WDS'06 Proceedings of Contributed Papers, Part III, 187–192, 2006. *ISBN 80-86732-86-X © MATFYZPRESS*

A Comparison of Advanced Poisson Equation Solvers Applied to the Particle-In-Cell Plasma Model

Z. Pekárek and R. Hrach

Charles University, Faculty of Mathematics and Physics, Prague, Czech Republic.

Abstract. In this contribution we estimate the performance of various Poisson equation solvers applied to the Particle-in-Cell plasma models. The solvers determine the practical usability of complex PIC models, especially in three dimensions.

The performance is measured on 2D models with grids of various sizes, the methods studied are SOR, conjugate gradients, LU decomposition and multigrid methods.

The results confirm the efficiency of the LU decomposition method on smaller grids. The advantages of using it as a part of the multigrid method on larger grids are discussed as well.

Introduction

In contemporary plasma physics there exist two main approaches to the computer modeling. The particle models provide a deeper insight into the actual processes taking place in plasma, they however require substantially more computational resources than the fluid models.

The performance of particle models is often accelerated by a number of techniques such as the Particle-in-Cell (PIC) method. Nevertheless, the resources required to acquire the results of the full model are still such that some additional simplifications must be implemented. Fortunately, a large class of plasma models has an intrinsic spatial symmetry which leads to a decrease in the number of spatial dimensions required to solve the problem.

However, there are problems and models which do not yield to such a simplification, either due to a complex spatial structure or due to the presence of the magnetic field. The examples can be found in many areas of plasma physics, e.g. magnetrons or advanced tokamak edge plasma probes such as the Katsumata probe.

Modeling such phenomena using a particle model is a challenging task due to the intrinsic size of the problem, pronounced mainly in the number of particles present in the region in question and in the number of the discretization grid nodes, which determines the precision of the model.

In order to obtain the results of a complex model in a reasonable time frame some sophisticated methods of numerical mathematics must be used. Since in the PIC models most of the time is typically spent computing the Poisson equation, that part of the code must be accelerated in the first place.

The PIC method used to model stationary plasma systems consists of a series of time steps which lead to a dynamic equilibrium. Each time step consists of several stages:

1. computation of the space charge density ϱ from the charged particle positions,
2. computation of the potential U from the density using the Poisson equation

$$\Delta U = -\frac{\varrho}{\varepsilon}$$

3. computation of the field intensity from the potential,
4. evolution of the particle positions in time using the forces calculated from the intensity,
5. removing particles which left the spatial region, inserting new particles, perform collisions in PIC/MC method, etc.

The density, potential and field intensity are discretized to a spatial grid in order to avoid the costly direct computations of the forces between particles. Due to the vast number of particles the position of grid nodes is often chosen to be equidistant in each dimension to accelerate the computation of density and forces (stages 1 and 4). This large number of particles also essentially rules out other methods of solving PDEs, such as the finite elements method, because any performance gain from more effective unstructured grids would be offset by far more complex computation of density and forces.

The contribution of each stage to the total computational burden depends on the ratio of the number of particles to the number of grid nodes. The number of grid nodes is mainly governed by the number of spatial dimensions used in the model. Thus, in 1D model the computation of potential is negligible, since the number of particles typically far exceeds the number of grid nodes, which leads to stages 1 and 4 demanding most of the computation time. In reasonably sized 2D systems the time spent solving the Poisson equation is comparable to the time spent on the particle ensamble. In 3D models the number of grid nodes typically exceeds the number of particles in the model (especially when using the macroparticle technique to decrease the number of particles in the model) and most of the computation time is spent solving the Poisson equation. This leads to the necessity of using some sophisticated methods of numerical mathematics to accelerate this computationally demanding part of the code.

Solvers of the Poisson equation

The Poisson equation discretized onto a finite differenced spatial grid can be in essence transformed to a system of linear equations, which can be described by a sparse matrix [*Press et al.*, 2002]. Numerical mathematics provides various methods with which the solution can be found. These methods can be divided into two groups: the direct and the iterative methods.

Iterative methods obtain the solution through a series of iterative steps which decrease the error in the estimated solution. This defect of the solution, which can be defined as $\mathbf{d} = \mathcal{L}\mathbf{U} - \mathbf{b}$ [*Press et al.*, 2002], can be estimated during the computation and the iterative refinement can be stopped when some preset criterion, as determined by the required accuracy of the model, is reached.

Iterative solvers are particularly useful in the PIC method, because at each time step the solution from the previous time step can be used as the initial estimate of the new solution. The importance of this benefit, however, strongly depends on fluctuations of the density between successive time steps.

