A PARALLELIZED VLASOV-FOKKER-PLANCK-SOLVER FOR DESKTOP PCS

P. Schönfeldt*, M. Brosi, J. L. Steinmann, and A.-S. Müller
Karlsruhe Institute of Technology , Karlsruhe, Germany

Abstract

In order to simulate the dynamics of an electron bunch due to the self-interaction with its own coherent synchrotron radiation it is a well established method to numerically solve the Vlasov-Fokker-Planck equation. In this paper we present a modularly extensible program that uses OpenCL to massively parallelize the computation, allowing a standard desktop PC to work with appropriate accuracy and yield reliable results within minutes. We provide numerical stability studies over a wide parameter range and compare our numerical findings to known results.

INTRODUCTION

Improvements in the online study of the micro-bunching instability [1] created a demand for fast simulations that are able to map the dynamics of micro-bunching over a wide range of physical parameters. Also the simulation tool itself should be well understood so that the influences of both the simulated physics and of numerical effects can be studied and separated.

The phenomenon of micro-bunching happens in the longitudinal phase space which is spanned by the position z and the energy E. Taking the particle density $\psi(z,E)$ of electrons in a storage ring to be a smooth function, its evolution can be described by the Vlasov-Fokker-Planck equation. It reads

$$\frac{\partial\psi}{\partial t}+\frac{\partial H}{\partial p}\frac{\partial\psi}{\partial q}-\frac{\partial H}{\partial q}\frac{\partial\psi}{\partial p}=\frac{2}{\tau_d}\frac{\partial}{\partial p}\left(p\psi+\frac{\partial\psi}{\partial p}\right), \quad (1)$$

with the generalized coordinates $q = z/\sigma_{z,0}$, and $p = (E-E_0)/\sigma_{\delta,0}$, the Hamiltonian H, the reference particle's energy E_0, and the damping time τ_d. The quantities $\sigma_{\delta,0}$ and $\sigma_{z,0}$ describe energy spread and bunch length, both in the equilibrium state that exists for small bunch charges.

To solve this partial differential equation there is a widely used formalism by Warnock and Ellison [2]. It uses a grid to discretize $\psi(q,p)$ and assumes that the collective force due to self-interaction with the bunch's own coherent synchrotron radiation is constant for small time steps. The perturbation due to the collective effects is described as a perturbation to the Hamiltonian

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

where Q_c is the charge involved in the perturbation, and V_c is the potential due to the collective effect, which can be expressed in terms of an impedance Z_c.

It is then possible to use the homogeneous solution, which in unperturbed case is represented by a rotation in phase space, and add the influence of diffusion and damping as a particular solution. To model the perturbation, the influence of V_c is applied as a 'kick' along the energy axis.

IMPLEMENTATION

The program discussed in this paper splits the direct implementation of each of the simulation steps $f:(q,p)\rightarrow(q',p')$ that compose one time step (rotation, kick, damping and diffusion) into two steps. This method – by construction – produces the same results as the single-step implementation. All information on the actual coordinates (q,p) is only needed by the first half simulation step. We call its result a 'source map' (SM). In the second half step one dimension of the original transformation $f:\mathbb{R}^2\rightarrow\mathbb{R}^2$ is no longer needed, yielding $f_{SM}:\mathbb{R}^1\rightarrow\mathbb{R}^1$. Practically speaking, this map holds the information which data of the current simulation step contributes to a grid point for the next simulation step. For many functions – such as rotation – the SM will look the same for the whole runtime of the program. For that reason it only has to be computed once and can be kept for multiple usages.

The source map formalism does not only allow to keep intermediate results, it also gives a handy interface to implement arbitrary functions that act on the phase space. Furthermore the reduction of the problem's dimension may also lead to a speed up. Results of a benchmark of the computational performance are shown in Fig. 1.

