EFFICIENT 3D SPACE CHARGE CALCULATIONS WITH ADAPTIVE DISCRETIZATION BASED ON MULTIGRID*

G. Pöplau†, U. van Rienen, Rostock University, Rostock, Germany

Abstract

Precise and fast 3D space-charge calculations for bunches of charged particles are still of growing importance in recent accelerator designs. A widespread approach is the particle-mesh method computing the potential of a bunch in the rest frame by means of Poisson's equation.

An adaptive discretization following the particle density distribution is implemented in the GPT tracking code together with a multigrid Poisson solver. The disadvantage of this approach is that the adaptivity is not achieved by a scheme that directly 'fits' into the multigrid algorithm.

In this paper we investigate a new approach to an adaptive discretization: the self-adaptive multigrid algorithm. The goal is that the adaptive mesh is constructed hierarchically, where the distribution of mesh lines is calculated by an error estimator. The algorithm will be investigated for ellipsoidal particle distributions of different lengths and compared to the adaptive GPT mesh. Furthermore, the performance of both adaptive discretization schemes will be tested with simulations for the rf gun of the European XFEL.

INTRODUCTION

The simulation of the dynamics of high-brightness charged particle bunches demand the fast calculation of 3D non-linear space charge fields with an accuracy that matches the quality of the bunch. The particle-mesh method is a widespread model for space charge calculations. Here, adaptive discretization techniques are often required in order to satisfy both computational demands: accuracy and fast performance. Nevertheless, adaptive discretizations are implemented only in a few software packages together with space charge calculations. For instance, the FFT Poisson solver that is often applied allows only an equidistant mesh. The GPT code (General Particle Tracer) sets the mesh lines dynamically based on the charge density [1], a substantial improvement over equidistant meshes. The Poisson solver MOEVE (**M**ultigrid for n**o**n-**e**quidistant grids to sol**v**e Poisson's **e**quation) is an efficient implementation that adapts multigrid for such non-equidistant meshes [2].

In this paper we aim to set the next step in optimizing the grid creation process for our multigrid Poisson solver MOEVE. We here propose to use the multigrid procedure itself to assign the optimal mesh line positions. This self-adaptive multigrid method was firstly described and investigated for space charge calculations in [3]. Recently this approach has been implemented in the tracking code GPT. With numerical test cases we compare these two adaptive discretization schemes: the adaptive GPT mesh and the self-adaptive multigrid algorithm. The numerical error and CPU time are investigated for ellipsoidal particle bunches of different lengths. Finally, the performance of both adaptive discretizations is tested with simulations for the rf gun of the European XFEL.

* Work supported by DFG under contract number RI 814/18-1
† gisela.poeplau@uni-rostock.de

PARTICLE-MESH MODEL IN GPT

In the tracking code GPT several space charge models are implemented [1]. The 3D model we consider here is based on the particle-mesh method (see [4] and citations therein). Hereby the bunch is modelled as a certain distribution of macro particles. All fields are calculated in the electrostatic approximation in the rest frame of the bunch, implicitly assuming only a few percent energy spread. After the transformation into the rest frame a mesh is constructed around the bunch and the charge of the particles is assigned to the mesh points. Now, the potential φ can be obtained from Poisson's equation given by

$$\begin{aligned} -\Delta\varphi &= \frac{\varrho}{\varepsilon_0} && \text{in } \Omega \subset \mathbb{R}^3, \\ \varphi &= 0 && \text{on } \partial\Omega_1, \\ \frac{\partial\varphi}{\partial n} + \frac{1}{r}\varphi &= 0 && \text{on } \partial\Omega_2, \end{aligned} \tag{1}$$

where ϱ the space charge distribution, ε_0 the dielectric constant and r the distance between the centre of the bunch and the boundary. Usually, the domain Ω is a rectangular box constructed around the bunch. On the surface $\partial\Omega = \partial\Omega_1 \cup \partial\Omega_2$ ($\partial\Omega_1 \cap \partial\Omega_2 = \emptyset$) perfectly conducting boundaries ($\partial\Omega_1$) or open boundaries ($\partial\Omega_2$) can be applied.

