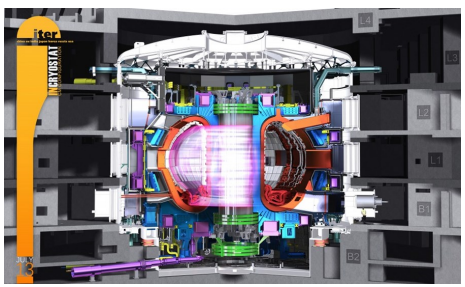
ALPINE: space charge computation in the exascale area

S. Muralikrishnan (PSI/Jülich), M. Frey (St. Andrew Univ) , A. Vinciguerra (ETH), M. Ligotino (ETH), A. Cerfon (NYU), M. Stoyanov (ORNL), R. Gayatri (LBL) and A. Adelmann (PSI)

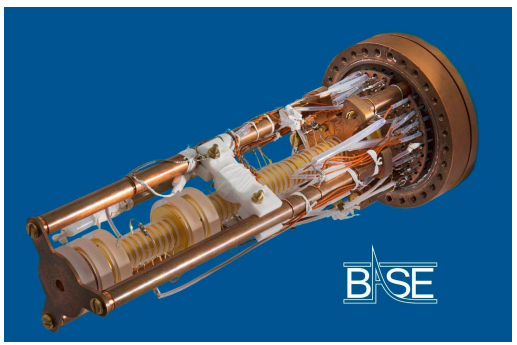
October 4. 2022

Scientific Motivation for ALPINE

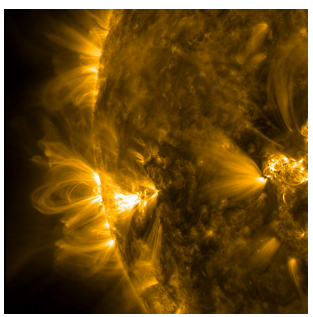
- Kinetic simulations of plasma play an important role in modelling nuclear fusion, particle accelerators, particle physics, astrophysical phenomena such as solar flares etc.



