



Deposited via The University of York.

White Rose Research Online URL for this paper:

<https://eprints.whiterose.ac.uk/id/eprint/221009/>

Version: Published Version

---

**Conference or Workshop Item:**

Bennett, Keith, Morris, Stuart, Goffrey, T. et al. (2024) Re-engineering the EPOCH PIC code in C++. In: High Power Laser Christmas Meeting, 16-18 Dec 2024, Abingdon.

---

**Reuse**

Items deposited in White Rose Research Online are protected by copyright, with all rights reserved unless indicated otherwise. They may be downloaded and/or printed for private study, or other acts as permitted by national copyright laws. The publisher or other rights holders may allow further reproduction and re-use of the full text version. This is indicated by the licence information on the White Rose Research Online record for the item.

**Takedown**

If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing [eprints@whiterose.ac.uk](mailto:eprints@whiterose.ac.uk) including the URL of the record and the reason for the withdrawal request.

# Re-engineering the EPOCH PIC code in C++

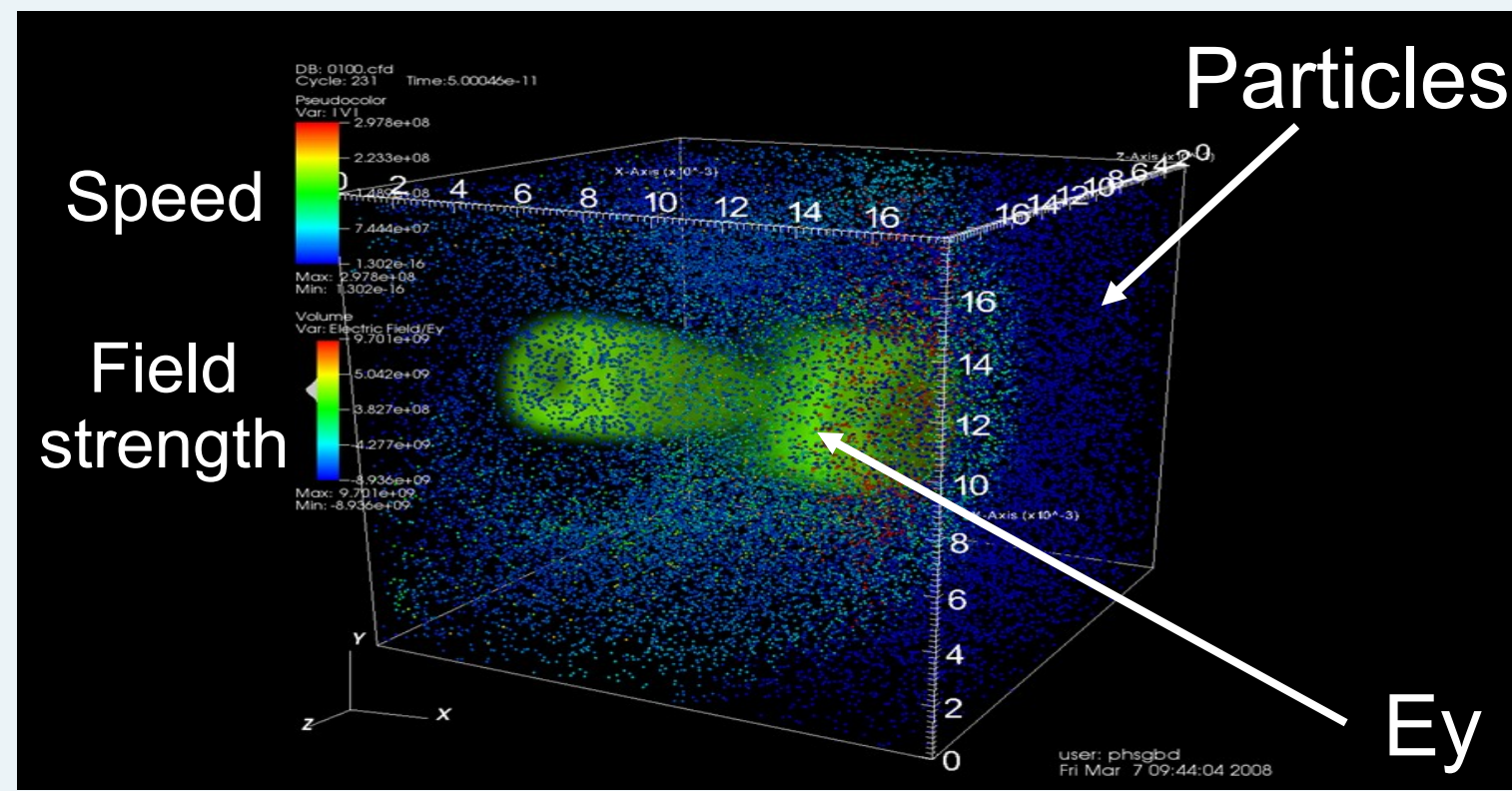
K. Bennett, S. Morris, T. Goffrey, T. Arber - *University of Warwick*