For the solution of the Poisson equation we applied the discretization with second order finite differences. This leads to a linear system of equations of the form

$$L_h u_h = f_h, \tag{2}$$

where u_h denotes the vector of the unknown values of the potential and f_h the vector of the given space charge density at the grid points. The step size h indicates a certain refinement level and the operator L_h is the discretization of the Laplacian.

ADAPTIVE MESHING

The Adaptive GPT Mesh

The adaptive GPT mesh (first in release 2.7) is an adaptive discretization that sets the mesh lines dynamically due

to the charge density in the bunch [1]. The number of mesh lines is chosen accordingly to the number and the distribution of the particles, respectively. Efficient space charge calculations can be performed with the MOEVE Poisson solver that has been constructed especially for such non-equidistant meshes. The adaptive GPT mesh is very reliable but the construction is very complex. For example it has to be ensured that neighboring step sizes are differ not more than a certain factor in order to ensure the convergence of the multigrid Poisson solver [2, 4].

The Self-Adaptive Multigrid Mesh

The self-adaptive multigrid algorithm is a new approach for the construction of adaptive discretizations for space charge calculations, recently implemented in the software package MOEVE. This approach starts with a relatively coarse grid and refines it according to a certain criterion. In MOEVE, the τ-criterion is applied (see [3] and citations therein). Now, this algorithm has been implemented in GPT.

In order to give a short overview of the method we have to introduce some notations. The step sizes h and $2h$ refer to the step sizes on the fine and the next coarser grid (usually with double mesh size), respectively. The operators I_h^{2h} and $\widehat{I}_h^{2h}$ denote different restriction operators. For the numerical tests of the next section the injection was chosen for $\widehat{I}_h^{2h}$ and the full weighting restriction for I_h^{2h}. The τ-criterion is based on the so-called (h,$2h$) relative truncation error τ_h^{2h} with respect to the restriction operators I_h^{2h} and $\widehat{I}_h^{2h}$. It is defined by

$$\tau_h^{2h} := L_{2h}\widehat{I}_h^{2h}u_h - I_h^{2h}L_h u_h \,. \qquad (3)$$

By means of the refinement criterion a hierarchy of locally refined grids can be generated. The self-adaptive multigrid scheme is given as follows:

Algorithm: Self-Adaptive Multigrid

1. Start on a relatively coarse mesh.
2. Perform a few multigrid cycles on equation (2).
3. Calculate τ_h^{2h}.
4. Add mesh lines locally, where $|\tau_h^{2h}| > \varepsilon$.
5. Proceed from 2. as long as $|\tau_h^{2h}| > \varepsilon$.

Main advantages of this approach are that the generated hierarchy of meshes now matches the hierarchy of meshes of multigrid and the values τ_h^{2h} are provided directly by the multigrid algorithm.

RESULTS

In this section we are going to discuss some test cases in order to compare the adaptive GPT mesh and the self-adaptive multigrid scheme. Hereby, the investigation of ellipsoidal particle distributions allows the measurement of the CPU time as well as the calculation of the numerical error. We study short and long bunches, which usually pose a problem to the Poisson solvers. Next, the performance of the two adaptive discretization methods are tested with simulations for the rf gun of the European XFEL.