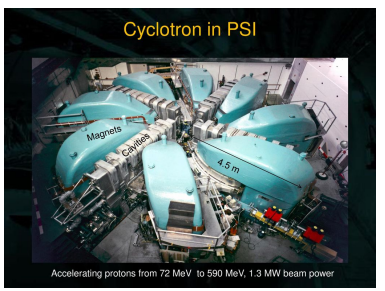
(a) Source: iter.org



(b) Source: CERN



(c) Source: NASA



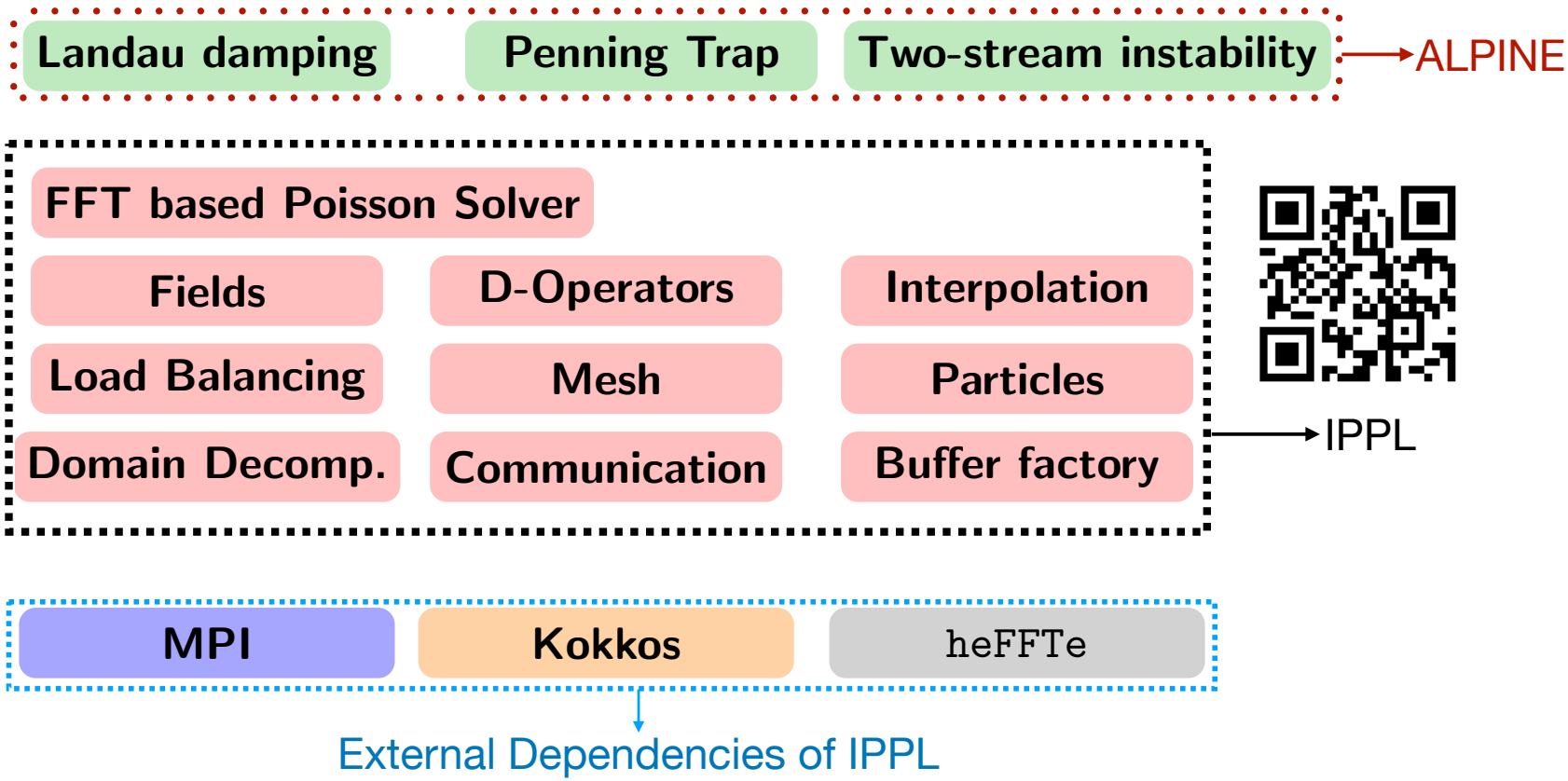
(d) Source: PSI

Towards exascale computing

- Performance Portable implementation of existing/novel numerical algorithms is critical for running these simulations on modern heterogeneous computing architectures

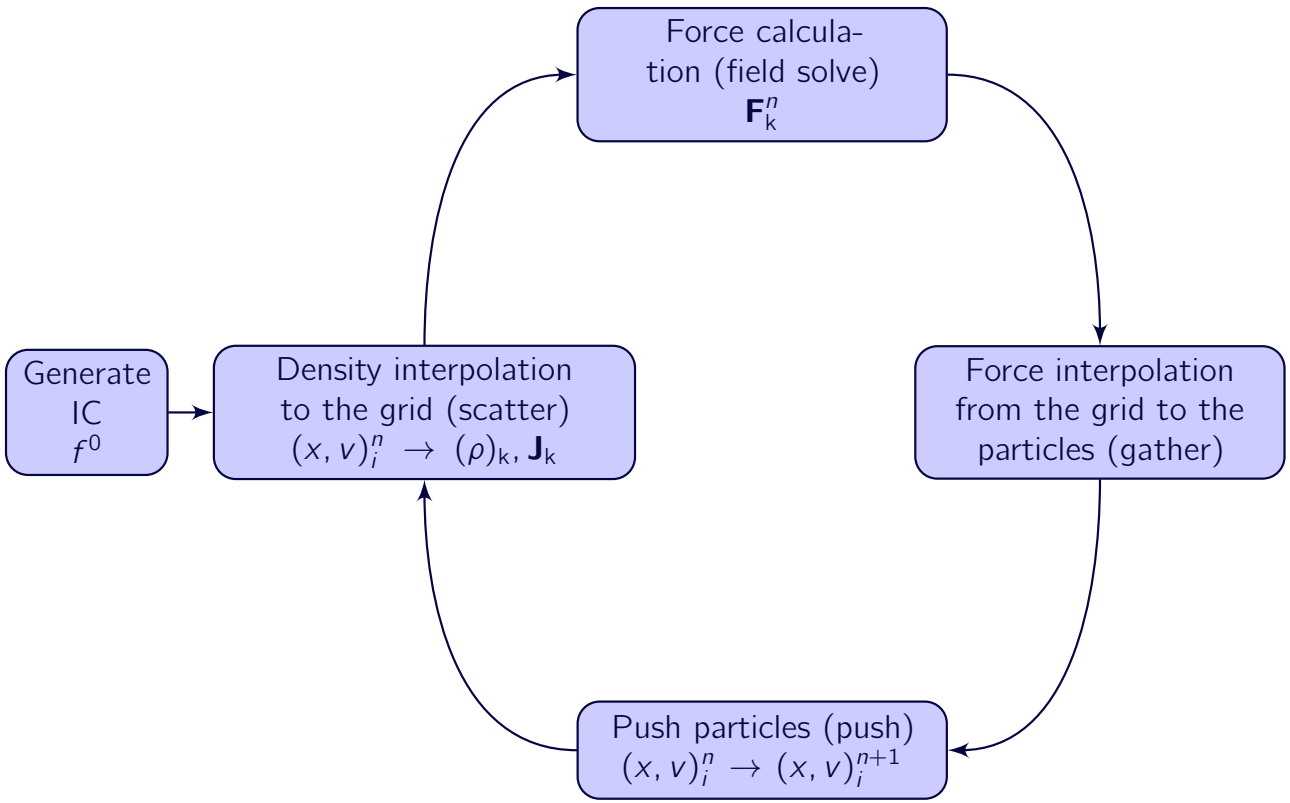
ALPINE (A set of performance portable pLasma physics Particle-in-cell mINi-apps for Exascale computing)

