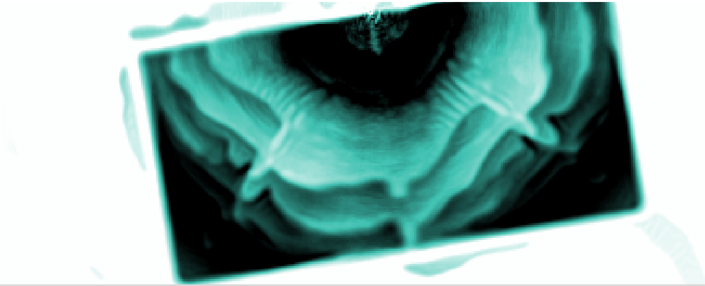
A Parallelized Vlasov-Fokker-Planck Solver for Desktop PCs

Patrik Schönfeldt | 8th of February 2016

Institute for Photon Science and Synchrotron Radiation

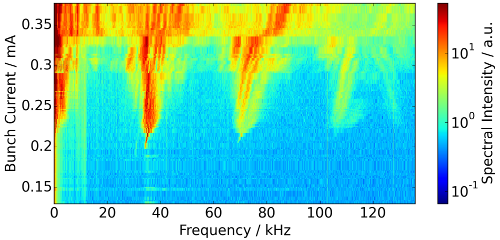


Motivation

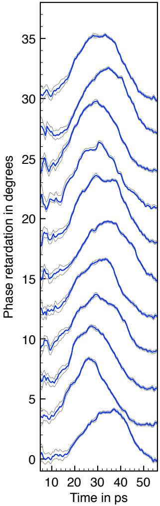
- Multitude of bursting measurements for different machine settings
- Map dynamics, for all the settings
- Nonlinear dynamics explains
  - Coherent synchrotron radiation
  - Above threshold substructures self-amplify
  - Microbunches in longitudinal phase space
- Vlasov-Fokker-Planck solvers well tested
- Speed up to use Desktop PCs

Motivation

- Multitude of bursting measurements for different machine settings
- Map dynamics, for all the settings
- Nonlinear dynamics explains
  - Coherent synchrotron radiation
  - Above threshold substructures self-amplify
  - Microbunches in longitudinal phase space
- Vlasov-Fokker-Planck solvers well tested
- Speed up to use Desktop PCs



[M. Brosi et al.]



[N. Hiller et al.]

Vlasov-Fokker-Planck Equation

$$\frac{\partial \psi}{\partial t} + \frac{\partial H}{\partial p}\frac{\partial \psi}{\partial q} - \frac{\partial H}{\partial q}\frac{\partial \psi}{\partial p} = \frac{2}{\tau_D}\left(\frac{\partial}{\partial p}(p\psi) + \frac{\partial^2 \psi}{\partial p^2}\right)$$

describes longitudinal beam dynamics

$\psi(q, p, t)$: Charge density
t: Time
p: Energy coordinate
q: Space coordinate
τ_D: Damping time

Hamiltonian

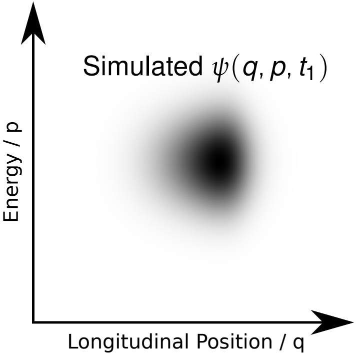
$$H(q,p,t) = \underbrace{H_e(q,p,t)}_{\text{external fields}} + \underbrace{H_c(q,t)}_{\text{collective effects}}$$
$$= \frac{1}{2}\left(q^2 + p^2\right) + Q_c \times V_c(Z_c, q, t)$$

Q_c: Charge
V_c: Potential
Z_c: Impedance

Vlasov-Fokker-Planck-Solvers

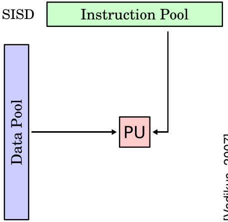
$$\frac{\partial \psi}{\partial t} + \frac{\partial H}{\partial p}\frac{\partial \psi}{\partial q} - \frac{\partial H}{\partial q}\frac{\partial \psi}{\partial p} = \frac{2}{\tau_D}\left(\frac{\partial}{\partial p}(p\psi) + \frac{\partial^2 \psi}{\partial p^2}\right)$$