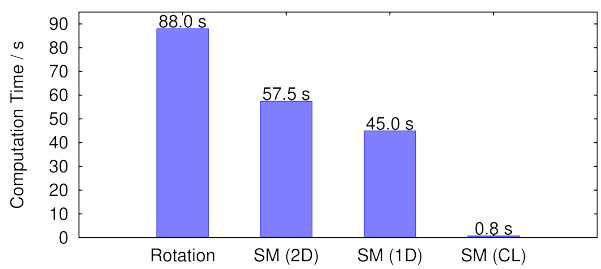

Figure 1: Computational time needed by an Intel Core i5 4258U for 1000 cubically interpolated rotation steps using different implementations. In this test case the grid has 512 points per axis, no optimizations (besides SM) are used.

In this case source mapping alone (SM 2D) brings a speedup of one third, while reducing the dimension (SM 1D)

* patrik.schoenfeldt@kit.edu

in total halves the runtime. A further speedup is achieved by using parallelization (OpenCL [3]), which we implemented for the abstract SM (SM CL). A second advantage of OpenCL is that it can utilize not only multi-core CPUs but also graphic processors. In total, a non-optimized program takes more than 9 days for a typical simulation run. Using the method described above, it can be reduced to 80 minutes – using a customer grade graphics card.

NUMERICAL ARTIFACTS

There are different contributions to numerical inaccuracies. For real numerical artifacts we identified two main sources: interpolation and numerical derivatives. As an example we take a current distribution that for a simulation run that uses cubic interpolation stays in a pseudo-stable state, which, however, exists above the expected bursting threshold (at $S_{CSR} = 0.5$ [4]). The complete set of simulation parameters is listed in Table 1.

Table 1: Parameters for an example run to check for numerical artifacts. For the given set no artifacts were observed. Changing to quadratic interpolation however, triggered the occurrence of (non-physical) structures with a period length of one grid cell.

Parameter	Value
Grid points per axis	256
Steps per T_s	4000
Interpolation method	cubic
Impedance model	free space
CSR strength S_{CSR}	0.516
Damping time	200 T_s

We also run a simulation with the same parameters except that quadratic interpolation is applied. If there was no influence of the different interpolation schemes, one would expect no difference between the two runs. However, as Fig. 2 depicts, this is not guaranteed.

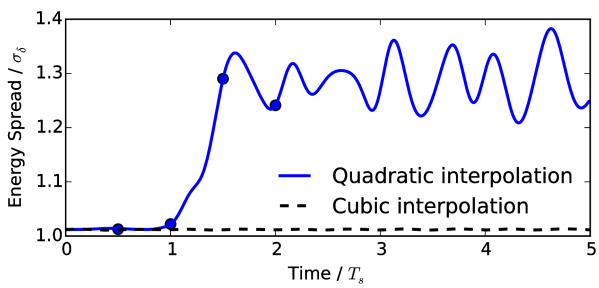

Figure 2: Evolution of the energy spread over time. For the case where cubic interpolation was used (dashed black line) the energy spread stays at $\sigma_E \approx 1\sigma_{\delta,0}$. When using quadratic interpolation (solid blue line) an increase can be observed at $T = 1\ T_s$.

The energy spread simulated using quadratic interpolation rapidly increases to a higher value after one synchrotron period ($T = 1\ T_s$). Such a behavior can be interpreted as evidence that the simulated conditions are above the micro-bunching threshold [4]. However, we want to track the evolution of the charge distribution, so we have to avoid numerical artifacts as they occur in the run using quadratic interpolation (depicted in Fig. 3): The period length of the ripples is exactly two grid cells, and they continue to exist even to a point where they create negative charge densities. In this particular case, the instability is clearly triggered by the artifacts (overshoots) of the interpolation.

