)

- The various teams

- **Overview of main activities**

- Neural-Parareal

- Foundation Models

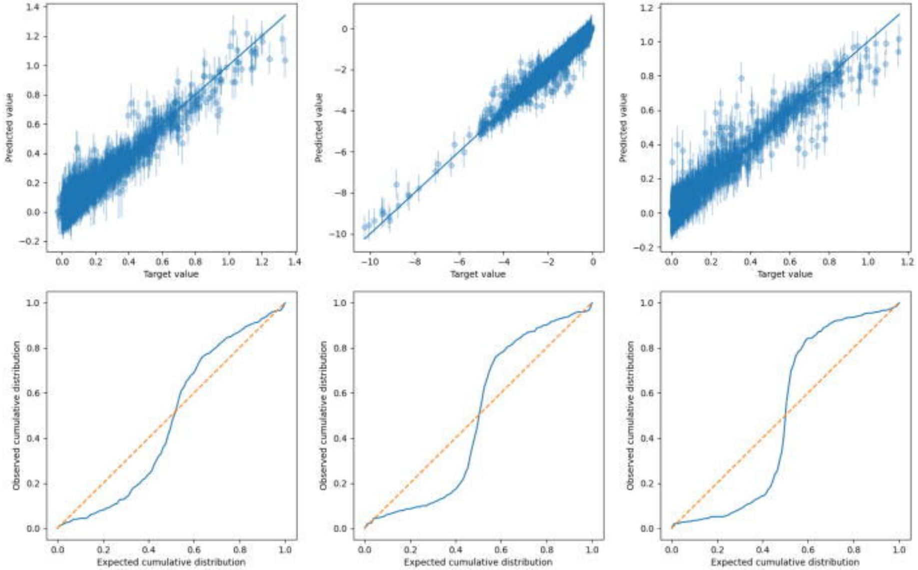
AI activities at UKAEA: MTM stability Surrogates for Integrated Models

Will Hornsby, in collaboration with Digilab ltd

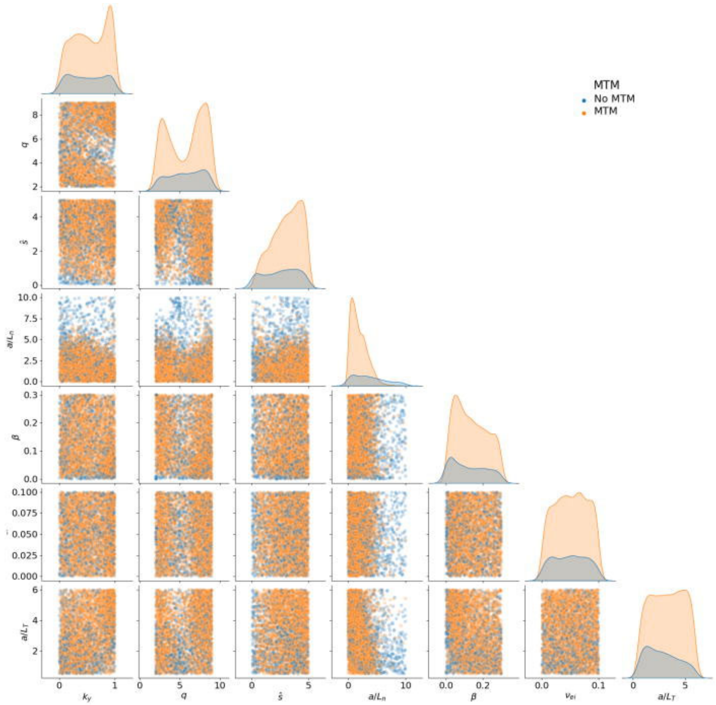
Using Gaussian Processes to emulate MTM stability with GS2

[W.Hornsby et al., *Phys. Plasmas 31, 012303 (2024)*]

digiLab

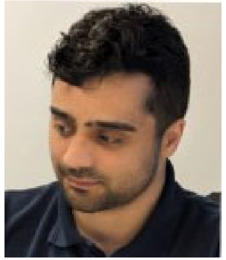


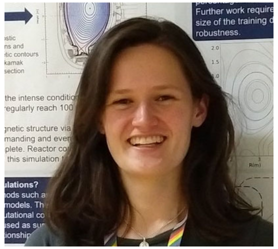
(a) Growth Rate. (b) Mode Frequency. (c) Electron Heat Flux.



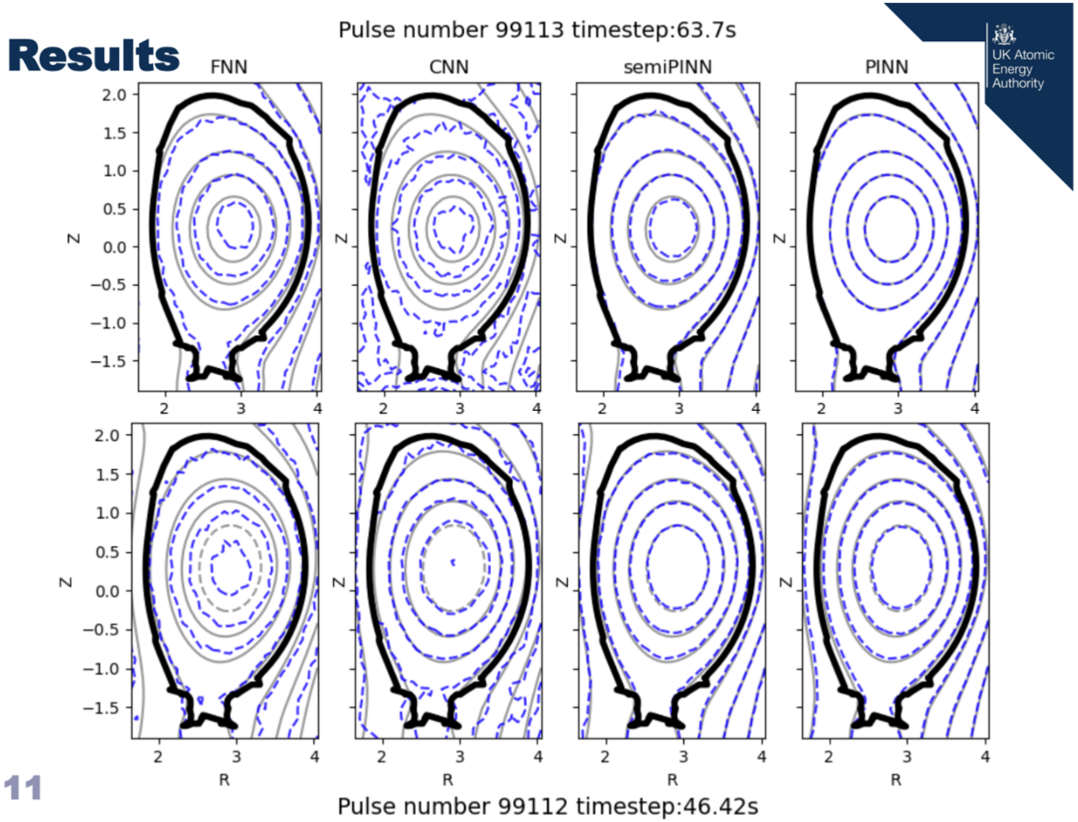
AI activities at UKAEA: MTM stability Surrogates for Integrated Models

Nico Amorisco (UKAEA)
Steve Etches (UKAEA)
Emily Lewis (UCL PhD)
Omar El-Zobaidi (Placement)





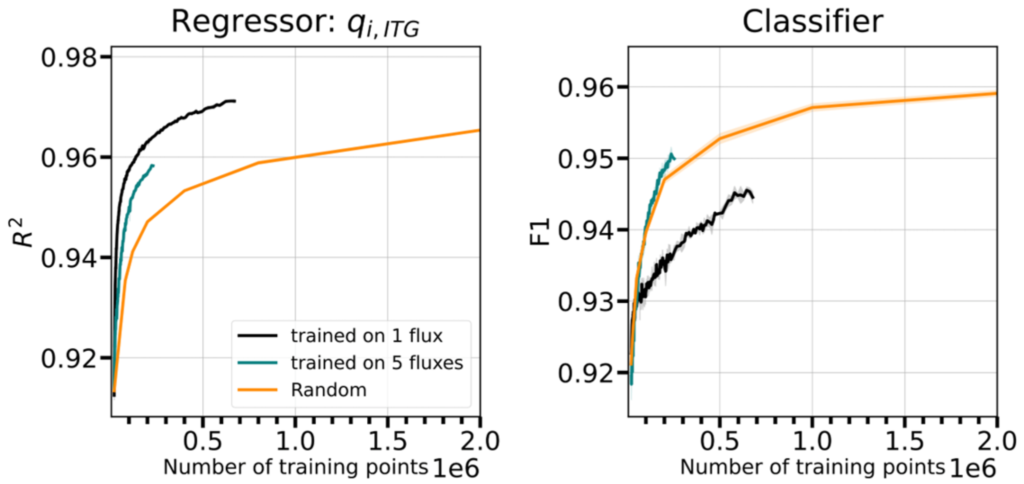
[N.Amorisco et al., "FreeGSNKE: A Python-based dynamic free-boundary toroidal plasma equilibrium solver", Phys. Plasmas 31, 042517 (2024)]()