- Well established algorithm by Warnock and Ellison (2000)
- For small time steps solve sequentially:
  1. LHS: Harmonic Oscillator (Rotation)
  2. RHS: Damping and Diffusion
  3. Perturbation due to wake field
- Direct implementation:
  One run takes days
- Parallelize and optimize each substep



Parallelization

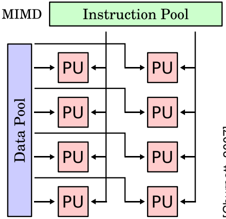
Typical Single-Core Workload:



[Vadikus, 2007]

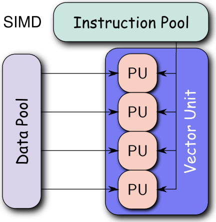
- Sequential programming: One computation at a time
- Parallel programming:
  - Multiple Data and/or
  - Multiple Instructions
- New/extended programming language needed

Typical Multi-Core Workload:



[Cburnett, 2007]

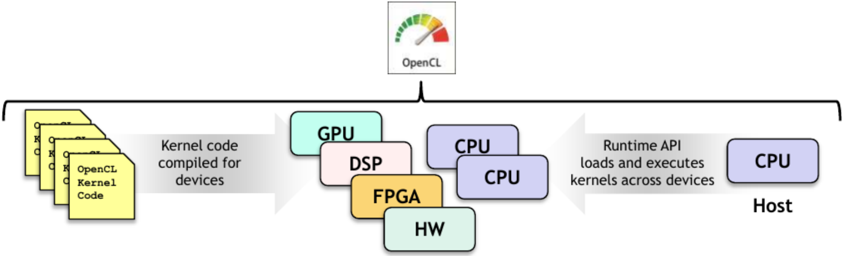
Typical GPU Workload:



[Cburnett, 2015]

OpenCL

- C-library for parallel programming
- High level interface to query, select, and initialize devices
- Abstract handling of devices (like GPU, Multi-Core CPU)
- OpenCL C language for parallel code, automatic adaption
- Same code runs on any supported hardware



[Khronos Group, 2015]

Optimize three steps of
Vlasov-Fokker-Planck Solver

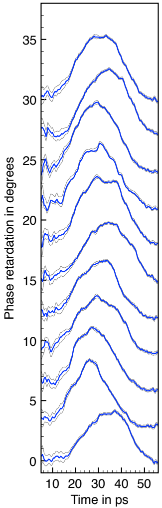
1. Rotation
2. Damping and Diffusion
3. Perturbation due to wake field

- Reduce computation time

Optimize three steps of
Vlasov-Fokker-Planck Solver

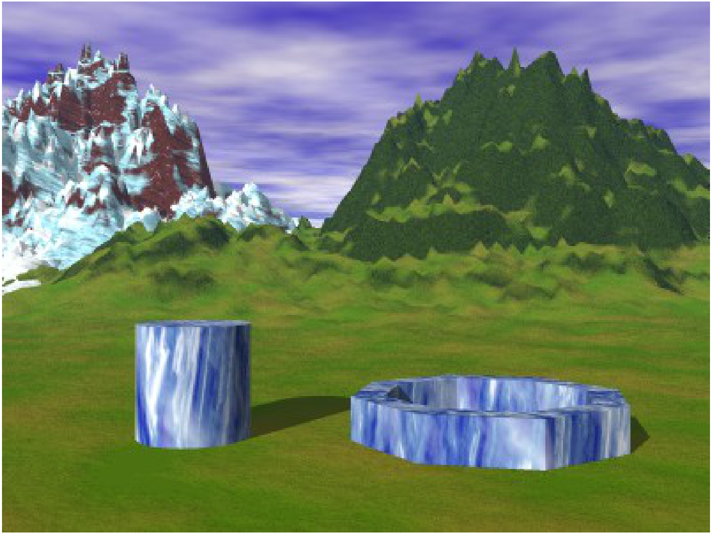
1. Rotation
2. Damping and Diffusion
3. Perturbation due to wake field