S. A. Wright, A. Naden, S. Bulut - *University of York*

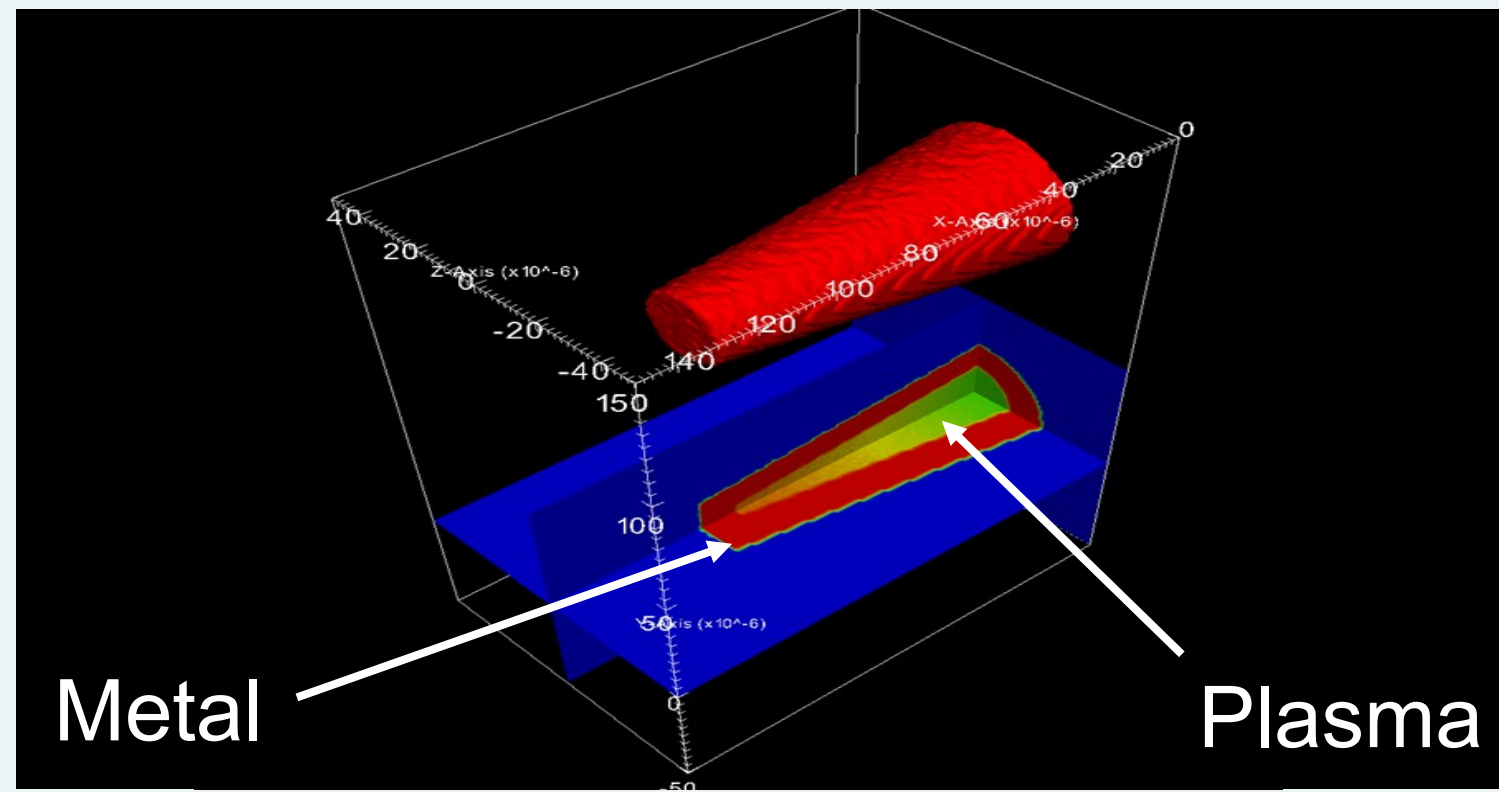


## The Extendable PIC Open Collaboration (EPOCH)

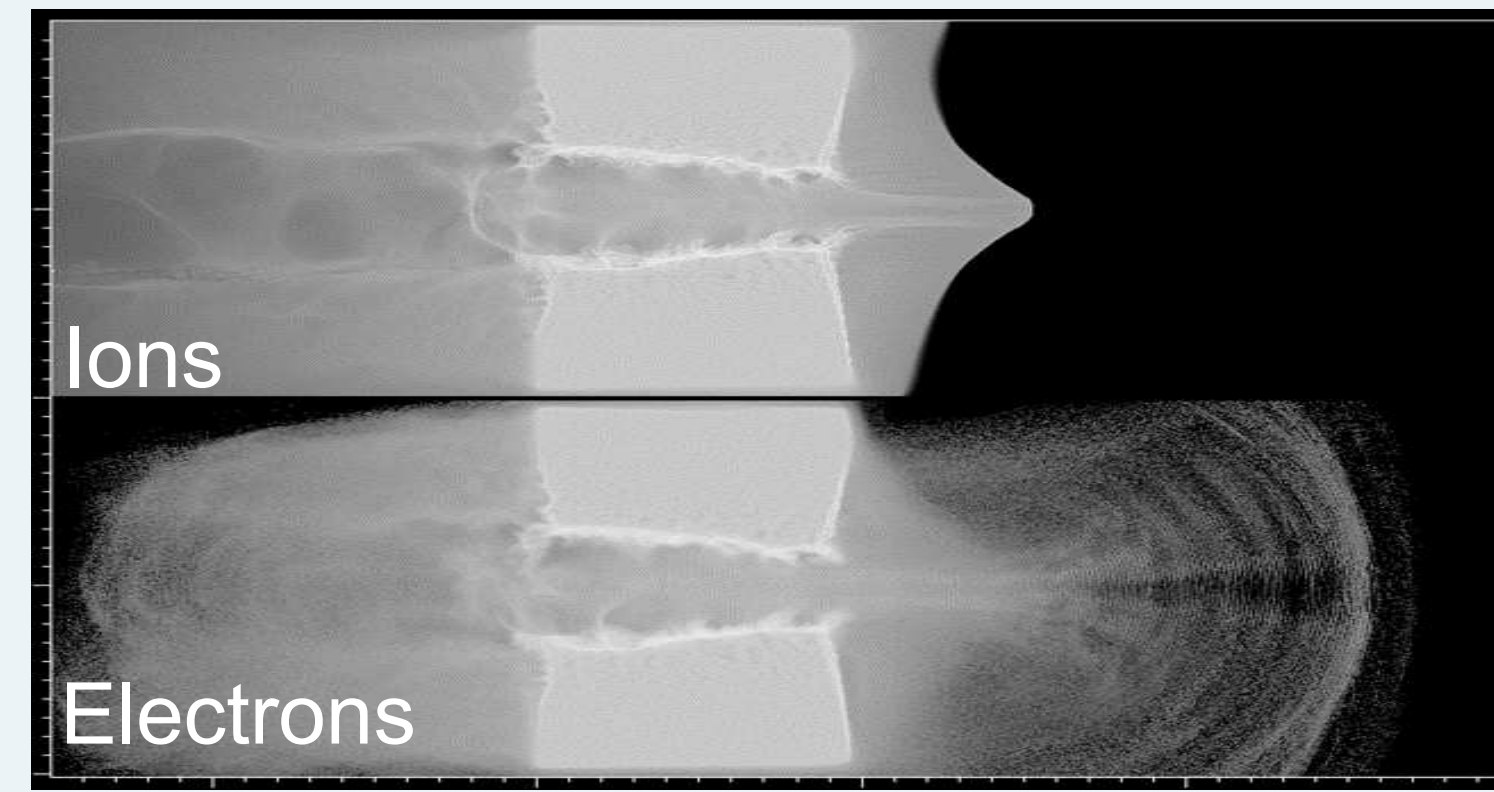
- EPOCH [1] is a relativistic EM-PIC code, which takes a simple text-file as input.
- EPOCH is written in F95 with MPI and scales to 32,000 cores on ARCHER2
- It has evolved since 2015 to include QED, radiation, ionisation, collisions, and cylindrical geometry.
- The code has been cited in over 1300 publications



3D simulation of a laser in underdense plasma (wakefield)



Setup of a 3D plasma-filled metal cone target for fast ignition simulation



2D simulation of a foil burn-through experiment (species density plotted)

It's new!

