

Efficient 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 suffers 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 efficiently through the use of an Unequally Spaced Fast Fourier Transform (USFFT) algorithm. After a spectral field 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 fields 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 suffers from a finite grid instability, in which sufficiently 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 finite 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], filtering [14], and high-order Galerkin methods [13], but many of these schemes suffer 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 benefit of eliminating numerical heating, but suffer from several drawbacks. They generally do not conserve momentum [17], sometimes suffer from increased noise, and heavily restrict the choice of particle shape [8]. However, these methods have seen widespread use due to their efficiency 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 finite 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 field 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 field 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 efficient via the USFFT. In Section 3, we analyze the results of several numerical experiments, comparing our code to an implementation of conventional PIC and providing experimental confirmation of its energy conservation. A proof of energy conservation in the continuous-time limit, as well as momentum conservation, is given in Section 4. In Section 5, we present an analysis of the algorithm's asymptotic time complexity (in both one and two dimensions) and a possible parallelization scheme along with the results of several scaling experiments. Section 6 contains a summary of our results and a discussion of possible generalizations.

2. Numerical Methods

For a collisionless electrostatic plasma, the time evolution of the distribution function $f(\mathbf{x}, \mathbf{v})$ is given by the Vlasov-Poisson system:

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \frac{\partial f}{\partial \mathbf{x}} + \frac{q}{m} \mathbf{E} \cdot \frac{\partial f}{\partial \mathbf{v}} = 0, \quad (1)$$

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\epsilon_0}. \quad (2)$$

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 fields are tracked on a spatial grid with N_g points. Each timestep begins by calculating the charge density $\rho(x)$ at the grid points from the particle positions (denoted \mathbf{X}_i). This is accomplished via a deposition of charge on the grid, such that

$$\rho_{\mathbf{x}} = \sum_i q S(\mathbf{X}_i - \mathbf{x}) \quad (3)$$

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

$$\tilde{\rho}_{\mathbf{k}} = \sum_{\mathbf{x}} \rho_{\mathbf{x}} e^{-i\mathbf{k} \cdot \mathbf{x}}, \quad (4)$$

$$\tilde{\mathbf{E}}_{\mathbf{k}} = \frac{1}{i\mathbf{k}\epsilon_0} \tilde{\rho}_{\mathbf{k}}, \quad (5)$$

$$\tilde{\phi}_{\mathbf{k}} = \frac{1}{k^2 \epsilon_0} \tilde{\rho}_{\mathbf{k}}, \quad (6)$$

$$\mathbf{E}_{\mathbf{x}} = \sum_{\mathbf{k}} \tilde{\mathbf{E}}_{\mathbf{k}} e^{i\mathbf{k} \cdot \mathbf{x}}. \quad (7)$$

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

$$\mathbf{F}_i = \sum_{\mathbf{x}} q \mathbf{E}_{\mathbf{x}} S(\mathbf{X}_i - \mathbf{x}). \quad (8)$$

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:

$$\mathbf{X}_i^{n+1} = \mathbf{X}_i^n + \mathbf{v}_i^{n+1/2} \Delta t \quad (9)$$

$$\mathbf{v}_i^{n+1/2} = \mathbf{v}_i^{n-1/2} + \frac{\mathbf{F}_i^n}{m} \Delta t. \quad (10)$$

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 $\rho(\mathbf{x})$ 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 $\tilde{\rho}$ directly from the particle positions, as

$$\tilde{\rho}_{\mathbf{k}} = \int_0^L d\mathbf{x} e^{-i\mathbf{k}\cdot\mathbf{x}} \rho(\mathbf{x}), \quad (11)$$

$$= \int_0^L d\mathbf{x} e^{-i\mathbf{k}\cdot\mathbf{x}} \sum_{i=1}^{N_p} q S(\mathbf{x} - \mathbf{X}_i), \quad (12)$$

