RECENT DEVELOPMENTS FOR EFFICIENT 3D SPACE CHARGE COMPUTATIONS BASED ON ADAPTIVE MULTIGRID DISCRETIZATIONS*

G. Pöplau†, U. van Rienen, Rostock University, Rostock, Germany
S.B. van der Geer, Pulsar Physics, Eindhoven, The Netherlands
M.J. de Loos, TU Eindhoven, Eindhoven, The Netherlands

Abstract

Efficient and accurate space-charge computations are essential for the design of high-brightness charged particle sources. Recently a new adaptive meshing strategy based on multigrid was implemented in GPT and the capabilities were demonstrated. This new meshing scheme uses the solution of an intermediate step in the multigrid algorithm itself to define optimal mesh line positions. In this paper we discuss further developments of this adaptive meshing strategy. We compare the new algorithm with the meshing scheme of GPT where the mesh line positions are based upon the projected charge density.

INTRODUCTION

The design of charged particle accelerators and beamlines heavily relies on numerical simulations. When non-linear space-charge effects, path-length differences and non-linear optics are significant, the favorite design method is solving the relativistic equations of motion of a large number of sample (macro) particles through the electromagnetic fields of the set-up. A major complication is that for high-brightness beams also mutual Coulomb interactions, known as space charge forces, need to be taken into account.

The particle-mesh method is applied for space charge computations if stochastic effects are negligible. For the efficiency of the method the construction of the mesh is essential. Both a good approximation to the particle distribution and the performance of the solver is determined by the mesh. In [6] the authors investigate an adaptive meshing strategy based on multigrid. The advantage of this approach is that the results of multigrid algorithm can be used for the refinement of a given mesh. In our previous work we showed the potential of the algorithm, whereas in this paper we focus on reliability and predictable performance over a wide parameter range.

In this paper we give a new approach for the successive refinement of a mesh based on multigrid that is much more stable. Furthermore another objective could be achieved: More mesh lines near the head and tail of cigar-shaped hard-edged bunches. Such bunches have extreme field gradients close to the edges that cannot be detected by an algorithm that is just based on the charge density. The new method detects these large fluctuations automatically and puts more mesh lines at the desired locations.

PARTICLE-MESH MODEL IN GPT

In the tracking code GPT several space charge models are implemented [4]. The 3D model we consider here is based on the particle-mesh method (see [3] and citations therein). Hereby the bunch is modelled as a certain distribution of macro particles. All fields are computed 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 and the charge of the particles is assigned to the mesh points. Now, Poisson's equation

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

is solved for the potential φ. Further ϱ denotes the space charge distribution and ε_0 the dielectric constant. The domain Ω is a rectangular box constructed around the bunch. In GPT several boundary conditions are implemented, but in this paper we have restricted the investigations to Dirichlet boundaries, i. e. the surface $\partial\Omega$ is assumed to be perfectly conducting.

For the solution of the Poisson equation we applied a discretization with second order finite differences. This leads to a linear system of equations of the form $L_h u_h = f_h$, 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 allocates the mesh lines dynamically due to the charge density in the bunch [4]. The number of mesh lines is chosen according to the number and the distribution of the particles, respectively. Efficient space charge computations can be performed with the MOEVE Poisson solver that has been constructed especially for such non-equidistant meshes [1]. The adaptive GPT mesh is highly

* Work supported by BMBF under contract number 05K10HRC
† gisela.poeplau@uni-rostock.de

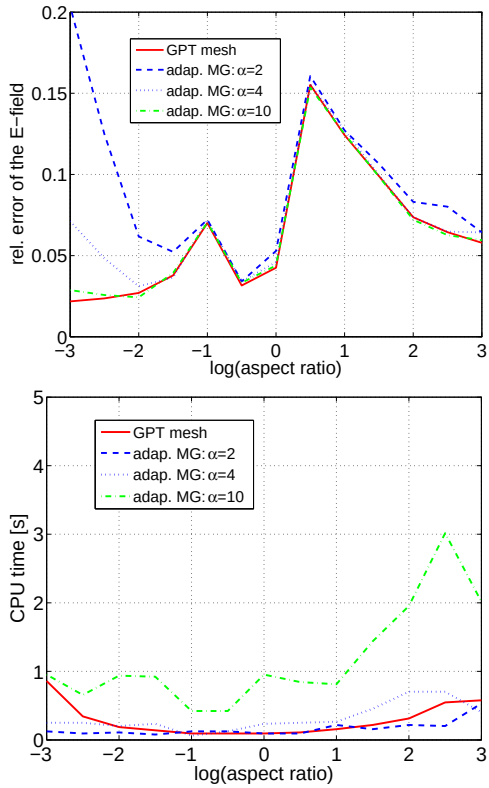

Figure 1: Relative error of the electric field (top) and CPU time (bottom) for the current GPT mesh and the new adaptive meshing scheme.

reliable but the construction is very complex. For example it has to be ensured that neighboring step sizes do not differ more than a certain factor in order to ensure the convergence of the multigrid Poisson solver [3].

