E cient Fourier Basis Particle Simulation

Matthew S. Mitchell[a], Matthew T. Miecnikowski[a], Gregory Beylkin[b], Scott E. Parker[a]

[a]Department of Physics, University of Colorado, Boulder, CO 80303 USA
[b]Department of Applied Mathematics, University of Colorado, Boulder, CO 80303 USA

Abstract

The standard particle-in-cell algorithm su ers from grid heating. There exists a gridless alternative which bypasses the deposition step and calculates each Fourier mode of the charge density directly from the particle positions. We show that a gridless method can be computed e ciently through the use of an Unequally Spaced Fast Fourier Transform (USFFT) algorithm. After a spectral eld solve, the forces on the particles are calculated via the inverse USFFT (a rapid solution of an approximate linear system) [1, 2]. We provide one and two dimensional implementations of this algorithm with an asymptotic runtime of $O(N_p + N_m^D \log N_m^D)$ for each iteration, identical to the standard PIC algorithm (where N_p is the number of particles and N_m is the number of Fourier modes, and D is the spatial dimensionality of the problem) We demonstrate superior energy conservation and reduced noise, as well as convergence of the energy conservation at small time steps.

Keywords: Numerical, Plasma, Particle-in-cell, Energy conserving, Momentum conserving, Fourier transform

1. Introduction

A common approach to numerically solving the Vlasov-Poisson system is to represent the distribution function using particles, with elds solved on a grid and interpolated at the particle positions [3, 4]. This scheme, known as the particle-in-cell (PIC) method, has been enormously successful for simulating plasmas and is used in a wide variety of applications, but does not conserve total energy [5]. Energy-conserving schemes based on variational formulations have been proposed [6, 7], but generally do not conserve momentum because of a lack of translational invariance [8], though the momentum error can be kept small in many cases due to the choice of integrator [9].

In addition to the lack of energy conservation, PIC also su ers from a nite grid instability, in which su ciently high Fourier modes experience exponential growth [10, 11] due to coupling with lower modes. This phenomenon causes numerical heating, which saturates when the Debye length is on the order of the grid spacing [10, 8]. The nite grid instability is of particular relevance to problems involving cold plasmas or multiple length scales [10, 12]. Various approaches have been proposed to reduce this instability, such as the use of smoother particle shapes [13], grid jiggling [12], ltering [14], and high-order Galerkin methods [13], but many of these schemes su er from issues such as a lack of charge conservation and high computational costs.

Several energy-conserving particle algorithms [6, 7] based on the Lagrangian formulated by Low [15] have been suggested as alternatives to PIC. Charge-conserving approaches [9, 16] using implicit time integration have also been proposed. Energy-conserving algorithms have the bene t of eliminating numerical heating, but su er from several drawbacks. They generally do not conserve momentum [17], sometimes su er from increased noise, and heavily restrict the choice of particle shape [8]. However, these methods have seen widespread use due to their e ciency and simplicity.

It has been demonstrated that exact energy and momentum conservation in a particle code can be achieved by depositing charge on a truncated Fourier basis [8]. This method has also been shown to eliminate the nite grid instability and reduce coupling between modes [18]. However, due to the poor scaling and high computational cost of this approach ($O(N_m N_p)$, where N_m is the number of Fourier modes and N_p is the number of particles), it has not been seriously considered in practice.

We present a similar algorithm to the one proposed in [8], in which we model the charge density as a sum of shape functions in continuous space and perform the eld solve with a truncated Fourier series. However, we reduce the computation time to $O(N_p + N_m \log N_m)$, which is equivalent to that of conventional PIC with a spectral eld solve, by making use of an Unequally Spaced FFT (USFFT) [1, 2].

This paper is organized as follows: we begin by reviewing the standard PIC method in Section 2. We then propose a gridless algorithm based on [8] and demonstrate that it can be made e cient via the USFFT. In Section 3, we analyze the results of several numerical experiments, comparing our code to an

$$r \quad \mathrm{E} = \frac{}{0}: \tag{2}$$

2.1. Particle-in-Cell Method

The Vlasov-Poisson system is commonly solved via the particle-in-cell (PIC) algorithm [3, 4], which tracks particles in continuous phase space, while elds are tracked on a spatial grid with N_g points. Each timestep begins by calculating the charge density (x) at the grid points from the particle positions (denoted X_i). This is accomplished via a deposition of charge on the grid, such that

