Plasma Science & Fusion Center

A Cross Machine and Parametric Neural Operator Surrogate Model for MHD Simulations

Qiyun Cheng[1], Cesar Clauser[1], E.d.D. Zapata-Cornejo[1], Nathaniel Ferraro[2], Cristina Rea[1]

1. Plasma Science & Fusion Center, MIT
2. Princeton Plasma Physics Laboratory, Princeton

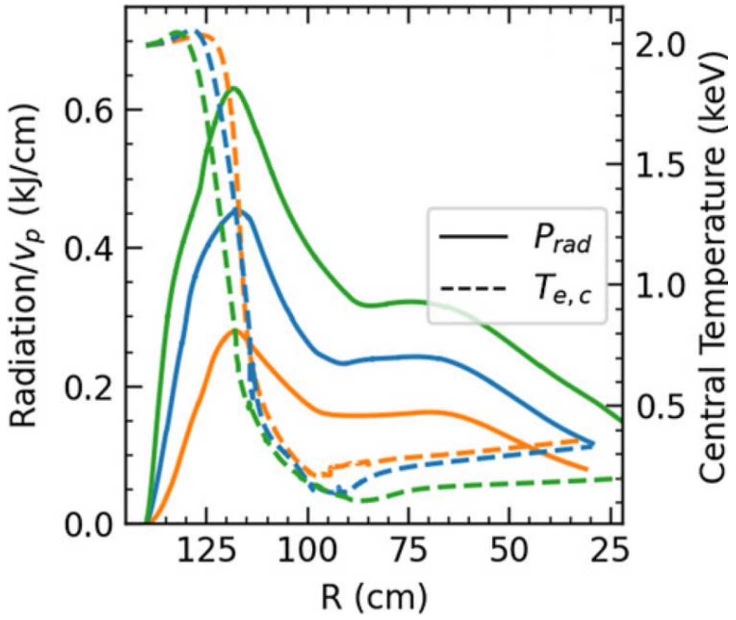
12/12/2025

Motivation & Research Background

- Future fusion device (e.g., ARC™) designs require large-scale parameter optimizations (e.g., disruption mitigation optimization)

JET runaway electrons damage.
https://www.iter.org/newsline/-/2234

Unmitigated disruptions in ITER is expected to become of the order 1 in 10^4 discharges [1].



Different **pellet injection velocities** (three colors) lead to varying **radiated power (P_{rad})** and the **plasma electron central temperature ($T_{e,c}$)** [2].

[1] E.J. Strait et al., *Nucl. Fusion* **59** 112012 (2019)
[2] C.F. Clauser et. al., *Nucl. Fusion*, **61**, 116003 (2021)

Motivation & Research Background

- High-fidelity nonlinear MHD simulations (e.g., M3D-C1) are essential but computationally expensive
  - E.g., one 3D M3D-C1 simulation with 8 toroidal planes (Data provided by Rishabh Datta)
  - One time step ~ 53s, 20~30k time steps for several milliseconds
  - On Perlmutter with 8 nodes for ~ 2 weeks
- Neural network-based surrogate models offer a potential approach to accelerate this optimization

Outlines

- Introduction of Operator Learning
- Generalization Methods of Neural Operator Surrogates
  - Parameter generalization
  - Geometry generalization
  - Demonstration
- M3D-C1 Solver Acceleration

Neural Operator & Operator Learning

Neural Operator: Neural networks for PDE surrogate modeling

PDE in operator form: $\mathcal{O}(\boldsymbol{p})[u(\boldsymbol{x})] = S(\boldsymbol{x})$

Solution of the field function:

$$u(\boldsymbol{x}) = \underline{\mathcal{O}^{-1}(\boldsymbol{p})}[S(\boldsymbol{x})]$$

Solution Operator

Computational expensive matrix inverse

Steady state electron temperature equation:

$$n_e(T_e \nabla \cdot \boldsymbol{v} + \boldsymbol{v} \cdot \nabla T_e) + \sigma_e T_e$$
$$= (\gamma - 1)(\nabla \cdot (\kappa_\perp \nabla T_e) + \nabla \cdot (\kappa_\parallel \boldsymbol{bb} \cdot \nabla T_e)$$
$$+ \eta J^2 + Q_{rad} + Q_{...})$$

Operator form:

$$\mathcal{O}(v, \sigma_e, \kappa_\perp, \kappa_\parallel)[T_e(r, z)] = S(r, z)$$

Neural operator (N.O.) as a surrogate of the solution operator

$$u_{N.O.}(\boldsymbol{x}) = \mathcal{O}_{N.O.}^{-1}(\boldsymbol{p})[S(\boldsymbol{x})] \approx \mathcal{O}^{-1}(\boldsymbol{p})[S(\boldsymbol{x})]$$