Emily Lewis, PhD at UCL:
Surrogate of plasma equilibrium

AI activities at UKAEA: Active Learning & Optimisation

Lorenzo Zanisi (UKAEA)
Enrico Crovini (Imperial PhD)
Theo Brown (UCL PhD)
Catherine Siddle (Grad-Scheme)

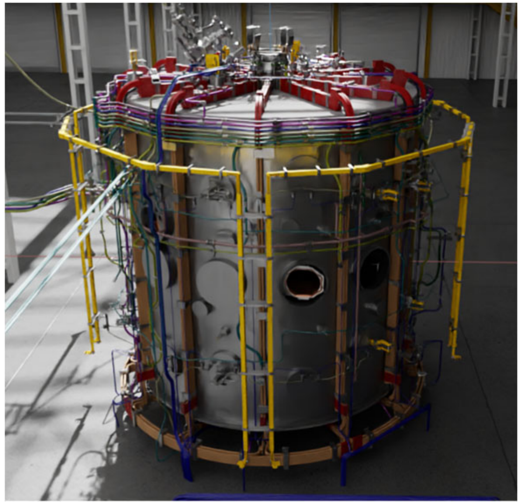


Active Learning for Qualikiz, [L. Zanisi et al 2024 Nucl. Fusion 64 036022]()

[E.Crovini et al. "Automatic JOREK calibration via batch Bayesian optimization", Physics of Plasmas 31, 063901 (2024)]()

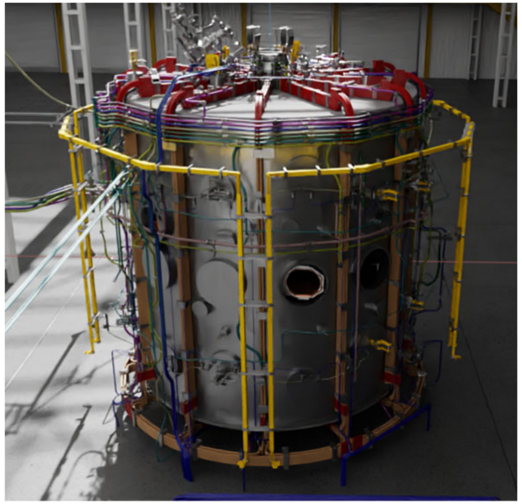
AI activities at UKAEA: Advanced Visualisation & AI

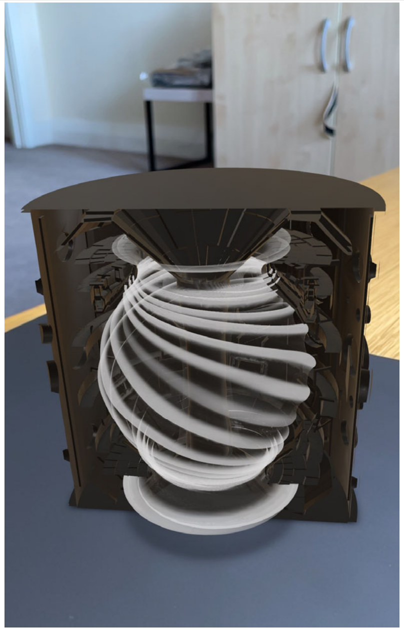
Nitesh Bhatia (UKAEA)
Ekin Ozturk (Imperial PhD)



AI activities at UKAEA: Advanced Visualisation & AI

Nitesh Bhatia (UKAEA)
Ekin Ozturk (Imperial PhD)

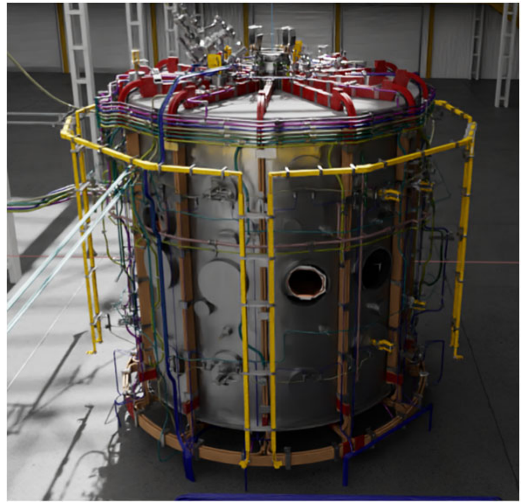


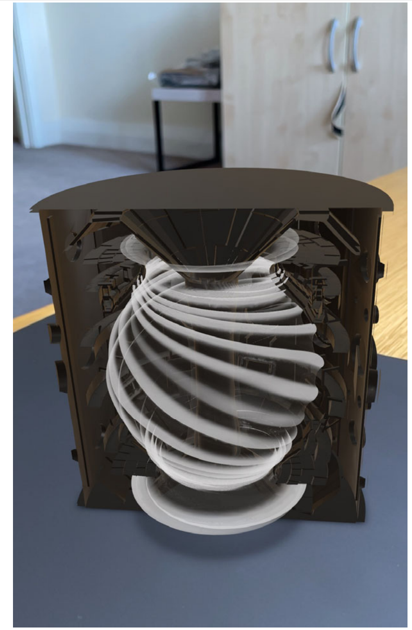


https://niteshbhatia008.github.io/nb-webxr-viewer/

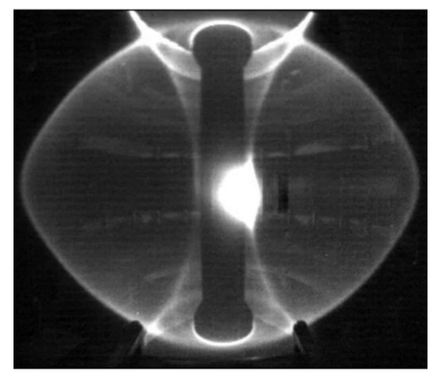
AI activities at UKAEA: Advanced Visualisation & AI

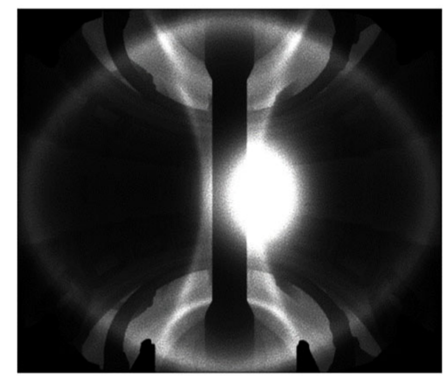
Nitesh Bhatia (UKAEA)
Ekin Ozturk (Imperial PhD)





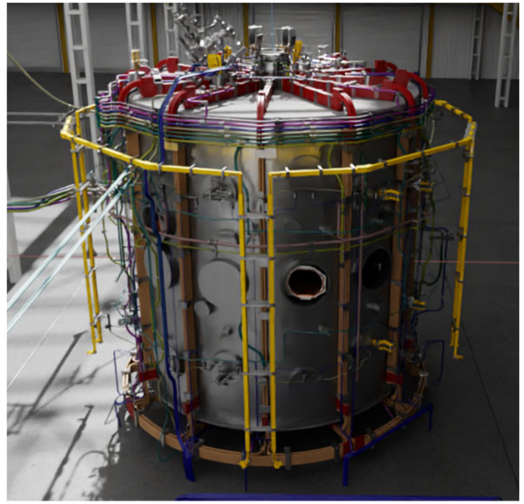
https://niteshbhatia008.github.io/nb-webxr-viewer/

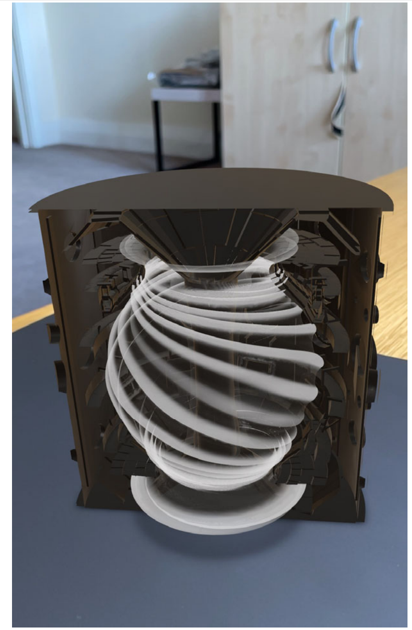




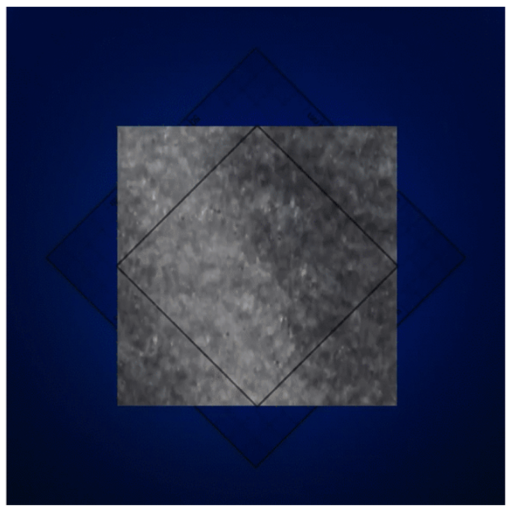
AI activities at UKAEA: Advanced Visualisation & AI

