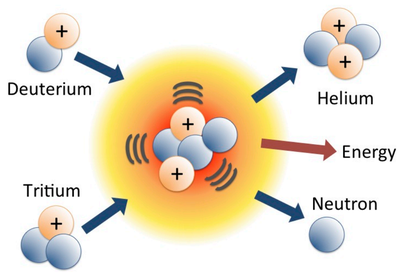
NEOPIC: HPC Strategies for FNO-DSE in Kinetic Plasma Simulation

Chelsea John, Sriramkrishnan Muralikrishnan, Jeffrey Kelling, Paul Gibbon, Andreas Herten

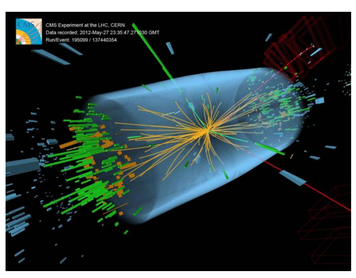
JÜLICH Forschungszentrum

NEOPIC

- Neural operator framework for particle-based kinetic plasma simulations
- Applications in nuclear fusion and particle accelerator modelling
- High-fidelity simulation costly with fine grid and many particles
- Grid solvers introduce numerical artifacts
- Neural operator surrogates are discretisation invariant and fast



Deuterium-Tritium Fusion
Ref: E.Schuster, Plasma Control Laboratory 2022



Proton-Proton collision at LHC
Ref: timeline.web.cern.ch, 2012

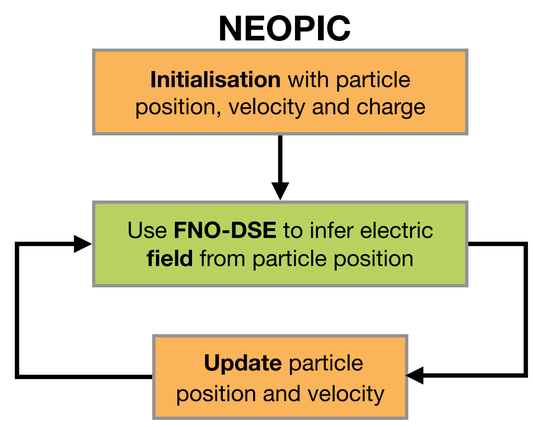
Algorithm

- **Neural Operator Particle-In-Cell (NEOPIC)**: Gridless particle method with neural operator for field evaluation
- **Non-uniform Fourier Neural Operator (FNO-DSE)**[1]: Handles non-uniform particle data and Fourier modes using **Vandermonde matrix**

Vlasov-Poisson System

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_x f + \frac{q}{m}(\mathbf{E} + \mathbf{v} \times \mathbf{b}) \cdot \nabla_{\mathbf{v}} f = 0$$

$$-\nabla^2 \phi = \frac{\rho}{\epsilon_0}, \mathbf{E} = -\nabla \phi$$



[1] Lingsch et al. *Beyond Regular Grids: Fourier-Based Neural Operators on Arbitrary Domains*. arXiv:2305.19663, 2024.

Memory Bottleneck in 3D Vandermonde Transform

Full 3D Vandermonde Matrix Transform

$$V \in \mathbb{C}^{B \times K_x \times K_y \times K_z \times N_p}, \textit{with } \mathcal{O}(K_x K_y K_z N_p) \textit{ memory}$$

$$\hat{u}_{b,c,k_x,k_y,k_z} = \sum_p u_{b,c,p} V_{b,p,k_x,k_y,k_z}$$

u: real-space input — $\hat{u}$: Fourier-space input
V: Vandermonde matrix — b: batch size
p: number of particles — c: number of channels
k_x, k_y, k_z: Fourier modes in X, Y, Z

Model Config

- Parameters: 268 Million TF32/FP32 (~1 GB)
- Fourier Layers: 4
- Channels: 32
- Fourier modes (Kx, Ky, Kz): 32
- Input/Output shape: [b=time step, 3, p]