$$= q \tilde{S}_{\mathbf{k}} \sum_{i=1}^{N_p} e^{-i\mathbf{k}\cdot\mathbf{X}_i}, \quad (13)$$

where \mathbf{k} is of the form $2\pi(\frac{n_x}{L_x}\hat{\mathbf{x}} + \frac{n_y}{L_y}\hat{\mathbf{y}} + \frac{n_z}{L_z}\hat{\mathbf{z}})$, for integers $n_x, n_y, n_z \in [-\frac{N_m}{2}, \frac{N_m}{2}]$, N_m being the number of modes (analogous to the number of grid cells in PIC). $\tilde{S}_{\mathbf{k}}$ denotes the Fourier transform of the shape function, given by

$$\tilde{S}_{\mathbf{k}} = \int_0^L d\mathbf{x} e^{-i\mathbf{k}\cdot\mathbf{x}} S(\mathbf{x}). \quad (14)$$

A unique feature of the PIF method is the ability to use a physical particle shape given by $S(\mathbf{x})$ as an alternative to filtering. A natural choice is a Gaussian particle shape with a spatial width chosen as a filtering parameter. Alternatively, a physical delta-function particle shape can be used without any mathematical complication. It is important to note that there is no computational penalty for using higher order shape functions with PIF. Because convolution with the shape function is simply a multiplication in Fourier space and $\tilde{S}_{\mathbf{k}}$ is typically evaluated analytically (and the results can be reused between timesteps), the step time is independent of the weighting scheme. This permits the use of arbitrary shape functions with no additional computational cost.

We now perform the field solve in a manner identical to conventional PIC, finding $\tilde{\mathbf{E}}_{\mathbf{k}}$ and $\tilde{\phi}_{\mathbf{k}}$ from Eq. (5) and (6). Then, using a similar approach to the deposition, we can determine the forces on the particles by summing over all modes of \mathbf{E} :

$$\mathbf{F}_i = \int_0^L d\mathbf{x} q \mathbf{E}(\mathbf{x}) S(\mathbf{x} - \mathbf{X}_i), \quad (15)$$

$$= \sum_{\mathbf{k}} e^{i\mathbf{k}\cdot\mathbf{X}_i} q \tilde{\mathbf{E}}_{\mathbf{k}} \tilde{S}_{\mathbf{k}}. \quad (16)$$

2.3. Unequally Spaced Fast Fourier Transform (USFFT)

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

$$\sum_i e^{-i\mathbf{k}\cdot\mathbf{X}_i}, \quad (17)$$

which must be performed for every mode, and

$$\sum_{\mathbf{k}} \tilde{S}_{\mathbf{k}} \tilde{\mathbf{E}}_{\mathbf{k}} e^{i\mathbf{k}\cdot\mathbf{X}_i}, \quad (18)$$

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

$$f(x) = \sum c_m \delta(x - x_m) \quad (19)$$

that is,

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

in an interval $|\xi| \leq c$. 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 $|\xi| \leq c$. 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 be 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 $|\xi| \leq c$ and rapidly decays to zero for $|\xi| > c$. 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 (however, the accuracy estimates in this paper are overly pessimistic), while the USFFT uses b-splines [1]. The gain in speed using this approach is significant and it is used in higher dimensions in a similar manner. Importantly, the error is controlled (see [1]) so that the algorithm can be used as a ‘‘black box.’’

In one dimension, the USFFT algorithm computes the sum

$$\tilde{g}_n = \sum_{l=1}^{N_p} g_l e^{\pm 2\pi i x_l n} \quad (21)$$

for all $-N_m/2 \leq n \leq N_m/2$, 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

$$g(x_l) = \sum_{n=-N_m/2}^{N_m/2-1} \tilde{g}_n e^{\pm 2\pi i x_l n}, \quad (22)$$

where \tilde{g}_n are arbitrary complex coefficients.