[Muralikrishnan et al., 2022]

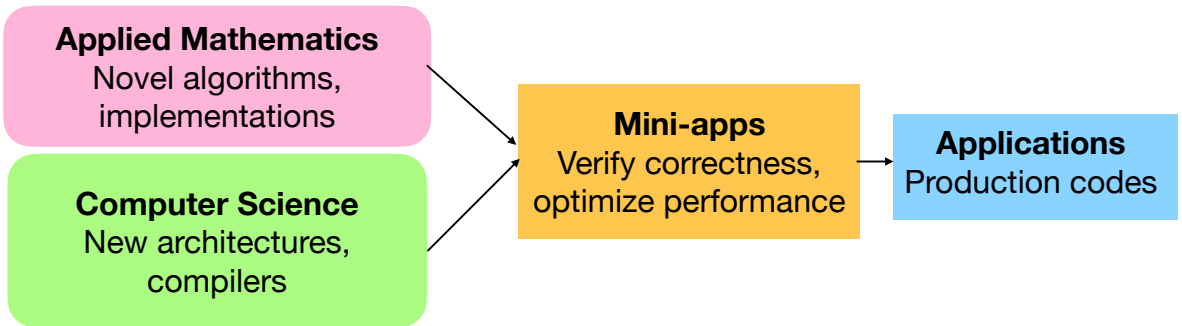


Simplified Particle-In-Cell (PIC)

With $f^0 \in \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^1$ we denote the initial phase space. The discrete force field is $\mathbf{F}_\mathrm{k}$ with $\mathbf{k} = \{i, j, k\}$. A particle i is denoted by $(\mathbf{x}, \mathbf{v})_i \in \mathbb{R}^3 \times \mathbb{R}^3$ and pushed from time step n to $n+1$:



Why Mini-apps?



- Lighter, easy to read code which serves as a proxy for real applications (OPAL Object Oriented Parallel Accelerator Library)
- Sandbox for implementing and testing novel algorithms [Muralikrishnan et al., 2021]
- Availability of reference results: optimization of algorithms, implementation while ensuring correctness
- Provide reliable performance expectations for the real applications [a]

[a] Heroux, M. A., et al. "Improving performance via mini-applications." Sandia National Laboratories, Tech. Rep. SAND2009-5574 3 (2009).

PAUL SCHERRER INSTITUT

Setup for Scaling Studies

Strong scaling

- Case A: 512^3 grid, $Np = 1,073,741,824$, 8 particles per cell
- Case B: 1024^3 grid, $Np = 8,589,934,592$, 8 particles per cell

PAUL SCHERRER INSTITUT

Setup for Scaling Studies

Strong scaling

- Case A: 512^3 grid, $Np = 1,073,741,824$, 8 particles per cell
- Case B: 1024^3 grid, $Np = 8,589,934,592$, 8 particles per cell

