

REPTIL – Status of development and overview of capabilities

IBS in the XFEL injector

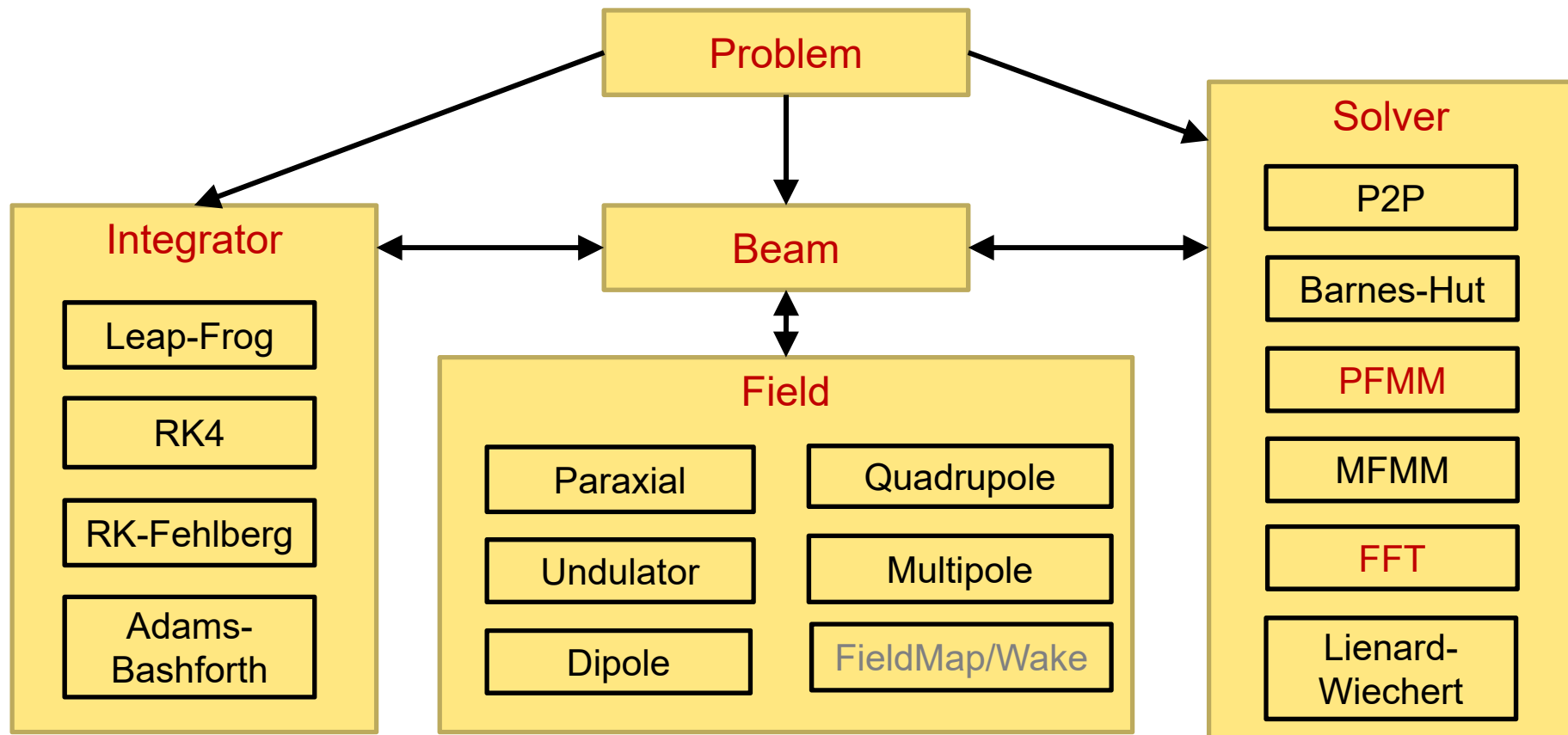
- Overview
- Description
 - <Problem>
 - <Beam>
 - <Solver>
 - <Integrator>
 - <Field>
- Convergence & accuracy
- Serial performance
- Shared & mixed memory scaling
- Distributed memory scaling

Overview

- **REPTIL** - **RE**lativistic **P**article **T**racking for **I**njectors & **L**inacs
- Background of development since more than 10 years (discontinuously) with contributions from Y. Chen, S. Schmitt
- These pieces and more were recently put together into a general-purpose tracking code because:
 - Cannot handle special problems (like IBS etc) without taking into account the rest of all effects along the beam line
 - We need a modern, high performance tool that is able to cope with large/huge problem sizes using new hardware capabilities
- Need interaction with users (Desy) to improve the code and to fully exploit its capabilities

Overview

- Structure of the code



Description

- The <Problem> menu

```
<Problem>  
  Name = XFEL  
  Type = tracking // tracking, loadonly or analysis  
<\Problem>
```

tracking – does the tracking

loadonly – load in the beam and field files and produces output files

analysis – computes slice energy spread and emittances and outputs them

Description

- The <Beam> menu

```
<Beam>
  File   = BSA1d_3ps_250pC_10k.ini
  Type   = astra    // astra for single file input, astra_partition for cluster
  Charge = 0.25e-09 // scale total charge to this number
<\Beam>
```

astra – only astra file-type is presently supported

astra_partition – multiple astra files (one separate file per MPI partition)

00000-astrafile.ini is read in from processor #0

00001-astrafile.ini is read in from processor #1

...

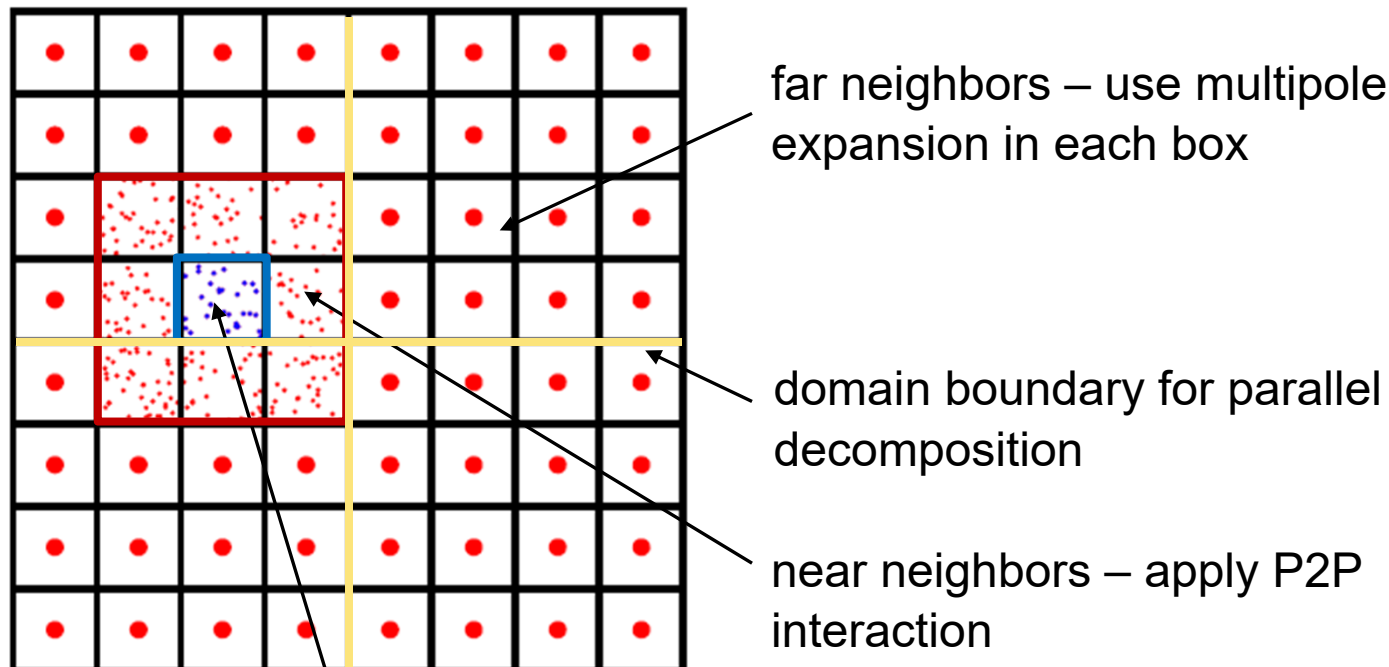
Description

- The <Solver> menu

```
<Solver>
  Type      = fft      // none, fft, mfmm, pfmm, bhtree, p2p or lw
  Nx        = 20      // number of mesh points for fft and mfmm
  Ny        = 20
  Nz        = 20
  IGF       = true    // use igf in fft solver
  NCrit     = 70      // number of bodies per box for mfmm and pfmm
  Theta     = 0.25    // accuracy criterion for bhtree
  Smooth    = 0       // mesh filter for charge density in mfmm and fft
  Alpha     = 0.5     // temporal IIR for the field data
  PSize     = 50e-6   // body size for p2p and bhtree
  Cathode   = true    // image charge for cathode at z=0
<\Solver>
```

Description

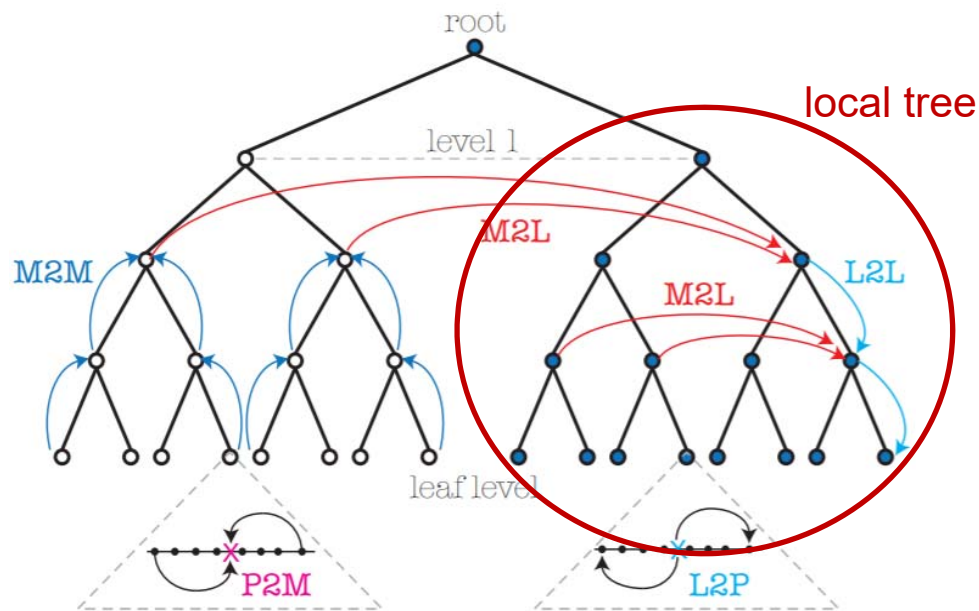
- The particle-FMM (PFMM) solver



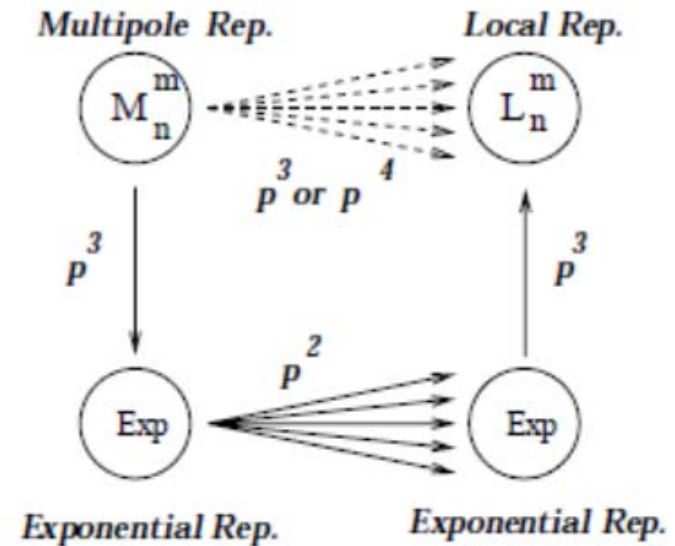
“NCrit” particles / box. Ncrit ~ 50-100

Description

- The particle-FMM (PFMM) solver



Hierarchic computation of mult. expansion
(Rohklin & Greengard, 1987)