The computational cost of the USFFT is $O(N_p + N_m \log N_m)$ in one dimension, and $O(N_p + N_m^D \log N_m^D)$ for the higher-dimensional implementations. It is important that the accuracy of the USFFT is user-controlled, and since the accuracy is guaranteed by the algorithm, one can use the USFFT in a manner similar to a conventional FFT (tight accuracy estimates can be found in [1]).

The USFFT algorithm uses the FFT as one of the steps, and for this reason, it is helpful to estimate the computational cost of the USFFT in terms of that of the FFT. The double precision USFFT in one dimension is approximately 2.5 – 3 times slower than the FFT of the same size. In two dimensions, the factor is approximately 10 – 20.

We use the USFFT by setting $g_l = 1$ and $x_l = X_l/L$ for all particles in 1D, so that we can efficiently compute the sum in Eq. (13). Similarly, by taking \tilde{g}_i as the amplitude of each mode of the electric field, we can compute the sum in Eq. (16) (in two and three dimensions, these must be done separately for each component of the electric field). As a result, PIF has the same computational complexity as standard PIC. We analyze this further in Section 5.

3. Numerical Tests

In this section, we test the behavior of the two methods on two classic kinetic problems in plasma physics for which PIC has been well verified with theory, namely the two stream instability and linear Landau damping. Results for the two stream instability presented in Sec. 3.1 are using a 1D implementation. Results for Landau damping are presented for both 1D and 2D implementations in Sec. 3.2.

3.1. Two Stream Instability

An implementation of PIF in one and two dimensions was created, using a serial implementation of the USFFT [1]. The performance and energy conservation of the code were compared to an implementation of conventional 1D and 2D PIC with a linear particle shape, first order interpolation, and a spectral field solve. Shared memory parallelism was implemented in both codes

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 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 effective grid spacing equal to approximately $0.88\lambda_D$. Fig. 1 plots the normalized field, 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 error shown in Fig. 2 happens at nonlinear saturation. Although the number of grid cells used is lower than would be seen in practice, this was chosen to simply illustrate a specific result—that PIF conserves energy *regardless of the number of modes used*, and requires fewer modes to reproduce physical behavior.

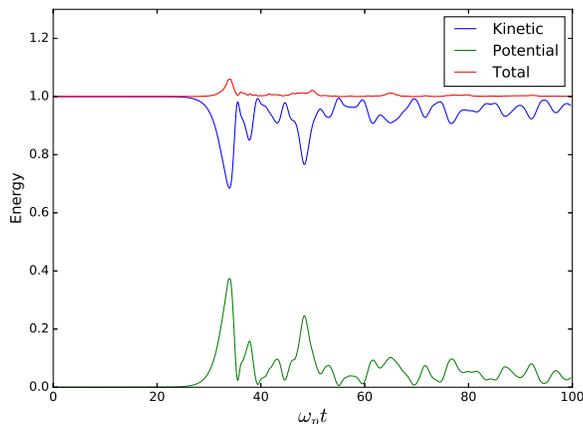


Fig. 1: Kinetic, potential, and total energy for the PIC method applied to the 1D two stream instability. Total energy is not conserved.

Aside from the difference in energy conservation, the two codes produced similar results. Fig. 3 illustrates the evolution of phase space over time in the PIF and PIC simulations, which we see produce qualitatively similar results. The two beams are represented by different colors, but all particles are identical. Furthermore, momentum was conserved within both the PIF and PIC simulations to within floating point error.