Weak scaling

- For GPUs: 256×128^2 grid and 8 particles per cell is the base case for 1 node/GPU. The max. grid size and particles are 2048^3 and $Np \approx 69$ billion at 2048 GPUs
- For CPUs: 512×256^2 grid and 8 particles per cell is the base case for 1 node. The max. grid size and particles are 4096×2048^2 and $Np \approx 138$ billion at 512 nodes = 16,384 cores

PAUL SCHERRER INSTITUT

Setup for Scaling Studies

Strong scaling

- Case A: 512^3 grid, $Np = 1,073,741,824$, 8 particles per cell
- Case B: 1024^3 grid, $Np = 8,589,934,592$, 8 particles per cell

Weak scaling

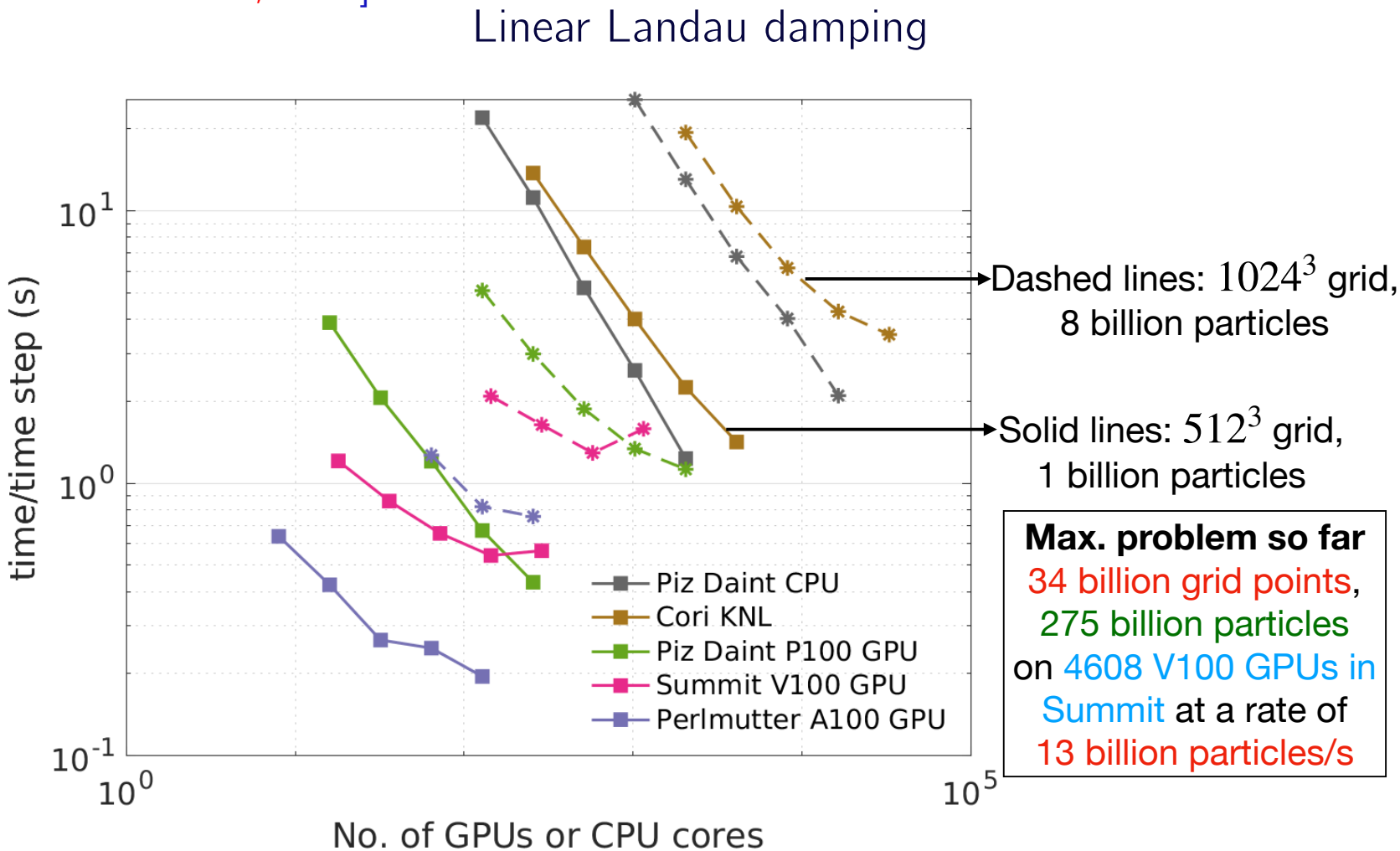
- For GPUs: 256×128^2 grid and 8 particles per cell is the base case for 1 node/GPU. The max. grid size and particles are 2048^3 and $Np \approx 69$ billion at 2048 GPUs
- For CPUs: 512×256^2 grid and 8 particles per cell is the base case for 1 node. The max. grid size and particles are 4096×2048^2 and $Np \approx 138$ billion at 512 nodes = 16,384 cores