- Reduce computation time
- Avoid numerical artifacts



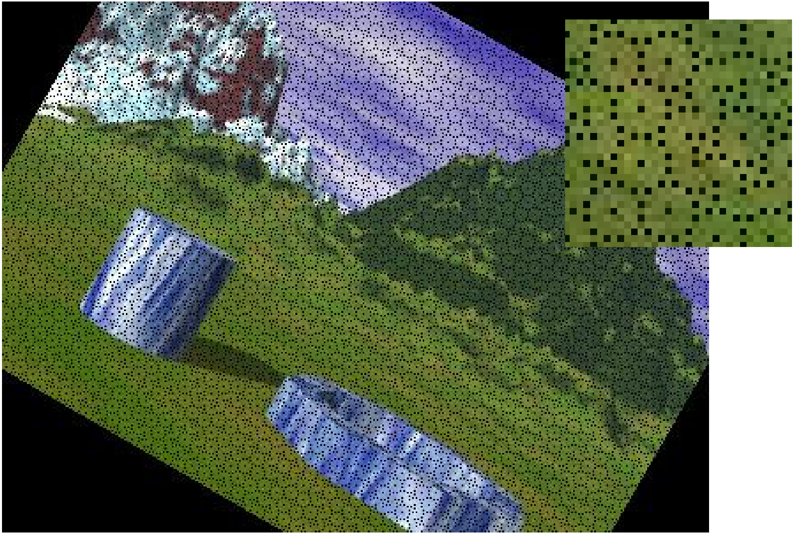
[N. Hiller et al.]

Rotation



[http://polymathprogrammer.com/2008/10/06/image-rotation-with-bilinear-interpolation/]

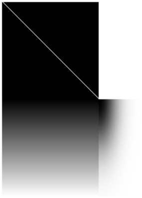
Rotation – Rotation Matrix and Rounding



[http://polymathprogrammer.com/2008/10/06/image-rotation-with-bilinear-interpolation/]

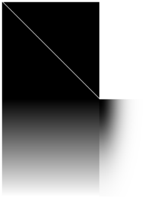
Rotation – Linear Interpolation

Test Pattern

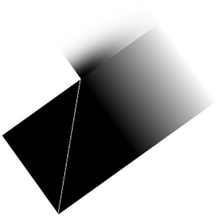


Rotation – Linear Interpolation

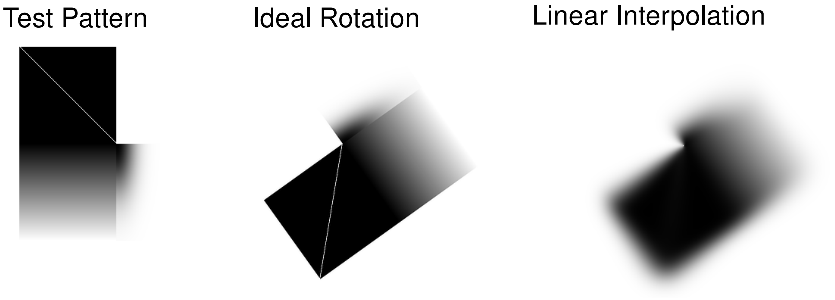
Test Pattern



Ideal Rotation

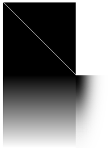


Rotation – Linear Interpolation

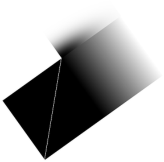


Rotation – Quadratic/Cubic Interpolation

Test Pattern

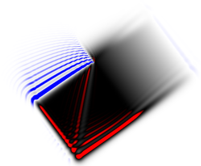


Ideal Rotation

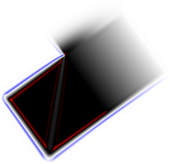


- Higher order interpolation polynomials
- Less Diffusion
- Added Structures
- Highlighted: Values exceed range of test pattern