Matrix-Free Factorised Contraction

$$V(k_x, k_y, k_z, p) = F^x_{k_x,p} F^y_{k_y,p} F^z_{k_z,p}, \textit{ where } F^i_{k_i,p} \in \mathbb{C}^{B \times K \times P}$$

$$\hat{u}_{b,c,k_x,k_y,k_z} = \sum_p u_{b,c,p} V(k_x, k_y, k_z, p)$$

```
Fx = torch.exp(-1j * kx[None, :, None] * x[:, None, :])  # (B, Kx, Np)
Fy = torch.exp(-1j * ky[None, :, None] * y[:, None, :])  # (B, Ky, Np)
Fz = torch.exp(-1j * kz[None, :, None] * z[:, None, :])  # (B, Kz, Np)

out = torch.einsum("bcp, bkp, blp, bmp -> bcklm", u, Fx, Fy, Fz)
# Output: [B, C, Kx, Ky, Kz]
```

Memory Comparison NVIDIA A100 40 GB with Input: [1, 3, 100000]

Component	Full 3D Vandermonde Matrix	Matrix-Free Factorised Contraction
Vandermonde (Complex FP32)	26.2 GB	73.2 MB
Activations/ Buffers / Cache	34.21 GB	5.5 GB
Model Weights,Parameters, Gradients (Adam, TF32/FP32)	4.0 GB	4.0 GB
Total	> 64.41 GB (OOM)	~9.6 GB

Parallel Schemes

- **Distributed Data Parallel (DDP)**: batch (time-step) parallelism with replicated model for speed
- **Tensor Parallelism (TP)**: splits particle positions across devices to reduce memory load
- **Spectral aggregation**: post Fourier transform, an all-reduce across TP ranks combines full Fourier spectrum from all particles

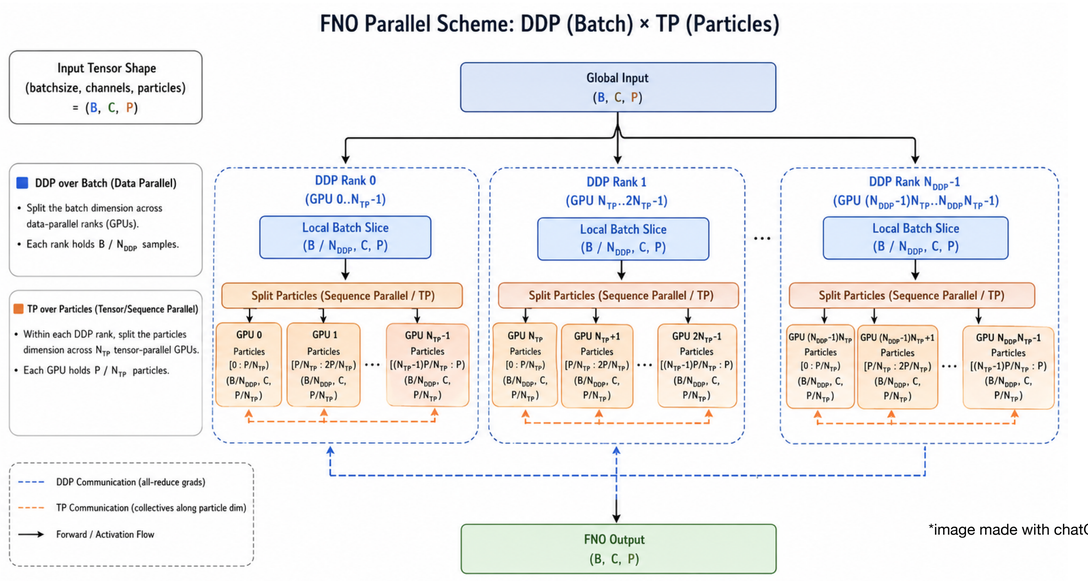
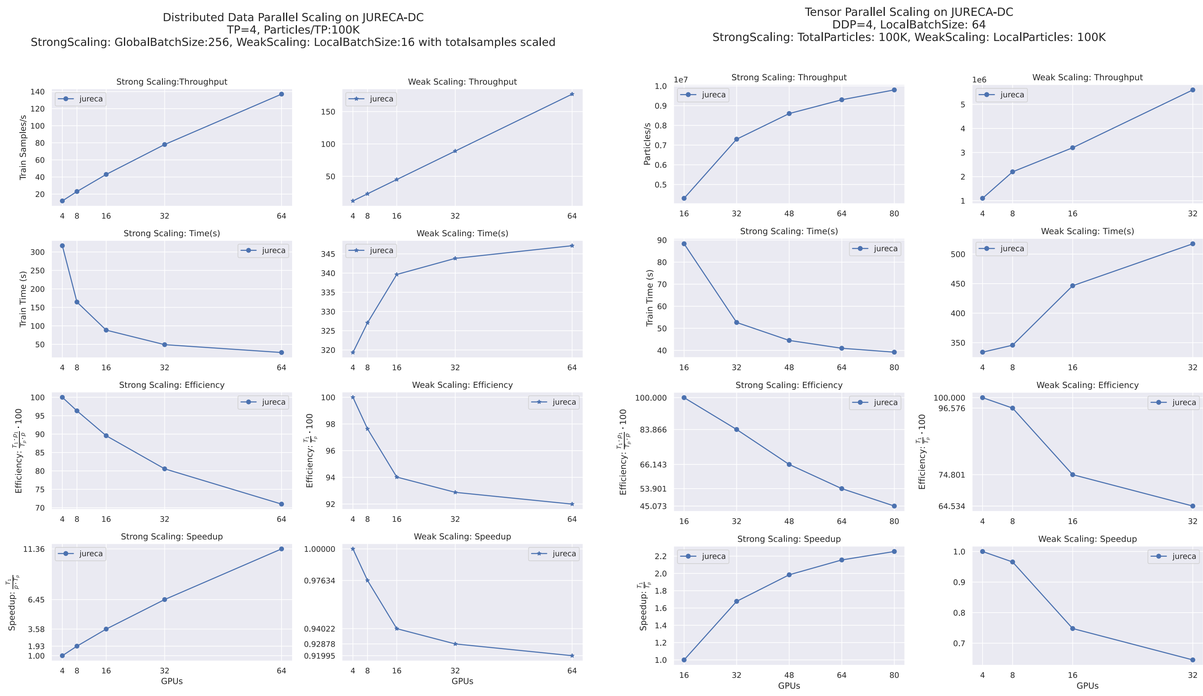