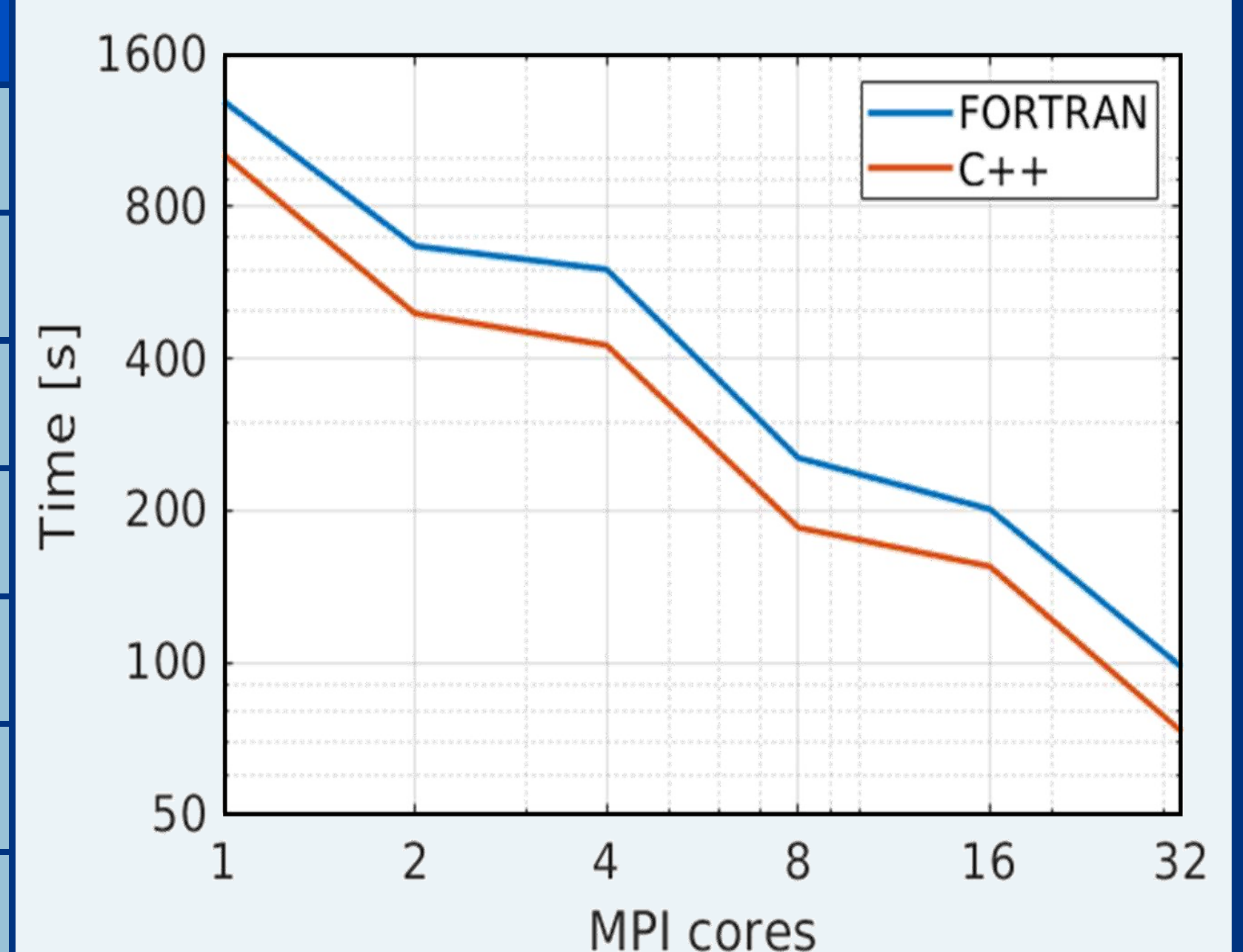
## C++ upgrade

- Advantages:
  - C++ templating: allows 1D, 2D and 3D in the same code
  - Derived classes: easily add features, like new physics packages
  - Modern HPC tools:
    - CPU/GPU portability library
    - High performance parallel data input/output
    - Portable particle/mesh data conventions
- Will be easier to implement run-time diagnostics
- Reads the same input decks as the FORTRAN code
- New documentation and examples provided

## Scaling tests

- Performance tested up to 32 cores against FORTRAN code
- Scaling comparable between two codes, with C++ 30-40% faster
- Comparison performed on 2D laser hole-boring example

