Contents

Chapter 1

Method

(REWRITE)

As the previous chapter described the need for numerical plasma models this chapter goes through the theory behind a Particle-In-Cell model, with a focus on multigrid as a Poisson solver. First there is a general overview of a PiC model and the different building blocks needed. Then there is an overview of the normalization scheme designed to minimize floating point operations, FLOPS. Domain partitioning as a strategy to parallelize the model is then considered. How the multigrid solver works and is structured follows, before the parallelization issues for the multigrid solver are considered. Lastly there is an overview of boundary conditions and the special considerations they have in a multigrid solver.

1.1 Particle-in-Cell

Particle based plasma simulations have been in use since the 1960s, (Verboncoeur, 2005), and part of this project was to design a massively parrallel implementation, with a focus on the Poisson solver. The aim of this chapter is to describe simple and fast PiC model, with good scaling properties, as a baseline and rather add in extra functionality later. Due to this it uses an electrostatic model and ignores relativistic effects, which makes it faster and more suited to certain tasks, such as space plasmas and plasma discharges. For an example of a modern relativistic full electromagnetic model see Sgattoni et al. (2015).

The first particle based plasma calculations was was done by Dawson, 1962 and Buneman (1959). They computed the electrical force directly between the particles leading to a computational scaling of $\mathcal{O}((\#particles)^2)$. Since a large number of particles is often wanted the PiC method seeks to improve the scaling by computing the force on the particles from an electric field instead. The electric field is computed from the charge distribution obtained from the plasma particle distribution. For an electrostatic model, which this thesis focuses on, this is usually done by solving the Poisson equation, eq. (1.1), over the whole domain,

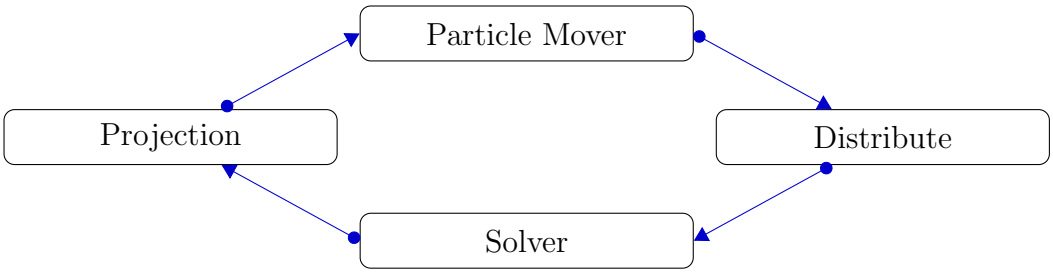

Figure 1.1: Schematic overview of the PIC cycle. The mover moves all the particles and updates the velocities. Next the charges are distributed to a charge density grid by a distribute algorithm. The solver then obtains the electric field, and magnetic field in a full electromagnetic model. Lastly the field values are projected onto the particles.

Ω. The input to the solver is the charge density, ρ, and the output is the potential, ϕ.

$$\nabla^2\phi = -\rho \qquad \text{in} \qquad \Omega \tag{1.1}$$

A PiC model has 4 main components: the mover, the weighting scheme, the field solver and the distributer. See fig. 1.1 for an overview of the PiC cycle. The mover is responsible for moving the particles and updating the velocities of the particles. The distribute module computes a charge distribution from the particle distribution, This is often done with 1st order interpolation, resulting in second order accuracy. Different order interpolation can also be used. The solver then computes the electric field. For an electrostatic case a multigrid algorithm is often used but there described in section 1.1.2. Lastly the fields are projected onto the particles.

1.1.1 Movers

The mover in a PiC model has the task of moving all the particles according to the velocity of the particles, as well as the electric and magnetic fields. An often used mover is the *Leapfrog* algorithm (Birdsall and Langdon, 2004), derived from a forward finite difference discretization of the timestep. Then the velocity is shifted half a timestep forward improving the accuracy, with no extra computations needed compared with *Euler* integration. When the magnetic force also needs to be considered the *Boris* algorithm **boris** is often used, employing rotations to effieciently deal with the cross product. While the aforementioned algorithms are explicit, implicit algorithms also exists (cite), which imposes less restrictions on the choice of the spatial resolution.

1.1.2 Field Solvers

The Poisson equation, **??**, is a well known and investigated problem. Here we will mention some advantages and disadvantages of different possible field solvers before we describe our choice of a multigrid solver.

Spectral Methods

(NOTE TO SELF: Shen no fourier transform, find different source, Name what $AandB$ are.) The spectral methods is based on Fourier transforms of the problem and solving the problem in it's spectral version, see (Shen, 1994), for an implementation of an spectral Poisson solver. They are efficient solvers that can be less intricate to implement, but can be inaccurate for complex geometries.