The Self-Adaptive Multigrid Mesh

Recently, a self-adaptive discretization based on multigrid has been implemented in GPT. The main idea is to start with a relatively coarse grid and to refine it step by step according to a certain criterion, the so-called τ-criterion. The τ-criterion and the extended τ-criterion together with the Poisson solver of MOEVE were discussed in [2] and [6], respectively. Then the mesh is refined if the τ-values exceed a certain value i. e. a mesh line is added if the maximum of τ-values in a plain in x-, y- and z-direction exceeds a certain value. Since the particle distribution is not smooth this approach provides in certain cases a mesh line distribution that does not sufficiently approximate the distribution of the particles. Such as very long bunches posed a problem.

In this paper we investigate a new approach for a better exploitation of the τ-values. Instead of the maximum value we choose the ℓ_2-norm. This approach gives a better average over the particle distribution in a certain plane. Thus, the weight of local spikes is much lower.

In order to describe our method in more detail we have to introduce some notations according to [5], where the adaptive multigrid method with the τ-criterion is described. 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. In our implementation 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 $\tau_h^{2h} := L_{2h}\widehat{I}_h^{2h}u_h - I_h^{2h}L_h u_h$. More details can be found in [5].

The values $\tau_h^{2h}(i,j,k)$ are now available at the mesh points (i,j,k) with $i = 0,\ldots,N_x-1$, $j = 0,\ldots,N_y-1$ and $k = 0,\ldots,N_z-1$, where N_x, N_y and N_z are the numbers of mesh lines in x-, y- and z-direction, respectively. For the extended τ-criterion [6] we introduce the differences of the values τ_h^{2h} for neighboring mesh points $D\tau_h^{2h} = (D_x\tau_h^{2h}, D_y\tau_h^{2h}, D_z\tau_h^{2h})$. Here, $D_x\tau_h^{2h} = \tau_h^{2h}(i+1,j,k) - \tau_h^{2h}(i,j,k)$ with $i = 0,\ldots,N_x-2$ denotes the differences in x-direction, D_y and D_z are defined analogously.

For the new τ-criterion we have taken the ℓ_2-norm of the τ-values in each plane in x-, y- and z-direction, respectively, i. e.

$$\|\tau_h^{2h}(x(i))\|_2^2 = \sum_{j,k}(\tau_h^{2h}(i,j,k))^2, \quad i = 0,\ldots,N_x-1 \tag{2}$$

and the ℓ_2-norm of the differences

$$\|D_x\tau_h^{2h}(i)\|_2^2 = \sum_{j,k}(D\tau_h^{2h}(i,j,k))^2, \; i = 0,\ldots,N_x-2. \tag{3}$$

The new 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 $L_h u_h = f_h$.
3. Compute τ_h^{2h} and $D\tau_h^{2h}$.
4. Add a mesh line locally in x-direction between point x_i and x_{i+1}, if
   $$0.5(\|\tau_h^{2h}(x(i))\|_2 + \|\tau_h^{2h}(x(i+1))\|_2) + \delta\|D_x\tau_h^{2h}(i)\|_2 > \varepsilon.$$
   Analogously, add mesh lines locally in y- and z-direction
5. Proceed from 2. as long as $N_x \cdot N_y \cdot N_z$ is smaller than α times the number of particles ($\alpha >= 1$).

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.

The term in step 4. of the algorithm computes an average of the norm of τ-values and of the norm of the difference of τ-values. Whereas the term $0.5(\|\tau_h^{2h}(x(i))\|_2 + \|\tau_h^{2h}(x(i+1))\|_2)$ is an indicator for the particle distribution, the term $\delta\|D_x\tau_h^{2h}(i)\|_2$ detects the sharp edges. We take this second term with weight $0 < \delta < 1$ that the influence of the edges are not overestimated. Our simulations were performed with $\delta = 0.2$.