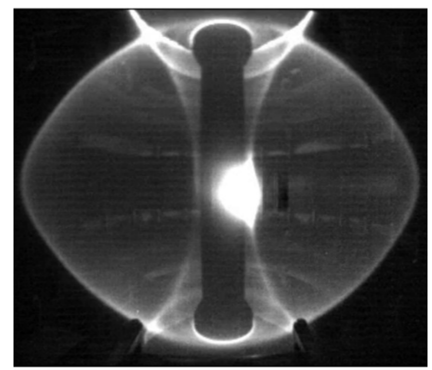
Nitesh Bhatia (UKAEA)
Ekin Ozturk (Imperial PhD)

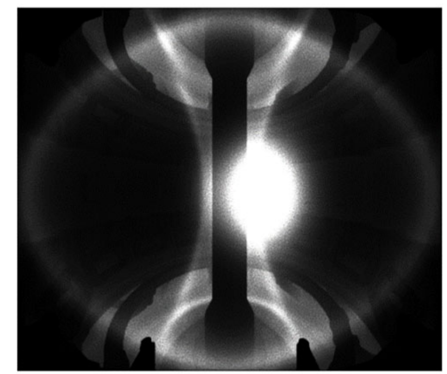




https://niteshbhatia008.github.io/nb-webxr-viewer/

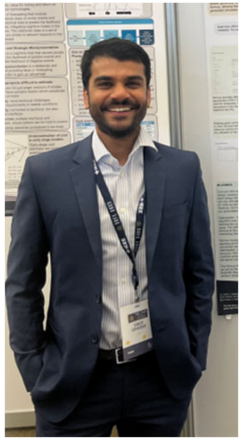


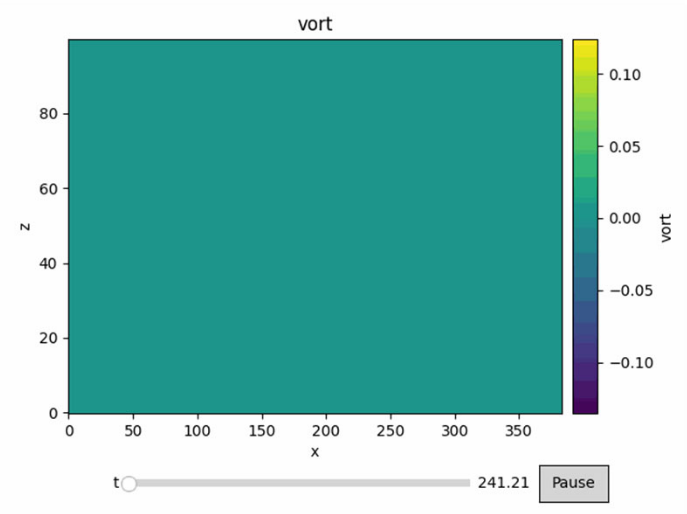




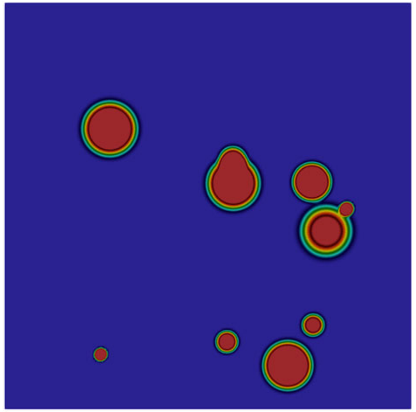
AI activities at UKAEA: Neural Operators & PINNs

Vignesh Gopakumar (UKAEA)
Naomi Carey (UKAEA Apprenticeship)
Daniel Brennand (UKAEA Apprenticeship)





STORM/BOUT++

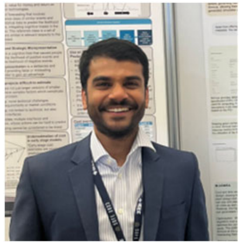


JOREK

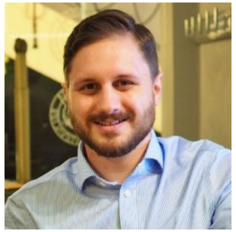
[*V.Gopakumar et al. 2024 Nucl. Fusion 64 056025*]
[*N.Carey et al., IAEA-FEC 2023*]

AI activities at UKAEA: Foundation Models

Samuel Jackson (UKAEA)
Vignesh Gopakumar (UKAEA)
Naomi Carey (UKAEA Apprenticeship)
Lorenzo Zanisi (UKAEA)
Nathan Cummings (UKAEA)
Johannes Brandstetter (JKU)





Main collaborations:
- Linz
- Turing Institute
- IBM

Outline

Foreword: Only a small fraction of all activities (and co-authors)

- The various teams

- Overview of main activities

- **Neural-Parareal**

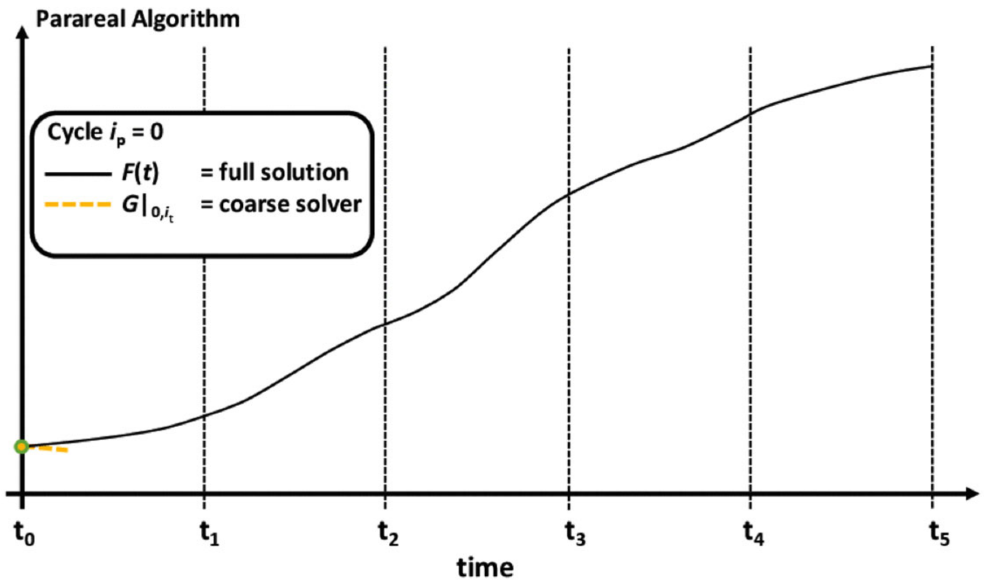
- Foundation Models

Neural-Parareal: Basics of Parareal

Split time domain into parallel windows [J.-L. Lions et al., Comptes Rendus de l'Académie des Sciences, Série I. 332 (7): 661–668 (2015)]