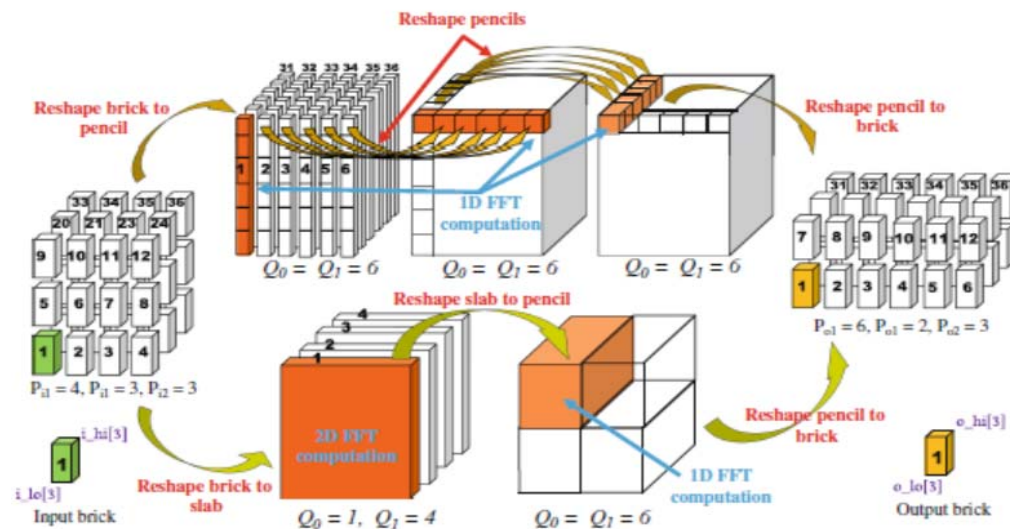


Exponential expansions for M2L
(Greengard & Rohklin, 1996)

MPI-parallelization needs to be improved – presently, local tree on each process...

Description

- The FFT solver
 - Applies Hockney's algorithm on doubled domains using (optionally) IGF and shifted Green functions for the cathode charge
 - Based on the parallel 3D-FFT library – HeFFTe¹
 - Backend FFTW, Intel MKL and CuFFT (**to be tested**)



Decomposition strategies for parallel FFT:

The input data is reshaped such that each processor performs one 1D-FFT at a time¹

¹ A. Ayala, et al., “heFFTe: Highly Efficient FFT for Exascale,” (ICCS 2020)

Description

- The <Integrator> menu

```
<Integrator>
  Type      = ab4      // lf2, rk, rkf or ab4
  TStep     = 1e-14    // start time step
  MinStep   = 1e-14    // minimum time step
  MaxStep   = 1e-09    // maximum time step
  NStep     = 100000   // maximum number of time steps
  TEnd      = 1e-06    // end integration time
  ZEnd      = 15.5     // end position for time integration
  Tolerance = 1.0e-7   // accuracy for time step adaption
<\Integrator>
```

- 4th order Adams-Bashforth + 5th order Adams-Moulton for error estimation
- Modified Nordsieck technique for automatic time step adaption
- **Only one solver call / time step (factor 4 faster than RKF)**

Description

- The <Monitor> menu

```
<Monitor>  
  NSlice = 300    // number of slices for slice analysis  
  ZSlice = 15.0  // write slice quantities every ZSlice-interval  
  ZData  = 10e-3 // write projected quantities every ZData-interval  
  ZVerb  = 10e-3 // output info every ZVerb-interval  
  ZSave  = 15.0  // save a complete distribution every ZSave-interval  
  ZWrite = 0.0   // write distribution in vtk format every ZWrite-interval  
<\Monitor>
```

Description

- The <Field> menu – axial cavity fields and solenoids

```
<Field>
  Name      = RF-Gun
  Type      = axial_tm_cavity
  File      = rz_GUN_FEM-003-531-531-filt3.txt
  Frequency = 1.3e+9
  Amplitude = 56.3e+6
  Phase     = 219.92 // 223.92-4
  ZShift    = 0.0
  Filter    = 0
  Order     = 1
  Write     = false
<\Field>
```

TM-mode cavity field

E_z on axis at equidistant points

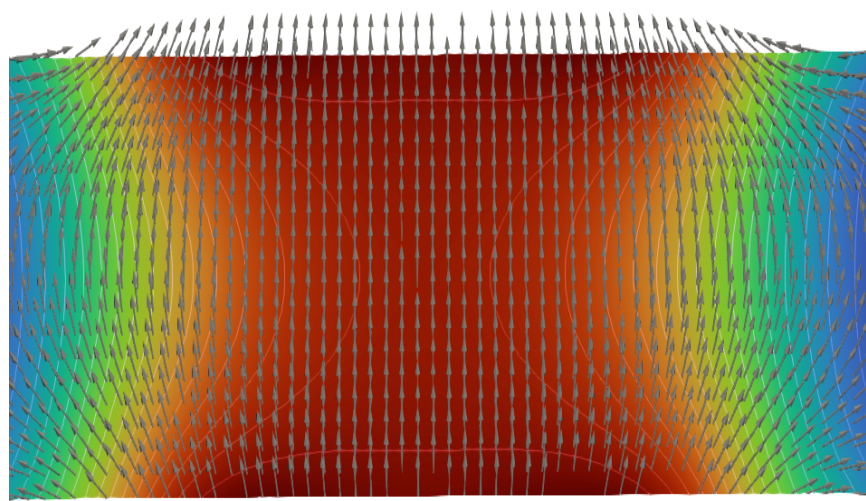
Absolute cavity phase (no auto-phasing)

Output field on axis and its derivatives for debugging

Description

- The <Field> menu – dipoles

```
<Field>
  Name      = Dipole
  Type      = ideal_dipole
  ZMin      = 14.0
  ZMax      = 14.2
  Strength  = 1
  Gap       = 0.1
  XOffset   = 0
  YOffset   = 0
  XRotation = 0
  YRotation = 0
  Write     = true // write field distribution in vtk format
<\Field>
```

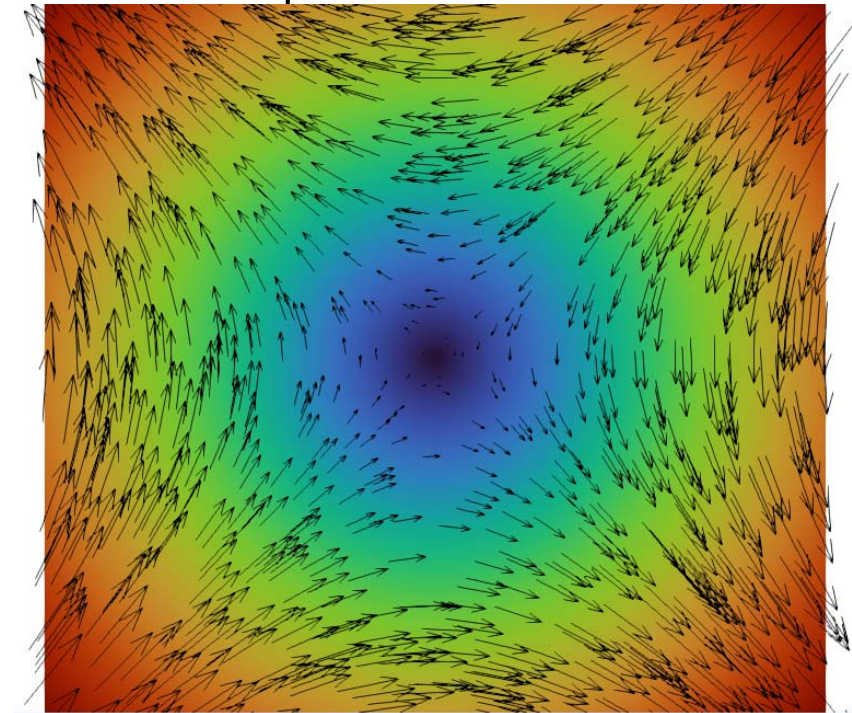


- Enge's model (with one parameter) for fringe fields
- Yet to be tested in “real” simulations

Description

- The <Field> menu – quadrupoles

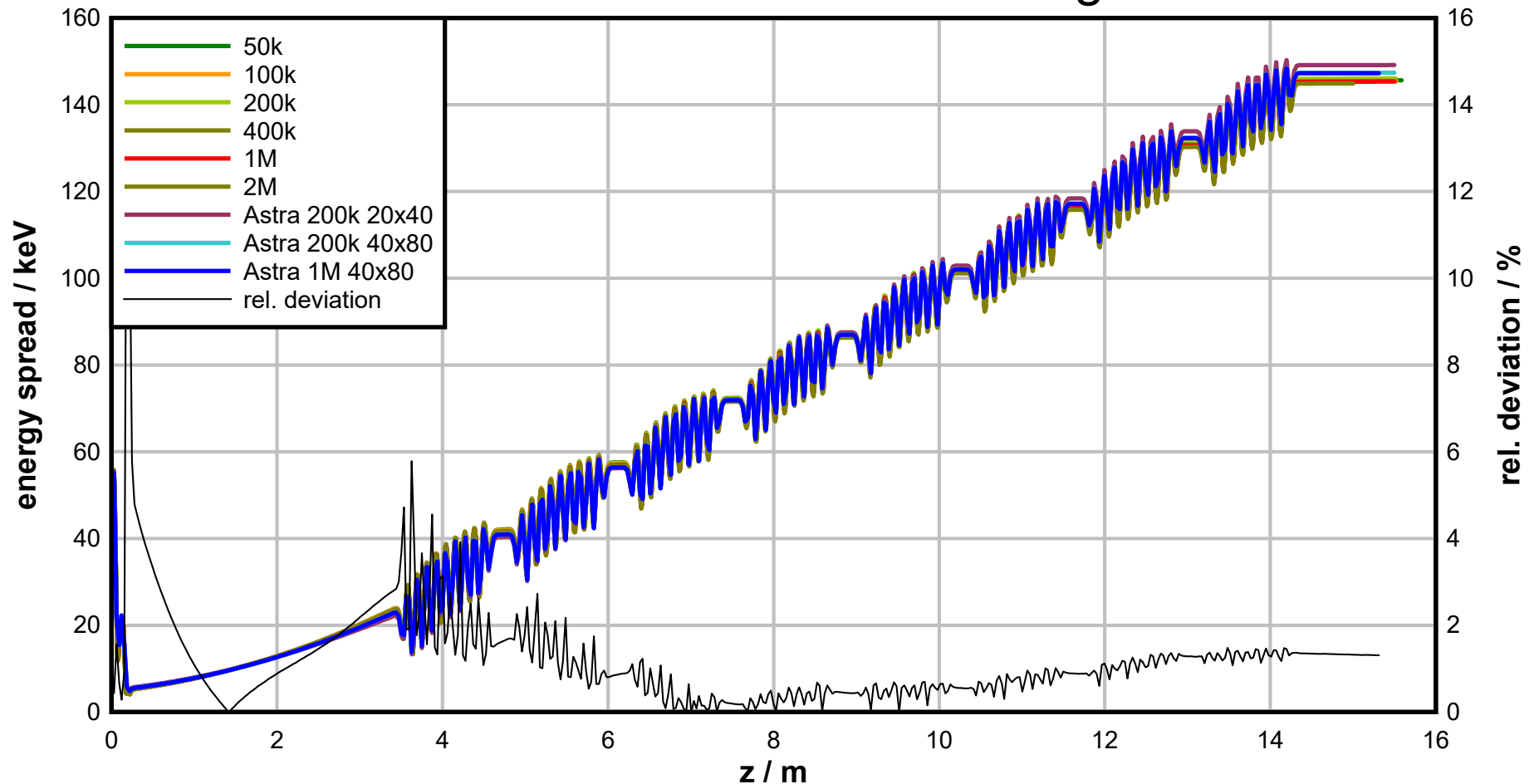
```
<Field>  
  Name      = Q.37.I1  
  Type      = ideal_quadrupole  
  Gradient  = -50  
  Bore      = 1.0e-07  
  ZMin      = 14.5  
  ZMax      = 14.7  
  XOffset   = 0  
  YOffset   = 0  
  XRotation = 0  
  YRotation = 0  
  Write     = false  
<\Field>
```