Piz Daint

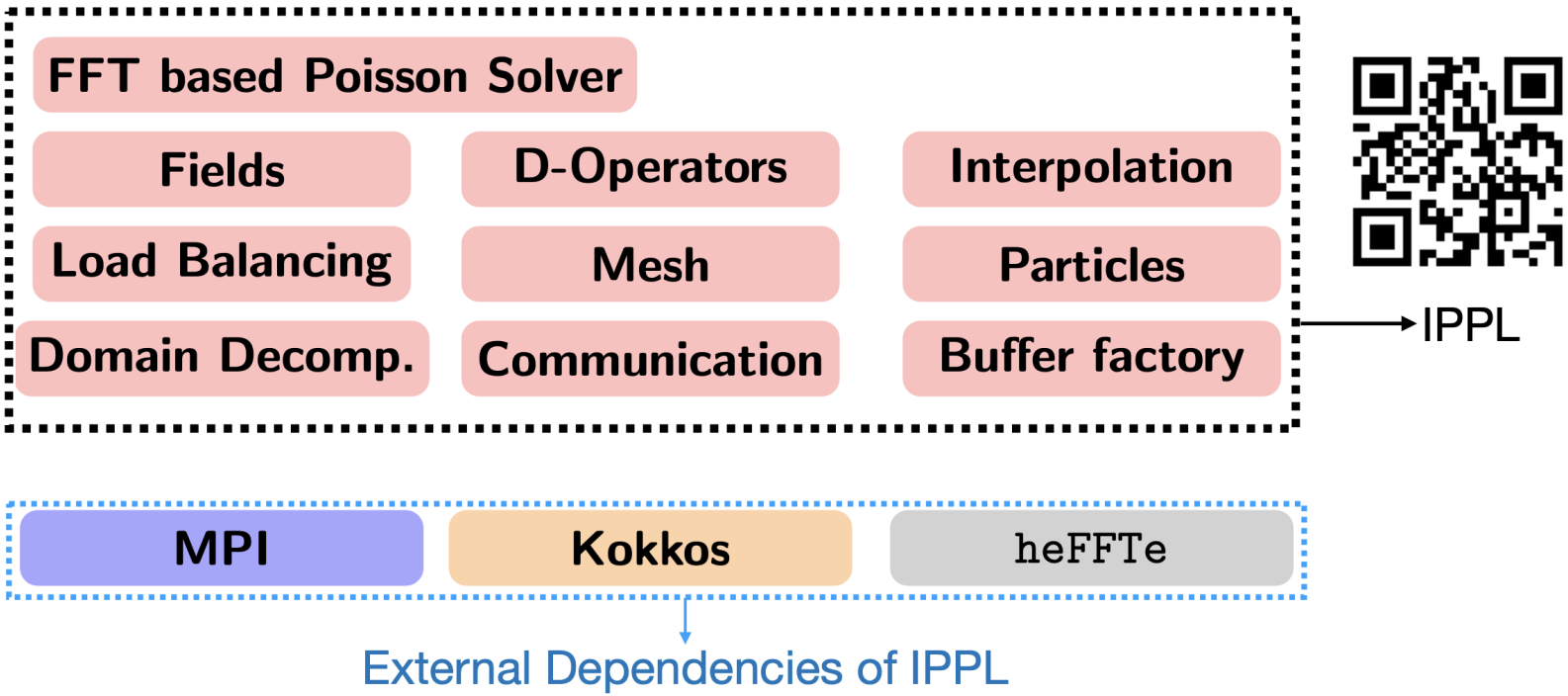
- GPUs: 1 P100 GPU/node. We use 1 MPI rank per GPU
- CPUs: 1 MPI rank and 32 OpenMP threads per node

Comparison across different architectures

[Muralikrishnan et al., 2022]



Independent Parallel Particle Library (IPPL V. 2.0)

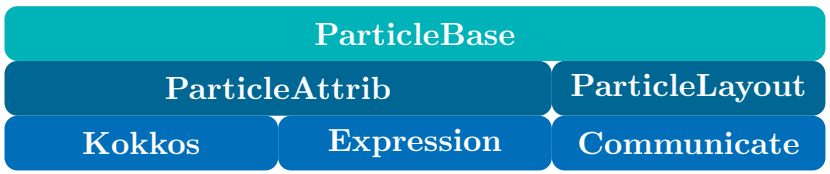


PAUL SCHERRER INSTITUT

Goals in the development of IPPL 2.0 (from 1.0 !)