3.2. Landau Damping

Our 1D implementations of PIF and conventional PIC were used to simulate a Maxwellian plasma of 10000 electrons with a neutralizing ion background, with a sinusoidal density perturbation of the second Fourier mode given by $n_1 = a \cos(4\pi x/L)$. This was performed for both 32 grid cells in PIC and 32

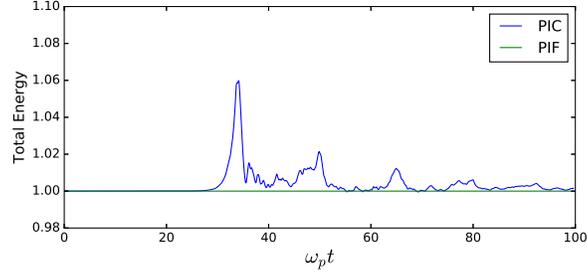


Fig. 2: Comparison of total energy over time in 1D PIF and PIC simulations of the two stream instability, demonstrating the superior energy conservation of PIF. For PIF, the maximum deviation from the initial energy was 4×10^{-6} .

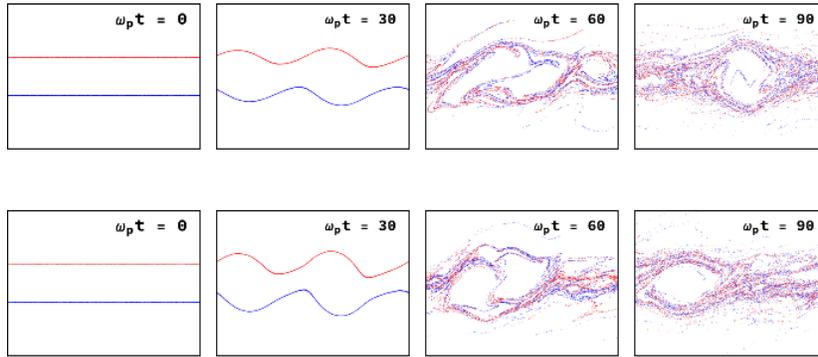


Fig. 3: Time evolution of phase space for PIC (top) and PIF (bottom) simulations of a two stream instability. The beams are represented by different colors to demonstrate mixing, but all particles are identical.

modes in PIF ($\Delta x = 1.0\lambda_D$) and for 128 grid cells in PIC and 128 modes in PIF ($\Delta x = 0.25\lambda_D$). The observed damping rate of the second mode of the E-field was compared to the theoretical rate, illustrated in Fig. 4 and Fig. 5.

At small grid spacings, we see that the results of the PIC simulation converge to those of PIF. PIF, however, accurately reproduces the theoretical damping rate regardless of the number of modes present. This is a somewhat contrived example (in practice, the grid spacing would not be equal to the Debye length), but it demonstrates that PIF can accurately reproduce physical behavior even when few modes are kept.

A similar test case was run in two dimensions with 10000 particles, 128 grid cells/modes, and $\Delta x = 0.25\lambda_D$. Similar behavior was observed, and significant grid heating was found in the two dimensional version of PIC, while PIF demonstrated perfect energy conservation (Fig. 6).

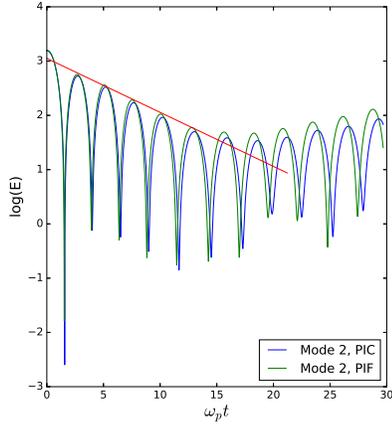


Fig. 4: Landau damping of the 2nd Fourier mode in a 32-cell simulations. We see agreement between the theoretical rate and the rate produced by PIF, but not that of PIC.

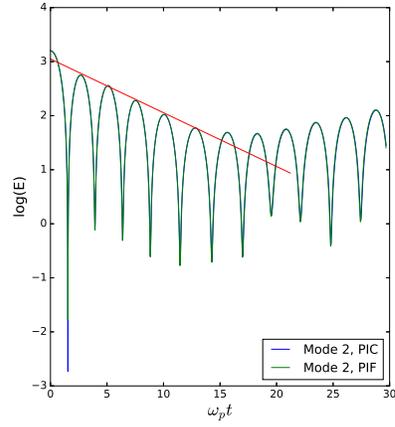


Fig. 5: Landau damping of the 2nd Fourier mode in a 128-cell simulations. The results of both algorithms agree with the damping rate predicted by theory.