- Enge's model (with one parameter) for fringe fields
- Yet to be tested in “real” simulations

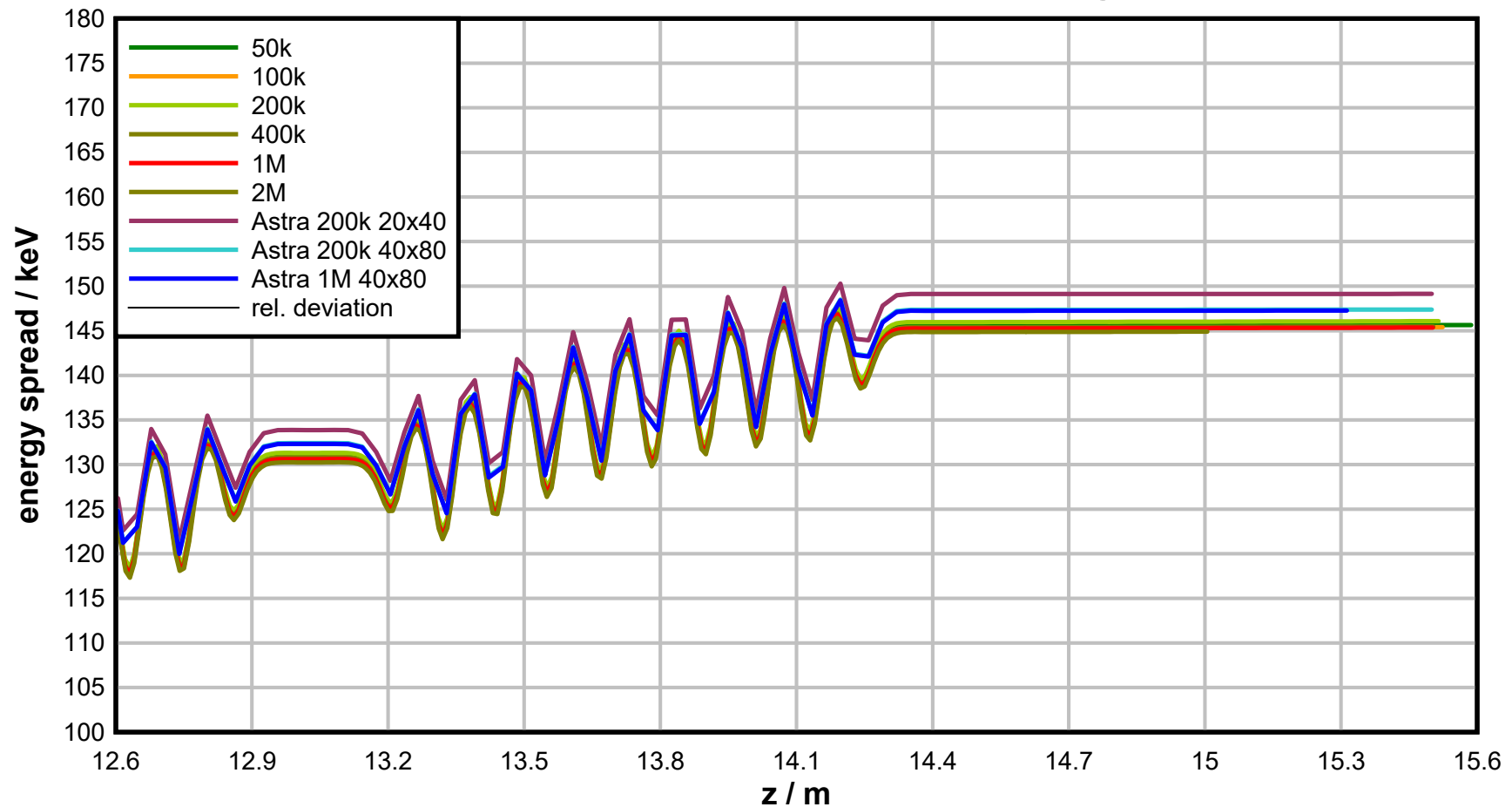
Convergence & Accuracy

The PFMM-solver with AB4-integrator



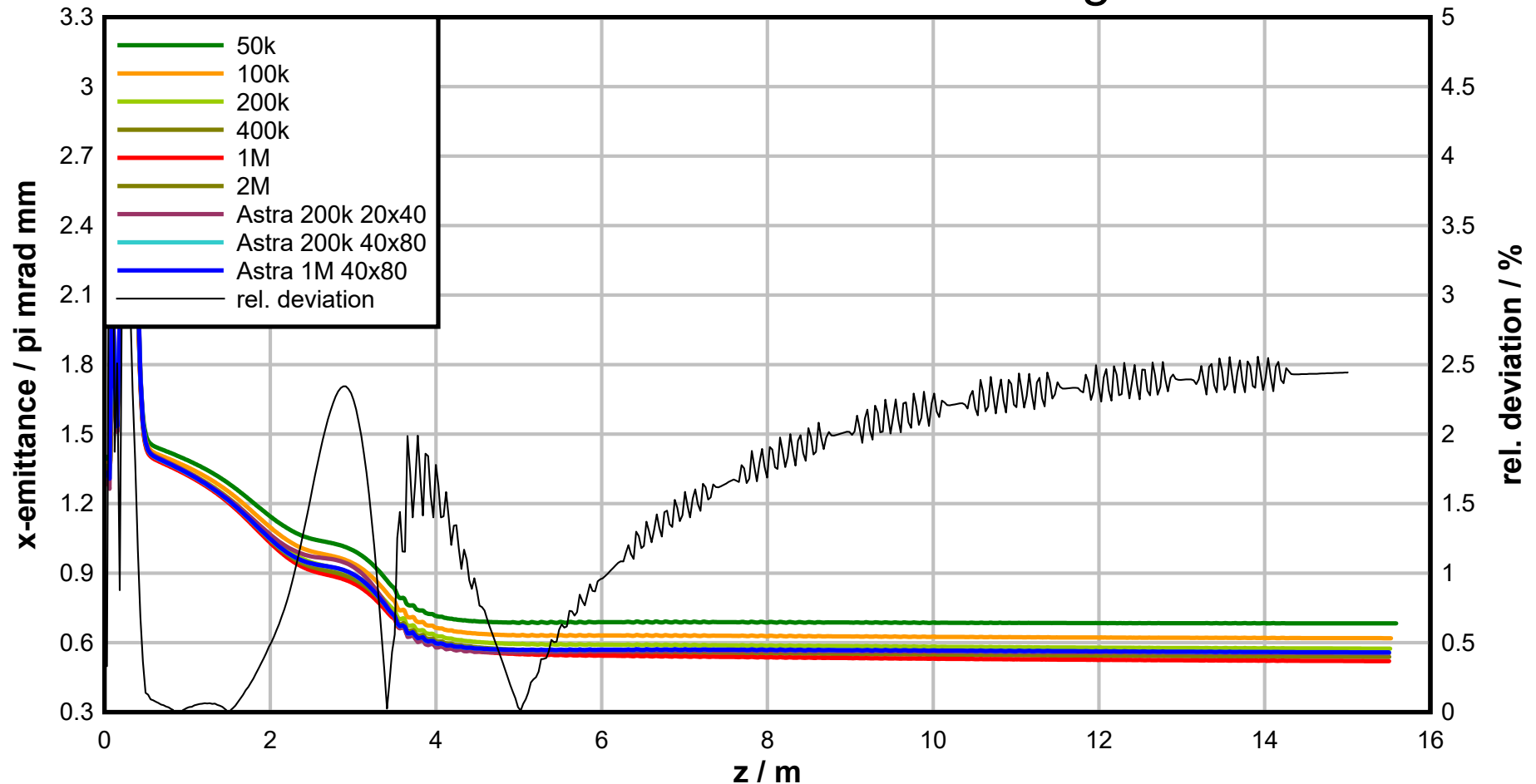
Convergence & Accuracy

The PFMM-solver with AB4-integrator



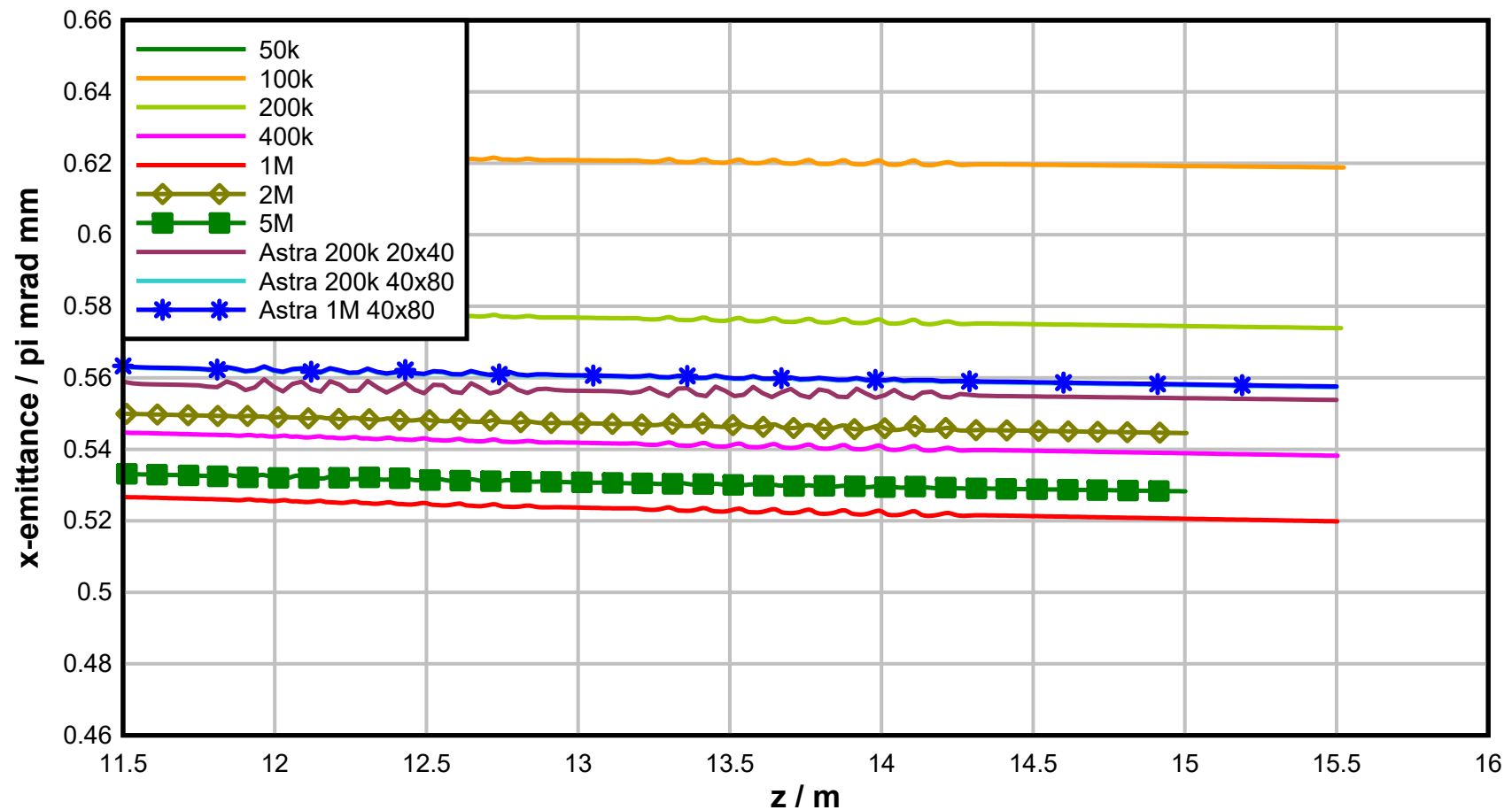
Convergence & Accuracy