- ✓ Enable **performance portability with Kokkos** (i.e. replace field and particle containers with Kokkos data structures)
- ✓ Upgrade to minimum `C++17` **standard**
- ✓ Keep **expression templates**
- ✓ **Keep** changes to the **user interface** to a minimum
- ✓ **Simplify code** (IPPL 1.0 $\approx$ 89k LoC vs IPPL 2.0 $\approx$ 20k LoC)
- dimension independence

Particles in IPPL 2.0



- **Attributes:**
  - Struct of Kokkos::Views
  - Expression templates
  - Easily added to application
- **Communication:**
  - Particle layout classes
  - De-/serialize Kokkos::View<**char***>
  - Pre-allocated buffers

```
using namespace ippl;

template<class PLayout>
struct Bunch
: public ParticleBase<PLayout> {
    Bunch(PLayout& playout)
    : ParticleBase<PLayout>(playout)
    {
        // add application attributes
        this->addAttribute(R);
        this->addAttribute(V);
        this->addAttribute(mass);
        this->addAttribute(charge);
    }

    ~Bunch() { }
    ParticleAttrib<double> mass, charge;
    ParticleAttrib<Vector<double>> R, V;
};
// compiles to single Kokkos kernel
bunch->R = bunch->R + dt * bunch->V;
```

