CTTS Meeting at the Annual APS DPP Conference
October 20, 2019, Ft. Lauderdale, FL

Progress with Numerical Aspects of Pellet / SPI Codes

Roman Samulyak, James Yuan, Nicholas Bosviel, Nizar Nailtho

Stony Brook University

Massively Parallel Lagrangian Particle Code

- The most time and memory consuming algorithms in the LP code:
  - Construction of octrees for particle distribution and searching these octrees for finding neighbors of each individual particle for PDE stencil operations
  - Quadtree and binary tree-based algorithms for kinetic models

- In the LP production code, the octree algorithm is parallelized using **OpenMP** and runs on a single multicore node, which limits the total number of particles in simulations.

- Because of adaptivity of the LP code, a typical multicore supercomputer node is sufficient for fully resolved simulation of a single pellet or a small number of pellet fragments

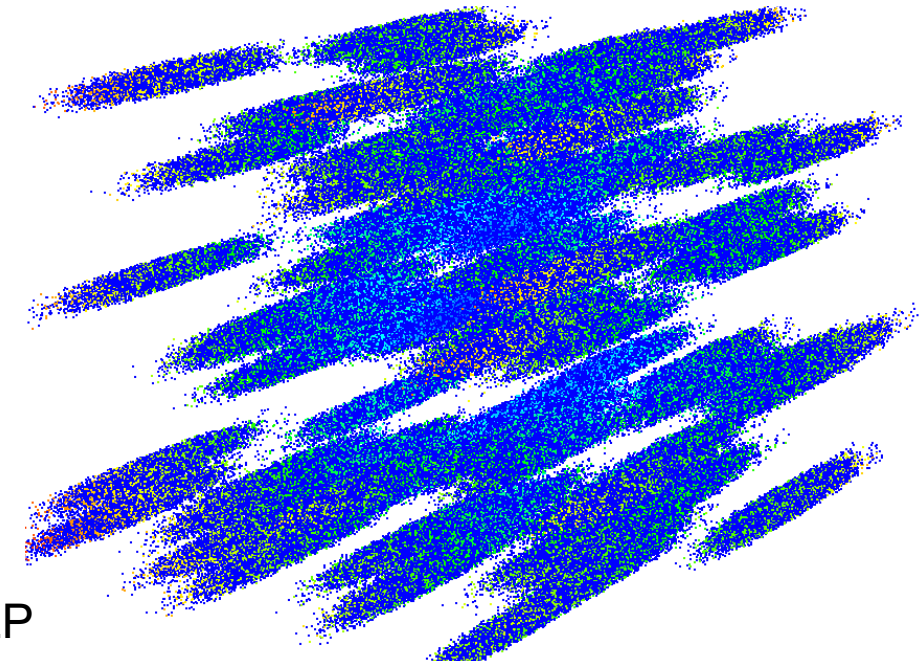
- In the SPI scenario, a large pellet is expected to shatter into hundreds of fragments. For example, it is estimated that a large pellet shatters into 250+ of fragments in DIII-D experiments. The corresponding simulations requires generating very large number of particles (10^7+), making simulations on a single multicore node very difficult or impossible (unless performed on the most advanced supercomputer architecture).

Massively Parallel Lagrangian Particle Code (cont)

- We have completed **MPI parallelization of the LP code** based on P4EST (Parallel Forest of K-trees)

- P4EST software library, developed in the past within an ASCR-funded project of Omar Ghattas
- P4EST library enables a dynamic management of a collection of adaptive octrees on distributed memory supercomputers
- P4EST scales to hundreds of thousands of processor cores.

- We have successfully implemented main particle data structures in the LP code compatible with P4EST and linked the codes. The new code has been tested over multiple MPI nodes with several millions of particles (it is capable of handling much larger numbers of particles).