The PFMM-solver with AB4-integrator



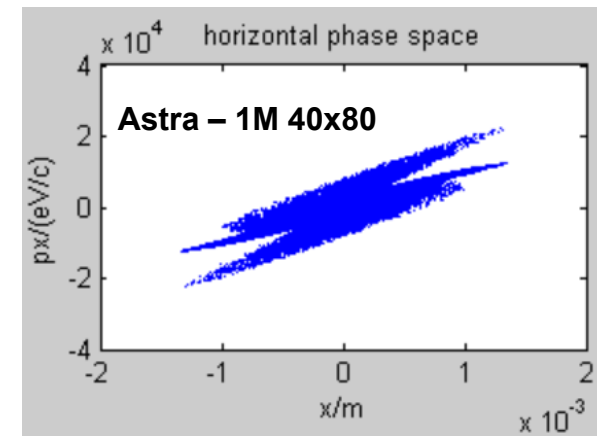
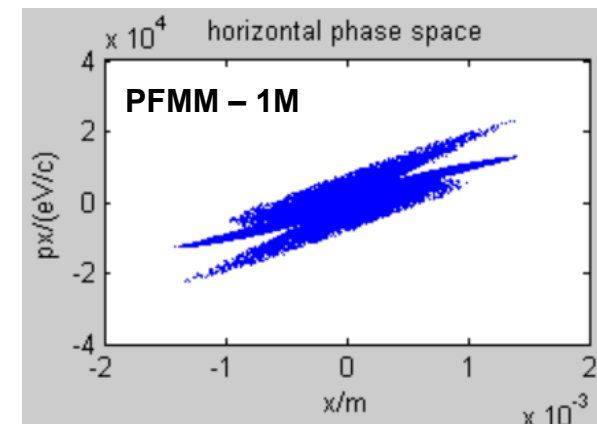
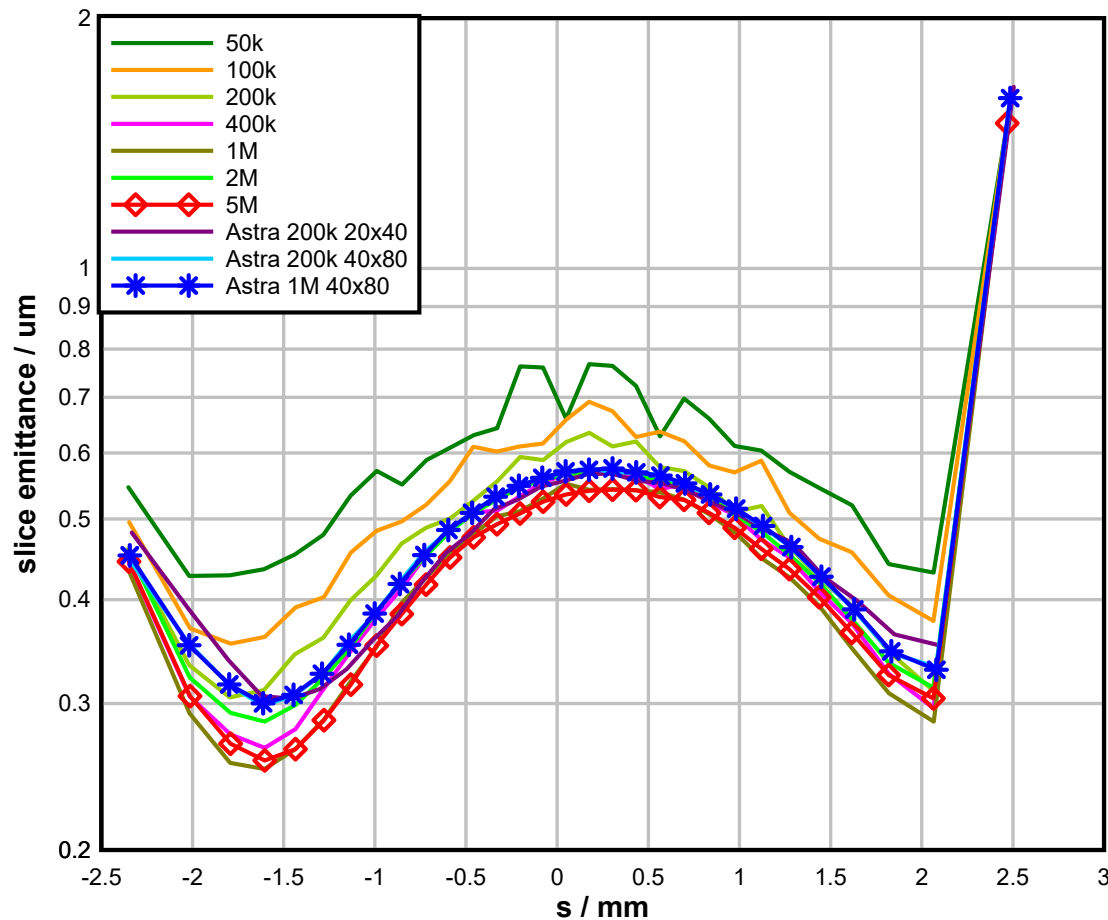
Convergence & Accuracy

The PFMM-solver with AB4-integrator



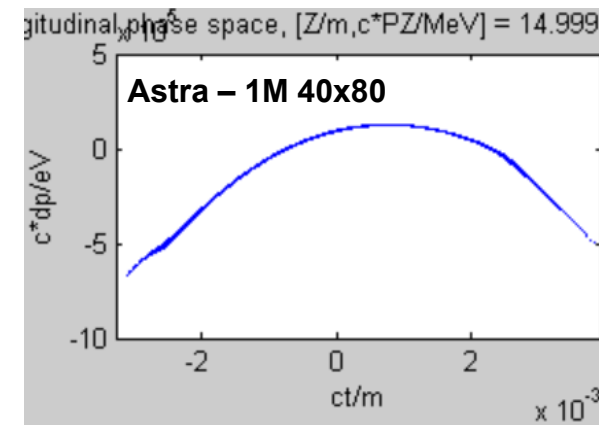
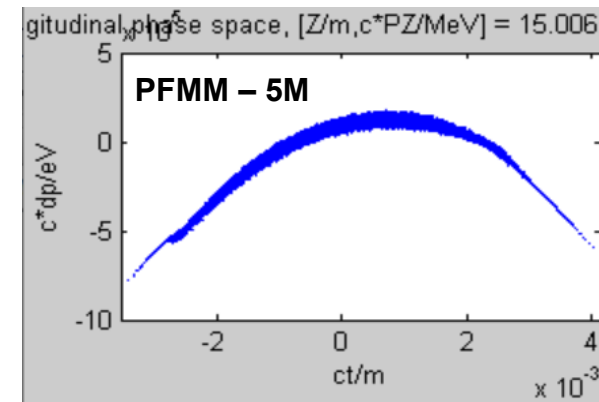
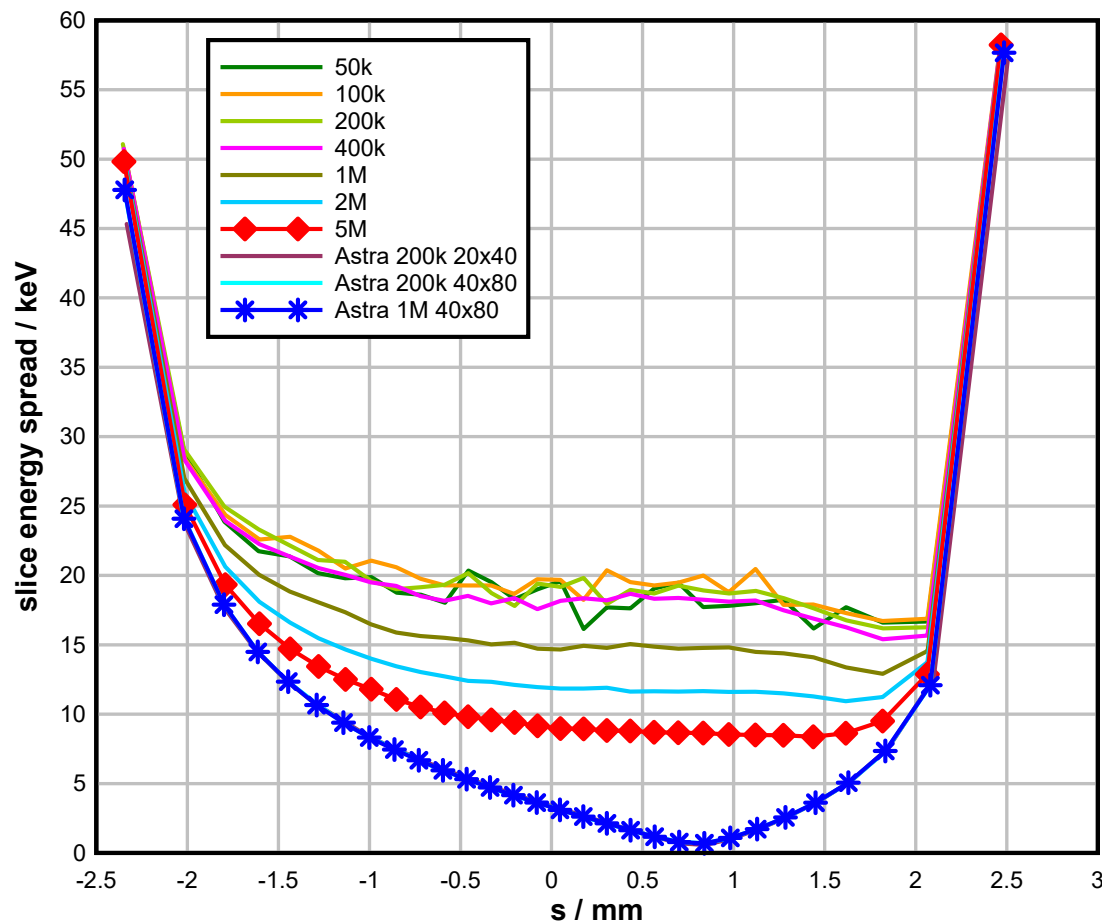
Convergence & Accuracy