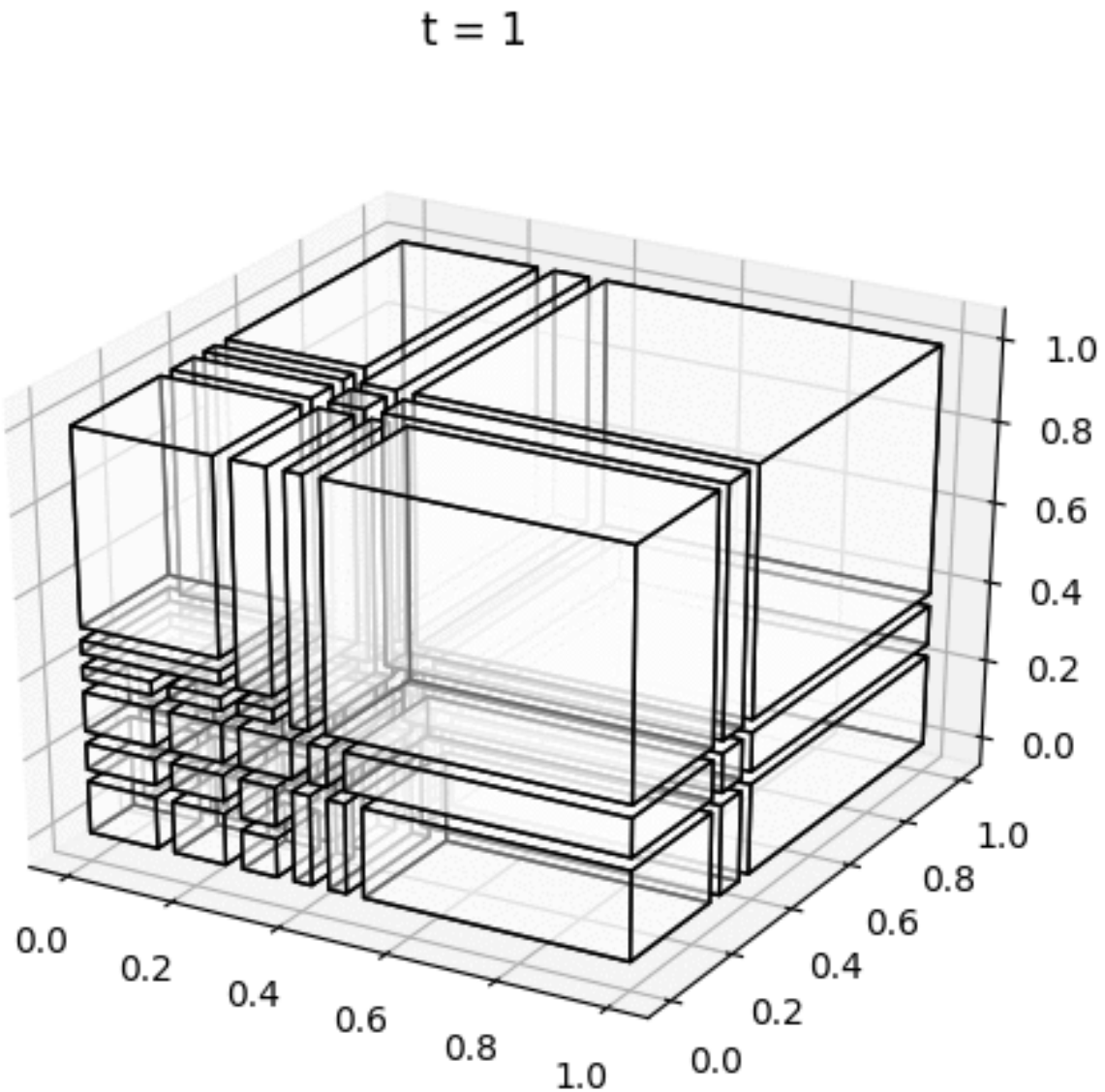
PAUL SCHERRER INSTITUT

PSI

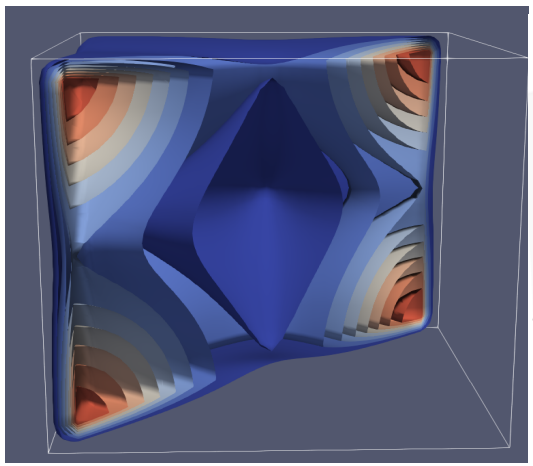
Example Movie

By preallocating the buffers used for field and particle communications and hence avoiding frequent cudaMalloc and cudaFree calls we are able to speedup the communication times in GPUs by $\mathcal{O}(10^3)$