$$_{\mathbf{x}} = \sum_i qS(\mathrm{X}_i \quad \mathrm{x}) \tag{3}$$

where S is some shape function corresponding to the weighting scheme. E and are then calculated at the grid points from , usually via spectral methods, as follows:

$$\tilde{}_{\mathbf{k}} = \sum_{\mathbf{x}} {}_{\mathbf{x}} e^{\ \ i\mathbf{k}\ \mathbf{x}}; \tag{4}$$

$$\bar{\mathrm{E}}_{\mathbf{k}} = \frac{1}{i\mathrm{k}\ {}_0}\tilde{}_{\mathbf{k}}; \tag{5}$$

$$\tilde{}_{\mathbf{k}} = \frac{1}{k^2\ {}_0}\tilde{}_{\mathbf{k}}; \tag{6}$$

$$\mathrm{E}_{\mathbf{x}} = \sum_{\mathbf{k}} \bar{\mathrm{E}}_{\mathbf{k}} e^{i\mathbf{k}\ \mathbf{x}}: \tag{7}$$

After the eld solve, the forces on the particles are calculated by interpolating the E- eld from the grid to the particle positions as

$$\mathrm{F}_i = \sum_{\mathbf{x}} q\mathrm{E}_{\mathbf{x}} S(\mathrm{X}_i \quad \mathrm{x}): \tag{8}$$

In this paper, we assume without loss of generality that the same weighting scheme is used for both deposition and interpolation, but this need not be the case.

The particles are then pushed forward, necessitating discretization in time. Our implementation of PIC (as well as our algorithm, which we discuss in section 2.2) makes use of the leapfrog integrator. Position and velocity are tracked at alternating timesteps, as follows:

$$\mathrm{X}_i^{n+1} = \mathrm{X}^n$$

2.2. Particle-in-Fourier Method

Our algorithm, termed Particle-in-Fourier (or PIF), begins by following established gridless algorithms [8]. Instead of depositing the charge on a grid, we treat (x) as a sum of shape functions in continuous space and \deposit" on a truncated Fourier basis. This can be accomplished by calculating each mode of ~ directly from the particle positions, as

$$
\begin{aligned}
\tilde{}_{\mathbf{k}} &= \int_0^L dx\, e^{\;i\mathbf{k}\;\mathbf{x}}\; (x); \qquad (11)\\
&= \int_0^L dx\, e^{\;i\mathbf{k}\;\mathbf{x}} \sum_{i=1}^{N_p} qS(x \quad X_i
\end{aligned}
$$

2.3. Unequally Spaced Fast Fourier Transform (USFFT)

Our algorithm relies on rapid evaluation of two computationally expensive sums:

$$\sum_i e^{-i\mathbf{k}\cdot\mathbf{x}_i}, \tag{17}$$

which must be performed for every mode, and

$$\sum_{\mathbf{k}} S_{\mathbf{k}}\mathbf{E}_{\mathbf{k}} e^{i\mathbf{k}\cdot\mathbf{x}_i}, \tag{18}$$

which must be performed for every particle. Naively, computing these sums requires $O(N_p N_m^D)$ operations for N_p particles and N_m modes in each direction. However, we can reduce the computational cost by using an Unequally Spaced FFT (USFFT) [1, 2].

Let us briefly describe the basics of USFFT algorithm (in one dimension, for simplicity). We want to compute the values of the Fourier transform of a generalized function

$$f(x) = \sum c_m \delta(x - x_m) \tag{19}$$

that is,

$$\hat{f}(\xi) = \sum_m c_m e^{-i\xi x_m}, \tag{20}$$

in an interval $|\xi| \le c$. A generic approach is to replace f with a smooth (i.e. many times differentiable) periodic function g, such that its Fourier transform accurately approximates the Fourier transform $\hat{f}$ in the interval $|\xi| \le c$. Once the function g and its values are available, we can then use the FFT to evaluate the Fourier transform $\hat{g}$ (instead of $\hat{f}$) at an equally spaced grid. Note that, due to the required smoothness of g, $\hat{g}$ must decay rapidly.

If we were to have direct access to the Fourier domain, then it would easy to construct g by simply multiplying $\hat{f}$ by a smooth function $\hat{w}$ (the so-called window function), such that it is close to 1 for $|\xi| \le c$ and rapidly decays to zero for $|\xi| \ge c$. Multiplying by $\hat{w}$ is equivalent to the convolution with w in space, thus leading to a relatively fast algorithm suggested in [19]. However, if the transition of $\hat{w}$ from one to zero is rapid, then the convolution with w will be computationally expensive; if the transition is gradual, then the interval in the Fourier domain becomes large and, again, the computational cost is significant. An algorithmic solution is to construct the window $\hat{w}$ as a ratio of two functions, a rapidly decaying numerator that is applied in space as a convolution, and a denominator, that is applied in the Fourier domain as a "compensating" factor.