The PFMM-solver with AB4-integrator



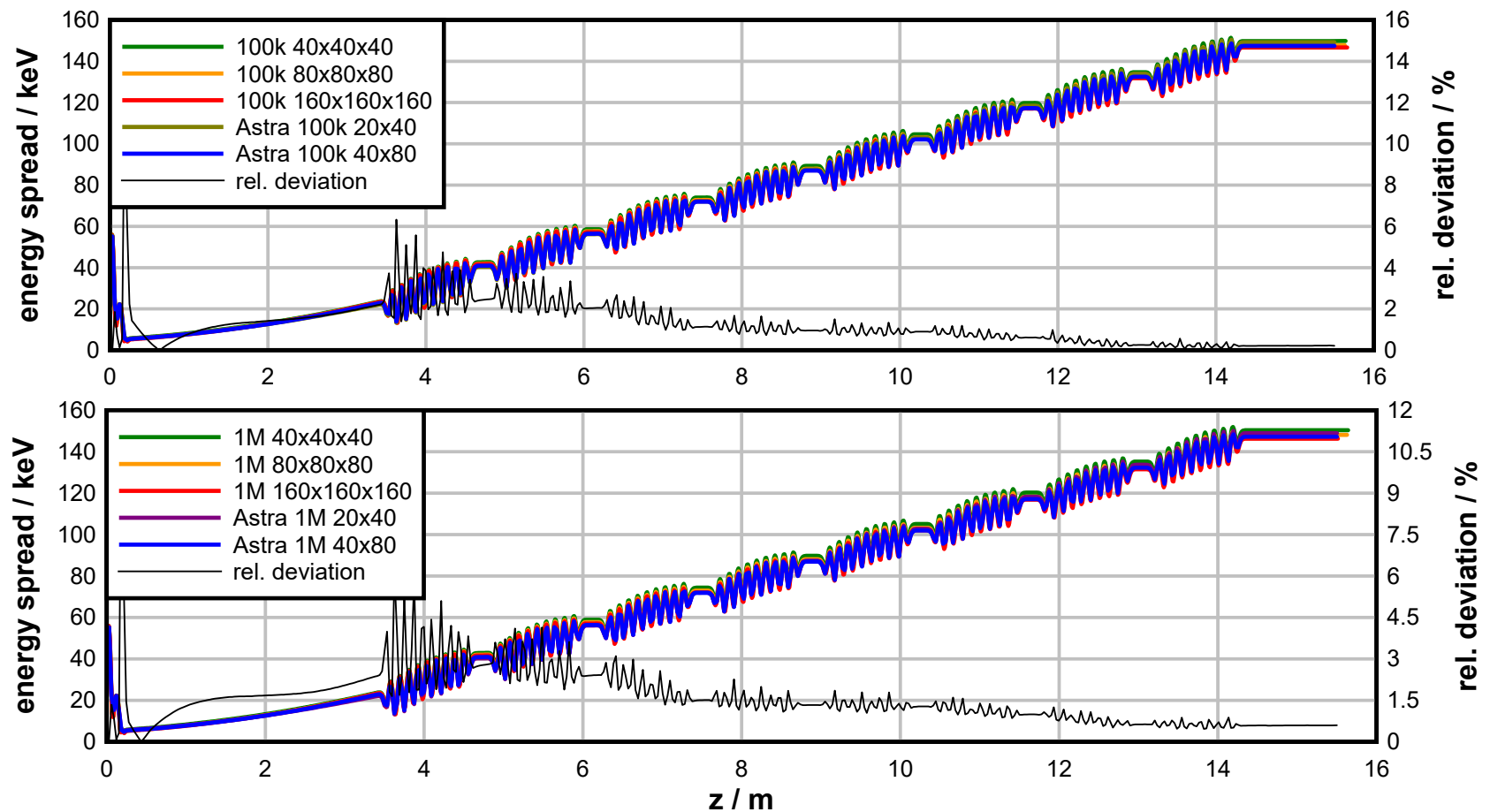
Convergence & Accuracy

The PFMM-solver with AB4-integrator



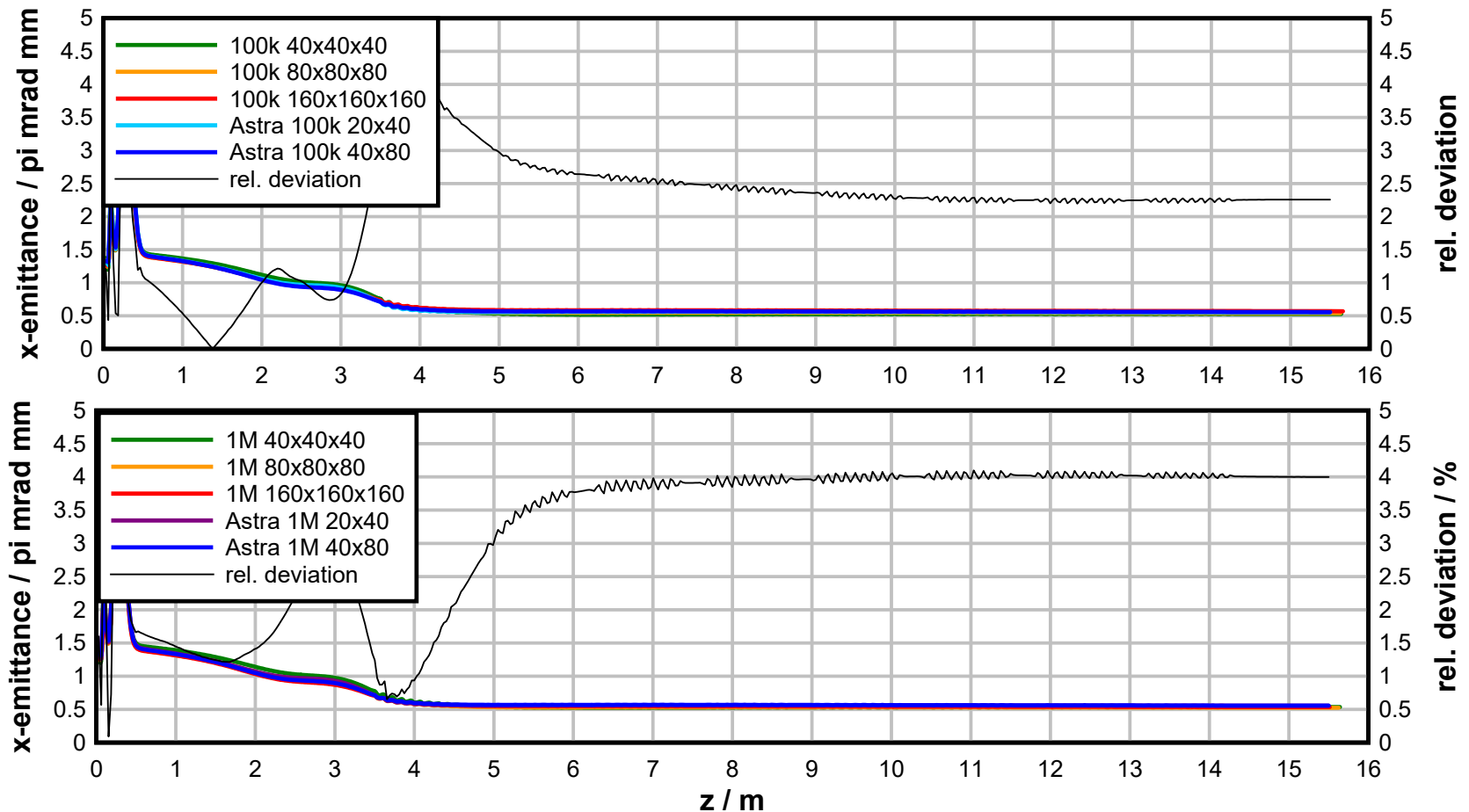
Convergence & Accuracy

The FFT-solver with AB4-integrator



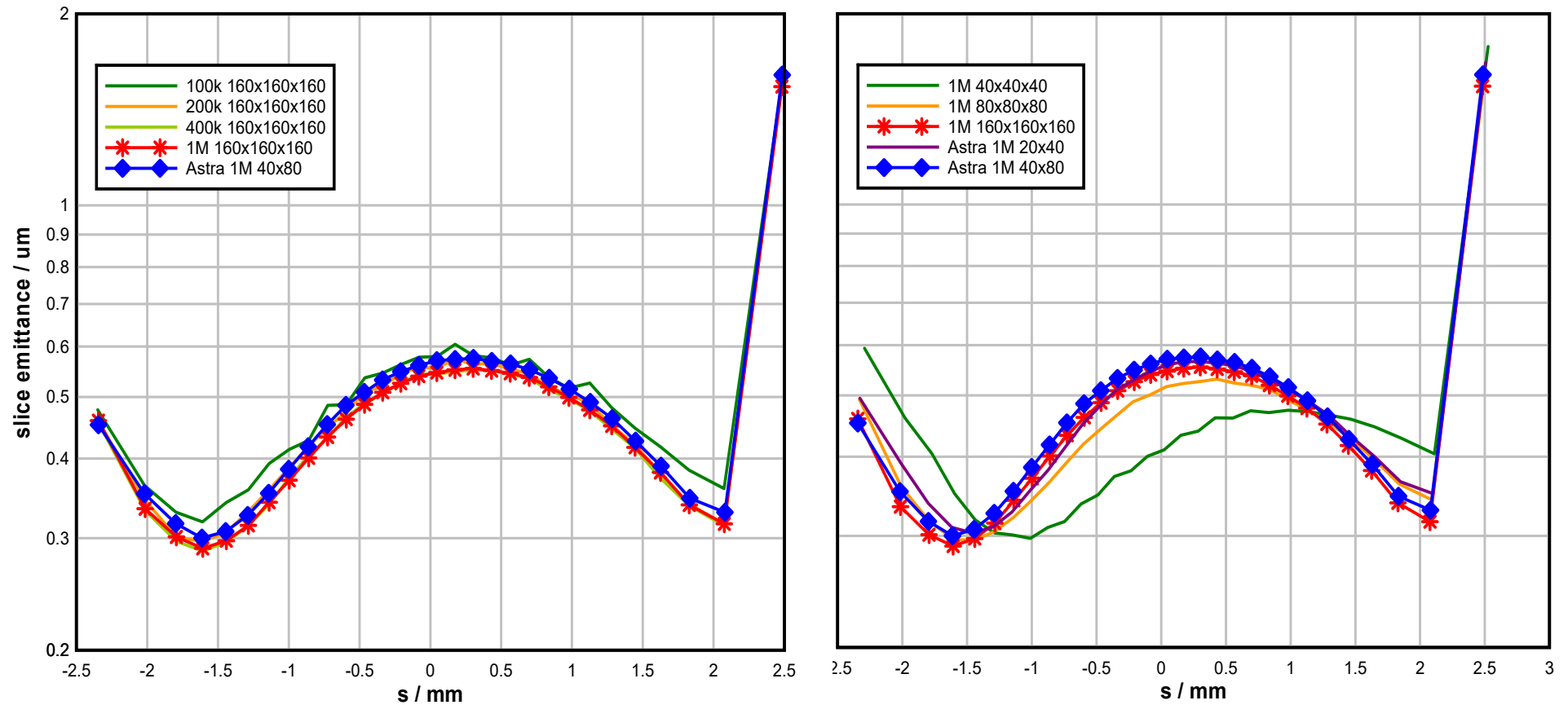
Convergence & Accuracy

The FFT-solver with AB4-integrator



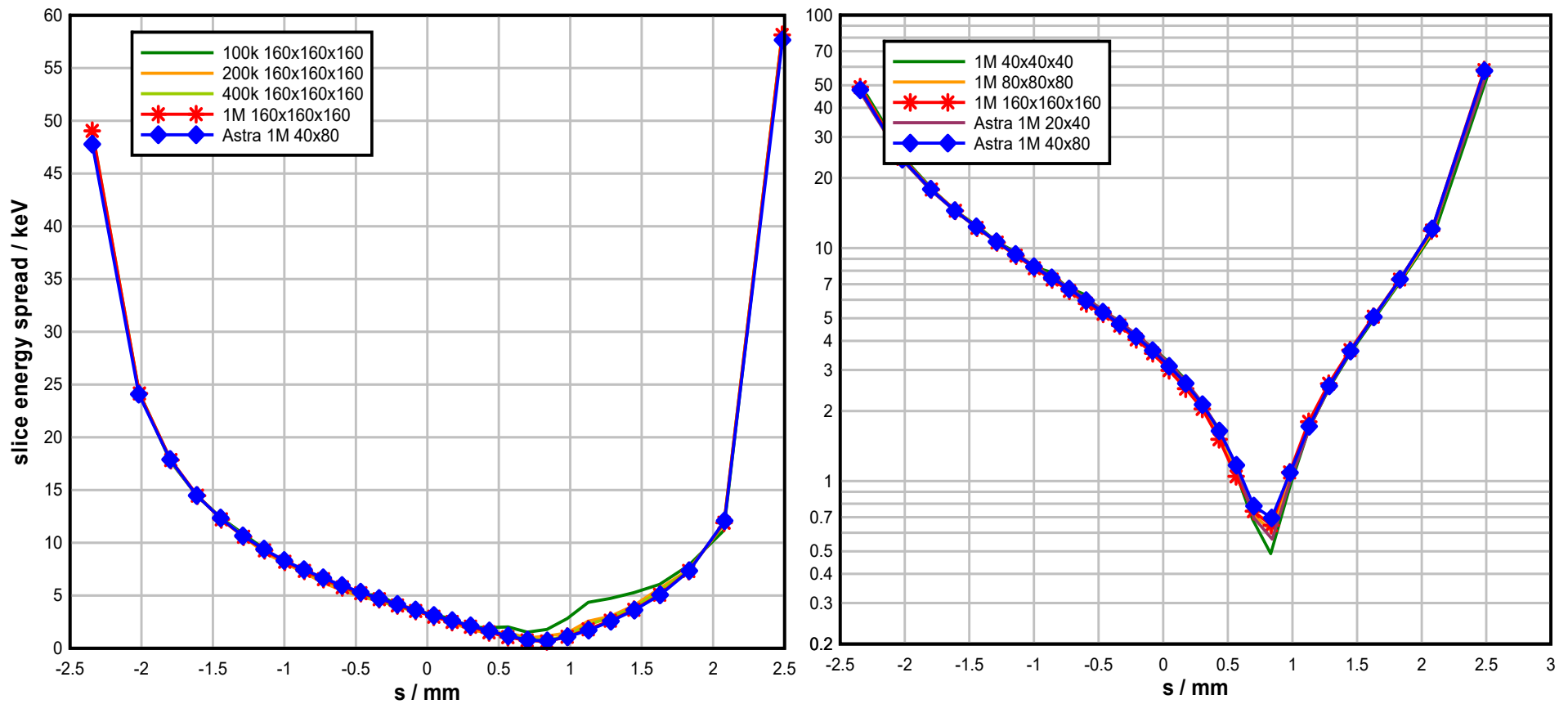