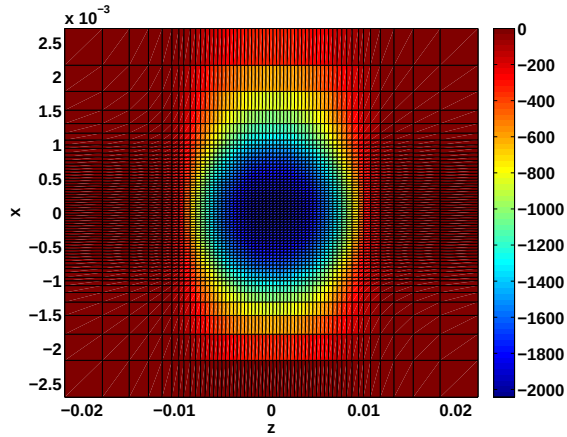

Figure 1: Adaptive GPT mesh for an ellipsoidal particle distribution with $c = 10.0$ mm. The plotted values are the potential at $y = 0.0$ m in the (x,z)-plane.

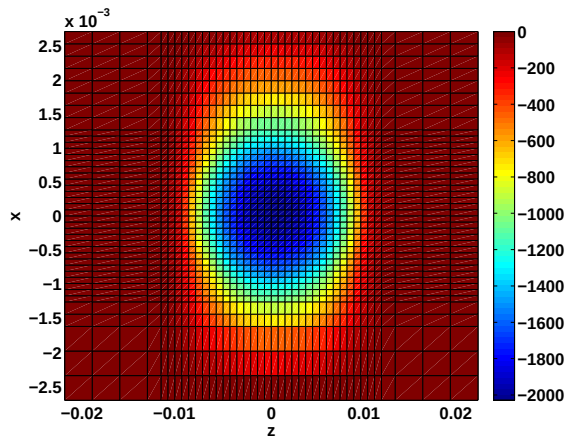

Figure 2: Self-adaptive multigrid mesh for an ellipsoidal particle distribution with $c = 10.0$ mm. The plotted values are the potential at $y = 0.0$ m in the (x,z)-plane.

Ellipsoidal Particle Distributions

First, the two different adaptive algorithms were applied to bunches of ellipsoidal shape with 100,000 uniformly distributed macro particles. The equatorial radii (transversal plane) were given with $a = b$ = 1 mm and the polar radius (longitudinal direction) with $c = 0.1$ mm, 1.0 mm, 10.0 mm. The bunch had a total charge of Q =1 nC. Self-adaptive multigrid was started with a coarse grid of $17 \times 17 \times 17$ mesh points. This coarse grid was refined iteratively with $\varepsilon = 0.5$. For the error of the solution

Table 1: Performance for ellipsoidal particle distributions: the adaptive GPT mesh and the self-adaptive multigrid mesh compared.

$N_x \times N_y \times N_z$	error, E-field	CPU time
ellipsoid with $c = 1.0$ mm		
adaptive GPT mesh		
$59 \times 59 \times 59$	$2.96 \cdot 10^{-2}$	1.06 s
self-adaptive multigrid		
$17 \times 17 \times 17$	$1.02 \cdot 10^{-1}$	0.48 s
$31 \times 31 \times 31$	$3.43 \cdot 10^{-2}$	0.62 s
$49 \times 51 \times 51$	$2.37 \cdot 10^{-2}$	0.76 s
ellipsoid with $c = 0.1$ mm		
adaptive GPT mesh		
$59 \times 59 \times 65$	$2.17 \cdot 10^{-1}$	1.09 s
self-adaptive multigrid		
$17 \times 17 \times 17$	$3.13 \cdot 10^{-1}$	0.41 s
$26 \times 26 \times 25$	$2.32 \cdot 10^{-1}$	0.66 s
$42 \times 42 \times 35$	$2.16 \cdot 10^{-1}$	0.53 s
ellipsoid with $c = 10.0$ mm		
adaptive GPT mesh		
$61 \times 61 \times 59$	$1.09 \cdot 10^{-1}$	1.06 s
self-adaptive multigrid		
$17 \times 17 \times 17$	$1.85 \cdot 10^{-1}$	0.50 s
$31 \times 31 \times 26$	$1.17 \cdot 10^{-1}$	0.50 s
$43 \times 43 \times 42$	$1.06 \cdot 10^{-1}$	0.75 s

the analytical values of the electric field E were compared to the numerically calculated values at the positions of the particles.