fast approximation with *Coarse-Solver*

correction using Fine-Solver



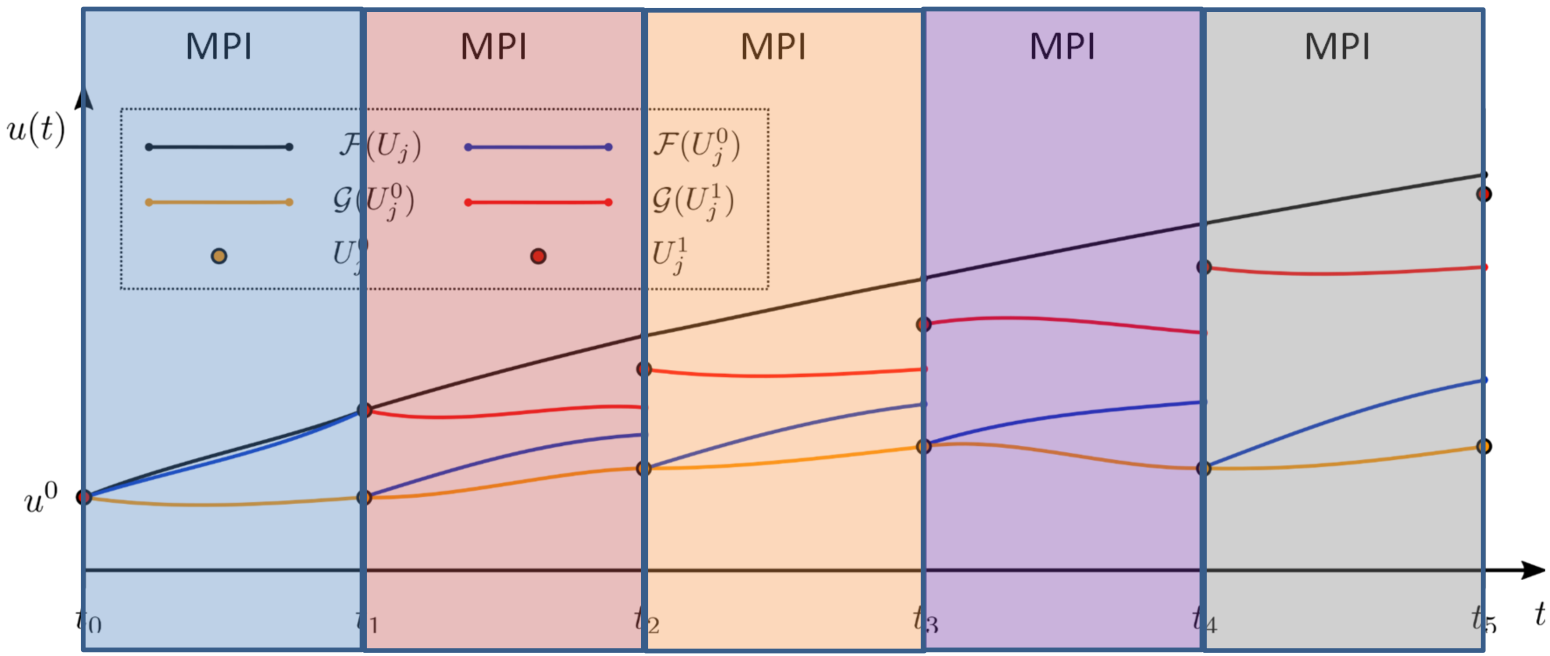
Neural-Parareal: Basics of Parareal

Basics of Parareal: split time domain into parallel windows

fast approximation with *Coarse-Solver*

correction using Fine-Solver

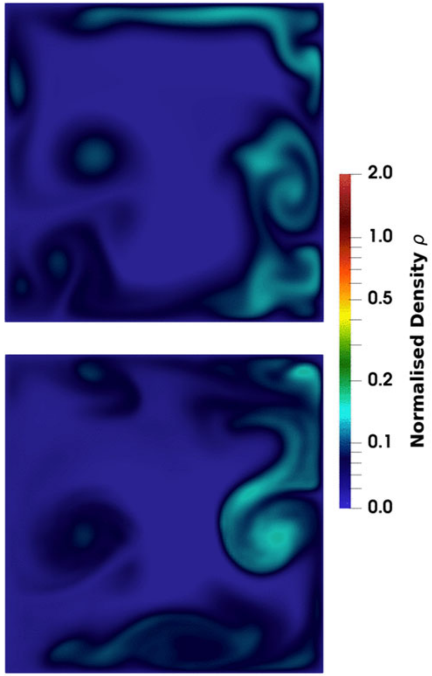
better coarse solver => few iterations => high speedup



***Neural-Parareal, S.Pamela (submitted to CPC)* [*https://arxiv.org/abs/2405.01355*](https://arxiv.org/abs/2405.01355)**

Neural-Parareal: Parareal with a Neural Operator

Basics of Parareal: split time domain into parallel windows
fast approximation with *Coarse-Solver*
correction using Fine-Solver
better coarse solver => few iterations => high speedup
=> use Machine Learning surrogates

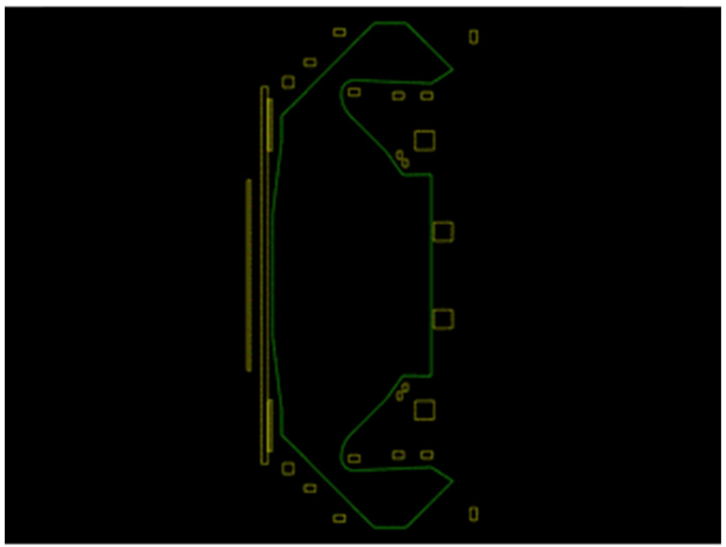


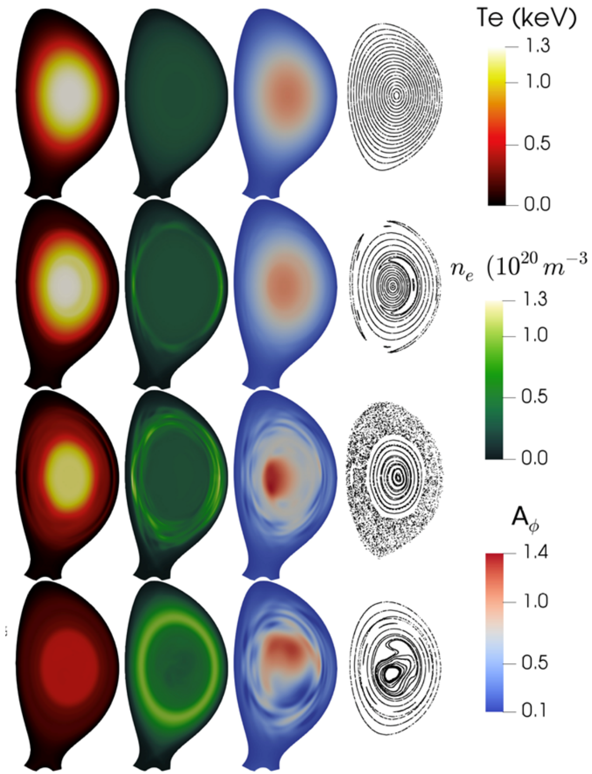
Top: ground truth
Bottom: neural PDE solver

Neural-Parareal, S.Pamela (submitted to CPC) https://arxiv.org/abs/2405.01355

Neural-Parareal: JOREK Demonstration

JOREK: non-linear MHD solver for tokamak plasmas [*jorek.eu*]
Comes with simplified models (basically NS in toroidal geometry, and Hasegawa-Wakatani)
Full simulations address things like plasma-edge filamentation, or disruption (loss of plasma control)





Neural-Parareal, S.Pamela (submitted to CPC) ***https://arxiv.org/abs/2405.01355***

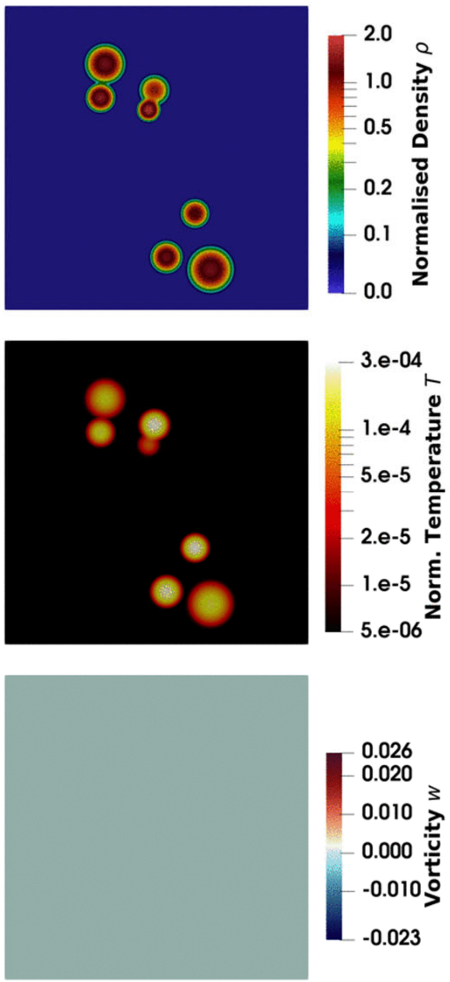
Neural-Parareal: Reduced Models in JOREK

Blobs with 3-variables model (Navier Stokes in torus)

- **ρ, T, Φ (stream function)**
- **Plus 1 auxiliary variable: vorticity $w = \nabla^2\Phi$**

Radially motion due to ∇p and toroidal geometry

Hotter blobs move faster



Neural-Parareal, S.Pamela (submitted to CPC) https://arxiv.org/abs/2405.01355

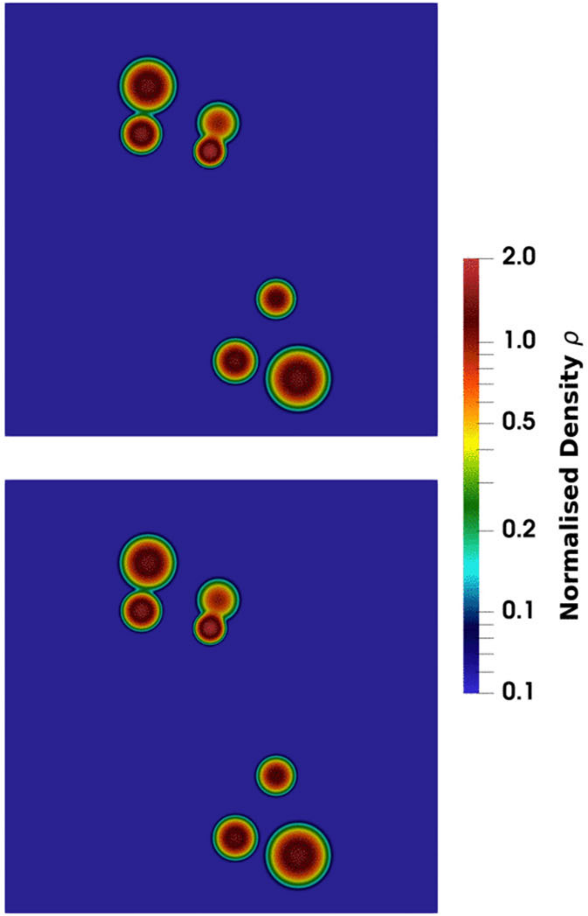
Neural-Parareal: Reduced Models in JOREK