There are many possible choices for the numerator of $\hat{w}$. [2] uses a Gaussian

In one dimension, the USFFT algorithm computes the sum

$$\tilde{g}_n = \sum_{l=1}^{N_p} g_l e^{-2\pi i x_l n} \tag{21}$$

for all $-N_m/2 \le n < N_m/2$, where $x_l \in [-1/2; 1/2]$ (equivalent to $x_l \in [0; 1]$) and g_l are arbitrary complex coefficients. The dual version of the USFFT rapidly evaluates the sum

$$g(x_l) = \sum^{N_m/2-1}$$

using OpenMP (see Section 5 for more details). Source code can be found at `https://github.com/matt2718/ftpic`.

The superior energy conservation of PIF becomes evident when we examine the the two stream instability. The 1D implementations of PIF and PIC were used to simulate the mixing of two counter-moving electron beams against a neutralizing background. Both codes were run with 10000 particles and 16 modes/grid cells for 20000 time steps, with the e ective grid spacing equal to approximately $0{:}88\ {}_D$. Fig. 1 plots the normalized eld, kinetic, and total energy of each system. It is clear that the total energy is not conserved for conventional PIC. Fig. 2 illustrates the total energy over time for both simulations, demonstrating energy conservation in PIF, but not in PIC. The peak in the energy

Fig. 2: Comparison of total energy over time in 1D PIF and PIC simulations of the two stream instability, demonstrating the superior energy conservation of PIF. For PIF, the maximum deviation from the initial energy was 4 10 ^{6}.

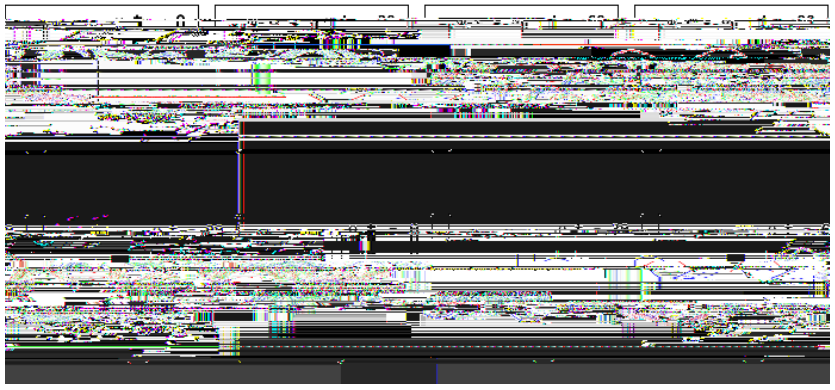

Fig. 3: Time evolpn/F25 5.9776gDo Q Q424o Q Q4phas30(olpnspac30(olpnfF,)-olpni)-1(olpn(top)1(olpnIC)-2olpni)F1(olpn(b

Fig. 6: Comparison of total energy over time in 2D PIF and PIC simulations of Landau damping, demonstrating the superior energy conservation of PIF in two dimensions.

and

$$
\begin{aligned}
\frac{d}{dt}T &= \sum_i m\underline{\mathrm{v}}_i \mathrm{v}_i && (29)\\
&= \sum_i \left(\sum_k q S_k E_k e^{ikX_i} \right) \mathrm{v}_i && (30)\\
&= \sum_i \sum_k \frac{q\mathrm{v}_i}{ik\ _0} S_k \tilde{\ }_k e^{ikX_i}. && (31)
\end{aligned}
$$

Eq. (31) is the negative complex conjugate of Eq. (28). Because the eld quantities are Hermitian in Fourier space, and the sum in k runs from $\frac{2}{L}N_m$ to $\frac{2}{L}N_m$ (and because both the kinetic and eld energies must be real), Eqs. (28) and (31) must sum to 0, so total energy is conserved.

