Modeling and Simulating the Vlasov Poisson Problem using Symplectic Integrators

Hannah Haider[1]

[1]Mechanical and Aerospace Engineering Department, University of California, San Diego

Fall 2022/Spring 2023-Summer 2023

Abstract

Symplectic numerical and model reduction techniques can be leveraged to evolve Hamiltonian systems over long-time scales. The following report specifically discusses Hamiltonian systems that model charged particle or plasma dynamics. Stemming from the full order model of the Vlasov Maxwell equations, the Poisson equation further simplifies the flow by assuming a self-consistent electrostatic field resulting in collisionless, self-gravitating plasma flow (applicable to galactic environments). Furthermore, its Hamiltonian system formulation can be used in conjunction with symplectic integrators to advance the charged particles' positions and velocities through long-time simulations. A detailed background is provided to walk through the simulation of the Vlasov Poisson system, and the results of repeated simulations are provided. Additionally, the model's stability and accuracy is evaluated as a function of the Hamiltonian over time, and a general conclusion on the computational burden of the simulation is discussed. Finally, future directions regarding the algorithm's development are suggested, including the incorporation of symplectic model reduction techniques to reduce the computational burden of running the PIC simulation.

Contents

1 Introduction

Plasma physics modeling is an imperative research field for a variety of engineering disciplines. For instance, computational modeling of the fourth state of matter, plasma, can lead to potential breakthroughs in energy systems and space weather. The adoption of plasma physics supports current global initiatives toward a net zero carbon emissions future and the ultimate transition to a fully renewable grid, with energy conversion processes like low temperature plasma-aided combustion showing promise. Additionally, time dependent predictions of plasma flow behavior are critical for an accurate understanding of the effects of space weather; it is particularly useful in predicting the behavior of solar wind flow and, thus, the onset of geomagnetic storms before potential ground and space-based technology blackouts.

The inherent advantages of researching plasma dynamics is readily apparent and growing, however, the focus of this report is on the modeling of collisionless plasma flow for space weather purposes. Research and development of geometric numerical techniques to simulate charged particle kinetics is growing, but the computational complexity and expense of the high dimensional, Particle-In-Cell (PIC) simulations remains a challenge. Collisionless plasma dynamic models are most accurately approximated by the Vlasov Maxwell equations which assumes a self consistent electromagnetic field governs the interaction between particles [8], as in the case for long range plasma flow in space. The Vlasov Maxwell equations are computationally problematic for a number of reasons: the high dimensionality of the model, nonlinearity, and the importance of conserving infinitely many quantities in time [4]. Therefore, it is of interest to firstly adopt a surrogate formulation known as the Vlasov Poisson equation to serve as a test bed for symplectic techniques. Rather than coupling the Maxwell equations describing the self consistent electric and magnetic fields, the Vlasov Poisson equation assumes an electrostatic field, or one which the magnetic field is presumed to be zero. This further justifies our studies in the domain of space weather as the assumption implies the flow behavior is self-gravitating [4].

Despite the adoption of a surrogate model, the computational expense of solving the Vlasov Poisson problem hinders further developments within the field. In order to make reliable predictions of plasma dynamics, repeated simulations of the model are needed to 1) understand the flow behavior, 2) allow for eventual model reduction, and 3) overall, to ensure accuracy for the accessibility of the scientific community. Therefore, this paper focuses on simulating a Vlasov Poisson model by primarily following the work in [7], [12] and begins the process of investigating symplectic model reduction techniques to alleviate the computational burden of each simulation. In particular, an independently made algorithm is proposed to solve the Vlasov Poisson equation by exploiting its Hamiltonian formulation, and repeated simulations are run to understand the flow behavior and test the accuracy of the model.

Further research in the dimensionality reduction of the Vlasov Poisson problem are also proposed. The simulation of the high dimensional model provides empirical data on the Hamiltonian which can be used to drive a reduced order model (ROM). Similarly to the use of symplectic integrators to simulate the full, high dimensional model, symplectic preserving model reduction techniques are discussed as potentially alleviating the computational cost of a full simulation whilst maintaining accuracy and stability of the solution.

2 Theoretical Background

2.1 The Full Order Model: Vlasov-Maxwell Equations

The Vlasov and Maxwell equations can be coupled to approximate nonlinear, charged particle dynamics in time. Specifically, the Vlasov Maxwell equations, or the Full Order Model (FOM), describe the evolution of a distribution of charged particles which generate a self-consistent electromagnetic field due to their collisionless nature [8]. The assumption of collisionless charged particle kinetics means that the FOM is not applicable to collision-based plasma kinetic modeling, such as in combustion; collisionless flow implies that the self-generative electromagnetic field induces a force, thus motion, and is characteristic to plasma flow in celestial mechanics.