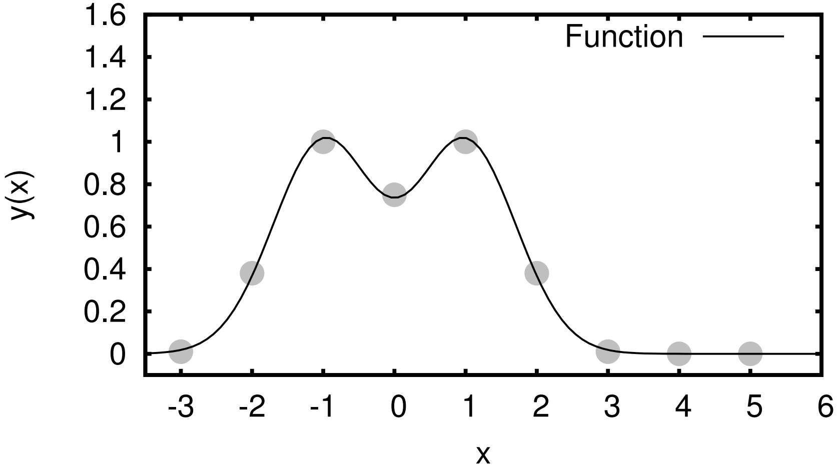
Quadratic Interpolation



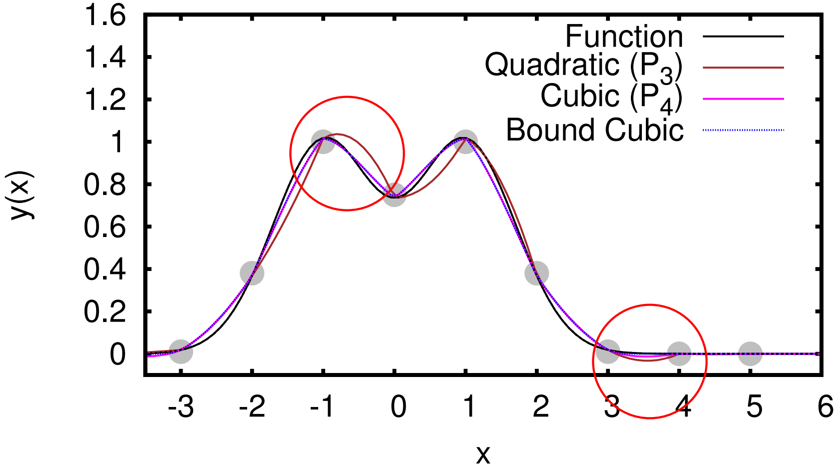
Cubic Interpolation



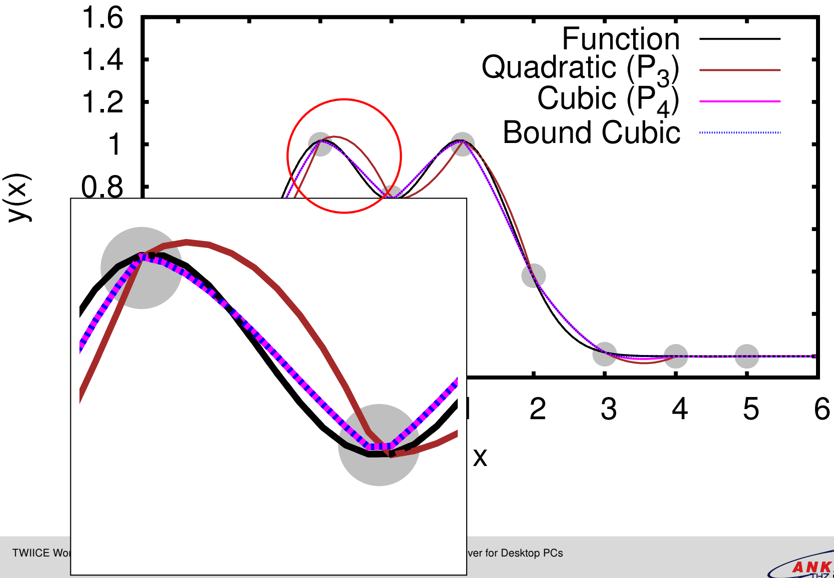
Over-/ Undershooting



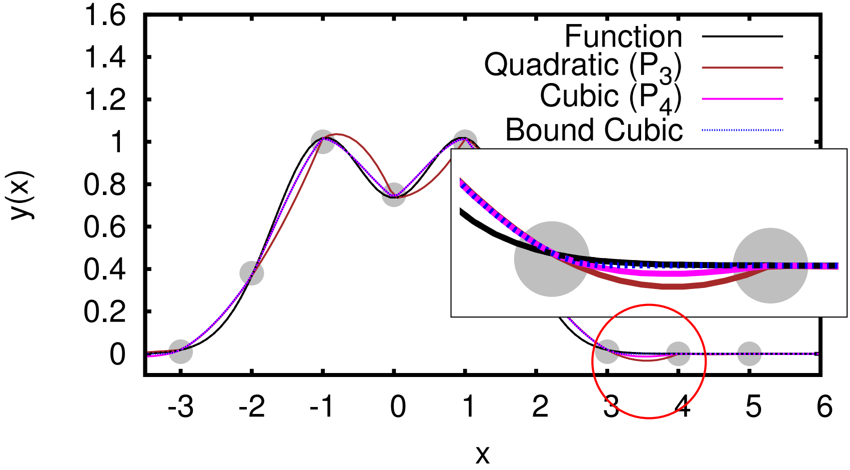
Over-/ Undershooting



Over-/ Undershooting



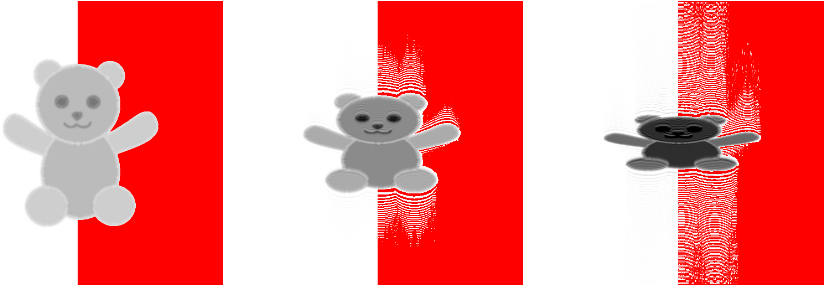
Over-/ Undershooting



Damping

- Damping reduces vertical size
- Integral is preserved
- Red background to improve contrast

Damping



$$\left.\frac{\partial f(x)}{\partial x}\right|_{x=x_0} \approx \frac{f(x_0+\Delta x) - f(x_0-\Delta x)}{2\Delta x} \left(= \left.