RESULTS

One of the most stringent tests in our suite of analytical benchmarks is the field-error of a hard-edged cylinder with an aspect ratio varying over 6 orders of magnitude. Typical rms errors are in the 10% range, as the field is almost singular near the end of bunches with extreme aspect ratios. Nevertheless we want the code to survive these cases, with controlled behavior regardless of the number of mesh lines and particles.

The results for this analytical test case is shown in Figure 1. It turns out that for every type of aspect ratio a set of parameters can be found that the relative error as well as the CPU time are comparable to the GPT meshing routine. Unfortunately, this is not the same set of parameters for all bunches. For low aspect ratios (10^{-1} to 10^{1}) we can achieve performing times that are up to a factor 2 faster than the routine of GPT. For longer or shorter bunches we have to spend more mesh lines in order to provide a certain accuracy, i. e. the factor α has to be chosen higher ($\alpha = 4$). The very long bunches with aspect ratio smaller than 10^{-2} require an even finer discretization ($\alpha = 10$).

Figure 2 represents the longitudinal field of a long bunch with aspect ratio $10^{-2.5}$ obtained with the GPT meshing routine and with the adaptive multigrid mesh, respectively. It can be observed that with the new meshing procedure the field at the edges is much better approximated due to edge detection of the algorithm.

CONCLUSION

In this paper we introduced a new adaptive meshing strategy based on multigrid. The main idea is to make use of quantities that are already provided by the multigrid algorithm - the τ-values, that give an estimation of the error. In our new approach we combine the ℓ_2-norm of τ-values with the ℓ_2-norm of the differences. This approach allows a refinement of the mesh due to the distribution of the particles as well as a refinement at edges of the distribution. The numerical investigations proof the potential of the new algorithm.

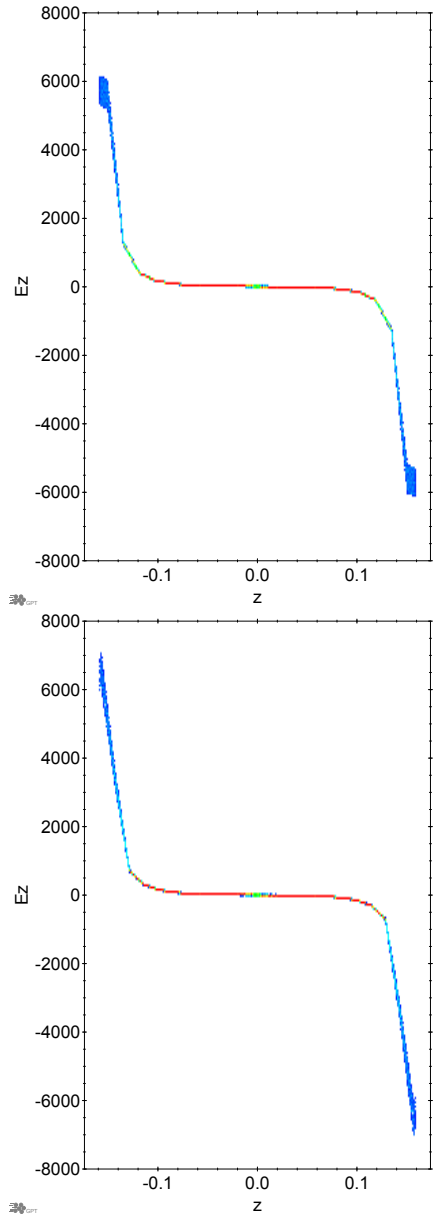

Figure 2: Longitudinal field of a long bunch with aspect ratio $10^{-2.5}$ obtained with the GPT mesh (top) and with the adaptive multigrid mesh (bottom).

REFERENCES

[1] G. Pöplau. "MOEVE 2.0: Multigrid Poisson Solver for Non-Equidistant Tensor Product Meshes". Universität Rostock, 2006.

[2] G. Pöplau and U. van Rienen. "Efficient 3D space charge calculations with adaptive discretization based on multigrid." In Proceedings of IPAC 2010 (1st International Particle Accelerator Conference), Kyoto, Japan, pages 1832–1834, 2010.

[3] 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.

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

[5] U. Trottenberg, C. Oosterlee, and A. Schüller. "Multigrid". Academic Press, San Diego, 2001.

[6] S.B. van der Geer, M.J. de Loos, G. Pöplau, and U. van Rienen. "Adaptive space-charge meshing in the general particle tracer code." In Proceedings of PAC 2011 (11th Particle Accelerator Conference), New York, USA, 2011.