4. Conservation Properties and Numerical Error

It is well-established that the standard PIC algorithm does not conserve energy; a full proof is given by Langdon [5]. We can prove that PIF conserves energy in the continuous-time limit by taking the field and kinetic energies to be

$$U_E = \frac{1}{2} \sum_k \tilde{\rho}_k \tilde{\phi}_k, \quad (23)$$

$$T = \sum_i \frac{1}{2} m \mathbf{v}_i^2, \quad (24)$$

and expanding their time derivatives as

$$\frac{d}{dt} U_E = \frac{d}{dt} \left(\frac{1}{2} \sum_k \tilde{\rho}_k \frac{\tilde{\rho}_k}{\epsilon_0 k^2} \right) \quad (25)$$

$$= \sum_k \frac{1}{\epsilon_0 k^2} \tilde{\rho}_k \dot{\tilde{\rho}}_k \quad (26)$$

$$= \sum_k \frac{1}{\epsilon_0 k^2} \tilde{\rho}_k \frac{d}{dt} \left(\sum_i q \tilde{S}_k e^{-ikX_i} \right) \quad (27)$$

$$= - \sum_i \sum_k \frac{q \mathbf{v}_i}{ik \epsilon_0} \tilde{S}_k \tilde{\rho}_k e^{-ikX_i}, \quad (28)$$

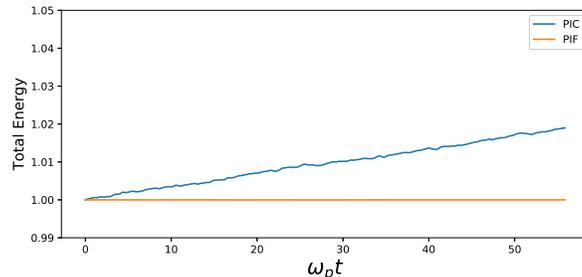


Fig. 6: Comparison of total energy over time in 2D PIF and PIC simulations of Landau damping, demonstrating the superior energy conservation of PIF in two dimensions.

and

$$\frac{d}{dt}T = \sum_i m \dot{\mathbf{v}}_i \mathbf{v}_i \quad (29)$$

$$= \sum_i \left(\sum_k q \tilde{S}_k \tilde{E}_k e^{ikX_i} \right) \mathbf{v}_i \quad (30)$$

$$= \sum_i \sum_k \frac{q \mathbf{v}_i}{ik\epsilon_0} \tilde{S}_k \tilde{\rho}_k e^{ikX_i}. \quad (31)$$

Eq. (31) is the negative complex conjugate of Eq. (28). Because the field quantities are Hermitian in Fourier space, and the sum in k runs from $-\frac{2\pi}{L}N_m$ to $\frac{2\pi}{L}N_m$ (and because both the kinetic and field 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 $O(\Delta t^2)$, in agreement with [8] and [18]. This holds in any number of dimensions, as illustrated in Fig. 7 and Fig. 8. Note that, although PIC is also quadratically convergent, this is only true for a sufficiently fine grid, while PIF displays quadratic convergence independent of the number of modes. For the parameters chosen here, we see that PIC and PIF disagree.

5. Performance and Scaling

The asymptotic runtime of the USFFT algorithm is $O(N_p + N_m \log N_m)$ [1]. Because the field solve and the particle push take $O(N_m)$ and $O(N_p)$ time respectively, the total asymptotic runtime of the particle-in-Fourier algorithm is $O(N_p + N_m \log N_m)$ for a single iteration. This is identical to the standard PIC algorithm, assuming the field solve is done spectrally. This scaling is demonstrated numerically in Fig. 9 and 10.