\frac{\partial P_3(x)}{\partial x}\right|_{x=x_0} \right)$$

Damping

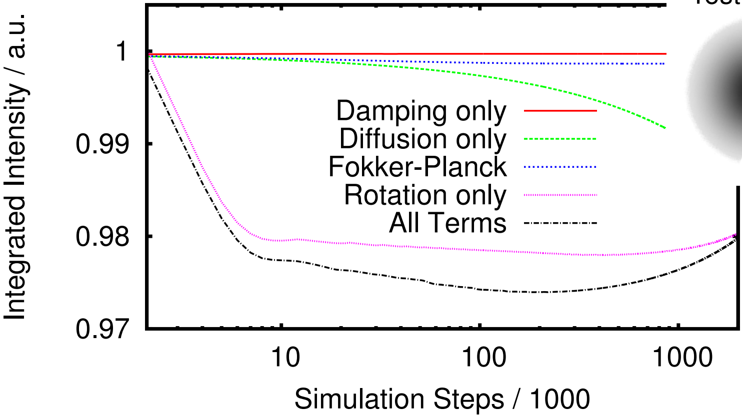
$$\left.\frac{\partial f(x)}{\partial x}\right|_{x=x_0} \approx \frac{-2f(x_0-\Delta x)-3f(x_0)+6f(x_0+\Delta x)-f(x_0+2\Delta x)}{6\Delta x}$$

$$\left(= \left.\frac{\partial P_4(x)}{\partial x}\right|_{x=x_0} \right)$$

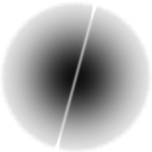
Diffusion

$$\left.\frac{\partial^2 f(x)}{\partial x^2}\right|_{x=x_0} \approx \frac{f(x_0 - \Delta x) - 2f(x_0) + f(x_0 + \Delta x)}{(\Delta x)^2}$$