Convergence & Accuracy

The FFT-solver with AB4-integrator



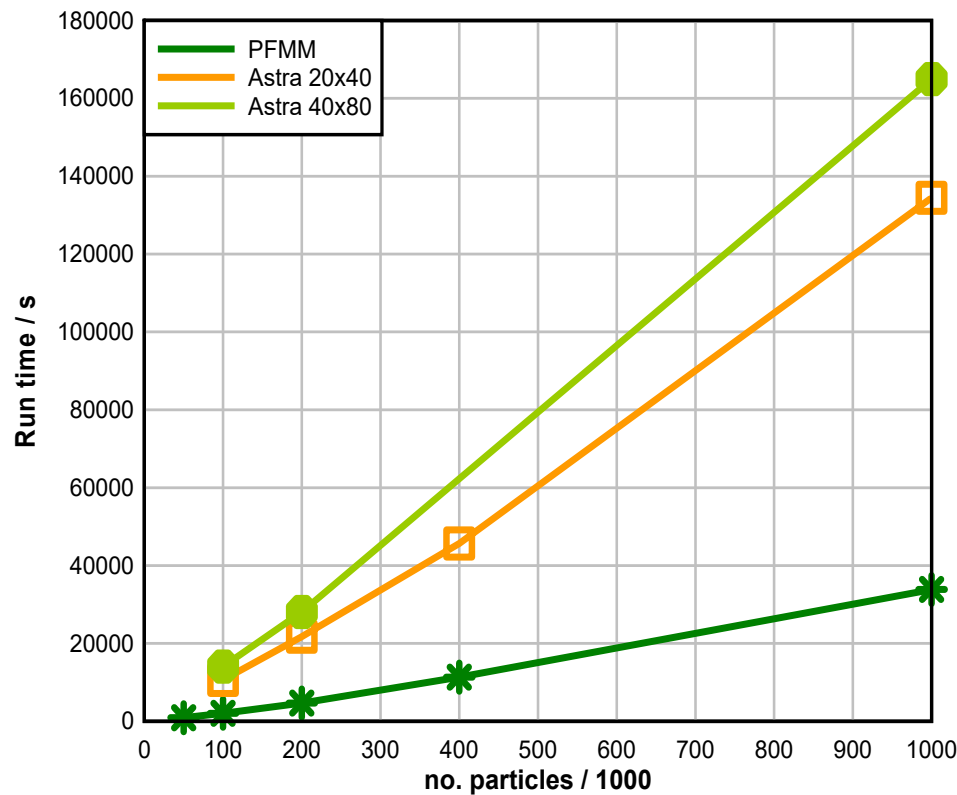
Convergence & Accuracy

The FFT-solver with AB4-integrator

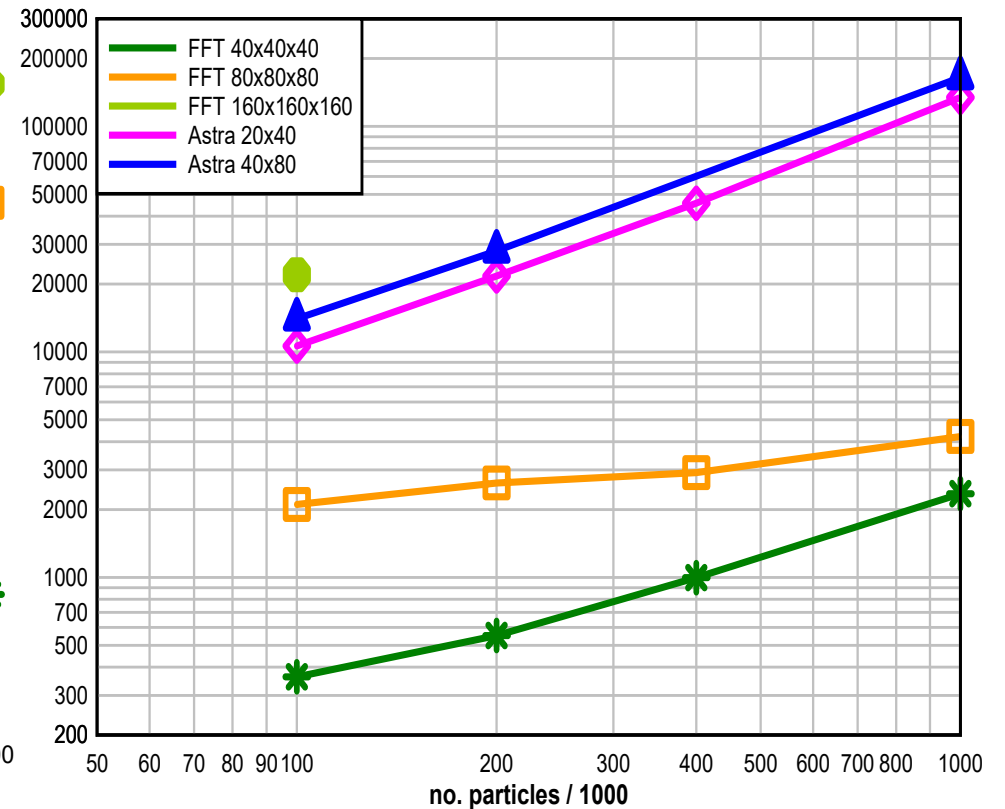


Serial performance

PFMM + AB4 vs. Astra

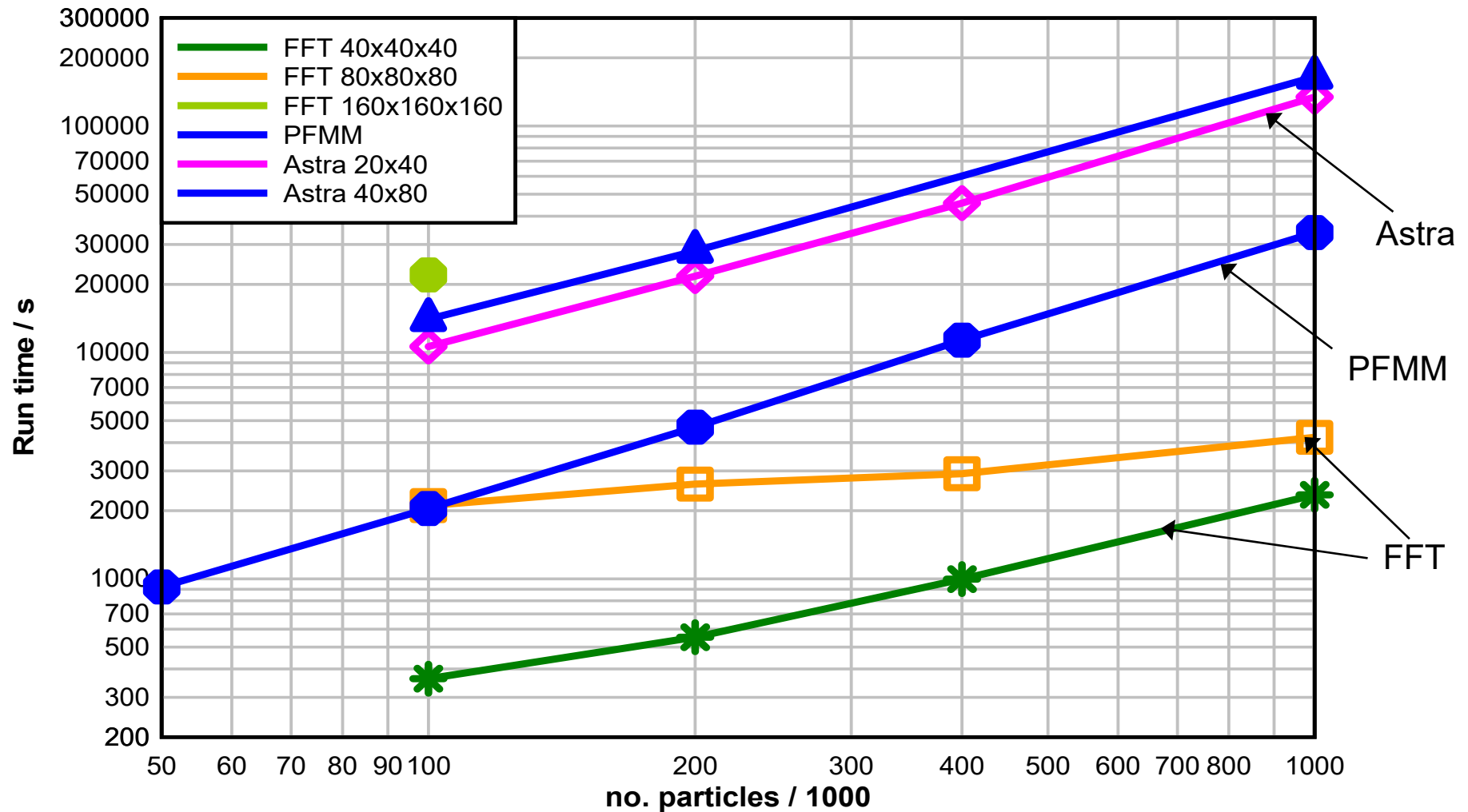


FFT + AB4 vs. Astra



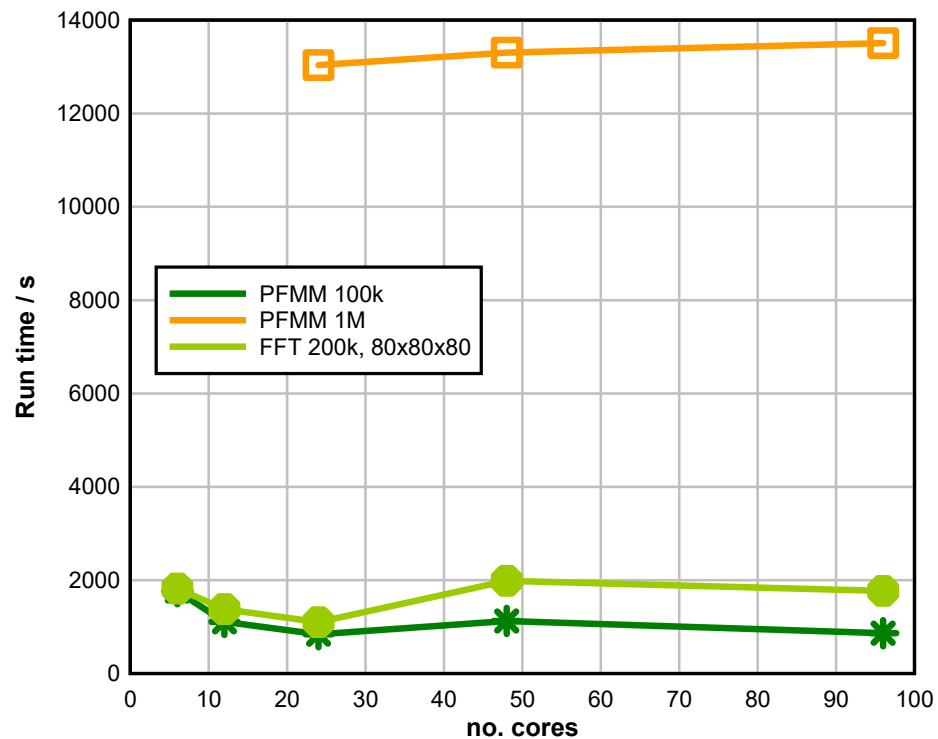