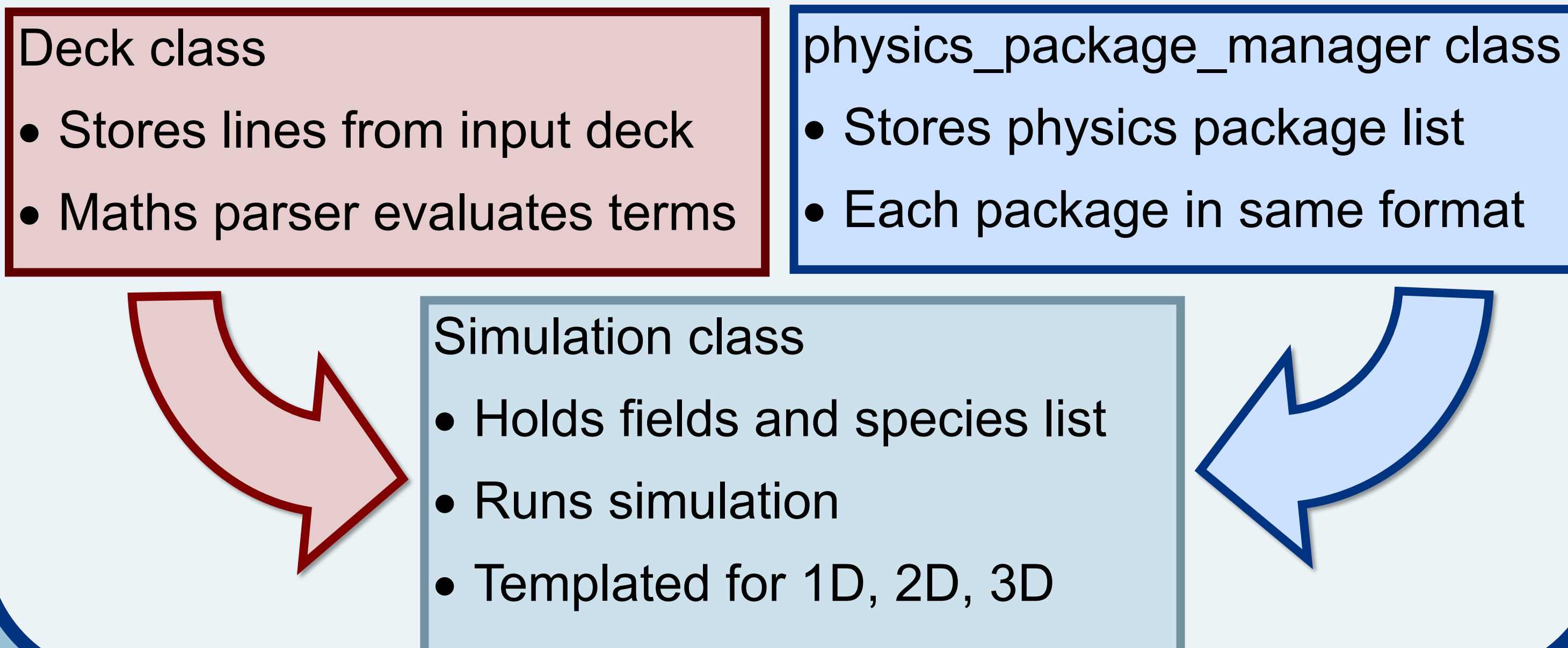
Simulation parameters
(500 x 500) cells
(25 x 25) $\mu\text{m}^2$ window
x BC: open
y BC: periodic
50 ppc (40 $e^-$ , 10 $C^{6+}$ )
$10^{22}$ $\text{Wcm}^{-2}$ Gaussian beam
5 $\mu\text{m}$ ionised C target
Pre-plasma, 2 $\mu\text{m}$ scale
100 fs simulated time