Take any source and give predicted field function

Existing Issues of Neural Operator Surrogates

$$u_{N.O.}(x) = \mathcal{O}_{N.O.}^{-1}(\boldsymbol{p})[S(\boldsymbol{x})]$$

(1) Data acquisition

Comprehensive parameter scans are infeasible

(2) Cross-machine generalization

Evolving device designs make it impossible to obtain sufficient data

(3) Accuracy degradation

Extrapolation performance remains low fidelity

(4) Limited interpretability

Safety-critical applications require transparent and explainable models

Parameter Generalization – Equation Recast

Train a **canonical neural operator** with $\boldsymbol{p}^*$

$$\mathcal{O}(\boldsymbol{p}^*)[u(\boldsymbol{x})] = S(\boldsymbol{x}) \quad \Longrightarrow \quad \mathcal{O}_{N.O.}^{-1}(\boldsymbol{p}^*)$$

$$T_e \nabla \cdot \boldsymbol{v}^* + \boldsymbol{v}^* \cdot \nabla T_e - \frac{(\gamma - 1)\kappa_\perp^*}{n_e^*} \nabla^2 T_e = S \qquad \underbrace{\left(\nabla \cdot \boldsymbol{v}^* + \boldsymbol{v}^* \cdot \nabla - \frac{(\gamma - 1)\kappa_\perp^*}{n_e^*} \nabla^2\right)}_{\mathcal{O}(\boldsymbol{p}^*)} T_e = S$$

Parameter Generalization – Equation Recast

Train a **canonical neural operator** with $\boldsymbol{p}^*$

$$\mathcal{O}(\boldsymbol{p}^*)[u(\boldsymbol{x})] = S(\boldsymbol{x}) \quad \Longrightarrow \quad \mathcal{O}_{N.O.}^{-1}(\boldsymbol{p}^*)$$

Given $\boldsymbol{p}'$, decompose the operator:

$$\mathcal{O}(\boldsymbol{p}')[u(\boldsymbol{x})] = \left(\underbrace{\mathcal{O}(\boldsymbol{p}') - \mathcal{O}(\boldsymbol{p}^*)} + \mathcal{O}(\boldsymbol{p}^*)\right)[u(\boldsymbol{x})] = S(\boldsymbol{x})$$

Residual operator: $\delta\mathcal{O}(\delta\boldsymbol{p})$

$$T_e \nabla \cdot \boldsymbol{v}^* + \boldsymbol{v}^* \cdot \nabla T_e - \frac{(\gamma - 1)\kappa_\perp^*}{n_e^*}\nabla^2 T_e = S \qquad T_e \nabla \cdot \boldsymbol{v}' + \boldsymbol{v}' \cdot \nabla T_e - \frac{(\gamma - 1)\kappa_\perp'}{n_e'}\nabla^2 T_e = S$$

$$\underbrace{T_e(\nabla \cdot \boldsymbol{v}' - \nabla \cdot \boldsymbol{v}^*) + (\boldsymbol{v}' \cdot \nabla T_e - \boldsymbol{v}^* \cdot \nabla T_e) - \left(\frac{(\gamma - 1)\kappa_\perp'}{n_e'}\nabla^2 T_e - \frac{(\gamma - 1)\kappa_\perp^*}{n_e^*}\nabla^2 T_e\right)}_{\delta\mathcal{O}(\delta\boldsymbol{p})[u(\boldsymbol{x})]}$$

Parameter Generalization – Equation Recast

Train a **canonical neural operator** with $\boldsymbol{p}^*$

$$\mathcal{O}(\boldsymbol{p}^*)[u(\boldsymbol{x})] = S(\boldsymbol{x}) \quad \Longrightarrow \quad \mathcal{O}_{N.O.}^{-1}(\boldsymbol{p}^*)$$

Given $\boldsymbol{p}'$, decompose the operator:

$$\mathcal{O}(\boldsymbol{p}')[u(\boldsymbol{x})] = \big(\underline{\mathcal{O}(\boldsymbol{p}') - \mathcal{O}(\boldsymbol{p}^*)} + \mathcal{O}(\boldsymbol{p}^*)\big)[u(\boldsymbol{x})] = S(\boldsymbol{x})$$

Residual operator: $\delta\mathcal{O}(\delta\boldsymbol{p})$

Equation Recast: $\mathcal{O}(\boldsymbol{p}^*)\,[u(\boldsymbol{x})] = S(\boldsymbol{x}) - \delta\mathcal{O}(\delta\boldsymbol{p})[u(\boldsymbol{x})]$

$$\Longrightarrow \quad u(\boldsymbol{x};\boldsymbol{p}') = \mathcal{O}_{N.O.}^{-1}(\boldsymbol{p}^*)\big[\underline{S(\boldsymbol{x}) - \delta\mathcal{O}(\delta\boldsymbol{p})[u(\boldsymbol{x})}]\big]$$

