Shortest PIC Code, As a Tutorial

Hua-sheng XIE (谢华生, huashengxie@gmail.com)
Department of Physics, Institute for Fusion Theory and Simulation,
Zhejiang University, Hangzhou, 310027, P.R.China
2012-01-01

Abstract

Providing an as short as possible PIC simulation code, which can be seen as an easy-following tutorial for beginners. However, my another motivation is to test the new method by solving Ampere's law instead of Poisson equation here.

1 Introduction

There are many good articles and books teach how to write a PIC code. Overviews can be found in [Birdsall1991], [Dawson1983], and [Hockney1988]. For space plasma, see [Büchner2003]; for laser, see [Wu2011]; on delta-f method, see [Sydora2003, 2007, 2009]. One can also find many good codes via the internet, e.g., [ES1D], [KEMPO] and [Lin_pic1d]. [PSTG] provides different more complicate codes, such as XOOPIC, which are widely applied, especially for low temperature and laser plasma. [Fitzpatrick2006] provided a simple tutorial. A free Pro evaluation version 2&1/2 EM PIC code [OOPIC] can be found in Tech-X Corporation's website. A similar simple PIC code as here is also provided in [PPLU], which is an integrated code with many modules for people to learn fundamental plasma physics. In fact, the MATLAB code provided in [Lapenta] has been very close to my final goal here.

2 PIC method

2.1 PIC cycle

Fig1 shows a typical PIC cycle.

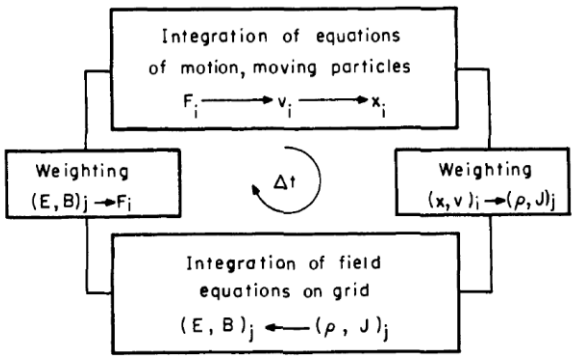

Figure 2-3a A typical cycle, one time step, in a particle simulation program. The particles are numbered $i = 1, 2, \ldots, NP$; the grid indices are j, which become vectors in 2 and 3 dimensions.

Fig1. A typical PIC cycle, from [Birdsall1991]

We need initial and diagnostic subroutines other than the above four steps. Since that these six steps are musts in a completed PIC code, to write the shortest PIC code, what we need do is to make each step as simple as possible. Then we'd better choose ES1D[1] model.

[1] As 1D model, many things are simplified a lot. And ES model, we do not need calculate the magnetic field $\mathbf{B}$. But, one should note that, in 1D system, the particles are not particles any more. In fact, they become charged planes if we see them in 3D.

2.2 Electrostatic 1D (ES1D)

Consider an unmagnetized, uniform, 1-dimensional plasma consisting of N electrons and N unit-charged ions. Now, ions are much more massive than electrons. Hence, on short time-scales, we can treat the ions as a static background, and only consider the motion of the electrons. Let x_i be the x-coordinate of the i-th electron. The equations of motion of the i-th electron are written (This part is mainly from [Fitzpatrick2006], for which is simple and well-documented):

$$\frac{dx_i}{dt} = v_i, \tag{0.1}$$

$$\frac{dv_i}{dt} = -\frac{eE(x_i)}{m_e}, \tag{0.2}$$

The field equation is Gauss's law (or, Poisson equation),

$$\frac{dE(x)}{dx} = \frac{e}{\varepsilon_0}\left[n_0 - n(x)\right]. \tag{0.3}$$

For simplicity, we solve (0.3) directly instead the traditional two steps method (0.4), i.e., to solve the Poisson equation and then to calculate the electric field $E(x)$,

$$\frac{d^2\phi(x)}{dx^2} = -\frac{e}{\varepsilon_0}\left[n_0 - n(x)\right], \quad E(x) = -\frac{d\phi(x)}{dx}. \tag{0.4}$$

2.3 Initial and normalization

We study two-stream instability as example.

$$f_0(x,v) = \frac{n_0}{2}\left\{\frac{1}{\sqrt{2\pi}v_{th}}e^{-(v-v_b)^2/2v_{th}^2} + \frac{1}{\sqrt{2\pi}v_{th}}e^{-(v+v_b)^2/2v_{th}^2}\right\}. \tag{0.5}$$