Top: **electrostatic model**
Bottom: electromagnetic model (RMHD)

Blobs with Reduced-MHD model
Used extensively for fusion

- **4 variables: ρ, T, Φ, ψ (magnetic potential)**
- **Plus 2 auxiliary variables:**
    - **Vorticity $w = \nabla^2\Phi$**
    - **Current $j = \nabla^2\psi$**

Effectively 6 variables
Behaviour is quite different



Neural-Parareal, S.Pamela (submitted to CPC) https://arxiv.org/abs/2405.01355

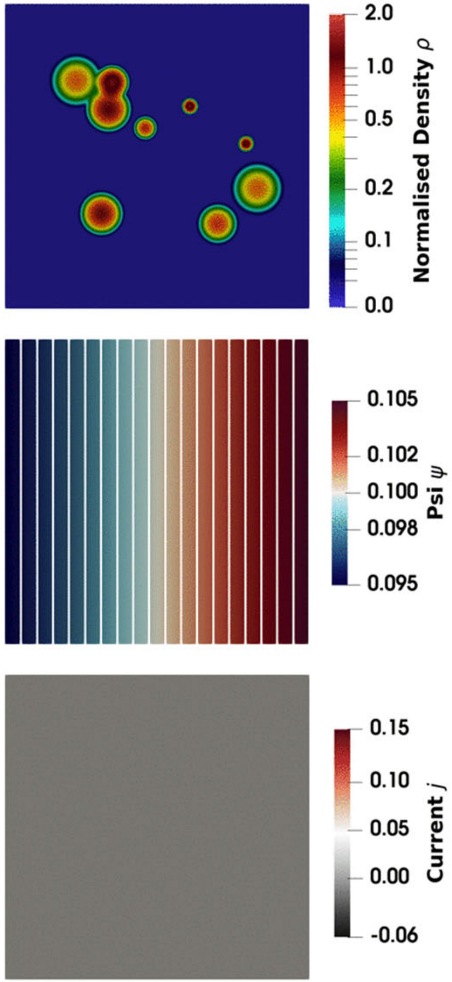
Neural-Parareal: Reduced Models in JOREK

Blobs with Reduced-MHD model
Used extensively for fusion

- **4 variables: ρ, T, Φ, ψ (magnetic potential)**
- **Plus 2 auxiliary variables:**
  - **Vorticity $w = \nabla^2\Phi$**
  - **Current $j = \nabla^2\psi$**

Effectively 6 variables
Behaviour is quite different

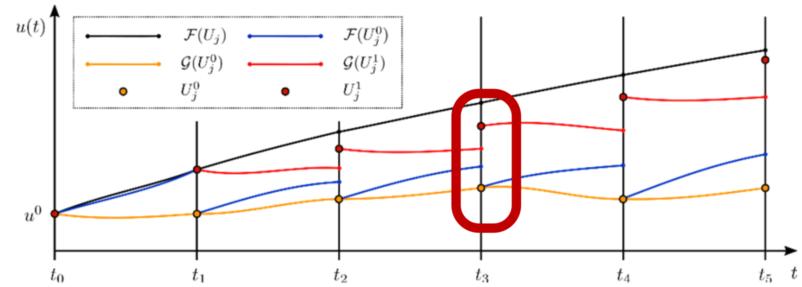
=> Blobs create their own internal current
=> which in turn affects velocity

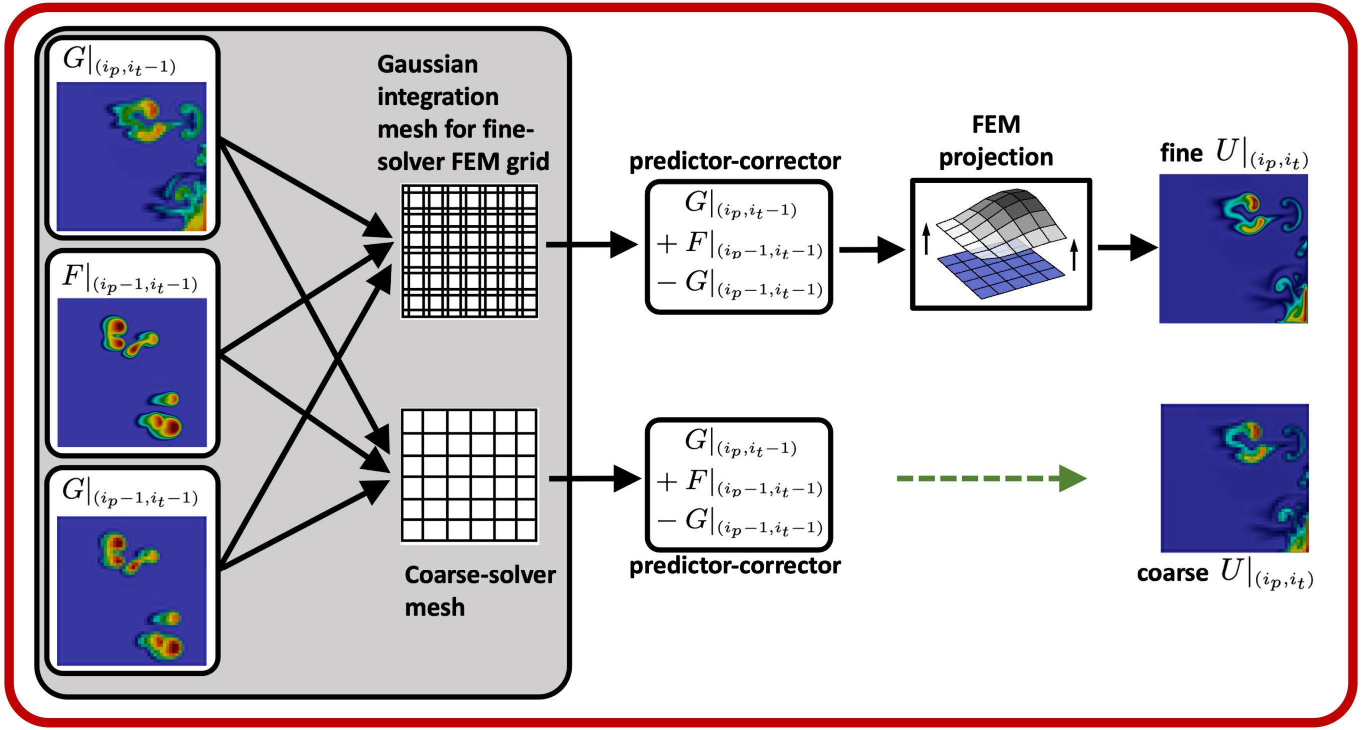


Neural-Parareal, S.Pamela (submitted to CPC) https://arxiv.org/abs/2405.01355

Neural-Parareal: Predictor-Corrector with JOREK

In theory, Parareal is "non-intrusive"
In practice, it requires a lot of work with the code's i/o
For FEM code, even more complex due to projection between resolutions





Neural-Parareal, S.Pamela (submitted to CPC) https://arxiv.org/abs/2405.01355

Neural-Parareal: Parareal with a Neural Operator

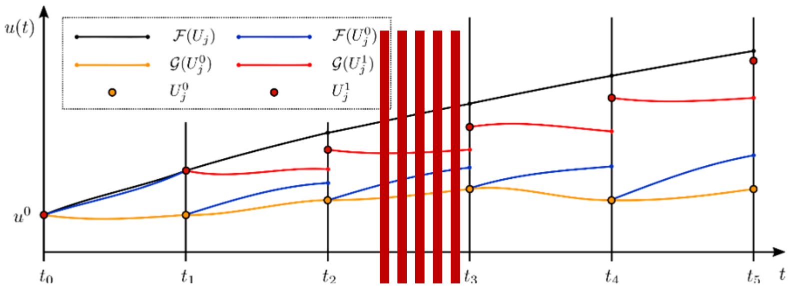
Except that a Neural PDE solver requires several input timesteps

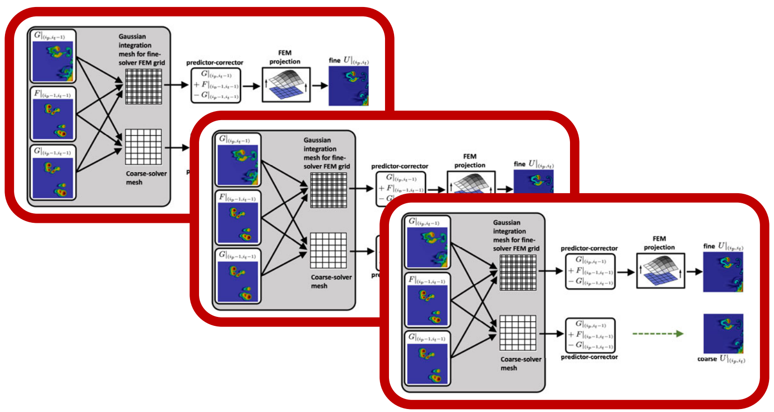
=> Need to apply the predictor-corrector to many timesteps

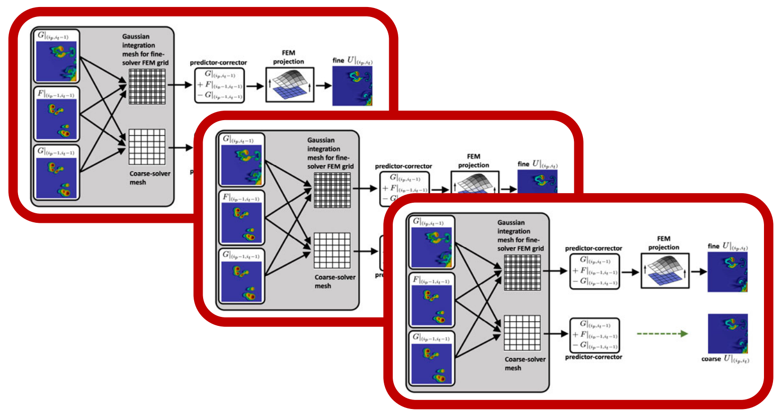
=> Even more i/o

=> Projections are costly => needs to be parallelised (otherwise can easily dominate workflow)

=> end up with a lot of extra data!