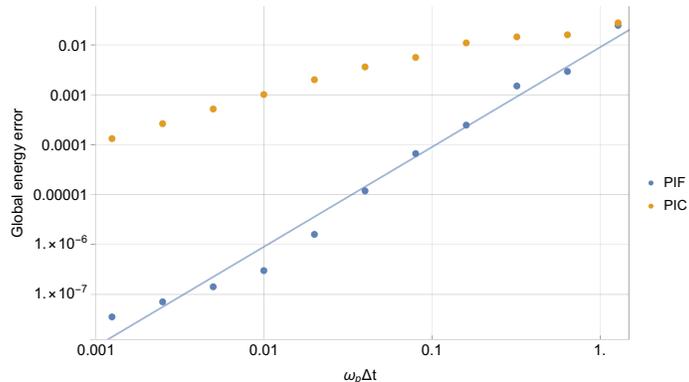


Fig. 7: 1D relative global error with respect to timestep size for a Landau damping test case with 32 grid cells/modes and 20,000 particles. In agreement with [8], we see global convergence of roughly $O(\Delta t^2)$ for PIF in the asymptotic regime.

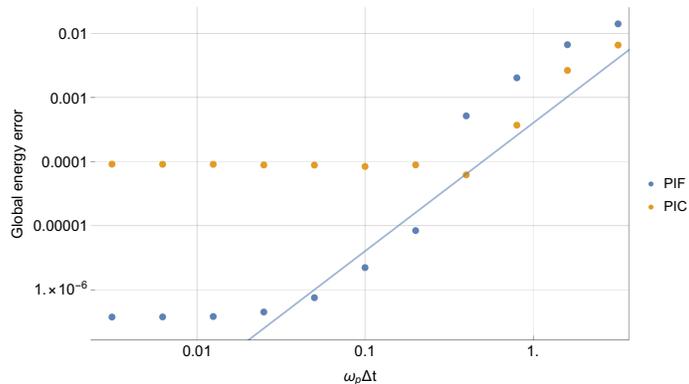


Fig. 8: 2D global error with respect to timestep size for a Landau damping test case with 32×32 grid cells/modes and 20,000 particles, also in agreement with literature.

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 difficult-to-parallelize scatter operations.

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

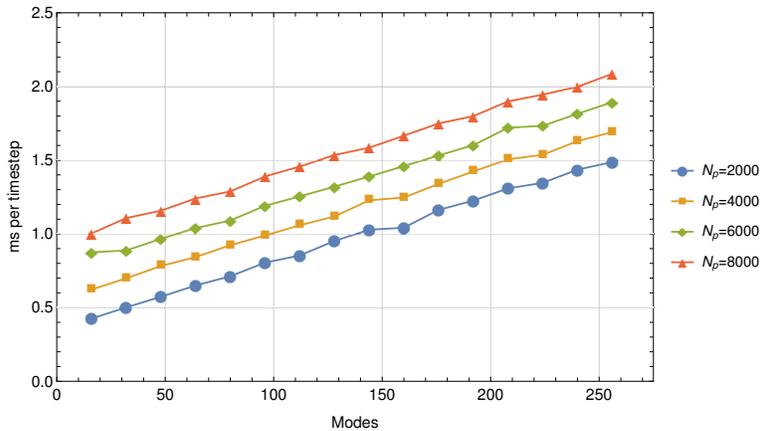


Fig. 9: 1D PIF performance with respect to particle number and mode number. We see asymptotic time complexity approximating $O(N_p + N_m \log N_m)$ as predicted by theory.