Time is normalized to ω_p^{-1}, where ω_p is the plasma frequency,

$$\omega_p^2 = \frac{n_0 e^2}{\varepsilon_0 m_e}. \tag{0.6}$$

Length is normalized to Debye length,

$$\lambda_D = \frac{v_{th}}{\omega_p}. \tag{0.7}$$

Density is normalized to n_0. The normalized equations take the form,

$$\frac{dx_i}{dt} = v_i, \tag{0.8}$$

$$\frac{dv_i}{dt} = -E(x_i), \tag{0.9}$$

$$\frac{dE(x)}{dt} = 1 - n(x). \tag{0.10}$$

The normalized unit velocity can be arbitrary and won't change the above normalized equations. However, for easy to

analysis the output results, we can use the thermal velocity v_{th}, then the initial distribution is (the coefficient is omitted)

$$f_0(x,v) = \frac{1}{2}\left[e^{-(v-v_b)^2} + e^{-(v+v_b)^2}\right]. \tag{0.11}$$

We solve the above system of equations in the domain $0 < x < L$, and consider periodic boundary conditions.

$$n(0) = n(L),\ \langle E(x)\rangle = 0. \tag{0.12}$$

Any particle which crosses the right boundary of the solution domain must reappear at the left boundary with the same velocity, and vice versa.

2.4 Particles to grid

This is the critical step for PIC. In order to obtain the electron number density $n(x)$ from the electron coordinates x_i we adopt a so-called particle-in-cell (PIC) approach. Let us define a set of J equally spaced spatial grid-points located at coordinates

$$x_j = j\Delta x \tag{0.13}$$

for $j = 0, J-1$, where $\Delta x = L/J$. Let $n_j \equiv n(x_j)$. Suppose that the i-th electron lies between the j-th and (j+1)-th grid-points: i.e., $x_j < x_i < x_{j+1}$. We let

$$n_j \to n_j + \left(\frac{x_{j+1} - x_i}{x_{j+1} - x_j}\right)/\Delta x, \tag{0.14}$$

and

$$n_{j+1} \to n_{j+1} + \left(\frac{x_i - x_{j+1}}{x_{j+1} - x_j}\right)/\Delta x, \tag{0.15}$$

Thus, $n_j \Delta x$ increases by 1 if the electron is at the j-th grid-point, $n_{j+1}\Delta x$ increases by 1 if the electron is at the (j+1)-th grid-point, and $n_j \Delta x$ and $n_{j+1}\Delta x$ both increase by 1/2 if the electron is halfway between the two grid-points, etc.

2.5 Grids to particle

Performing a similar assignment for each electron in turn allows us to build up the n_j from the electron coordinates (assuming that all the n_j are initialized to zero at the start of this process).

3 Example codes

In order to as simple as possible, we use MATLAB, which can shorten our code in great degree. For example, to give the initial Maxwellian (Gaussian) distribution, one need just use the randn() function. The code here is partly inherited from [Lapenta].