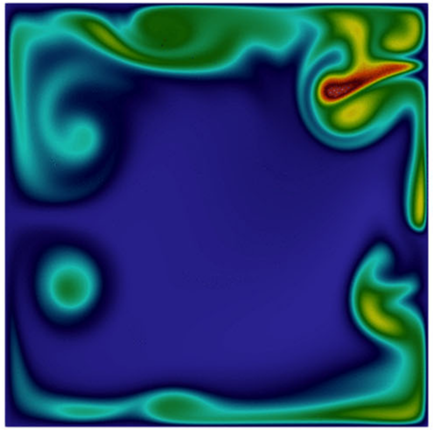


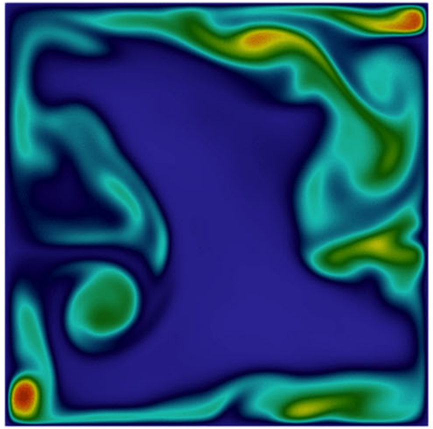


Neural-Parareal: Parareal Demonstration

Top: ground truth
Bottom: Parareal evolution (last timestep)

It works!
Looking at difference of last timestep with ground truth
(top is ground truth, bottom is parareal evolution)



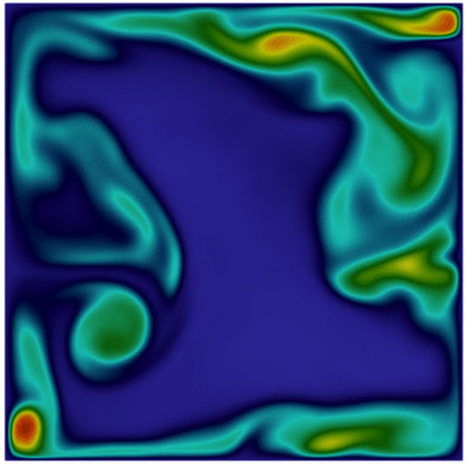


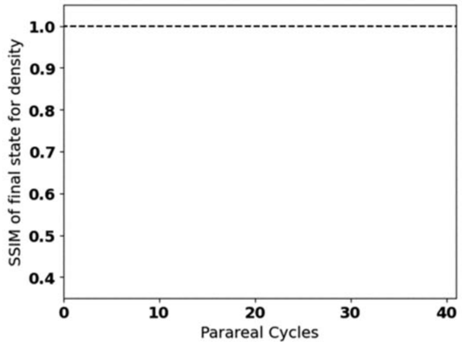
Neural-Parareal: SSIM Measure

Top: Parareal evolution (last time-step)
Bottom: Corresponding SSIM evolution

Using SSIM (Structural Similarity Index Measure)
Better than MSE for generic structures of blobs
“SSIM = 1” means 100% accuracy

No matter how bad your coarse solver, Parareal will always converge to SSIM=1 at final iteration





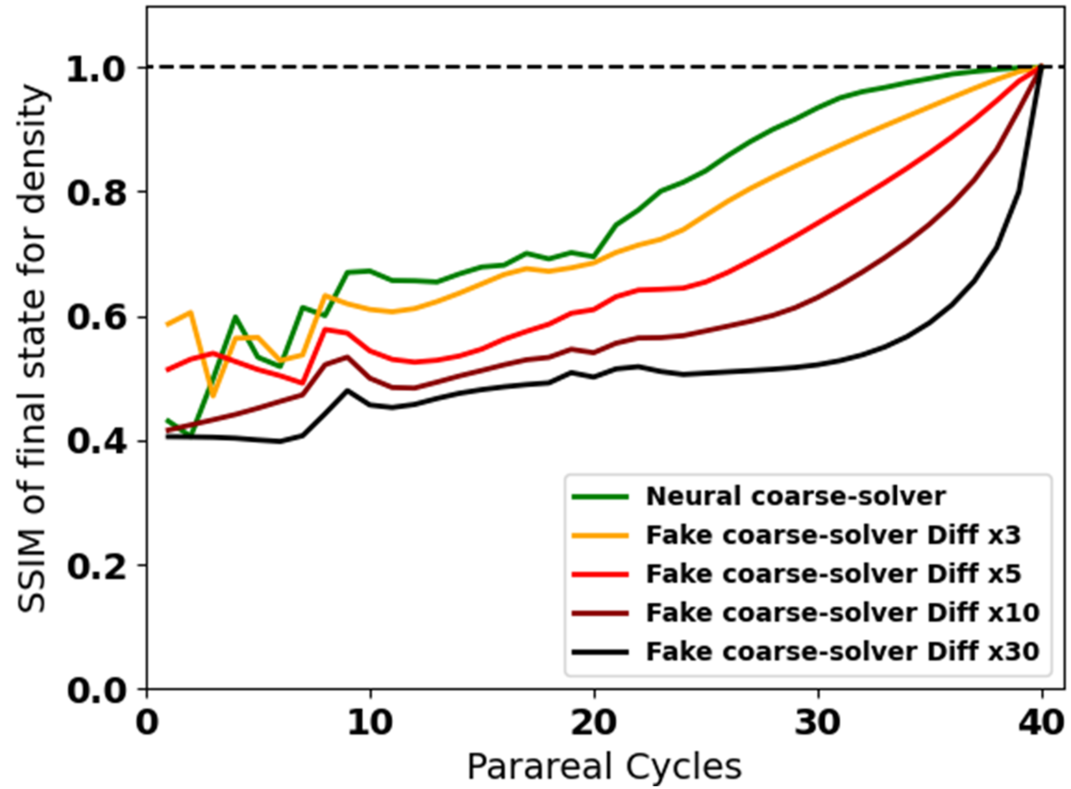
Neural-Parareal, S.Pamela (submitted to CPC) [*https://arxiv.org/abs/2405.01355*](https://arxiv.org/abs/2405.01355)

Neural-Parareal: Neural vs. Traditional Coarse Solver

"fake" coarse solver == JOREK itself, but with controlled difference

- Exactly same physics model
- Lower spatial resolution (half)
- Higher diffusion
  - => Diff x 30 is a bad coarse solver
  - => Diff x 3 is a good coarse solver

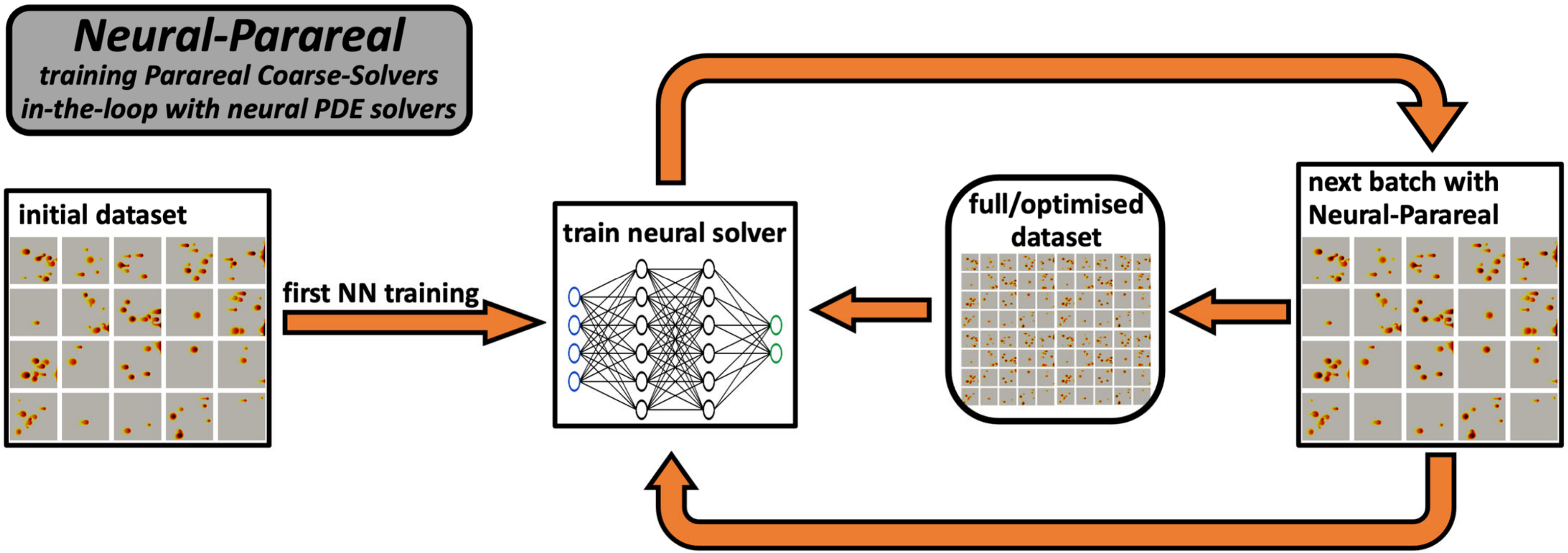
Neural Coarse solver gives better performance than Diff x 3
=> it works really well