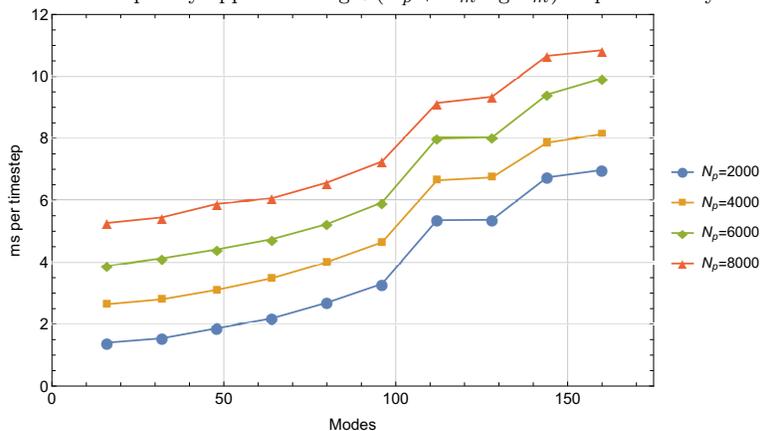


Fig. 10: 2D PIF performance with respect to particle number and mode number. As in the 1D case, we see agreement with the theoretical scaling of $O(N_p + N_m^2 \log N_m^2)$.

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 field 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-fields 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 Scientific 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.

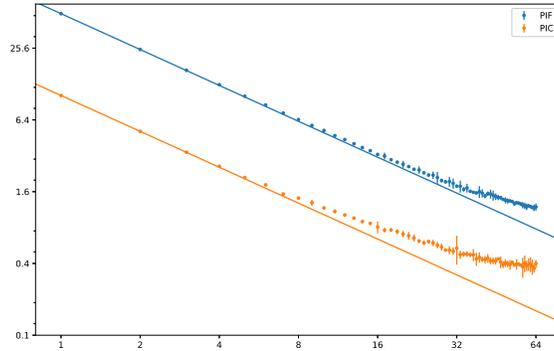


Fig. 11: 1D PIF has comparable strong scaling to 1D PIC as the number of threads is increased from 1 to 64, with 80000 particles and 64 grid cells.

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 difficulty of parallelizing the deposition step due to the scatter operations involved.

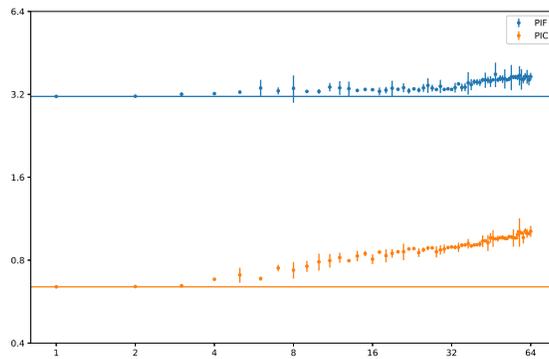


Fig. 12: 1D PIF has superior weak scaling to 1D PIC as the number of threads is increased from 1 to 64, with 20000 particles/thread and 64 grid cells.

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 verified them through several numerical experiments. In addition, we have verified that the results of the PIF model agree with theory for standard test cases such as Landau damping and the two stream instability.

We did find that PIF was slower than a reference implementation of PIC by a factor of approximately 2.9 in the 1D case and 6.8 in the 2D case. However, PIF demonstrated better weak and strong scaling by avoiding costly scatter operations during the deposition step. Furthermore, the computational cost of PIF is independent of the particle shape function, permitting the use of higher order or Gaussian weighting schemes without additional time.

We expect that the PIF algorithm can be generalized to three dimensional energy conserving electrostatic and electromagnetic models, as the field solves can still be performed with spectral methods [3]. This can be accomplished through the use of three dimensional USFFTs, still with a computational cost of $O(N_p + N_m^D \log N_m^D)$ [1]. In addition, we expect that our method could be generalized to δf based codes [20], which are of particular interest to fusion plasmas.

In some applications, the simulation model may be periodic in one or more directions, and one could use a PIF representation in the periodic directions and PIC in the non-periodic ones. For example, in a toroidal fusion plasma, a Fourier representation in the toroidal direction is appropriate. Such a mixed basis approach needs to be explored further.