Stability of Different Terms



Test pattern

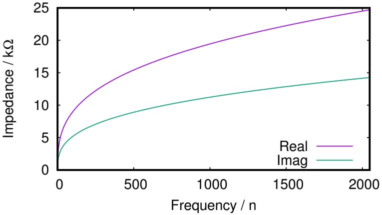


- Interpolation leads to losses at sharp edges
- Numerically stable state after 10 steps

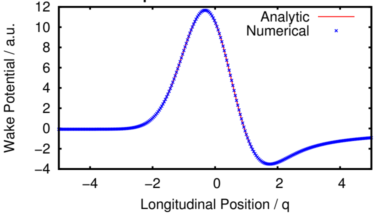
Wake Potential – Frequency Domain

$$V_c(q,t) = \Re\left(\int_0^\infty \mathrm{d}k\, Z(k) \int_{-\infty}^{\infty} \mathrm{d}q' \rho(q',t) e^{-ikq'} e^{ikq} \right)$$

[Khan, 2006]



Example: Gaussian bunch



- Impedances known analytically
- All functions continuous
- Fourier transform algorithms are optimized

Benchmark: Convergence

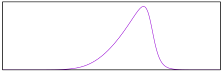
- Arbitrary pattern
- Charge below instability threshold
- Expected to find physical equilibrium

Benchmark: Convergence

- Arbitrary pattern
- Charge below instability threshold
- Expected to find physical equilibrium

Benchmark: Convergence

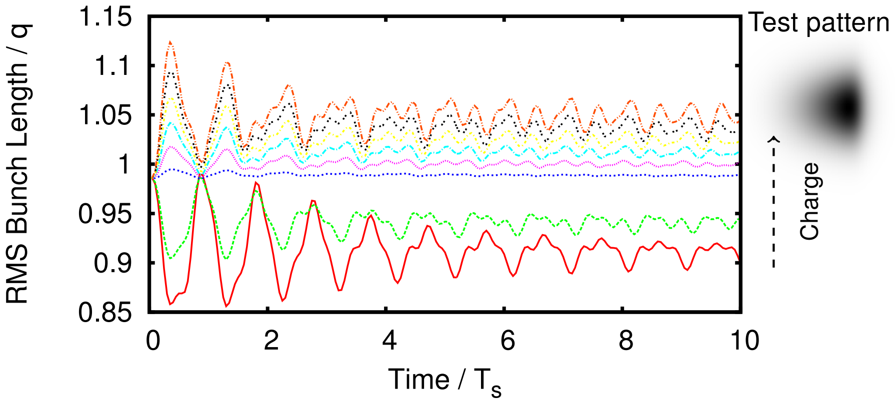
- Arbitrary pattern
- Charge below instability threshold
- Expected to find physical equilibrium



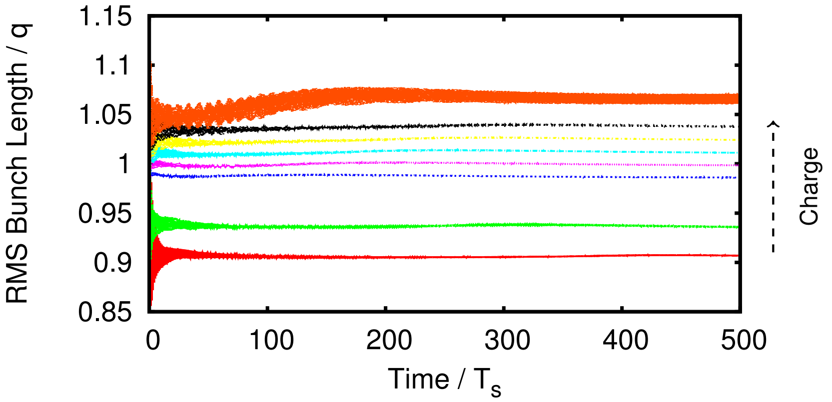
Haissinsky Distribution

Benchmark: Convergence

- Same starting distribution
- Varied bunch charge
  $\Rightarrow$ different steady state



Benchmark: Convergence



- Varied bunch charge
  $\Rightarrow$ different steady state ✓
- Microbunching not simulated
  (no seed applied)

Benchmark: Computation Times

Benchmark Scenario

- 500 synchrotron periods
- 512×512 mesh points
- 4000 steps per period

CPU-time for different tasks

- 59% Synchrotron Oscillation
- 23% Collective Force
- 18% Damping and Diffusion

Non-optimized program: 222h53’ (> 9 days)