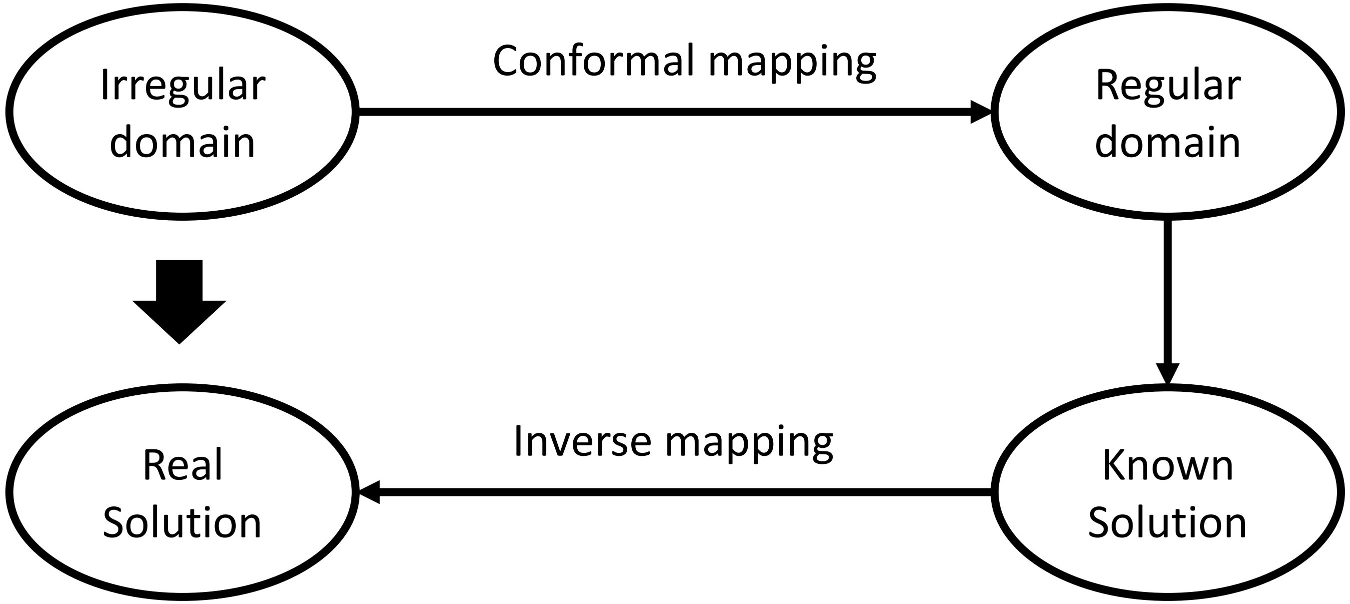
Absorb parameter variance into source term

Form an iterative scheme using canonical N.O.

The canonical N.O. is generalized to all parameters with preserved physics

Cross Machine Generalization – Domain Unification

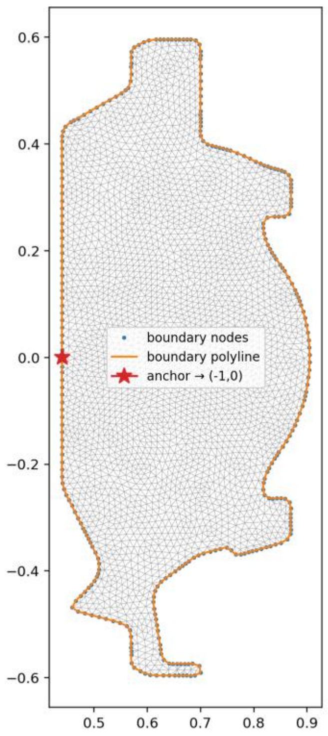
Solving a PDE analytically:



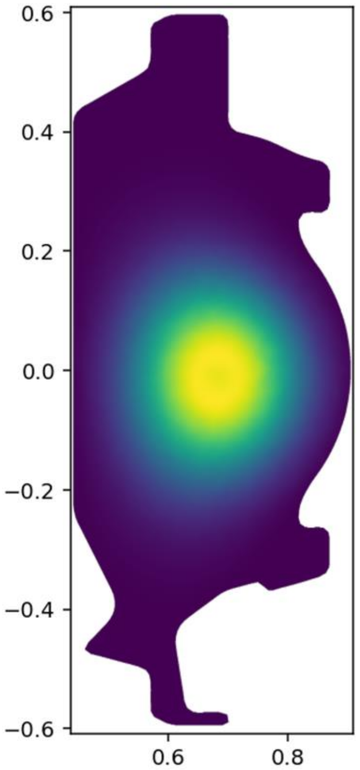
Cross Machine Generalization – Domain Unification

Cross machine geometry unification via **Harmonic Mapping**:

C-Mod 2D Mesh (Cross Section)

