Improving performance of multi-dimensional Particle-In-Cell codes for modelling of medium pressure plasma

Z Pekárek, M Lahuta and R Hrach
Faculty of Mathematics and Physics, Charles University, V Holešovičkách 2, Prague 8, Czech Republic.
E-mail: zdenek.pekarek@mff.cuni.cz

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, FACR and multigrid methods.

The results confirm the efficiency of the direct methods tested, namely the LU decomposition method and FACR. The advantages of using LU decomposition as a part of the multigrid method on larger grids are discussed as well.

1. Introduction
In contemporary plasma physics there exist two main approaches to the computer modelling. 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 devices 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.

Modelling 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:

(i) computation of the space charge density $\varrho$ from the charged particle positions,
(ii) computation of the potential $U$ from the density using the Poisson equation (1),

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

(iii) computation of the field intensity from the potential,
(iv) evolution of particle positions in time using forces calculated from the intensity,
(v) removing particles which left the spatial region, inserting new particles, performing 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 (i) and (iv)). 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.

2. 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 [1]. Numerical mathematics provides various methods for finding the solution. 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 until a preset threshold is met. This defect of the solution [1] has a real physical meaning (influences the accuracy of the PIC model, since it introduces non-physical fluctuations into the values of potential) and thus the threshold must be chosen carefully. Defects of direct solvers are usually far below any reasonable limits.

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 [9].

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 or require substantially more computational resources than the iterative methods.

2.1. Successive overrelaxation method - SOR
This simple and straight-forward iterative method is often the first choice for a Poisson equation solver. It is based on the Gauss-Seidel method, which is modified by adding a parameter $\omega$ accelerating the convergence. The choice of $\omega$ is essential for the good performance of the method. Our results were obtained using the Chebyshev acceleration [1].

SOR is significantly faster than the Gauss-Seidel method, however the solution is less smooth [2] and thus unsuitable for the multigrid method.
2.2. 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 [1]. It has slightly larger memory requirements than SOR and can be easily parallelized and vectorized.

2.3. 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. 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 smoothing 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.

2.4. LU decomposition method
This direct method is based on the actual factorisation of the matrix describing the whole system of linear equations [1]. 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 [3]. Despite its optimizations it is still very demanding in terms of computational resources, especially computer memory.

2.5. FACR
One of the fastest direct method for solving Poisson equation is FACR algorithm [4]. It is a combination of fast Fourier transform (FFT) and cyclic reduction (CR) [5]. For Dirichlet boundary conditions the number of grid points in each direction must be $2^k - 1$. Other boundary conditions (periodic, Neumann) and their combinations are possible. When there are points in the interior of grid with a given value of potential a so-called capacity matrix method must be included [6] because FACR itself cannot implement boundary conditions in the interior of computational domain. In this case the Poisson equation must be solved twice. First with zero charge density at the place of desired potential and secondly with corrected charge density calculated from capacity matrix. FACR method can be also extended into 3D.

3. 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 [6], [7].
Source of charged particles was the undisturbed plasma with Maxwell distribution of velocities and with different temperatures of electrons and ions, $T_e = 23210$ K and $T_i = 300$ K.

Trajectories of charged particles were calculated by the molecular dynamics technique (Verlet algorithm, different time steps of electrons and ions $\Delta t_e = 1 \times 10^{-11}$ s and $\Delta t_i = 1 \times 10^{-8}$ 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 $1 \times 10^{-7}$.

For the computation of charge density and evaluation of forces on individual particles the CIC weighting was used.

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

4. Results

![Figure 1.](image)

**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).

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 [2].

In this contribution we did not deploy the complete multigrid hierarchy, on the coarsest level of multigrid scheme we used the LU decomposition 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 $1025 \times 1025$ grid would construct additional grids of $513 \times 513$ and $257 \times 257$ 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 2](attachment:image.png)

**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. The total number of particles in the model was approximately $2 \times 10^6$.

5. 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.

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 $\omega$. The Chebyshev acceleration provides a reasonable boost to its performance when compared to the use of constant optimal value of $\omega$. It should be noted, however, that the traditional optimal value of $\omega$ used to find a one-time solution of a PDE can significantly differ from the empirically established optimal $\omega$ for PIC models [8].

The conjugate gradients method has similar disadvantages as SOR. Additional disadvantage stems from its larger memory requirements, which might also be the reason for its poor performance in comparison to SOR. 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, often used as standalone methods, without the need for a CG method.
From the methods evaluated the most effective ones are both direct methods. FACR is roughly 2 or 3 times faster than LU decomposition based method, with both methods being more than an order of magnitude faster than SOR. This difference is even more pronounced on larger grids.

The importance of additional speed of FACR in comparison with the LU solver is questionable, since for models with a larger number of particles the performance of LU solver has already only a limited impact on the overall performance of the model, cf. figure 2. Unless the FACR can be obtained as a generally available library with a simple and easy to use interface, such as can be constructed for the LU decomposition-based solver library, the benefit of a higher speed is negated by the need to implement the FACR method itself.

For the 2D PIC grids of reasonable sizes the LU method provides a fast 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 iterative methods would not be computable in a reasonable time frame.

The LU method unfortunately requires large computer resources, such as memory, so it is inconvenient to use it 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 illustrated 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.

6. Conclusion
The performance of Poisson equation solvers has crucial influence on practical usability of more complex PIC models. In this contribution we evaluated performance and ease of implementation of various solutions of this part of the model and identified the most effective methods 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.

References
[1] Press WH et al 2002 Numerical Recipes in C++ (Cambridge: Cambridge University Press)
[2] Hackbusch W 2003 Multi-Grid Methods and Applications (Berlin: Springer-Verlag)
[3] Davis TA 2004 ACM Trans. Math. Software 30(2) 196–9
[4] Swarztrauber PN 1977 SIAM Rev. 19(3) 490–501
[5] Buzbee BL, Golub GH and Nielson CW 1988 SIAM J. Num. Anal. 7(4) 627–56
[6] Hockney RW and Eastwood JW 1999 Computer Simulation Using Particles (Bristol: IOP Publishing)
[7] Hrach R 1999 Czech. J. Phys. 49 155
[8] Šimek J 2006 Thesis (Prague: MFF UK)
[9] Hrach R, Lahuta M, Pekárek Z and Šimek J 2006 Czech. J. Phys. 56 (Suppl. B) B990–5