Direct methods reach the solution in a single step, which must be carried out entirely. They do not need any initial estimate of the solution and are essentially independent of the fluctuations during evolution of the model.

Direct methods are often much faster in obtaining the solution, they are however much more complex to implement and/or require substantially more computational resources than the iterative methods.

Gauss-Seidel method

This method forms together with the Jacobi method a very basic way of iterative solving of the Poisson equation [*Press et al.*, 2002]. Each iterative step consists of the update performed on each of the 2D grid nodes inside the computational domain. It has several modifications differing in the order of nodes updated. One of them is the red-black (also known as checkerboard) ordering, which consists of two half-sweeps through the grid, each updating only the nodes with the appropriate "color" [*Hackbusch*, 2003].

Gauss-Seidel method is not fast enough to be used as a standalone method. However, it has some specific properties, such as good smoothing ability, which make it a good choice for the smoother subroutine of the multigrid method.

Successive overrelaxation method - SOR

This simple and straight-forward method is often the first choice for the Poisson equation solver. It is based on the Gauss-Seidel method, which is modified by adding a parameter ω acccelerating the convergence. The choice of ω is essential for the good performance of the method. Our results were obtained using the Chebyshev acceleration [*Press et al.*, 2002].

SOR is significantly faster than the Gauss-Seidel method, however the solution is less smooth [*Hackbusch*, 2003] and thus unsuitable for the multigrid method.

Conjugate gradients method

This iterative method is based on a rather general idea of minimizing a given vector, which in this case is the difference between the current iteration's estimate of the potential and the exact potential as defined by the boundary conditions and the charge density [*Press et al.*, 2002]. It has significantly larger memory requirements and can be easily parallelized and vectorized.

LU decomposition method

This direct method is based on the actual factorisation of the matrix describing the whole system of linear equations [*Press et al.*, 2002]. The matrix is fortunately sparse, i.e. most of its elements are zeros. The method pre-transforms the matrix into two triangular matrices which can be then repeatedly used to directly obtain the solution vector from the right hand side vector.

The LU decomposition implemented in our code is based on the library UMFPACK [*Davis*, 2004]. Despite its optimizations it is still very demanding in terms of computational resources and on the common computer architecture (32-bit memory addressing) it is very difficult to solve 2D grids larger than approximately 900×900 grid nodes.

Multigrid method

The basic problem with iterative methods such as Gauss-Seidel or SOR is that they can rapidly decrease defect components with the wavenumber comparable to the grid element size. The smoother components, however, remain largely unchanged and disappear only gradually. The remedy for this situation is to construct a series of increasingly coarse grids. On each of these grids a smoother method destroys the defect component intrinsic to the grid coarseness.

Multigrid method is not actually a single method, but a class of methods assembled from different building blocks. There are four main components: the restriction and prolongation subroutines take care of the transfer of the residual defect and the solution update between the grids of adjacent coarseness, the smoother subroutine improves the solution on each of the grids and the solver subroutine provides the solution on the coarsest level.

The choice of possible subroutines for each task is abundant and the optimal setup for the PIC model is still not entirely evaluated. For the purpose of this paper we use the full weighting interpolation for both restriction and prolongation, one step of Gauss-Seidel method with red-black ordering as a smoother and LU decomposition as the coarsest level solver. The successive levels of multigrid hierarchy differ in the number of nodes in each dimension by the factor of two, so in 2D model each coarser grid is four times smaller than the previous one.

Model

The performance of the solvers was evaluated on a standard 2D model of a cylindrical probe immersed in the positive column of DC glow Ar discharge. Some standard assumptions were made [*Hockney, Eastwood*, 1999], [*Hrach*, 1999]:

- Source of charged particles was the undisturbed plasma with Maxwell distribution of velocities and with different temperatures of electrons and ions, $T_\mathrm{e} = 23210$ K and $T_\mathrm{i} = 300$ K.
- Trajectories of charged particles were calculated by the molecular dynamics technique (Verlet algorithm, different time steps of electrons and ions $\Delta t_\mathrm{e} = 1 \cdot 10^{-11}$ s and $\Delta t_\mathrm{i} = 1 \cdot 10^{-8}$ s).
- Scattering of charged particles by neutrals was treated stochastically by Monte Carlo method. In order to speed-up simulations the null-collision technique was used. The pressure was $p = 133$ Pa, ionization coefficient was $1 \cdot 10^{-7}$.
- For the computation of charge density and evaluation of forces on individual particles the CIC weighting was used.