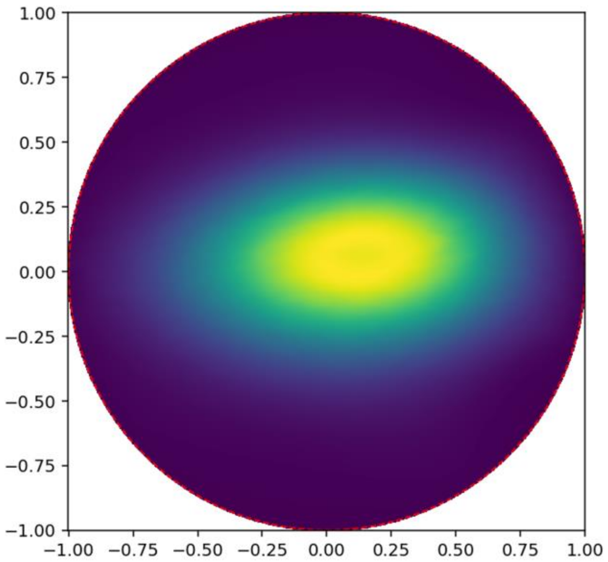


Mesh



Physics Space

Im(z)



Computational Domain

Re(z)

Boundary:

$$f\Big|_{\partial\Omega} = e^{i\theta}$$

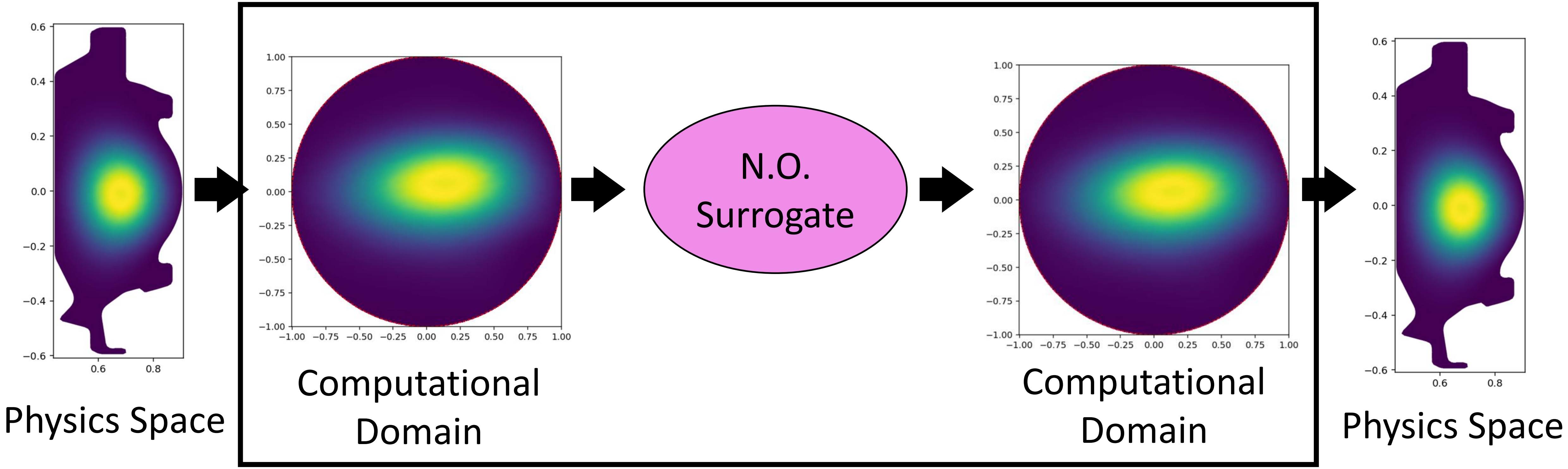
Harmonic functions:

$$\Delta u = \Delta v = 0$$

Harmonic Mapping is uniquely determined by the geometry

Cross Machine Generalization – Domain Unification

Cross machine geometry unification via **Harmonic Mapping**:

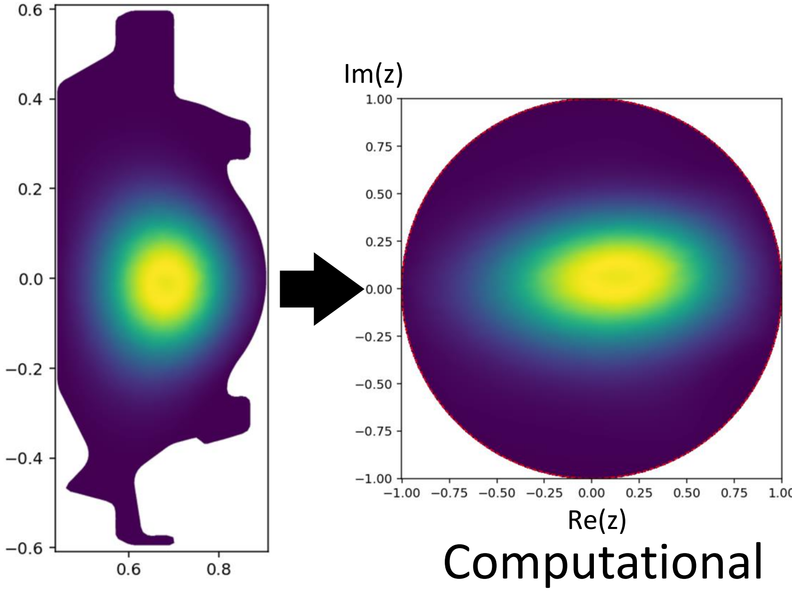


Separate the geometry analytically, achieving **geometry-agnostic** training

PDE is defined in the computational domain, same PDE for all machines (+ Jacobians)

Cross Machine Generalization – Domain Unification

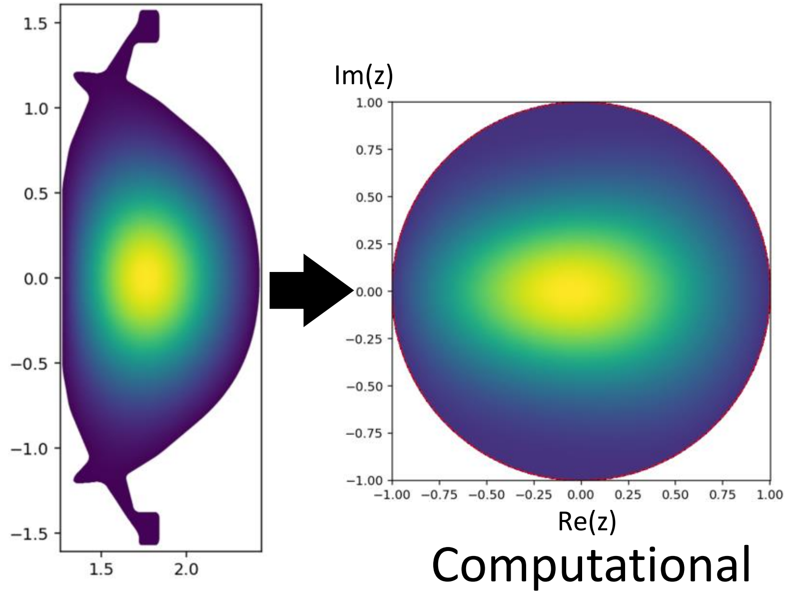
C-Mod



Physics Space

Computational Domain

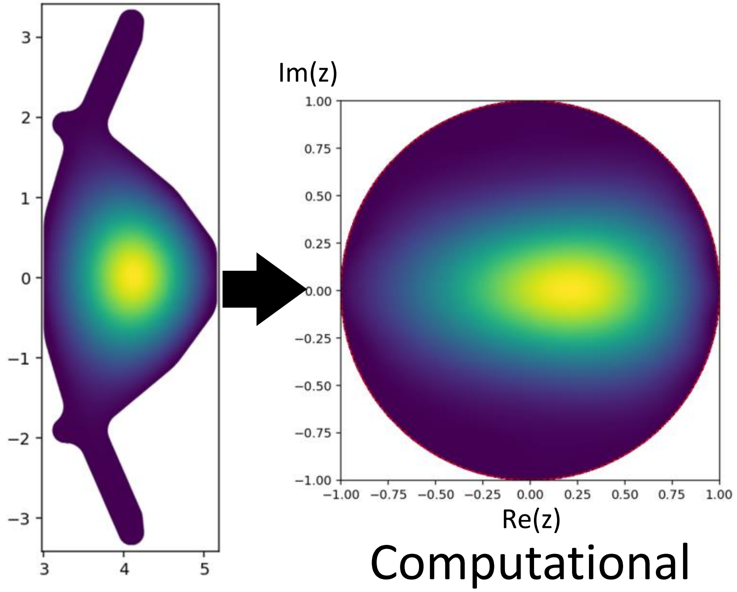
SPARC



Physics Space

Computational Domain

ARC_V2A



Physics Space

Computational Domain

Mappings and inverse mapping are **precalculated**

Different Jacobians can be **precalculated** and **absorbed** using Equation Recast

Demo – Fourier Neural Operator (FNO) based Surrogate

Simplified thermal quenching problem (single energy equation)

$$n_e\left(\frac{\partial T_e}{\partial t}+\cancel{T_e\nabla\cdot\boldsymbol{v}}+\cancel{\boldsymbol{v}\cdot\nabla T_e}\right)\cancel{+\sigma_e T_e}$$

$$=(\gamma-1)\left(\nabla\cdot(\kappa_\perp\nabla T_e)+\cancel{\nabla\cdot(\kappa_\parallel \boldsymbol{bb}\cdot\nabla T_e)}+\eta J^2\cancel{+Q_{rad}}+Q_{ext}\right)$$