3.1 Pices1d_x.m

```
1 close all;clear all;clc;
```

```
2
3 % parameters
4 L=8*pi; dt=.1; nt=2000; ntout=100; ng=32; np=20000;
5 vb=1.0; vt=0.3; xp1=1.0e-1; vp1=0.0;
6 wp=1; qm=-1; mode=2;
7 q=wp^2/(qm*np/L); rho_back=-q*np/L; dx=L/ng;
8
9 % initial loading for the 2 Stream instability
10 xp=linspace(0,L,np)';
11 vp=vt*randn(np,1)+(1-2*mod([1:np]',2)).*vb;
12
13 % Perturbation
14 vp=vp+vp1*sin(2*pi*xp/L*mode);
15 xp=xp+xp1*(L/np)*sin(2*pi*xp/L*mode);
16 p=1:np;p=[p p];
17
18 % Main computational cycle
19 for it=0:nt
20    % diagnosing
21    if(mod(it,ntout)==0)
22       plot(xp,vp,'b.','Markersize',2);
23       axis([0,L,-3*(abs(vt)+abs(vb)),3*(abs(vt)+abs(vb))]);
24       title(['Phase space plotting, vp-x, t=',num2str(it*dt)]);
25       xlabel('xp');ylabel('vp'); pause(0.2);
26       print(gcf, '-dpng', ['vp-x,t=',num2str(it*dt),'.png']);
27    end
28
29    % update xp
30    xp=xp+vp*dt;
31
32    % apply periodic bc on the particle positions
33    xp=xp./L+10.0;
34    xp=L.*(xp-floor(xp));
35
36    % projection p->g
37    g1=floor(xp/dx-.5)+1;g=[g1;g1+1];
38    fraz1=1-abs(xp/dx-g1+.5);
39    fraz=[fraz1;1-fraz1];
40
41    % apply bc on the projection
42    out=(g<1);g(out)=g(out)+ng;
43    out=(g>ng);g(out)=g(out)-ng;
44    mat=sparse(p,g,fraz,np,ng);
45    rho=full((q/dx)*sum(mat))'+rho_back;
46
47    % computing fields, dE/dx
```

```
48      Eg=zeros(ng,1);
49      for k=1:ng-1
50          Eg(k+1)=Eg(k)+(rho(k)+rho(k+1))*dx/2;
51      end
52      Eg(1)=Eg(ng)+rho(ng)*dx;
53      Eg=Eg-mean(Eg);
54
55      % projection q->p and update of vp
56      vp=vp+mat*qm*Eg*dt;
57  end
```

Total code is less than 60 lines, included several blank and comment lines. However, it is still possible to shorten the lines: 16, 37-45 and 48-53. If one does not need the diagnosing part, then the lines 20-28 can also be deleted. If one likes, he/she can shorten this code to less than 30 lines.

3.2 Results

Phase space distribution.