Simulation of SPI using massively parallel LP

Massively Parallel Lagrangian Particle Code: Verification

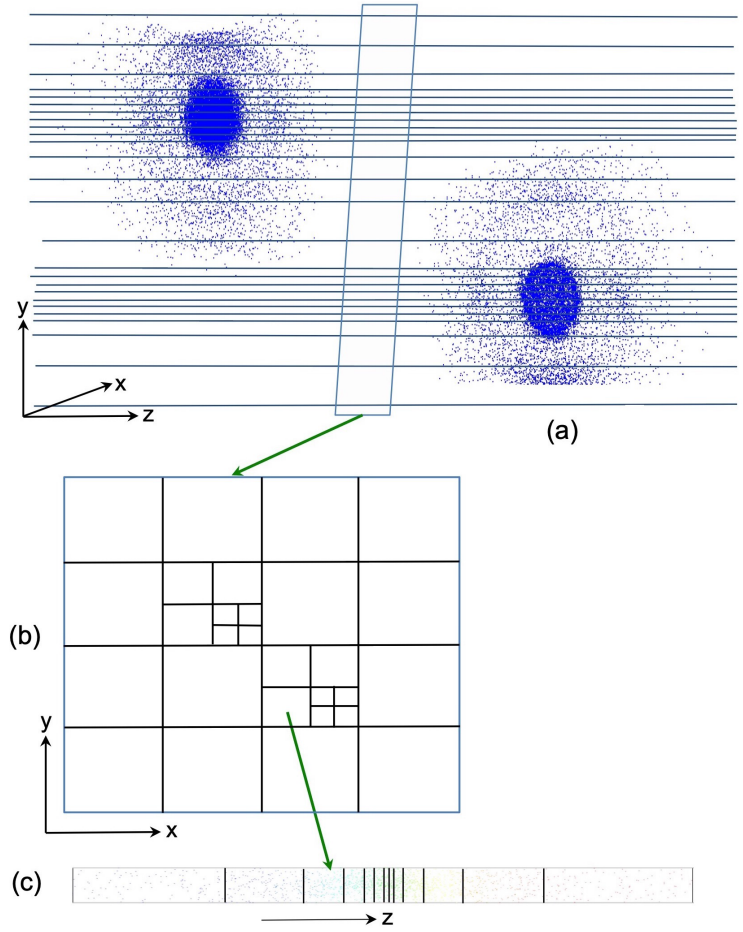
- The core LP code has been successfully verified using classical problems with analytic solutions (Gresho vortex, Yee vortex). It achieves 2nd order accuracy and identical results to the previous Open-MP parallelized code (at the same resolution)

- All physics modules of the pellet ablation code has been ported to the new code
- Pellet ablation simulation dive the same results as the Open-MP parallelized code

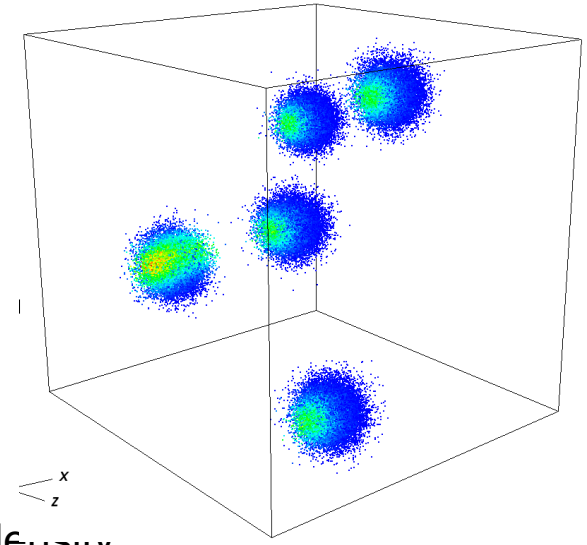
- "Practical scalability" test:
  - Open MP code: a well-resolved simulation of a single neon pellet ablation during 200 microseconds takes about 8 h on a single multicore node (24 cores)
  - **MPI code**: the same problem takes only **0.5 h using 40 MPI processes:** a significant improvement per computational core. The reason is on the next slide
  - High-resolution multi-fragment SPI simulation are expected to run on the order of 10 h using larger MPI runs

LP algorithms for integration of physics quantities along magnetic field lines

- Numerous LP algorithms (electron heating, grad B drift) require the integration of physics quantities along magnetic field lines
- How to compute line integrals for highly non-uniform and dynamic particle systems in the most accurate and efficient way?



a) 3D distribution of Lagrangian particles in SPI simulation. Horizontal lines schematically depict plasma density integral paths, adaptively refined near pellet fragments.



b) Quadtree data structure, built using Lagrangian particles projected to a transverse plane. Each quadtree cell contains one path for the plasma density integral.

c) Re-distribution of Lagrangian particles in each quadtree cell to 3D using their saved longitudinal coordinate and line integration of density based on a binary tree in the longitudinal direction.