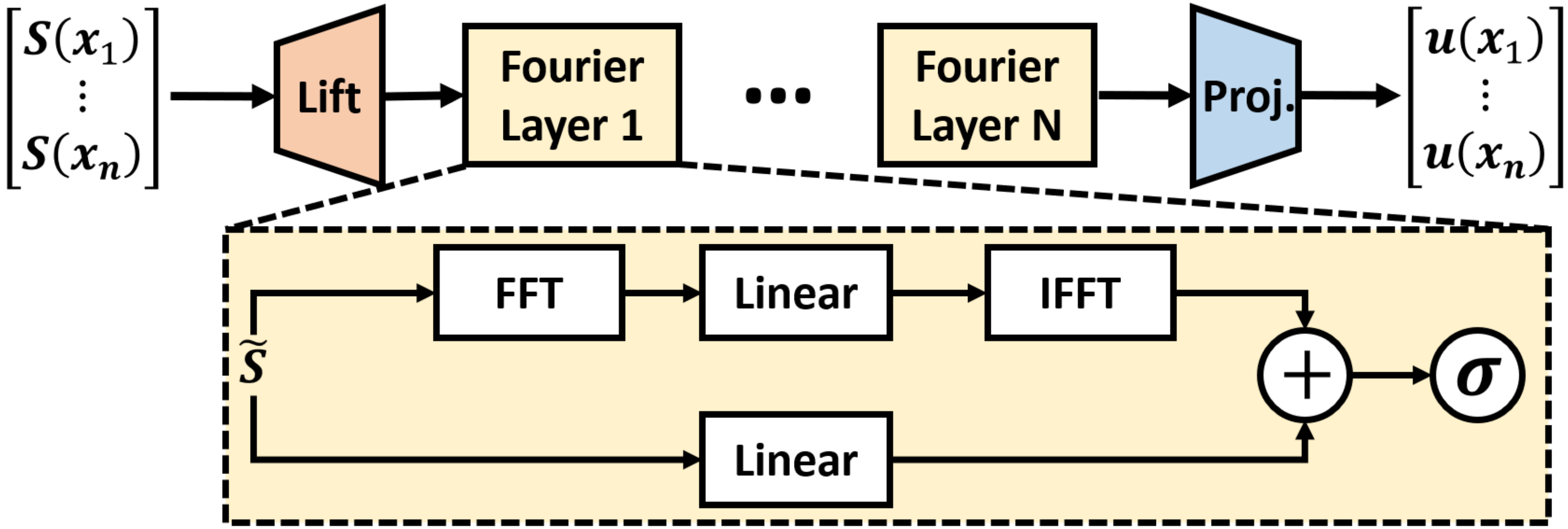
$$\Rightarrow\quad T_e^{i+1}=\underbrace{\left(\frac{1}{\delta t}-\frac{(\gamma-1)}{n_e}\kappa_\perp^*\nabla^2\right)}_{\text{Neural Operator}}{}^{-1}\left[\frac{\gamma-1}{n_e}(\eta J^2+Q_{ext})+\underbrace{\frac{1}{\delta t}T_e^i}_{\text{I. C.}}+\underbrace{\frac{\gamma-1}{n_e}(\kappa_\perp^{test}-\kappa_\perp^*)\nabla^2T_e^{i+1}}_{\text{Additional Source}}\right]$$

Other equation evolutions are disabled in M3D-C1 simulations

Demo – Fourier Neural Operator (FNO) based Surrogate

$$S(r,z) = \frac{\gamma-1}{n_e}(\eta J^2 + Q_{ext}) + \frac{1}{\delta t}T_e^i + \frac{\gamma-1}{n_e}(\kappa_\perp^{test} - \kappa_\perp^*)\nabla^2 T_e^{i+1}$$