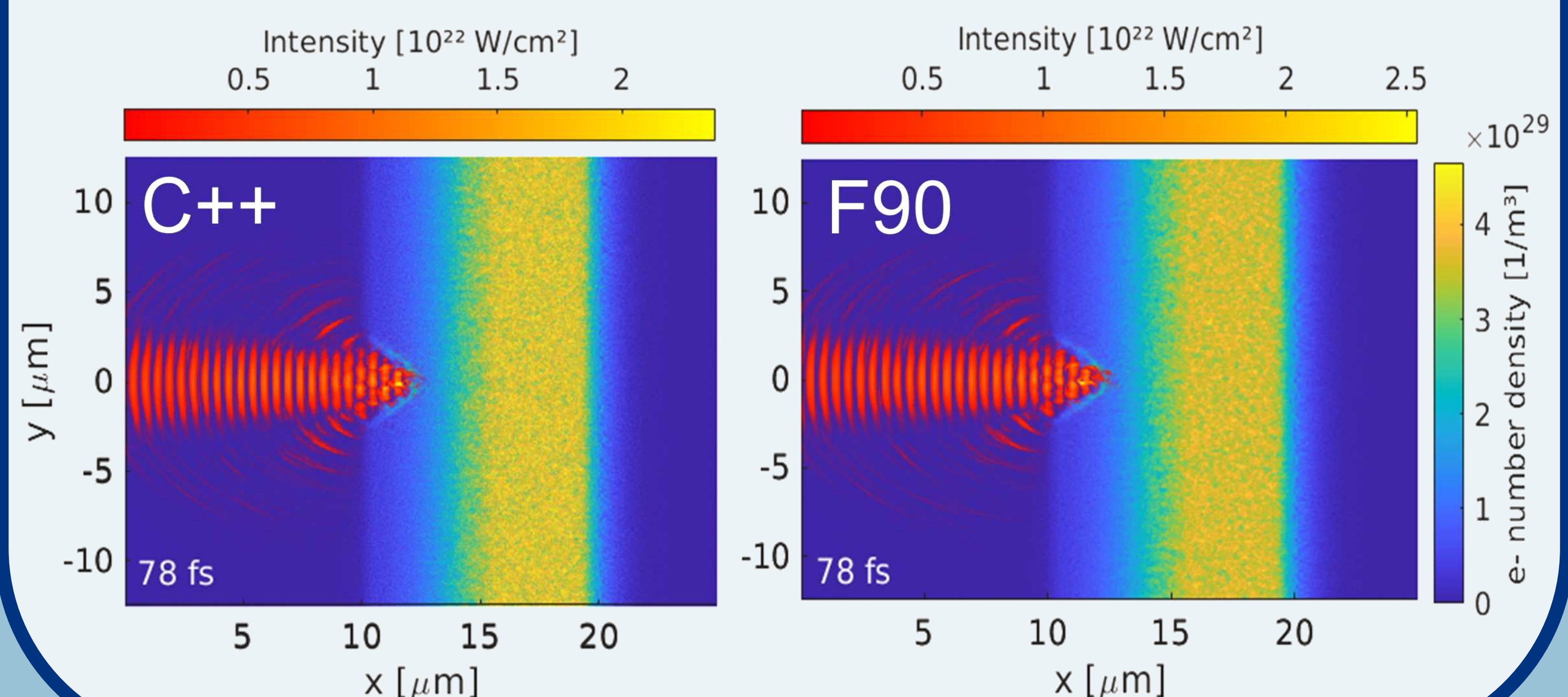
Simulation runtime as a function of MPI core count for FORTRAN and C++ code versions

## C++ structure

- Code is structured into classes, stored in a main Simulation class
- Various structures used to hold multiple derived lists
- Derived lists use polymorphism to allow easy extensibility

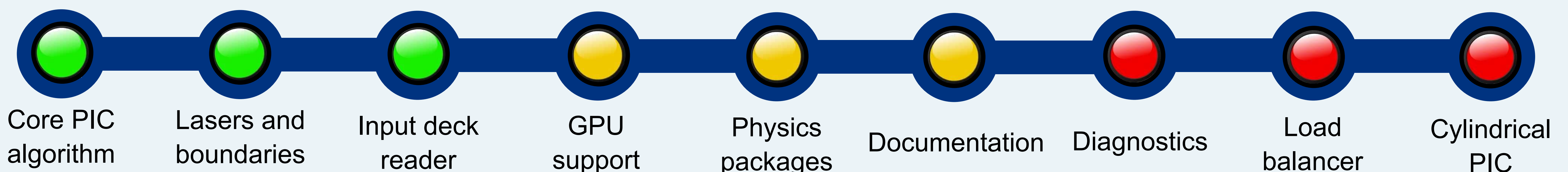


- Both codes yield the same hole-boring results:



## Project progress

- Visualisation of code progress. Green tasks are complete, yellow are works in progress, red are yet to start.



## Acknowledgements:

This project was supported by the EPSRC grants EP/W03008X/1 and EP/W029111/1



UNIVERSITY of York



Engineering and Physical Sciences Research Council

## References:

[1] T. D. Arber, *Plasma Phys. Control Fusion*, 57(11). (2015)