Dimensions of the model were $1 \cdot 10^{-2} \times 1 \cdot 10^{-2}$ m, the diameter of the probe was $2 \cdot 10^{-4}$ m. Initial number of particles was in all cases $1 \cdot 10^{6}$ of electrons and ions each. The stopping criterion for all iterative methods was a maximal point-wise defect of potential smaller than $1 \cdot 10^{-6}$ V [*Press et al.*, 2002].

Results

The performance of various Poisson equation solvers was measured on 2D square grids of various sizes, the grid nodes were equidistantly spaced. The number of grid nodes, including the border nodes, in each dimension was $2^n + 1$ where n denotes the level of the grid in a complete multigrid scheme [*Hackbusch*, 2003].

In this contribution we did not deploy the complete multigrid hierarchy, on the coarsest level of multigrid scheme we used the LU decompositition solver instead. The notation ”MG2” describes a multigrid method with two additional coarser grids in addition to the basic finest one, i.e. MG2 method on a 1025×1025 grid would construct additional grids of 513×513 and 257×257 nodes and on the latter one the exact solution would be found via LU decomposition during each iteration.

All computations were carried out on a Pentium 4 3.2 GHz PC with 1 GB of RAM.

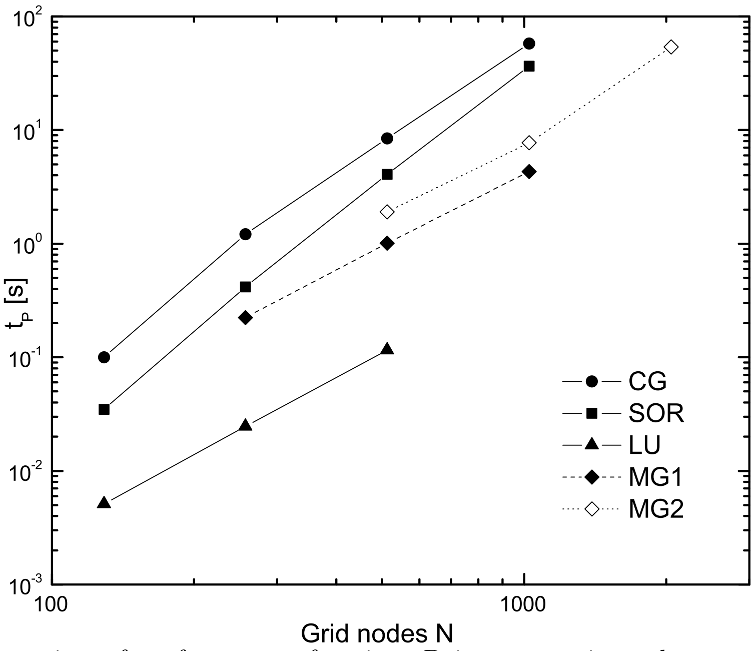

Figure 1. An overview of performance of various Poisson equation solvers applied to the 2D PIC model discretized on $N \times N$ nodes, t_P denotes the time spent computing the potential during each PIC time step. The methods listed are successive overrelaxation (SOR), conjugate gradients (CG), LU decomposition-based method (LU), two-grid multigrid (MG1) and three-grid multigrid (MG2). Both figures display the same data, the scaling differs.

Discussion

The methods presented in this paper cover a wide selection of methods used to solve a partial differential equation, especially the Poisson equation used in the PIC-based plasma models. One notable class of methods is absent in this paper – while the methods using the Fast Fourier Transform are very efficient in their performance, the task of their implementation is complex and still more complex is implementing additional boundary discontinuities in the spatial domain. Since this paper is concerned mainly with more readily available methods and the boundary discontinuities in the form of electrodes are abundant on PIC models, we decided to omit FFT methods.

The basic and perhaps most widespread method, SOR, provides reasonable performance on smaller grids. However, on larger grids the performance is insufficient. The performance also depends on the number of particles in the model and on the time step, with both these parameters affecting the fluctuations of the density between successive time steps of the model. This renders largely ineffective the intuitive model acceleration based on reduction of number of particles in the system.

The performance of SOR is extremely affected by the value of its parameter ω. The Chebyshev acceleration provides a reasonable boost to its performance when compared to the use of constant optimal value of ω.

The conjugate gradients method has similar disadvantages as SOR. Additional disadvantage stems from its large memory requirements, which might also be the reason for its poor performance in comparison to other methods. The performance of CG could be boosted through the use of specialized libraries for vector processing (BLAS) and through deployment of preconditioners, simplified solvers used to accelerate the convergence rate. These solvers can be, however, used as standalone methods, without the need for a CG method.