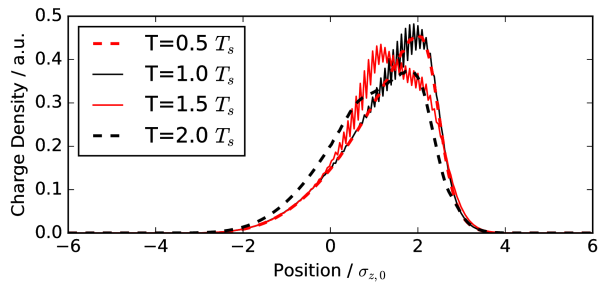

Figure 3: Bunch profiles of the simulation run using quadratic interpolation at selected points in time (cf. blue discs in Fig. 2). The bunch profiles computed during the initial increase of the energy spread (solid lines) show ripples with a period length of two grid cells. The earlier and later profiles (dashed lines) do not show such structures. This implies that the increase of energy spread is driven by a numerical instability.

For the numerical differentiation the same type of artifacts can occur. This can be explained by the fact that the algorithm usually used for numerical differentiation [5] is equivalent to differentiating the quadratic interpolation polynomial P_2

$$\left.\frac{\partial f(x)}{\partial x}\right|_{x_0} \approx \frac{f(x_0+\Delta x) - f(x_0-\Delta x)}{2\Delta x} = \left.\frac{\partial P_2(x)}{\partial x}\right|_{x_0}.$$

As a consequence, we target the problem by using the cubic interpolation polynomial P_3, and obtain

$$\left.\frac{\partial f(x)}{\partial x}\right|_{x_0} \approx \frac{-2f(x_0-\Delta x) - 3f(x_0)}{6\Delta x} + \frac{6f(x_0+\Delta x) - f(x_0+2\Delta x)}{6\Delta x} = \left.\frac{\partial P_3(x)}{\partial x}\right|_{x_0}. \quad (3)$$

In our tests we did not find any case where a more complex differentiation method than the one described above was needed to avoid artifacts. But note that also higher order polynomials show overshoots. To completely rule out this source of artifacts, one has to use clamped or saturated interpolation functions. In contrast to the differentiation, there were rare cases where we observed rotation artifacts that we had to suppress by clamping – which restricts interpolation to the range of the neighboring values.

ISBN 978-3-95450-147-2

CONVERGENCE STUDIES

In this section we compare the effect of different numerical settings, which – in an ideal world – would not effect the physical result. We also compare different implementations of the necessary simulation steps to solve Eq. 1. For example, it is possible to implement the rotation in phase space or to implement it as an energy-dependent drift together with a location dependent RF kick. In this paper the second approach will be referred to as 'Manhattan rotation'.

For the sake of simplicity we go back to the unperturbed case (meaning $H_c = 0$ in Eq. 2). So any starting distribution should converge to a normal distribution with $\sigma_q = \sigma_p = 1$.

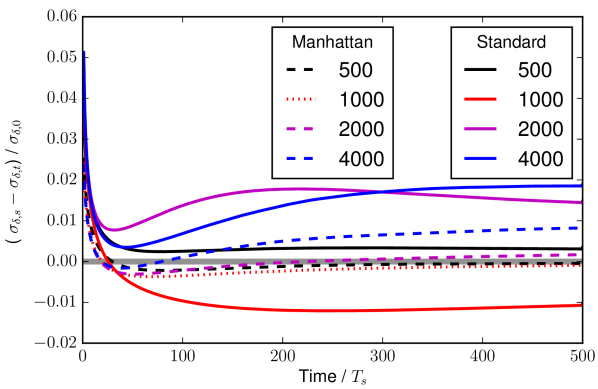

Figure 4: Damping of the RMS energy spread for different numbers of computation steps per synchrotron period. Every color represents a different number of simulation steps per synchrotron period. In absolute numbers all simulations deliver strictly monotonic functions, the (common) initial offset is explained by an imperfect starting distribution. To make the differences more clear, the theoretical behavior ($\sigma_{\delta,t} = A \times \exp(-t/\tau) + \sigma_{\delta,0}$) has been subtracted from the simulated one ($\sigma_{\delta,s}$). Manhattan rotation (dashed lines) reproduces the set values ($\tau = 150\,T_s$, solid grey line) better than standard rotation (other solid lines). Also note that the error increases when the number of steps per synchrotron period becomes too big.