- Windows 10 workstation - 2 x Intel Xeon 3.47 GHz, 6 cores / processor

Serial performance

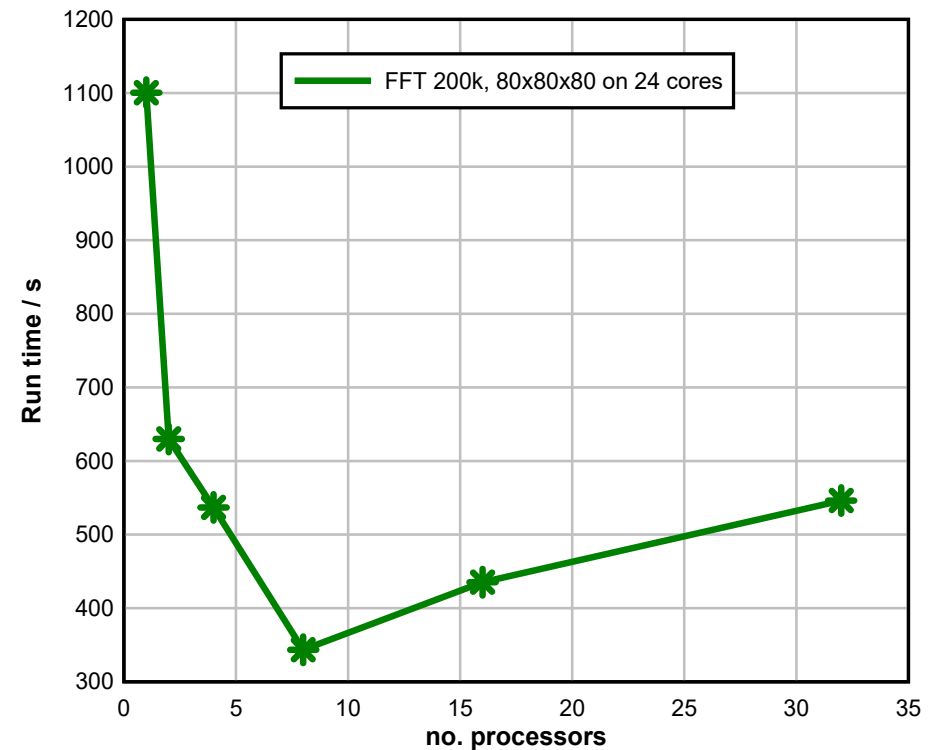


Shared & mixed memory scaling

vs. no. of cores (1 processor)



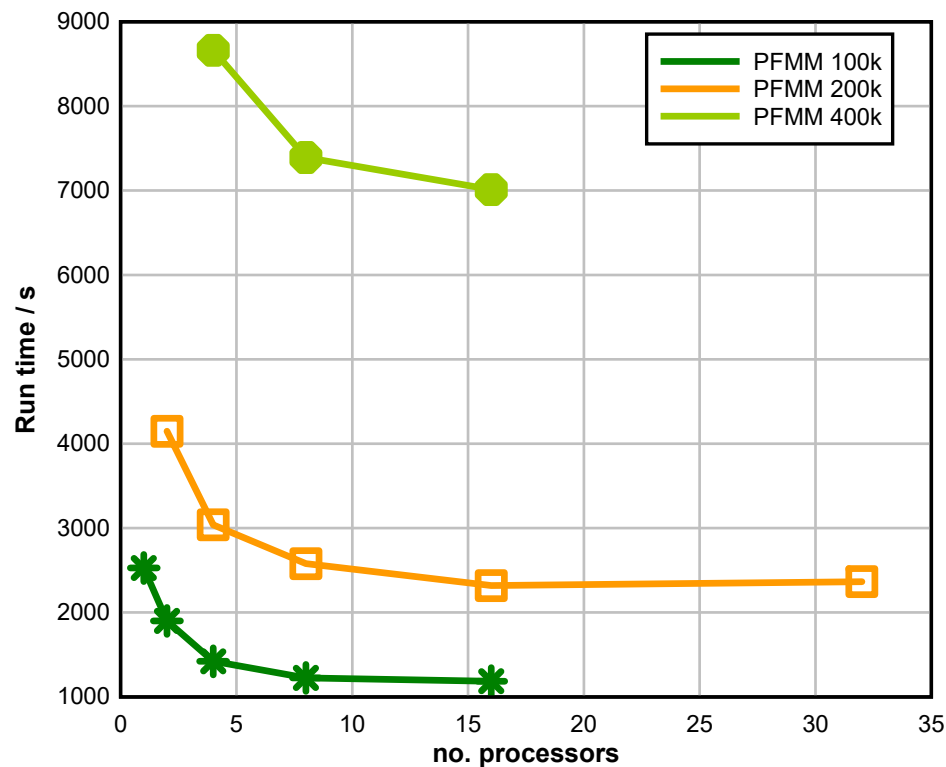
vs. no. of processors (24 cores)



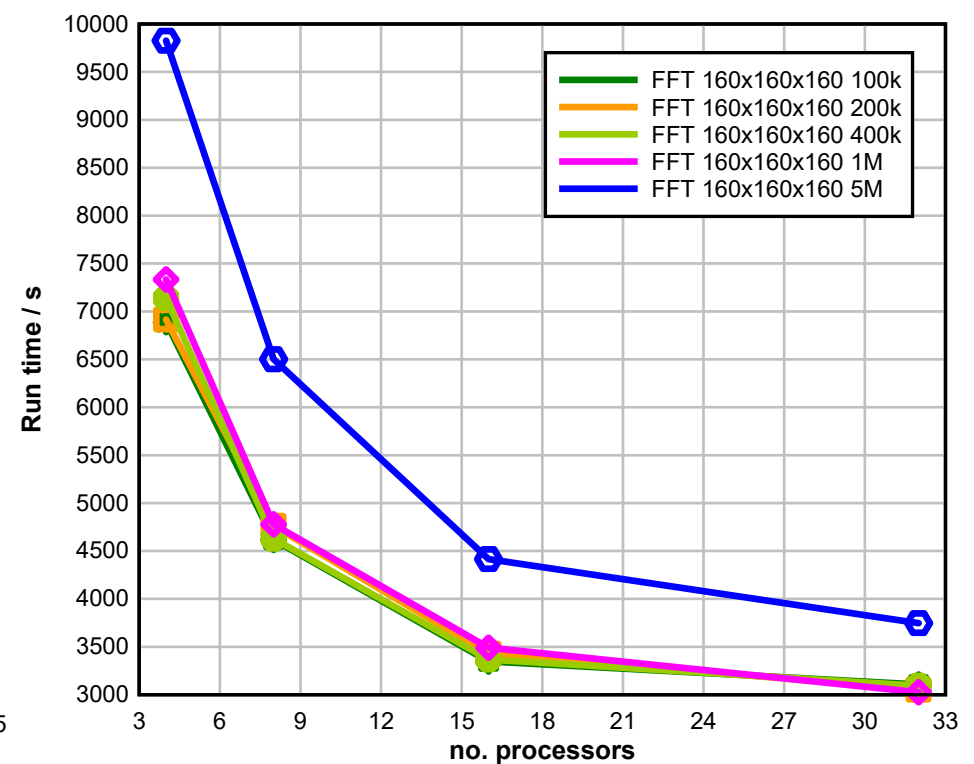
- HHLR – Lichtenberg cluster of the TU Darmstadt