We have shown energy conservation in the continuous time limit, but an actual simulation involves discretization in the time domain. Numerically, we observe that the global truncation timestep error in the total energy of PIF converges as $O(\ t^2)$, in agreement with [8] and [18]. This holds in any number

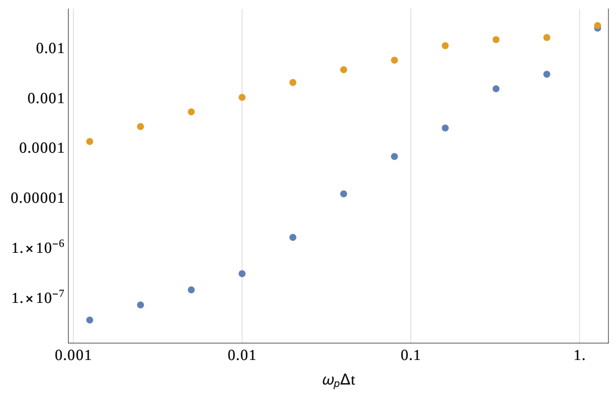

Fig. 7: 1D relative global error with respect to timestep size for a Landau damping test case with 32 grid cells/modes and 20,000 particles. In agreement with [8], we see global convergence of roughly $O(\ t^2)$ for PIF in the asymptotic regime.

Fig. 8: 2D global error with respect to timestep size for a Landau damping test case with 32 32 grid cells/modes and 20,000 particles, also in agreement with literature.

Though we only provide one and two-dimensional implementations, the same scheme can be extended to three dimensions, while still scaling well. In any number of dimensions, the generalized USFFT requires $O(N_p + N_m^D \log N_m^D)$ time (where D is the dimensionality of the system). As an ordinary D-dimensional FFT requires $O(N_m^D \log N_m^D)$ time, the scaling is identical to that of PIC regardless of the dimensionality.

In addition, because convolution with an arbitrary shape function corresponds to multiplication in Fourier space, increasing the width of the shape function beyond a single grid cell requires no additional time. This can provide a huge advantage for PIF as higher order shape functions (even Gaussian) are essentially free, whereas in PIC, higher order shape functions involve many more calculations and di cult-to-parallelize scatter operations.

For reasonable parameters, the N_p term dominates over the $N_m \log N_m$ term.

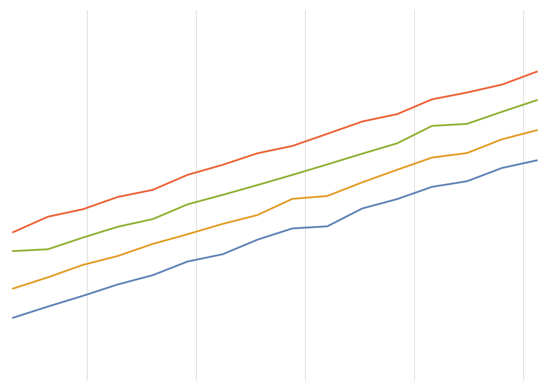

Fig. 9: 1D PIF performance with respect to particle number and mode number. We see asymptotic time complexity approximating $O(N_p + N_m \log N_m)$ as predicted by theory.

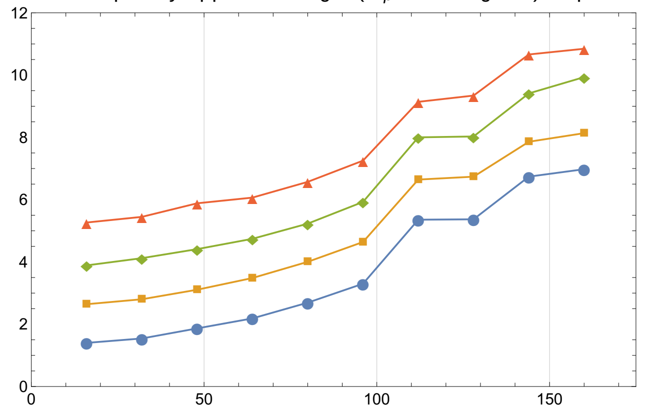

Fig. 10: 2D PIF performance with respect to particle number and mode number. As in the 1D case, we see agreement with the theoretical scaling of $O(N_p + N_m^2 \log N_m^2)$.

This suggests the following parallelization scheme for PIF (similar to one commonly used for PIC): we divide the particles between nodes, with each node performing the eld solve for its own particles (i.e. each node evaluates the serial USFFT independently for a fraction of the particles). Before the push step, the E- elds from every node are summed together. To test this scheme, we created a simple shared memory implementation of it in the one dimensional versions of PIF and PIC.