Neural-Parareal, S.Pamela (submitted to CPC) https://arxiv.org/abs/2405.01355

Neural-Parareal: Self-Improving Framework

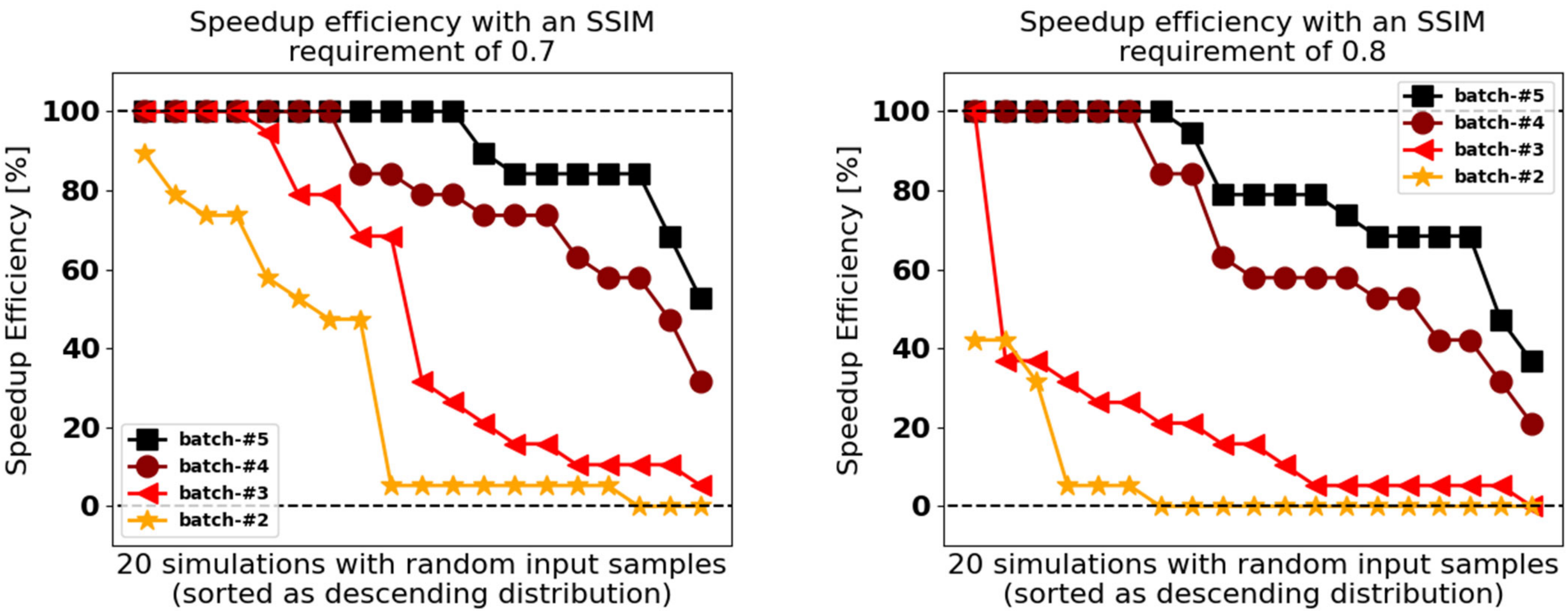
Since Parareal simulations produce a lot of data, train NN as we launch more simulations



Neural-Parareal, S.Pamela (submitted to CPC) [***https://arxiv.org/abs/2405.01355***](https://arxiv.org/abs/2405.01355)

Neural-Parareal: Main Result

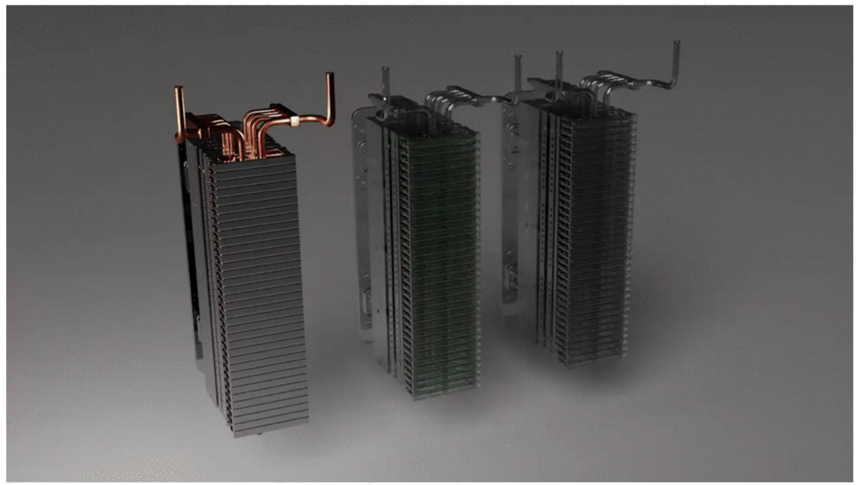
Ran 5 batches of 20 simulations
Speed-up increases significantly and quickly

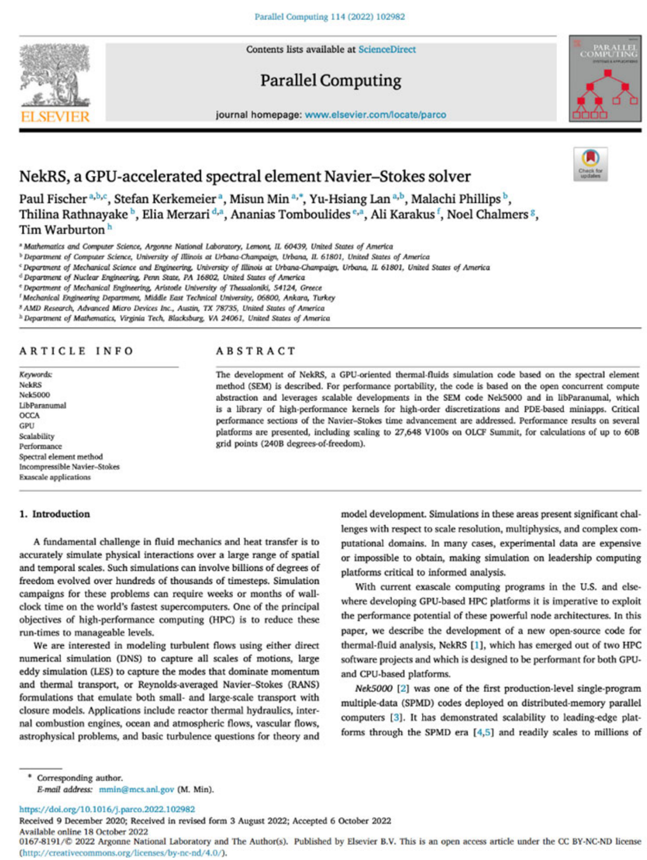


Neural-Parareal, S.Pamela (submitted to CPC) https://arxiv.org/abs/2405.01355

GPU usage and scale

- Moving away from CPU simulations
  - => Many fusion codes still require CPU's
  - => Need to move towards GPU codes (NekRS, CGYRO, XGC, OpenMC)
  - => Strong collaboration with US-NL (Exascale Computing Project ECP)





Parallel Computing 114 (2022) 102982

Contents lists available at ScienceDirect

Parallel Computing

journal homepage: www.elsevier.com/locate/parco

NekRS, a GPU-accelerated spectral element Navier–Stokes solver

Paul Fischer [a,b,c], Stefan Kerkemeier [a], Misun Min [a,*], Yu-Hsiang Lan [a,b], Malachi Phillips [b], Thilina Rathnayake [b], Elia Merzari [d,a], Ananias Tomboulides [e,a], Ali Karakus [f], Noel Chalmers [g], Tim Warburton [h]

[a] Mathematics and Computer Science, Argonne National Laboratory, Lemont, IL 60439, United States of America
[b] Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, United States of America
[c] Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801, United States of America
[d] Department of Nuclear Engineering, Penn State, PA 16802, United States of America
[e] Department of Mechanical Engineering, Aristotle University of Thessaloniki, 54124, Greece
[f] Mechanical Engineering Department, Middle East Technical University, 06800, Ankara, Turkey
[g] AMD Research, Advanced Micro Devices Inc., Austin, TX 78735, United States of America
[h] Department of Mathematics, Virginia Tech, Blacksburg, VA 24061, United States of America

ARTICLE INFO

Keywords:
NekRS
Nek5000
LibParanumal
OCCA
GPU
Scalability
Performance
Spectral element method
Incompressible Navier–Stokes
Exascale applications

ABSTRACT

The development of NekRS, a GPU-oriented thermal-fluids simulation code based on the spectral element method (SEM) is described. For performance portability, the code is based on the open concurrent compute abstraction and leverages scalable developments in the SEM code Nek5000 and in libParanumal, which is a library of high-performance kernels for high-order discretizations and PDE-based miniapps. Critical performance sections of the Navier–Stokes time advancement are addressed. Performance results on several platforms are presented, including scaling to 27,648 V100s on OLCF Summit, for calculations of up to 60B grid points (240B degrees-of-freedom).