*image made with chatGPT

Scaling

- Strong & weak scaling for DDP and TP
- DDP: time step parallelism
- TP: particle parallelism
- NVIDIA A100 40 GB GPUs of JURECA-DC [2]

[2] https://www.fz-juelich.de/en/jsc/systems/supercomputers/jureca.



Conclusion

- Full Vandermonde matrix construction becomes **memory-bound** as particle count increases
- **Matrix-free factorised contraction** reduces Vandermonde transform memory by **>300×**
- **DDP** replicates the model, enabling **time step parallelism**
- **TP** shards particles, **reducing memory pressure** and accelerating training
- **DDP strong scaling**: **70% efficiency, 11× speedup** on **64 GPUs (**4 TP ranks)
- **DDP weak scaling: 92% efficiency** on **64 GPUs** (16 DDP × 4 TP)
- **TP strong scaling: 45% efficiency, 2.2× speedup** on **80 GPUs** (4 DDP); communication limited
- **TP weak scaling: 64% efficiency** on **32 GPUs** (4 DDP × 8 TP)

Future Work

- FP64 precision training (emulation or brute force)
- Implement activation checkpointing/re-computation
- Foundational model for particle field prediction

Acknowledgment

NEOPIC (INF funding code: ZT-I-PF-5-242) is funded by the Initiative and Networking fund of the Helmholtz Association (Helmholtz AI project call). We thank the Helmholtz x Mila Fellowship for supporting collaboration with Mila AI Institute, and JSC for access to JURECA-DC compute resources.

HELMHOLTZ AI FUNDED PROJECT

JÜLICH Forschungszentrum

HZDR HELMHOLTZ ZENTRUM DRESDEN ROSSENDORF