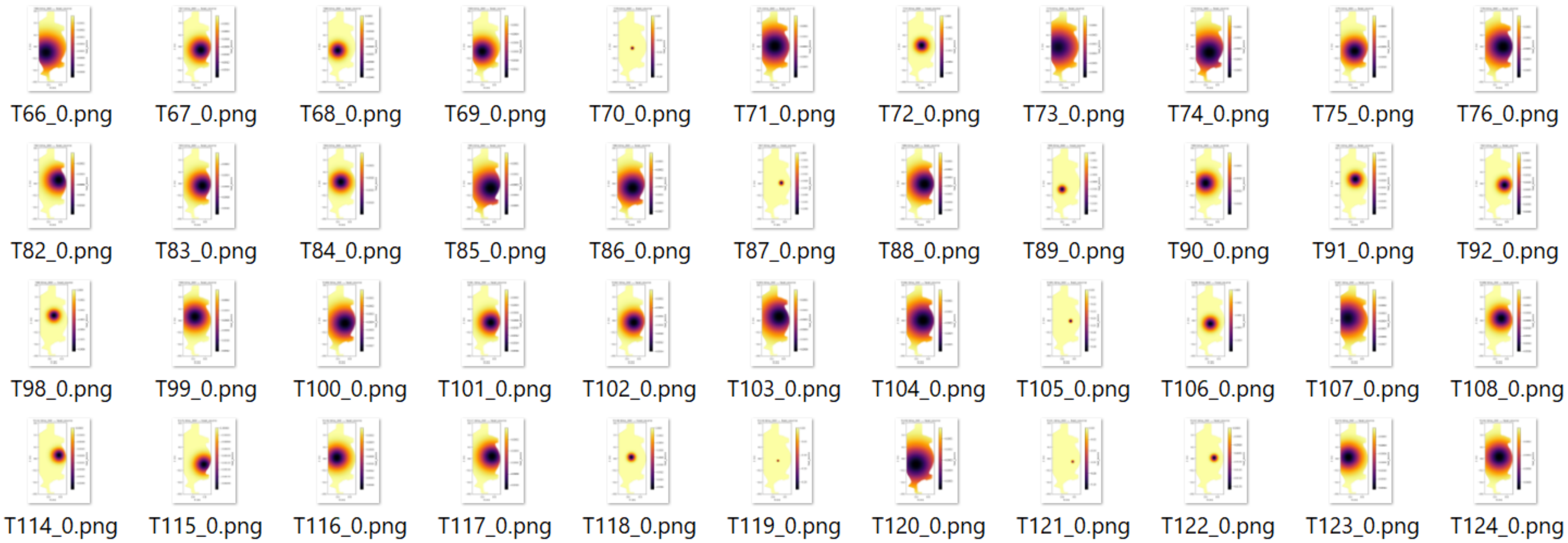
$$u(r,z) = T_e^{i+1}(r,z)$$



Fourier neural operator [1]

[1] Z. Li, et al., Fourier Neural Operator for Parametric Partial Differential Equations, International Conference on Learning Representations, 2021.

Demo – Fourier Neural Operator (FNO) based Surrogate



C-MOD Mesh
256 Gaussian sources
Random Var
Random Center

256 Sources × 5 time steps

$N_{train} = 256 \times 5 = 1280$

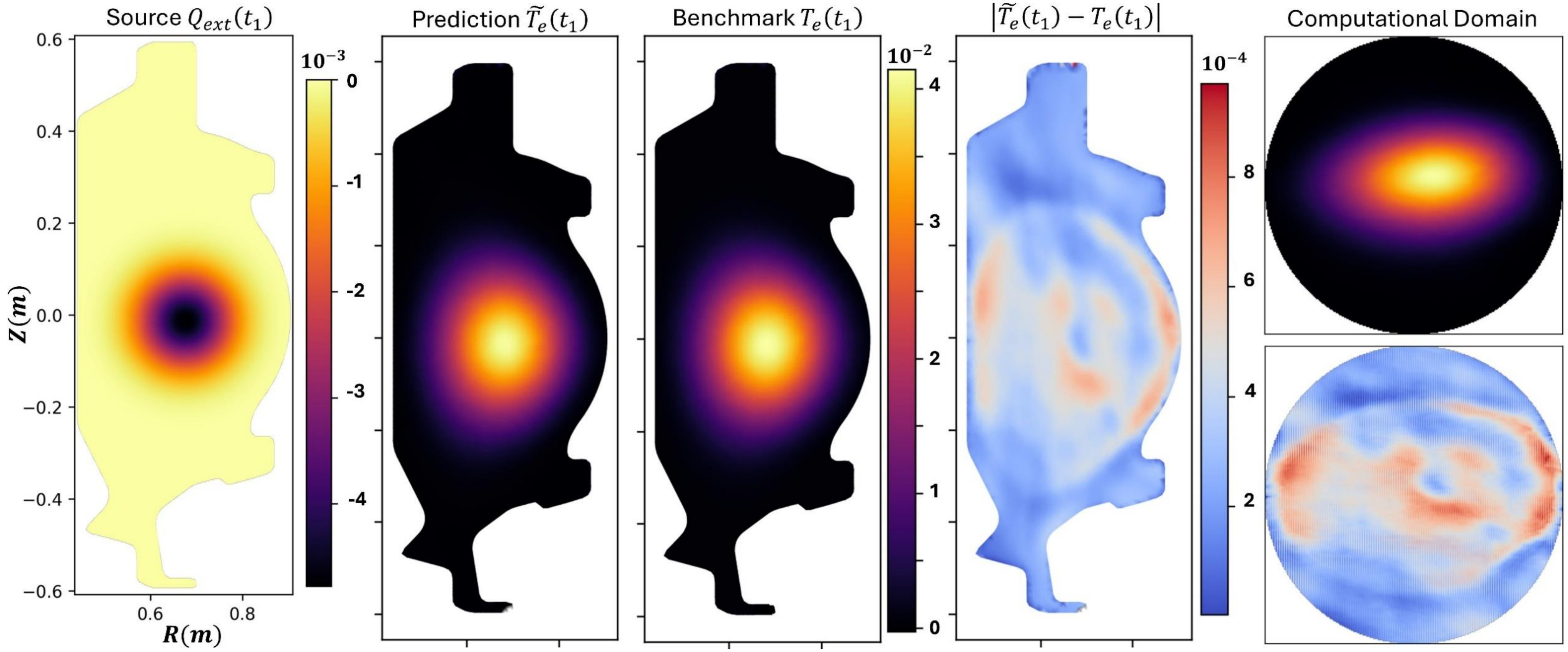
2 Nodes with 8 × A100 GPUs (MIT Engaging Cluster)