When looking for a solution with a spectral method we first rewrite the functions as Fourier series, which for the three-dimensional Poisson equation would be

$$\nabla^2 \sum A_{j,k,l} e^{i(jx+ky+lz)} = \sum B_{j,k,l} e^{i(jx+ky+lz)} \tag{1.2}$$

From there we get a relation between the coefficients

$$A_{j,k,l} = -\frac{B_{j,k,l}}{j^2+k^2+l^2} \tag{1.3}$$

Then we compute the Fourier transform of the right hand side obtaining the coefficients $B_{j,k,l}$. We compute all the coefficients $A_{j,k,l}$ from the relation between the coefficients. At last we perform a inverse Fourier transform of the left hand side obtaining the solution.

(1.4)

Finite Element Methods

The finite element, (FEM), is a method to numerically solve a partial differential equations (PDE) first transforming the problem into a variational problem and then constructing a mesh and local trial functions, see Alnæs et al., 2011 for a more complete discussion.

To transform the PDE to a variational problem we first multiply the PDE by a test function v, then it is integrated using integration by parts on the second order terms. Then the problem is separated into two parts, the bilinear form $a(u,v)$ containing the unknown solution and the test function and the linear form $L(v)$ containing only the test function.

$$a(u,v) = L(v) \qquad v \epsilon \hat{V} \tag{1.5}$$

Next we construct discrete local function spaces of that we assume contain the trialfunctions and testfunctions. The function space, $\hat{V}$, often consists of locally defined functions that are 0 except in a close neighbourhood of a mesh point, so the resulting matrix to be solved is sparse and can be computed quickly. The matrix system is then solved by a suiting linear algebra algorithm, before the solution is put together. The FEM method is very suited to tackling problems on complicated grids.

Multigrid

The multigrid method used to solve the Poisson equation and obtain the electric field is a widely used and highly efficient solver for elliptic equations, having a theoretical scaling of $\mathcal{O}(N)$ (Press et al., 1988), where N is the grid points. It is very well suited to simple geometries that can easily be translated to coarser problems. Due to this it we have chosen to use it as the solver in our PiC model. The multigrid method is based on iterative solvers such as Gauss-Seidel, **??**, these have the property that they quickly eliminate local errors in the solution, while far away influences takes longer to incorporate. Multigrid algorithms tries to lessen this problem by transforming this problem to a coarser grid so the distant errors gets solved in fewer iterations. Due to this it needs algorithms to transfer the problem between coarser and finer grids, which is called restrictors and prolongators. The multigrid algorithm is described in more detail in **??**.

Bibliography

Alnæs, M. S. et al. (2011). The FEniCS Manual. October. URL: http://mmc2.geofisica.unam.mx/acl/edp/Ejemplitos/FEniCs/fenics-manual-2011-10-31.pdf (visited on 04/21/2016).

Birdsall, C. K. and A. B. Langdon (2004). Plasma Physics via Computer Simulation. en. CRC Press. ISBN: 978-1-4822-6306-0.

Buneman, O. (1959). "Dissipation of Currents in Ionized Media". In: Physical Review 115.3, pp. 503–517. DOI: 10.1103/PhysRev.115.503. URL: http://link.aps.org/doi/10.1103/PhysRev.115.503 (visited on 08/19/2016).

Dawson, J. (1962). "One-Dimensional Plasma Model". In: Physics of Fluids (1958-1988) 5.4, pp. 445–459. ISSN: 0031-9171. DOI: 10.1063/1.1706638. URL: http://scitation.aip.org/content/aip/journal/pof1/5/4/10.1063/1.1706638 (visited on 08/19/2016).

Press, W. H. et al. (1988). "Numerical recipes in C". In: Cambridge University Press 1, p. 3. URL: http://journals.cambridge.org/abstract_S0269964800000565 (visited on 04/21/2016).

Sgattoni, A. et al. (2015). piccante: arXiv:1503.02464. DOI: 10.5281/zenodo.16097. URL: http://zenodo.org/record/16097 (visited on 08/22/2016).

Shen, J. (1994). "Efficient Spectral-Galerkin Method I. Direct Solvers of Second- and Fourth-Order Equations Using Legendre Polynomials". In: SIAM Journal on Scientific Comput 15.6, pp. 1489–1505. ISSN: 1064-8275. DOI: 10.1137/0915089. URL: http://epubs.siam.org/doi/abs/10.1137/0915089 (visited on 04/21/2016).

Verboncoeur, J. P. (2005). "Particle simulation of plasmas: review and advances". en. In: Plasma Physics and Controlled Fusion 47.5A, A231. ISSN: 0741-3335. DOI: 10.1088/0741-3335/47/5A/017. URL: http://stacks.iop.org/0741-3335/47/i=5A/a=017 (visited on 07/21/2016).