Acknowledgments

Research supported by the United States Department of Energy under grants DE-FG02-08ER54954 and de-sc0008801.

References

- [1] G. Beylkin, On the Fast Fourier Transform of Functions with Singularities, *Applied and Computational Harmonic Analysis* 2 (4) (1995) 363–381. doi:10.1006/acha.1995.1026.
URL <http://www.sciencedirect.com/science/article/pii/S1063520385710263>
- [2] A. Dutt, V. Rokhlin, Fast Fourier Transforms for Nonequispaced Data, *SIAM Journal on Scientific Computing* 14 (6) (1993) 1368–1393. doi: 10.1137/0914081.
URL <https://epubs.siam.org/doi/abs/10.1137/0914081>

- [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, Effects 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/S0021999170900240>
- [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/S0021999170900124>
- [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/S001046559190036K>
- [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. doi:10.1016/0021-9991(72)90048-4.
URL <http://www.sciencedirect.com/science/article/pii/S0021999172900484>
- [11] E. L. Lindman, Dispersion relation for computer-simulated plasmas, *Journal of Computational Physics* 5 (1) (1970) 13–22. doi:10.1016/0021-9991(70)90049-5.
URL <http://www.sciencedirect.com/science/article/pii/S0021999170900495>

- [12] J. W. Brackbill, G. Lapenta, A Method to Suppress the Finite-Grid Instability in Plasma Simulations, *Journal of Computational Physics* 114 (1) (1994) 77–84. doi:10.1006/jcph.1994.1150.
URL <http://www.sciencedirect.com/science/article/pii/S0021999184711508>
- [13] G. B. Jacobs, J. S. Hesthaven, High-order nodal discontinuous Galerkin particle-in-cell method on unstructured grids, *Journal of Computational Physics* 214 (1) (2006) 96–121. doi:10.1016/j.jcp.2005.09.008.
URL <http://www.sciencedirect.com/science/article/pii/S0021999105004250>
- [14] R. W. Hockney, S. P. Goel, J. W. Eastwood, Quiet high-resolution computer models of a plasma, *Journal of Computational Physics* 14 (2) (1974) 148–158. doi:10.1016/0021-9991(74)90010-2.
URL <http://www.sciencedirect.com/science/article/pii/S0021999174900102>
- [15] F. E. Low, A Lagrangian formulation of the Boltzmann-Vlasov equation for plasmas, *Proc. R. Soc. Lond. A* 248 (1253) (1958) 282–287. doi:10.1098/rspa.1958.0244.
URL <http://rspa.royalsocietypublishing.org/content/248/1253/282>
- [16] S. Markidis, G. Lapenta, The energy conserving particle-in-cell method, *Journal of Computational Physics* 230 (18) (2011) 7037–7052. doi:10.1016/j.jcp.2011.05.033.
URL <http://www.sciencedirect.com/science/article/pii/S0021999111003445>
- [17] A. B. Langdon, Energy-conserving plasma simulation algorithms, *Journal of Computational Physics* 12 (2) (1973) 247–268. doi:10.1016/S0021-9991(73)80014-2.
URL <http://www.sciencedirect.com/science/article/pii/S0021999173800142>
- [18] C. K. Huang, Y. Zeng, Y. Wang, M. D. Meyers, S. Yi, B. J. Albright, Finite grid instability and spectral fidelity of the electrostatic Particle-In-Cell algorithm, *Computer Physics Communications* 207 (2016) 123–135. doi:10.1016/j.cpc.2016.05.021.
URL <http://www.sciencedirect.com/science/article/pii/S001046551630145X>
- [19] W. H. Press, G. B. Rybicki, Fast algorithm for spectral analysis of unevenly sampled data, *The Astrophysical Journal* 338 (1989) 277–280. doi:10.1086/167197.
URL <http://adsabs.harvard.edu/abs/1989ApJ...338..277P>

- [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>