The only advantage of CG is that with a growing grid size its performance decreases less than the

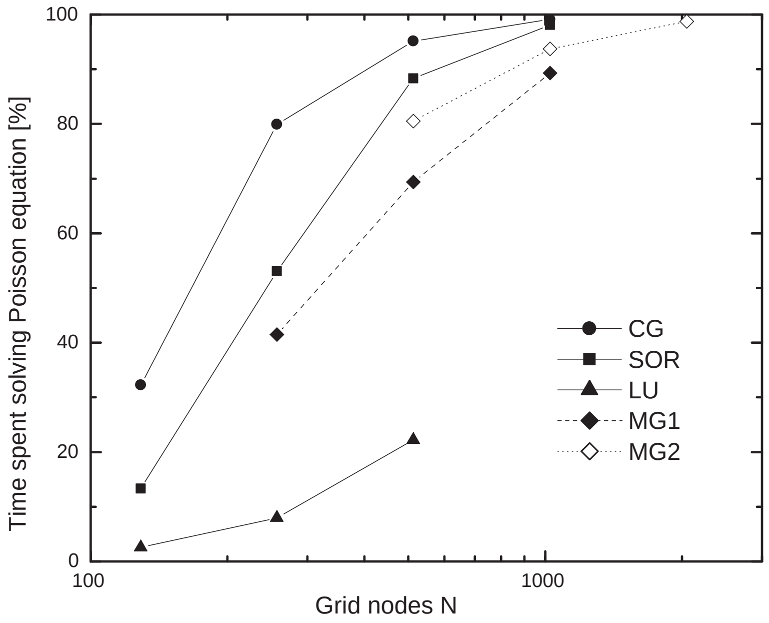

Figure 2. An overview of performance of various Poisson equation solvers applied to the 2D PIC model discretized on $N \times N$ nodes. The percentage denotes the ratio of time spent computing the potential and total computing time.

performance of SOR. However, in comparison with multigrid methods its performance on larger grids would be still insufficient.

From the methods evaluated the most effective one is the only truly direct method, which is based on LU decomposition. Fig. 1 shows that the LU solver is more than an order of magnitude faster than SOR on the grid of 257×257 nodes. This difference is even more pronounced on a larger grid.

For the 2D PIC grids of these reasonable sizes this method provides a fast monolithic Poisson solver with other advantageous properties, such as the insensitivity to the fluctuations of density. This combination of properties could be utilized to create models with longer relaxation times, which with previous methods would not be computable in a reasonable time frame.

The LU method unfortunately requires too much computer resources, such as memory, to be routinely used on large scale models. This leads to an idea of using the multigrid method to extend the exact solution of LU method to a larger grid. Such an extension is iterative by nature, so the coarsest grid solver is used repeatedly and the overall performance is crucially dependent on its efficiency. This is well ilustrated by the difference between MG1 and MG2 solvers, e.g. on the grid of 513×513 nodes. The MG1 solver is almost twice as fast as MG2, because MG1 solves the 257×257 grid exactly, while MG2 uses on this grid a Gauss-Seidel smoother and obtains the exact solution only on the still coarser grid.

This penalty together with memory requirements of large LU solvers still limits the use of multigrid methods on really large problems such as the full 3D models. To optimize the performance of these methods some further investigation into the optimal choices of building blocks of multigrid methods must be carried out. The obvious candidates for optimization are the restriction and prolongation subroutines. The order of node processing in the smoother subroutine and the number of smoothing steps might have some influence as well.

Conclusion

The performance of Poisson equation solvers has crucial influence on practical usability of more complex PIC models. In this contribution we evaluated performance of various approaches to solving this part of the model and identified the most effective method for mid-size 2D models and a promising method for large 3D models. The possibilities of further optimization were discussed as well.

Acknowledgments. The work is a part of the research plan MSM0021620834 that is financed by the Ministry of Education of the Czech Republic and partly was supported by the Grant Agency of Charles University Prague, Grants No. 296/2004 and 220/2006.

References

Davis T. A., *ACM Trans. Math. Software*, vol 30, no. 2, pp. 196-199, 2004

Hackbusch W.: Multi-Grid Methods and Applications, Springer-Verlag, Berlin, Germany, 2003.

Hockney R. W., Eastwood J. W.: Computer Simulation Using Particles, IOP Publishing, Bristol 1999.

Hrach R.: *Czech. J. Phys.* 49 (1999) 155.

Press W. H.*et al.*. Numerical Recipes in C++. Cambridge University Press, Cambridge, UK, second edition, 2002.