Benchmark: Computation Times

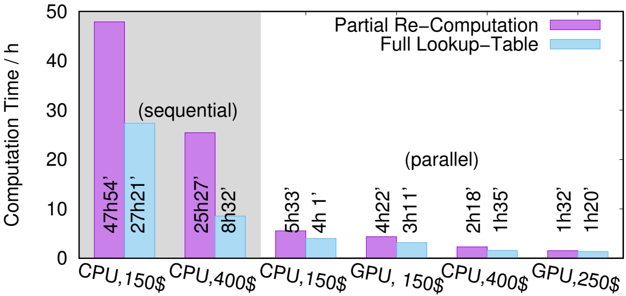
Benchmark Scenario

- 500 synchrotron periods
- 512×512 mesh points
- 4000 steps per period

CPU-time for different tasks

- 59% Synchrotron Oscillation
- 23% Collective Force
- 18% Damping and Diffusion

Non-optimized program: 222h53’ (> 9 days)



Acknowledgements

- **KIT THz-Team** (from IBPT, IMS, IPE, IPS and LAS):
  M. Balzer, E. Blomley, A. Borysenko, M. Brosi, E. Bründermann, M. Caselle, C. Chang*, N. Hiller, S. Höninger, M. Hofherr, E. Huttel, K.S. Ilin, V. Judin*, B. Kehrer, M. Klein*, S. Marsching, Y.-L. Mathis, M.J. Nasse, G. Niehues, A. Plech, J. Raasch, P. Rieger*, L. Rota, R. Ruprecht, M. Schedler, A. Scheuring, P. Schönfeldt, M. Schuh, P. Schütze*, M. Schwarz, M. Siegel, N.J. Smale, B. Smit, J. Steinmann, P. Thoma*, S. Walter, M. Weber, S. Wuensch, M. Yan, and A.-S. Müller
  **THz-Alumni*
- **For interesting discussions, good ideas and a lot of fun:**
  F. Caspers (CERN), S. Khan (DELTA), P. Peier, B. Steffen (DESY), H.-W. Hübers, A. Semenov (DLR), P. Kuske, G. Wüstefeld (HZB), V. Schlott (PSI) ,Y. Cai, J. Corbett, R. Warnock (SLAC), S. Bielawski, C. Evain, E. Roussel, C. Szwaj (U. Lille)

SPONSORED BY THE Federal Ministry of Education and Research

Summary

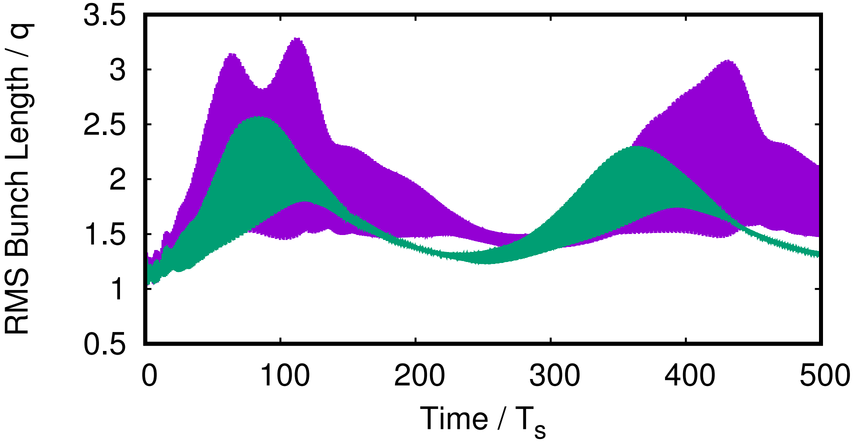
- Individually optimized terms to reduce artifacts
- Speedup of more than factor 150 (80 minutes instead of 9 days)
- Tested with free space CSR impedance
- Microbunching not simulated (no seed applied)
- Should be applicable for arbitrary impedances

Outlook

- Use other impedances than free space CSR
- Implement a seeding mechanism
- Maybe: Improve program latency/waiting times

Backup

Simulated Non-Equilibrium



Needed Accuracy