Distributed memory scaling

PFMM + AB4 – run time



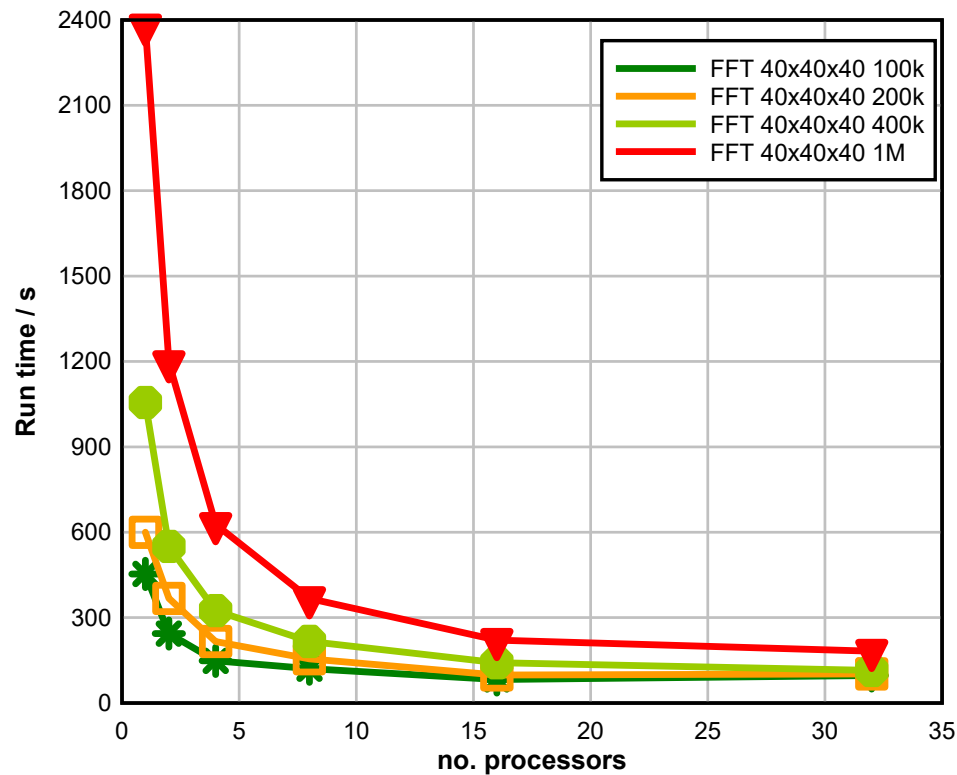
FFT + AB4 – run time



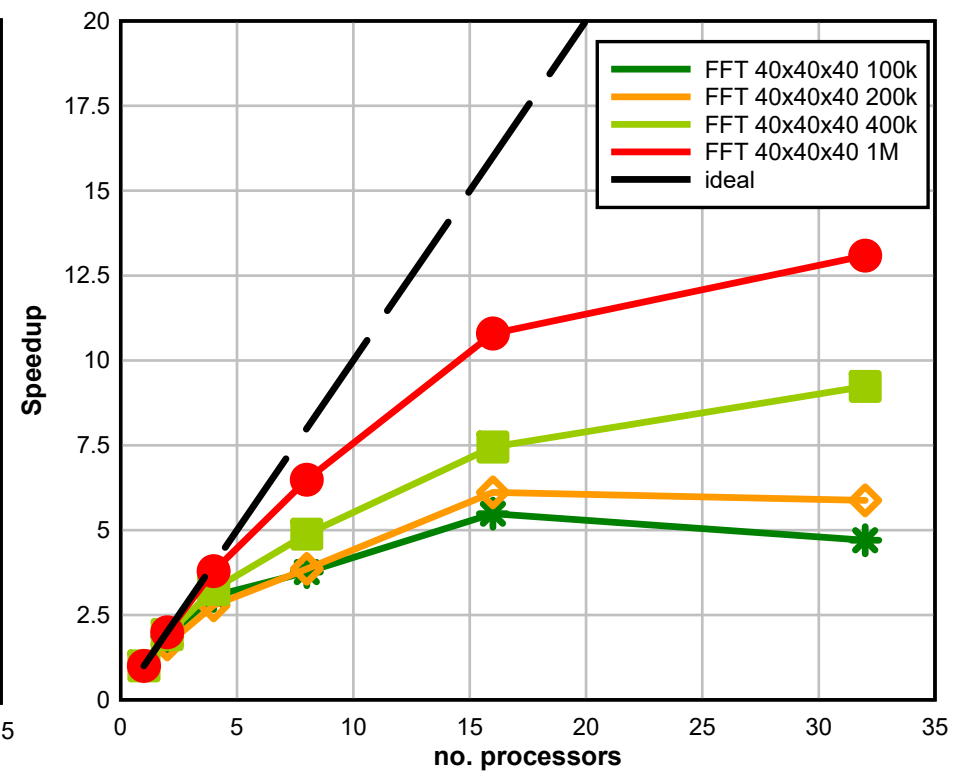
- TEMF Wakefield-cluster (Windows)

Distributed memory scaling

FFT + AB4 – run time



FFT + AB4 – speedup

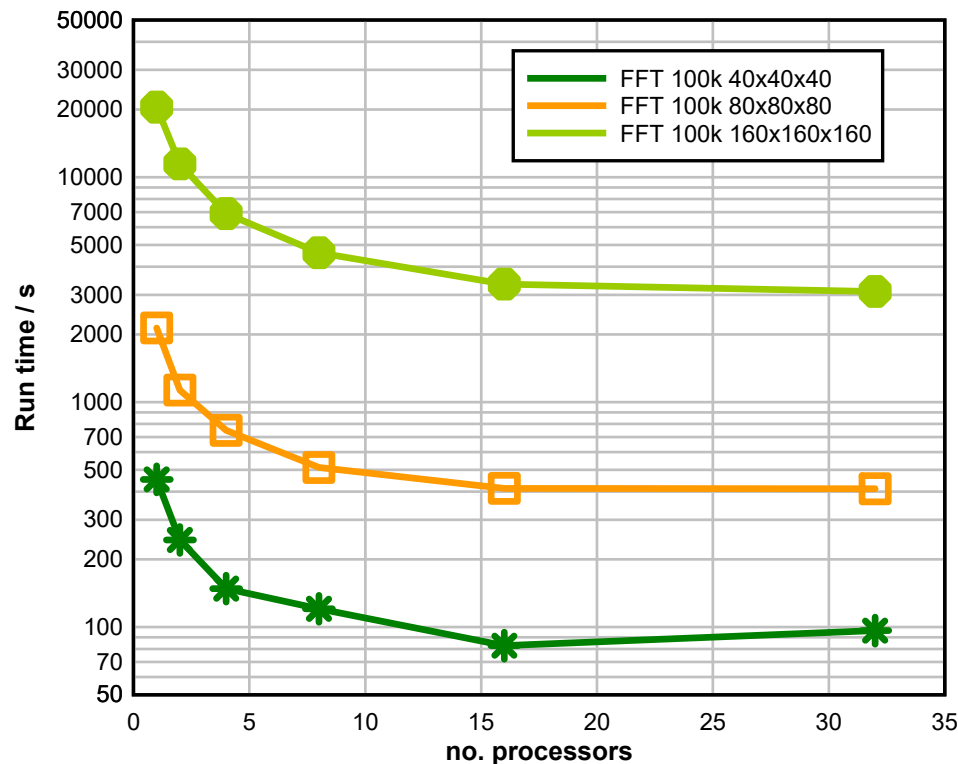


- TEMF Wakefield-cluster (Windows)

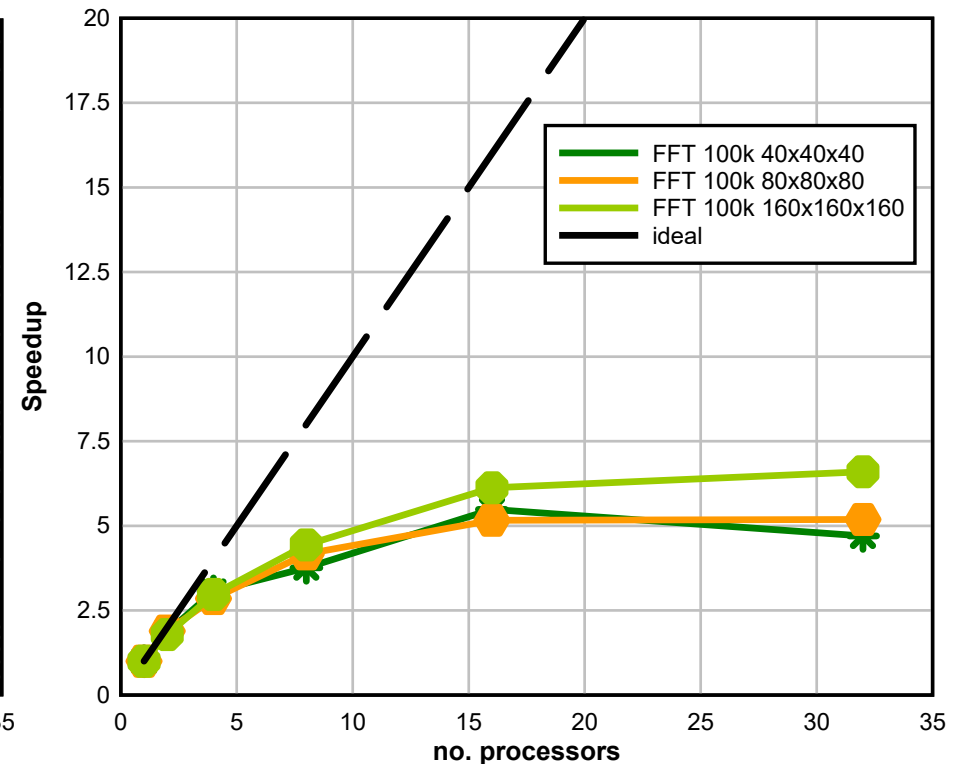


Distributed memory scaling

FFT + AB4 – run time



FFT + AB4 – speedup



- TEMF Wakefield-cluster (Windows)

Conclusions & Outlook

- Reptil offers a variety of tracking/solver techniques
- Excellent agreement of the FFT-solver with Astra
- More than an order of magnitude faster than Astra
- Even more speedup on parallel platforms
- Simulations with “real electrons” are quite feasible/easy
- More work is needed:
 - Particle noise / smoothing in the PFMM-solver
 - Improve parallelization efficiency
 - FFT-solver with CUDA backend
 - Hierarchic (tree-based) LW-solver
 - Wakefields and field maps
 - ...