This algorithm was serial in the old code and fully parallel in the new MPI code, giving extra performance

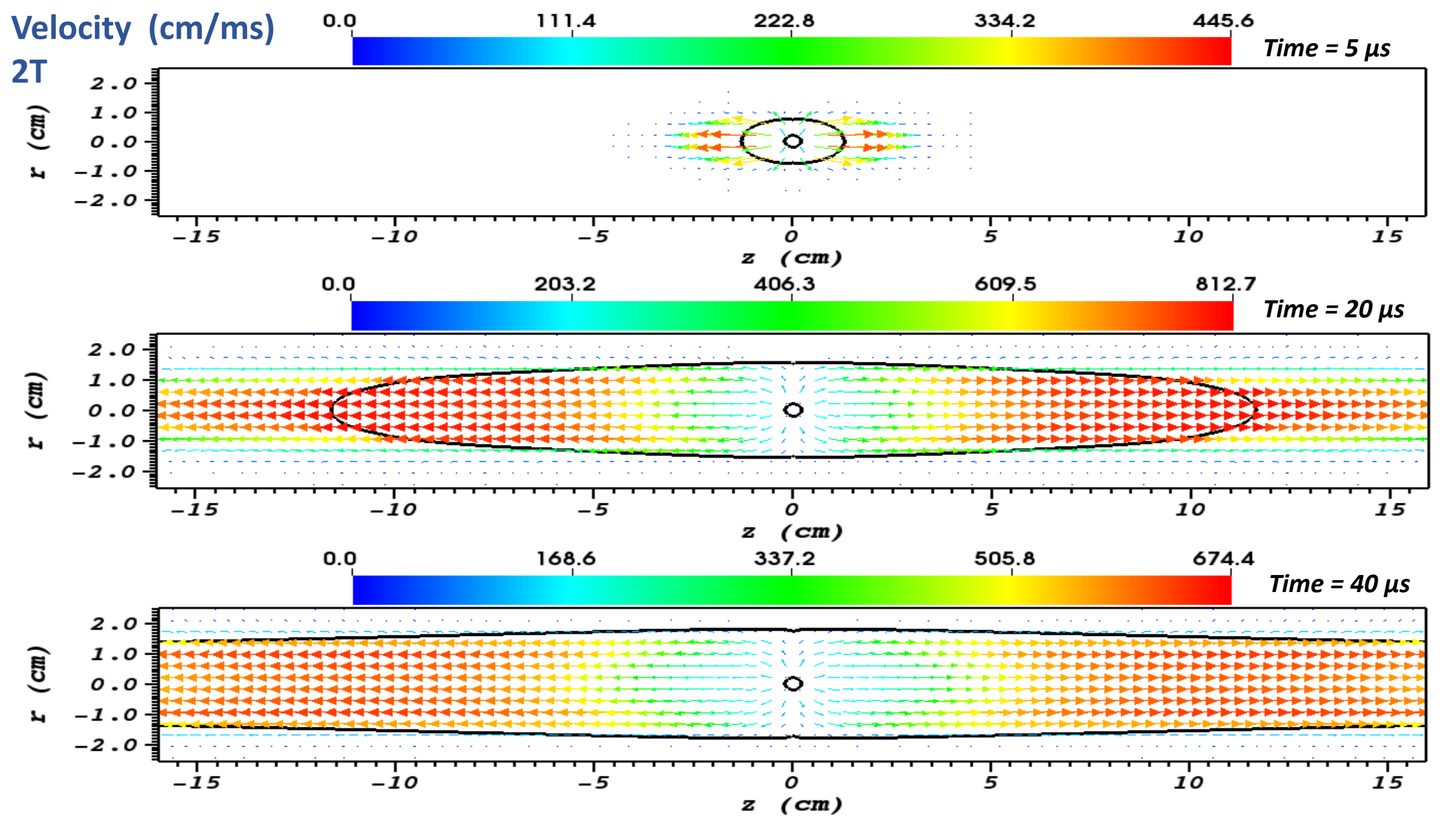
Explicit Tracking of Ablation Cloud – Plasma Interface in FronTier

- Past simulations of the pellet ablation with the FronTier code used interface tracking only for the pellet – ablated material interface
  - The interface between the ablated material and the ambient plasma was not tracked

- After implementing additional physics models that account for the properties of the ambient plasma in both codes, it became important to explicitly track the interface between the ablated material and the ambient plasma

- Such tracking was recently enabled in the FronTier code by developing new routines for the propagation of this interface

- While new simulation results show a big improvement in the transverse profiles of physics states in the ablation cloud (see sections on physics results), in particular the distribution of the longitudinal velocity in the transverse direction, that was previously very diffused near the cloud boundary, the ablation rate change due to tracking was small (within several %, depending on simulation conditions).


Velocity (cm/ms)
2T
0.0
111.4
222.8
334.2
445.6
Time = 5 μs
r (cm)
z (cm)
0.0
203.2
406.3
609.5
812.7
Time = 20 μs
r (cm)
z (cm)
0.0
168.6
337.2
505.8
674.4
Time = 40 μs
r (cm)
z (cm)