Table 1 represents the results for the two adaptive grids described above. It turns out that the same error as on the adaptive GPT mesh is achieved on the self-adaptive multigrid mesh but with less mesh lines. Consequently, the solution process requires less CPU time. Figure 1 and 2 show the adaptive GPT and the self-adaptive multigrid discretization, respectively.

The RF-Gun of the XFEL

The simulation of the rf gun for the XFEL were performed according to the parameters given in the technical design report [5]. The bunch had a charge of 1 nC and a radius of 1.5 mm. The temporal distribution was modelled as a flat top shape with 20 ps full width at half maximum (FWHM) and a rise and fall time of 2 ps. The particles of the bunch were started at the cathode and tracked 0.28 m downstream through the one and a half cell cavity (1.3 GHz, normal conducting with an accelerating field of 60 MV/m at the cathode) and the solenoid field (maximum field of 0.2 T centered at 0.4 m from the cathode). For self-adaptive multigrid the following parameters were set: $9 \times 9 \times 9$ mesh points for the coarsest grid and $\varepsilon = 0.5$.

Table 2 represents the CPU time of the simulations for the two different adaptive discretization schemes. It can be observed that the self-adaptive multigrid method requires less CPU time with increasing number of macro particles.

Table 2: Performance of the simulations for the rf gun of the XFEL: the adaptive GPT mesh compared to the self-adaptive multigrid mesh.

♯ of macro particles	adaptive mesh	CPU time
100,000	GPT	971 s
	MG	938 s
200,000	GPT	1815 s
	MG	1299 s

CONCLUSIONS

The construction of grids is a crucial task for the efficient application of the particle-mesh method for space charge calculations. In this paper we compared two different approaches for the construction of adaptive grids. The numerical test cases showed that the adaptive multigrid method generates meshes with less mesh lines, but the solution achieves the same accuracy than on the adaptive GPT mesh. The main advantage is that the construction process fits into the multigrid scheme. Hence, the new algorithm is more robust. The simulations for the rf gun of the European XFEL have shown an improvement with respect to CPU time for increasing number of macro particles.

ACKNOWLEDGMENT

The authors like to thank Marieke de Loos and Bas van der Geer (Pulsar Physics) for their support to implement the self-adaptive multigrid algorithm in GPT, fruitful comments and suggestions. Furthermore, we want to thank both that they made the simulation setup for the rf gun of the European XFEL available to us.

REFERENCES

[1] Pulsar Physics, Burghstraat 47, 5614 BC Eindhoven, The Netherlands, `www.pulsar.nl/gpt`. *General Particle Tracer (GPT)*, release 2.70, 2004.

[2] G. Pöplau, U. van Rienen, S.B. van der Geer, and M.J. de Loos. Multigrid algorithms for the fast calculation of space-charge effects in accelerator design. *IEEE Transactions on Magnetics*, 40(2):714–717, 2004.

[3] G. Pöplau and U. van Rienen. A self-adaptive multigrid technique for 3D space charge calculations. *IEEE Transactions on Magnetics*, 44(6):1242–1245, 2008.

[4] S.B. van der Geer, M.J. de Loos, O.J. Luiten, G. Pöplau, and U. van Rienen. 3D space-charge model for GPT simulations of high-brightness electron bunches. In M. Berz and K. Makino, editors, *Computational Accelerator Physics 2002 (Proceedings of the 7th International Computational Accelerator Physics Conference, East Lansing, Michigan, USA)*, number 175 in Institute of Physics Conference Series, pages 101–110, Bristol and Philadelphia, 2005. Institute of Physics Publishing.

[5] M. Altarelli et al., editor. *The European X-Ray Free-Electron Laser Technical design report*. DESY 2006-097, 2007.