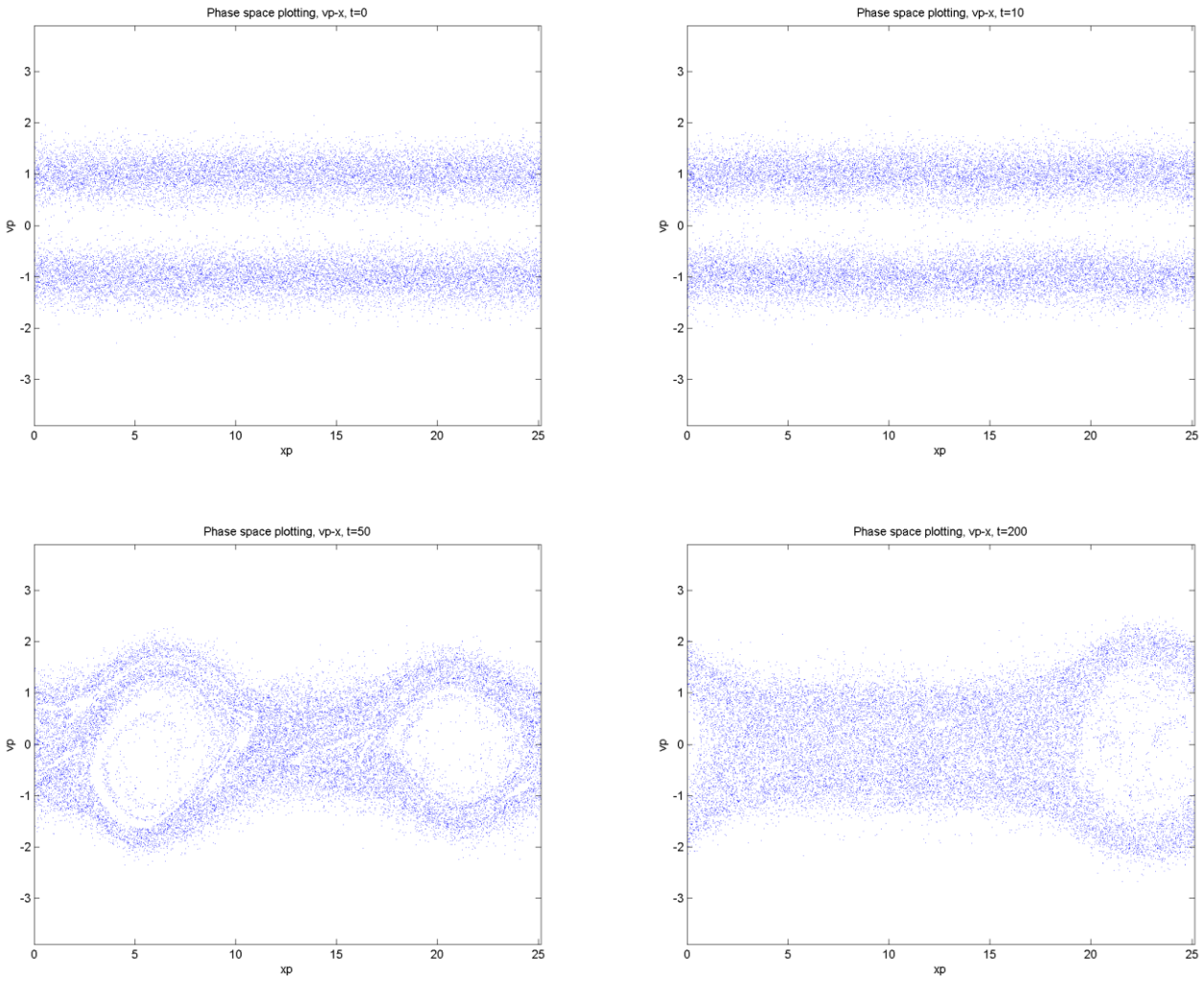

Fig2. Outputs of pices1d_x.m, phase space distribution of particles in two-stream PIC simulation at different time

4 Using Ampere's law

As I mentioned at the beginning, I hope to test the new method here, which by solving Ampere's law instead of Poisson equation.

When charges are conserved, i.e.,

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{0.16}$$

The Poisson equation (0.4) or Gauss's law (0.3) is equivalent to the below Ampere's law,

$$\frac{\partial \mathbf{E}}{\partial t} = -\frac{\mathbf{J}}{\varepsilon_0} \quad (\mathbf{J} = en\mathbf{u}), \tag{0.17}$$

.

5 References

[Birdsall1991] Birdsall, C. & Langdon, A., Plasma Physcis via Computer Simulation, IOP, 1991.

[Büchner2003] Büchner, J.; Dum, C. & Scholer, M., Space Plasma Simulation, Springer, 2003.

[Dawson1983] Dawson, J. M., Particle simulation of plasmas, Rev. Mod. Phys., American Physical Society, 1983, 55, 403-447.

[ES1D] http://www.crcpress.com/product/isbn/9780750310253

[Fitzpatrick2006] Fitzpatrick, R., Computational Physics: An introductory course The University of Texas at Austin, 2006, Chapter8, Particle-in-Cell Codes, http://farside.ph.utexas.edu/teaching/329/lectures/node96.html

[Hockney1988] Hockney & Eastwood, Computer Simulation using Particles, Adam Hilger, 1988.

[KEMPO] http://www.rish.kyoto-u.ac.jp/isss7/KEMPO/, EM1D.

[Lapenta] Lapenta, G., https://perswww.kuleuven.be/~u0052182/weather/pic.pdf , Particle In Cell Method: A brief description of the PIC Method.

[Lin_pic1d] http://phoenix.ps.uci.edu/zlin/pic1d/

[OOPIC] http://www.txcorp.com/downloads/index.php, OOPIC Pro evaluation version.

[PPLU] XIE, Hua-sheng, Plasma Physics Learn Utility, http://ifts.zju.edu.cn/forum/viewtopic.php?f=18&t=461, 2011.

[PSTG] http://ptsg.eecs.berkeley.edu/

[Sydora2003] Sydora, R. D., Low Noise Electrostatic and Electromagnetic Delta-f Particle-in-Cell Simulation of Plasmas, 2003, in [Büchner2003], P109.

[Sydora2007] Sydora, R. D. δf Particle-in-Cell Plasma Simulation Model: Properties and Applications, Advanced Methods for Space Simulations, edited by H. Usui and Y. Omura, pp. 47–60., 2007.

[Sydora2009] Sydora, R. D., Delta-F Particle-in-Cell Plasma Simulation Model 9th International School/Symposium for Space Simulations (ISSS9), University of Versailles-Saint-Quentin, France, July 4th, 2009, 2009.

[Wu2011] Wu, H.-C., JPIC & How to make a PIC code, arXiv, 2011, http://arxiv.org/abs/1104.3163.

[Test] http://ifts.zju.edu.cn/forum/viewtopic.php?f=18&t=258, solve the E field by different methods.

2012-01-01 15:25

huashengxie@gmail.com