In one example series of tests we observe the evolution of a normal distributed charge density with $\sigma_q = \sigma_p = 2.8$ where the damping time is set to $\tau = 150\,T_s$. The Number of simulation steps per T_s is varied between 500 and 4000. As depicted in Fig. 4 for all settings a clear exponential damping is observed. For Manhattan rotation the error on the reconstructed values is significantly (maximum 1%, usually $\ll 1\%$) lower than for standard rotation (1% up to 2.5%). Furthermore Manhattan rotation is more robust against changing step sizes.

In another example we study the evolution of a normal distributed charge density with $\sigma_q = \sigma_p = 1$. From the physics point of view it is expected that the distribution stays constant with time. What can be observed (see Fig. 5), however, is that σ_p converges to different values depending on the numerical parameters. Especially for standard rotation,

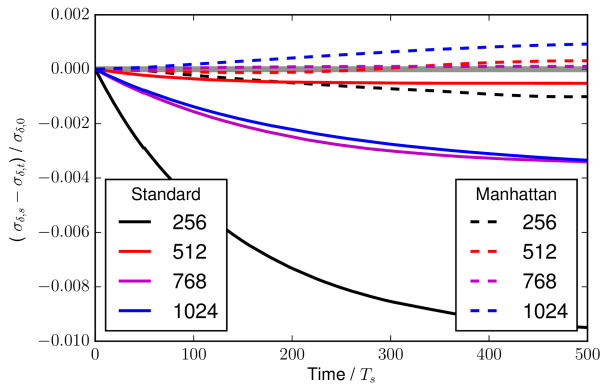

Figure 5: Evolution of the RMS energy spread for different grid sizes. Manhattan rotation (dashed lines) reproduces the expected value ($\sigma_{\delta,t} = \sigma_{\delta,0}$, solid grey line) better than standard rotation (other solid lines).

increasing the grid size first makes the result more accurate until an optimum is reached. Further increasing the grid size leads to an increased difference from the expected value. Again, as for different step sizes, Manhattan rotation produces the better and more reliable results also in this case.

SUMMARY

We introduced a Vlasov-Fokker-Planck-solver that uses OpenCL for parallelized computation. Doing this it can simulate the dynamics of the longitudinal phase space using desktop PCs more than 100 times faster than a non-optimized implementation, or even 150 times faster when using a dedicated (customer class) graphics card. Furthermore, we eliminated sources of numerical artifacts and have done numerical stability studies to show that relative errors can be reliably kept clearly below 1%.

ACKNOWLEDGMENTS

The authors thank G. Stupakov for his suggestion to investigate numerical effects of the 'Manhattan rotation', and M. Klein for giving us her code as a reference. M. Brosi, P. Schönfeldt, and J. L. Steinmann want to acknowledge the support by the Helmholtz International Research School for Teratronics (HIRST).

REFERENCES

[1] M. Brosi, M. Caselle, E. Hertle, N. Hiller, A. Kopmann, P. Schönfeldt, and A.-S. Müller, "Online studies of THz- radiation in the bursting regime at ANKA", in *proc of IPAC'15*, Richmond, VA, paper MOPHA042.

[2] R. L. Warnock and J. A. Ellison, "A general method for propagation of the phase space distribution, with application to the sawtooth instability", SLAC, Stanford, California, SLAC-PUB-8404, 2000.

[3] OpenCL website. https://www.khronos.org/opencl/.

[4] K. L. F. Bane, Y. Cai, and G. Stupakov, "Threshold studies of the microwave instability in electron storage rings", in *Phys. Rev. ST Accel. Beams*, vol. 13, p. 104402, 2010.

[5] William H. Press *et al.*, *Numerical Recipes 3rd Edition: The Art of Scientific Computing*, Cambridge University Press, New York, NY, USA, 3rd edition, 2007.

ISBN 978-3-95450-147-2