1. Introduction

A fundamental challenge in fluid mechanics and heat transfer is to accurately simulate physical interactions over a large range of spatial and temporal scales. Such simulations can involve billions of degrees of freedom evolved over hundreds of thousands of timesteps. Simulation campaigns for these problems can require weeks or months of wall-clock time on the world's fastest supercomputers. One of the principal objectives of high-performance computing (HPC) is to reduce these run-times to manageable levels.

We are interested in modeling turbulent flows using either direct numerical simulation (DNS) to capture all scales of motions, large eddy simulation (LES) to capture the modes that dominate momentum and thermal transport, or Reynolds-averaged Navier–Stokes (RANS) formulations that emulate both small- and large-scale transport with closure models. Applications include reactor thermal hydraulics, internal combustion engines, ocean and atmospheric flows, vascular flows, astrophysical problems, and basic turbulence questions for theory and model development. Simulations in these areas present significant challenges with respect to scale resolution, multiphysics, and complex computational domains. In many cases, experimental data are expensive or impossible to obtain, making simulation on leadership computing platforms critical to informed analysis.

With current exascale computing programs in the U.S. and elsewhere developing GPU-based HPC platforms it is imperative to exploit the performance potential of these powerful node architectures. In this paper, we describe the development of a new open-source code for thermal-fluid analysis, NekRS [1], which has emerged out of two HPC software projects and which is designed to be performant for both GPU- and CPU-based platforms.

Nek5000 [2] was one of the first production-level single-program multiple-data (SPMD) codes deployed on distributed-memory parallel computers [3]. It has demonstrated scalability to leading-edge platforms through the SPMD era [4,5] and readily scales to millions of

* Corresponding author.
E-mail address: mmin@mcs.anl.gov (M. Min).

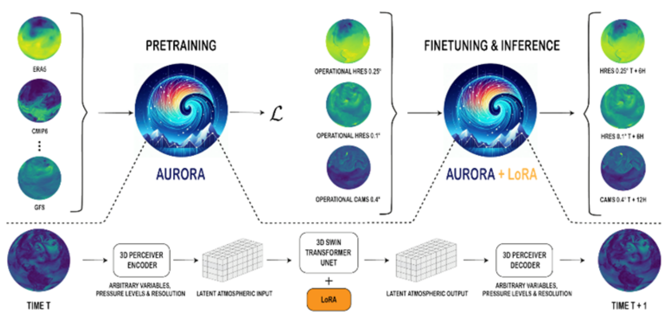
https://doi.org/10.1016/j.parco.2022.102982
Received 9 December 2020; Received in revised form 3 August 2022; Accepted 6 October 2022
Available online 18 October 2022

Andy Davis & Nitesh Bhatia
NekRS running on Frontier

GPU usage and scale

- Moving away from CPU simulations
  - => Many fusion codes still require CPU's
  - => Need to move towards GPU codes (NekRS, CGYRO, XGC, OpenMC)
  - => Strong collaboration with US-NL (Exascale Computing Project ECP)

- Transformers and Foundation models
  - => Most ML workflows are not GPU intensive
  - => Transformers are
  - => In visit at Linz (JKU) this week to learn from world experts
  - => Collaborations with JKU, Turing Institute, IBM, IAEA
  - => Strength of Transformers: they scale very well on GPUs
  - => Challenge of Transformers: data-hungry



arXiv:2405.13063v2 [physics.ao-ph] 28 May 2024

AURORA: A FOUNDATION MODEL OF THE ATMOSPHERE

Cristian Bodnar[*,1], Wessel P. Bruinsma[*,1], Ana Lucic[*,1], Megan Stanley[*,1], Johannes Brandstetter[3,†], Patrick Garvan[1], Maik Riechert[1], Jonathan Weyn[2], Haiyu Dong[2], Anna Vaughan[4], Jayesh K. Gupta[5,†], Kit Tambiratnam[2], Alex Archibald[4], Elizabeth Heider[1], Max Welling[6,†], Richard E. Turner[1,4], and Paris Perdikaris[1]

[1]Microsoft Research AI for Science
[2]Microsoft Corporation [3]JKU Linz [4]University of Cambridge [5]Poly Corporation [6]University of Amsterdam

[*]Equal contribution [†]Work done while at Microsoft Research

ABSTRACT

Deep learning foundation models are revolutionizing many facets of science by leveraging vast amounts of data to learn general-purpose representations that can be adapted to tackle diverse downstream tasks. Foundation models hold the promise to also transform our ability to model our planet and its subsystems by exploiting the vast expanse of Earth system data. Here we introduce Aurora, a large-scale foundation model of the atmosphere trained on over a million hours of diverse weather and climate data. Aurora leverages the strengths of the foundation modelling approach to produce operational forecasts for a wide variety of atmospheric prediction problems, including those with limited training data, heterogeneous variables, and extreme events. In under a minute, Aurora produces 5-day global air pollution predictions and 10-day high-resolution weather forecasts that outperform state-of-the-art classical simulation tools and the best specialized deep learning models. Taken together, these results indicate that foundation models can transform environmental forecasting.

1 Introduction

Deep learning foundation models have revolutionised various scientific domains, such as protein structure prediction (Abramson et al., 2024), drug discovery (Chithrananda et al., 2020), computer vision (Betker et al., 2023), and natural language processing (OpenAI, 2024). The key tenets of foundation models include *pretraining*, where a single large-scale neural network learns to capture intricate patterns and structure from a large corpus of diverse data; and *fine-tuning*, which allows the model to leverage its learned representations to excel at new tasks with limited training data (Bommasani et al., 2021; Brown et al., 2020).

The Earth system is a complex and interconnected network of subsystems, such as the atmosphere, oceans, land, and ice, which constantly interact in intricate ways. In a rapidly changing climate, accurate understanding of these subsystems becomes increasingly important. We envision that foundation models can revolutionise our ability to model subsystems of the Earth, and eventually the whole Earth.

Amongst the Earth's subsystems, the atmosphere stands out as particularly data-rich (Reichstein et al., 2019; Bauer et al., 2015) and therefore constitutes ripe ground for pretraining a foundation model. Classical atmospheric simulation approaches, such as numerical weather prediction (NWP), are costly and unable to exploit this wealth of data (Bauer et al., 2015). Recent deep learning approaches are cheaper, more flexible, and have shown great promise in specific prediction tasks with abundant training data (Lam et al., 2023; Bi et al., 2023; Chen et al., 2023a,b; Han et al., 2024; Kochkov et al., 2024; Lessig et al., 2023; Pathak et al., 2022; Bonev et al., 2023; Andrychowicz et al., 2023; Ham et al., 2019; Nguyen et al., 2023a,b). However, these methods struggle when atmospheric training data are scarce (Chantry et al., 2021) or heterogeneous (Reichstein et al., 2019), and they lack robustness in predicting extremes (Charlton-Perez et al., 2024). By learning generalizable representations from vast amounts of diverse data, foundation models have been able to overcome analogous challenges in other domains (Zhai et al., 2022; Radford et al., 2021; Bommasani et al., 2021; Nguyen et al., 2023a).

GPU usage and scale

- Moving away from CPU simulations
  => Many fusion codes still require CPU's
  => Need to move towards GPU codes (NekRS, CGYRO, XGC, OpenMC)
  => Strong collaboration with US-NL (Exascale Computing Project ECP)

- Transformers and Foundation models
  => Most ML workflows are not GPU intensive
  => Transformers are
  => In visit at Linz (JKU) this week to learn from world experts
  => Collaborations with JKU, Turing Institute, IBM, IAEA
  => Strength of Transformers: they scale very well on GPUs
  => Challenge of Transformers: data-hungry

- Several GPU clusters => portability is key
  - Leonardo (CINECA) Nvidia
  - CSD3 (Cambridge) Nvidia
  - Dawn (Cambridge) Intel
  - Isembard-AI (Bristol) Nvidia
  - LUMI (Finland) AMD
  - Frontier (USA) AMD

Conclusion

- AI & Machine Learning at UKAEA has ramped up over last 3 years
- Several projects running in collaboration with internal/external partners
- Integration into larger framework (eg. Neural-Parareal, JINTRAC)
- Currently on visit at Linz with Johannes Brandstetter
  => Starting Transformers and Foundation Models
  => will increase GPU usage a lot in near future

Thank you for your attention!