In order to verify that this scaling holds, the two 1D codes were run for 1000 time steps with 80000 particles and 64 grid cells/mode on a single 68-core Xeon Phi node of the Cori supercomputer at the National Energy Research Scienti c Computing Center (NERSC). Both codes were compiled with GCC. The number of OpenMP threads was varied from 1 to 64, with the problem size kept constant. For each data point, the code was run 4 times, with the variation in runtime indicated by the error bars in Fig. 11. Both algorithms exhibited similar strong scaling, as shown in Fig. 11. On average, PIF required 2.9 times as much time as PIC, and the two algorithms demonstrated comparable scaling.

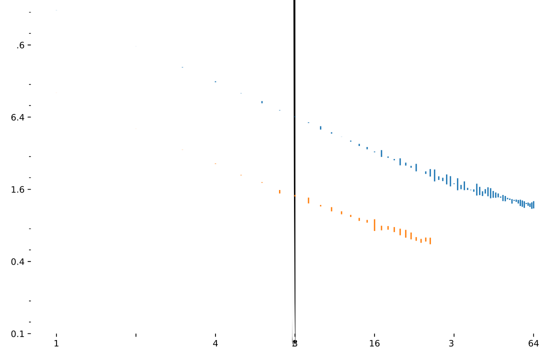

Fig. 11: 1D PIF has comparable strong scaling to 1D PIC as the number of threads is increased from 1 to 64, with 80000 particles and 64 grid cells.

Weak scaling was investigated by increasing the problem size, such that each run had 20000 particles per thread. The number of grid cells was kept constant between runs. Results are shown in Fig. 12. We see that PIF demonstrates better weak scaling than our implementation of PIC for large numbers of particles. We suspect that this is because of the di culty of parallelizing the deposition step due to the scatter operations involved.

Fig. 12: 1D PIF has superior weak scaling to 1D PIC as the number of threads is increased from 1 to 64, with 20000 particles/thread and 64 grid cells.

6. Discussion

We have demonstrated that the PIF gridless scheme is a feasible approach to plasma simulation, as it can be implemented with comparable performance and identical scaling to the conventional PIC method while conserving both energy and momentum in the continuous-time limit. We have provided an analysis of these conservation properties and veri ed them through several numerical

[3] C. K. Birdsall, A. B. Langdon, Plasma physics via computer simulation, McGraw-Hill, New York, 1985.

[4] R. W. Hockney, J. W. Eastwood, Computer simulation using particles, special student ed Edition, A. Hilger, Bristol [England] ; Philadelphia, 1988.

[5] A. B. Langdon, E ects of the spatial grid in simulation plasmas, Journal of Computational Physics 6 (2) (1970) 247{267. doi:10.1016/0021-9991(70)90024-0.
URL http://www.sciencedirect.com/science/article/pii/0021999170900240

[6] H. R. Lewis, Energy-conserving numerical approximations for Vlasov plasmas, Journal of Computational Physics 6 (1) (1970) 136{141. doi:10.1016/0021-9991(70)90012-4.
URL http://www.sciencedirect.com/science/article/pii/0021999170900124

[7] J. W. Eastwood, The virtual particle electromagnetic particle-mesh method, Computer Physics Communications 64 (2) (1991) 252{266. doi:10.1016/0010-4655(91)90036-K.
URL http://www.sciencedirect.com/science/article/pii/001046559190036K

[8] E. G. Evstatiev, B. A. Shadwick, Variational formulation of particle algorithms for kinetic plasma simulations, Journal of Computational Physics 245 (2013) 376{398. doi:10.1016/j.jcp.2013.03.006.
URL http://www.sciencedirect.com/science/article/pii/S0021999113001800

[9] G. Chen, L. Chacn, D. C. Barnes, An energy- and charge-conserving, implicit, electrostatic particle-in-cell algorithm, Journal of Computational Physics 230 (18) (2011) 7018{7036. doi:10.1016/j.jcp.2011.05.031.
URL http://www.sciencedirect.com/science/article/pii/S0021999111003421

[10] H. Okuda, Nonphysical noises and instabilities in plasma simulation due to a spatial grid, Journal of Computational Physics 10 (3) (1972) 475{486.

[20] S. E. Parker, W. W. Lee, A fully nonlinear characteristic method for gyrokinetic simulation, Physics of Fluids B: Plasma Physics 5 (1) (1993) 77{86. doi:10.1063/1.860870.
URL https://aip.scitation.org/doi/abs/10.1063/1.860870