$T_{train} \sim 2.2\ hours$

$\boldsymbol{\kappa_{\perp}^{*} = 1 \times 10^{-3}}$

Demo – Parameter Generalization

$\kappa_{\perp}^{test} = 5 \times 10^{-4}$



Neural Operator Infer_time = 14ms

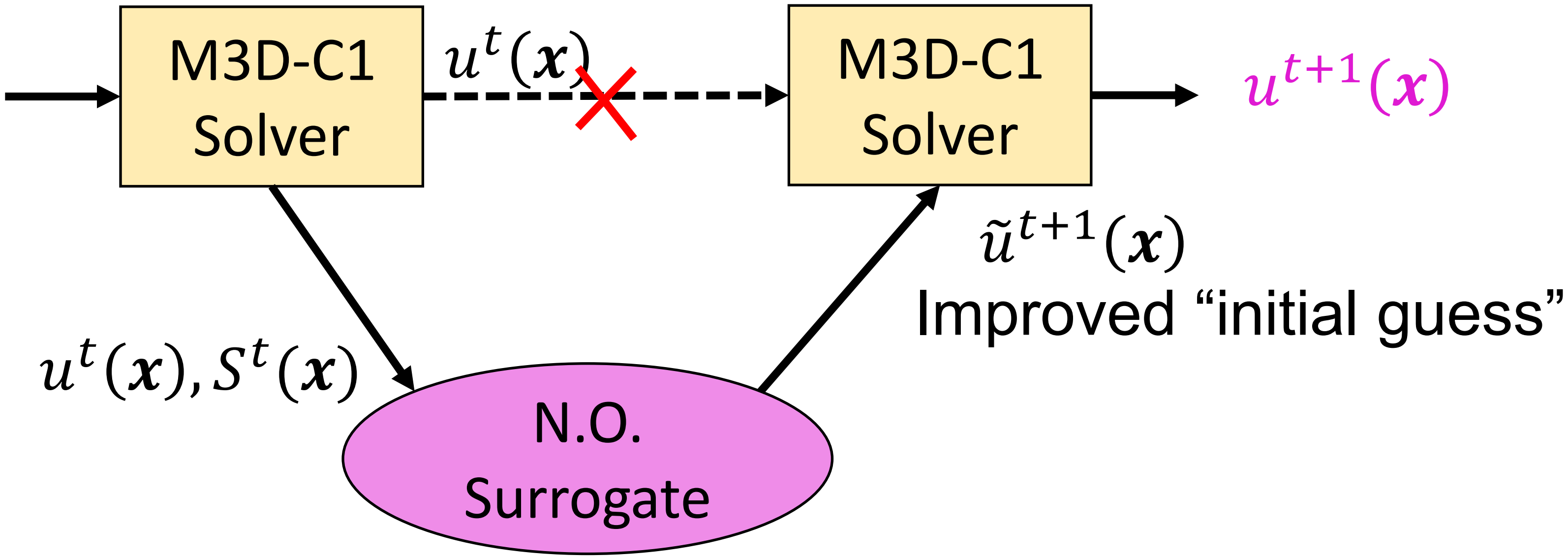
M3D-C1 Run_time = 2.4s

Fidelity Preservation – Preconditioner & Hybrid Solver

Hybrid solver - use neural operator surrogate as **Preconditioner**:

- Interpretable solutions
- Preserve full fidelity
- Reduce iterations
- Jump time steps

Accelerate M3D-C1 simulations



Only predict "next" time step

Conclusions and Discussions

- The neural operator surrogate successfully predicts temperature evolution across different parameters
- Its inference time is negligible compared to high-fidelity MHD solvers

On-going work:

- Extending to complete MHD equations and more complex plasma events
- Coupling the neural operator surrogate into the M3D-C1 solver

Potential of neural operator surrogates:

- Accelerate high-fidelity simulations
- Enable rapid prototyping for new system designs

chengq@psfc.mit.edu

MIT Disruption Group

This is work is supported by Commonwealth Fusion System.