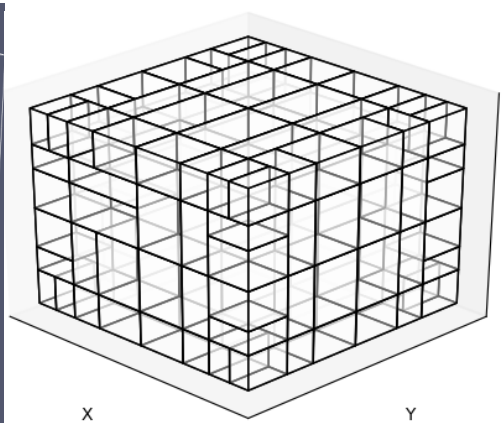
Domain Decomposition - works of Michael



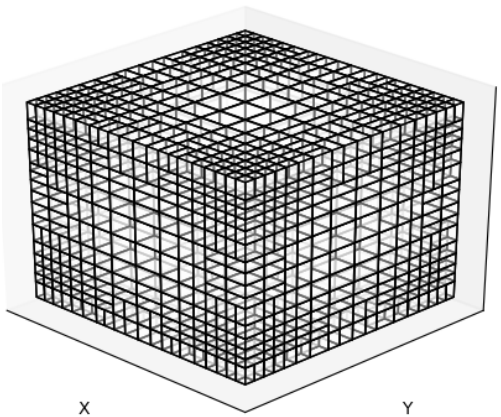
Nonlinear Landau Damping



(e) Initial density



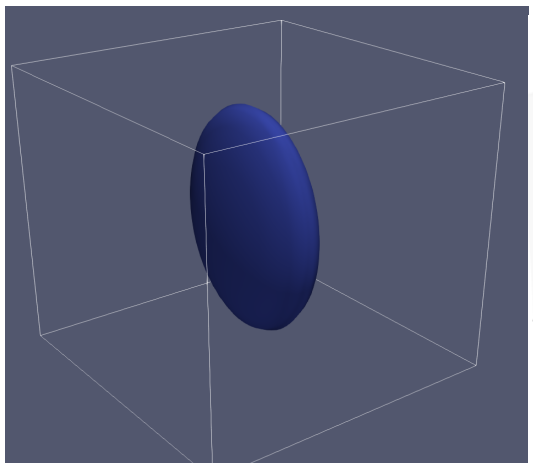
(f) 128 GPUs



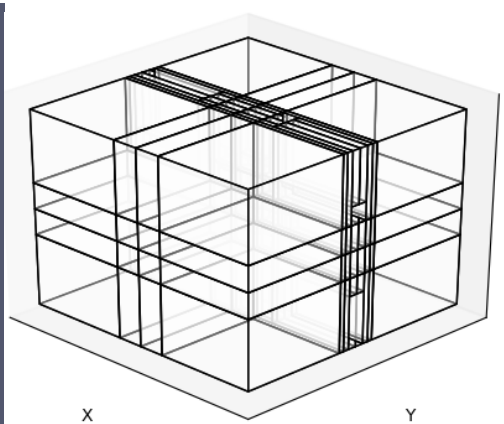
(g) 2048 GPUs

The non-uniform particle density requires particle load balancing to reduce memory requirements per GPU as well as communication costs

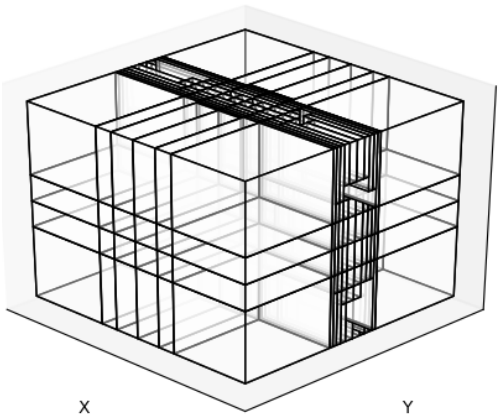
Penning Trap



(h) Initial density



(i) 256 GPUs



(j) 2048 GPUs

In this case simulations fail without particle load balancing due to highly clustered particle bunch. But the particle load balancing leads to large imbalance in the fields and hence impact Poisson solve

PAUL SCHERRER INSTITUT

Conclusions and Future work

Summary

- Presented portable, particle-in-cell based plasma physics mini-apps targeting exascale architectures
- Showed scaling up to thousands of CPU cores and GPUs on Piz Daint, Cori KNL, Summit and Perlmutter
- all is based on a performance portable version of IPPL

Where is the gap regarding space charge computations?

1. benchmarking efforts probing all aspects: physics, numerics and HPC capabilities
2. make efficient use of computing hardware (W/FLOPS)

What is needed to bridge this gap?

1. community benchmark effort (is hard)
2. we will base OPAL on IPPL and hence make it performance portable

Two roles to fill at my group in PSI

Hiring

Both for a two year post doc:

- one post doc (OPAL, IPPL, ALPINE, performance portable)
- one possibility for a Marie Skłodowska-Curie Fellowship on performance portable PDE solver

https://amas.web.psi.ch/people/aadelmann/pub/amas-post-docs.pdf

PAUL SCHERRER INSTITUT

References I

[Muralikrishnan et al., 2021] Muralikrishnan, S., Cerfon, A. J., Frey, M., Ricketson, L. F., and Adelmann, A. (2021).
Sparse grid-based adaptive noise reduction strategy for particle-in-cell schemes.
Journal of Computational Physics: X, 11:100094.

[Muralikrishnan et al., 2022] Muralikrishnan, S., Frey, M., Vinciguerra, A., Ligotino, M., Cerfon, A. J., Stoyanov, M., Gayatri, R., and Adelmann, A. (2022).
Alpine: A set of performance portable plasma physics particle-in-cell mini-apps for exascale computing, arxiv:2205.11052.

Acknowledgment

We would like to thank Marc Caubet Serrabou, Scientist HPCE group PSI for all the help with the installations, Peter Messmer from NVIDIA for fruitful discussions regarding GPU optimizations.
This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No. 701647 and from the United States National Science Foundation under Grant No. PHY-1820852.

We are an open source project and welcome any collaboration!