In order to approximate the FOM plasma dynamics, a distribution function f is used to represent the particle weights as a function of space $\mathbf{x}$, velocity $\mathbf{v}$, and time t. The Boltzmann function $f(\mathbf{x}, \mathbf{v}, t)$ represents the probability that a typical particle, in a set of computational macroparticles, has a position $\mathbf{x}$ and velocity $\mathbf{v}$ at an instant of time t [4]. In other words, f describes the distribution of particles or the plasma density. Thereafter, the Vlasov equation describes the evolution of f in time, with respect to its phase space $\mathbf{x}$ and $\mathbf{v}$:

$$\frac{d}{dt} f(\mathbf{x}, \mathbf{v}, t) = 0. \tag{1}$$

Expanding Vlasov's equation (1), an ordinary differential equation (ODE), to a partial differential equation (PDE) reveals the complex dynamics which influence the evolution of the distribution function f,

$$\frac{\partial f(\mathbf{x}, \mathbf{v}, t)}{\partial t} + \frac{d\mathbf{x}}{dt} \cdot \frac{\partial f(\mathbf{x}, \mathbf{v}, t)}{\partial \mathbf{x}} + \frac{d\mathbf{v}}{dt} \cdot \frac{\partial f(\mathbf{x}, \mathbf{v}, t)}{\partial \mathbf{v}} = 0. \tag{2}$$

Physicists will immediately recognize the full time derivatives of $\mathbf{x}$ and $\mathbf{v}$ in equation (2) as velocity $\mathbf{v}$ and acceleration $\mathbf{a}$, respectively. Using Newton's second law, $\mathbf{F} = m\mathbf{a}$, the acceleration of a charged particle can be simply described as $\mathbf{a} = \frac{\mathbf{F}}{m}$. To approximate the force on a charged particle, with assumed constant charge q and traveling in an electromagnetic field, the Lorentz force

$$\mathbf{F} = q(\mathbf{E} + \mathbf{v} \times \mathbf{B}), \tag{3}$$

is coupled into the system of equations. The Lorentz force specifically approximates the force imposed on a particle of charge q by combining the force that is magnitudedly proportional to the electric field $\mathbf{E}$ (parallel to flow) and the force proportional to the cross product of the velocity $\mathbf{v}$ and the magnetic field $\mathbf{B}$ (normal to flow). Dividing equation (3) by mass yields the acceleration on a particle of charge q, and can be substituted in equation (2) as follows:

$$\frac{\partial f(\mathbf{x}, \mathbf{v}, t)}{\partial t} + \mathbf{v} \cdot \frac{\partial f(\mathbf{x}, \mathbf{v}, t)}{\partial \mathbf{x}} + \frac{q}{m}(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \frac{\partial f(\mathbf{x}, \mathbf{v}, t)}{\partial \mathbf{v}} = 0. \tag{4}$$

In order to close the system presented in (4), equations which approximate the electromagnetic field are needed. The Maxwell equations are the fundamental set of PDEs which can be used to describe the behavior of the electric and magnetic fields in space and time [4]. Given a 1-dimensional problem, the divergence and curl of the electric and magnetic fields are approximated as

$$\nabla \cdot \mathbf{E}(\mathbf{x}, t) = \frac{\rho(\mathbf{x}, t)}{\epsilon_0}, \tag{5}$$

$$\nabla \cdot \mathbf{B}(\mathbf{x}, t) = 0, \tag{6}$$

$$-\nabla \times \mathbf{E}(\mathbf{x}, t) = \frac{\partial \mathbf{B}(\mathbf{x}, t)}{\partial t}, \tag{7}$$

$$\nabla \times \mathbf{B}(\mathbf{x}, t) = \epsilon_0 \mu_0 \frac{\partial \mathbf{E}(\mathbf{x}, t)}{\partial t} + \mu_0 \mathbf{j}, \tag{8}$$

where ρ, the charge density, and $\mathbf{j}$ are approximated by the following:

$$\rho(\mathbf{x}, t) = q \int_{\mathbf{v}} f(\mathbf{x}, \mathbf{v}, t) d\mathbf{v} \tag{9}$$

$$\mathbf{j} = q \int_{\mathbf{v}} f(\mathbf{x}, \mathbf{v}, t) d\mathbf{v}. \tag{10}$$

2.2 The Reduced-Physics Surrogate Model: Vlasov-Poisson

Geometric integration and model reduction are an ongoing research field, their application to the FOM can lead to improved space weather models in terms of accuracy and computational cost. However, the simulation of the FOM can be both computationally and physically complex, coupling equations (4)-(10), and the need for long-time integration increases simulation run time [7]. In order to provide leeway for an efficient and robust model of the Vlasov Maxwell system, a reduced-physics surrogate model is adopted to relieve some of the computational burden and serve as a test problem for different symplectic model reduction techniques.

The surrogate model incorporates an added assumption of zero magnetic field to derive the Vlasov Poisson system, essentially

$$\mathbf{B}(\mathbf{x}, t) = 0. \tag{11}$$

Substituting equation (11) into equation (7) yields

$$-\nabla \times \mathbf{E}(\mathbf{x}, t) = 0. \tag{12}$$

Recalling that the curl of a gradient of *any* vector is always equal to zero, we can simply represent the electric field $\mathbf{E}$ as the gradient of the electric potential Φ:

$$-\nabla \times (\nabla \cdot \Phi(\mathbf{x}, t)) = 0, \tag{13}$$

$$-\nabla \cdot \Phi = \mathbf{E}(\mathbf{x}, t). \tag{14}$$

By substituting equation (14) into equation (5) of the Maxwell equations, one arrives at the Poisson equation

$$\nabla \cdot (-\nabla \cdot \Phi(\mathbf{x}, t)) = \frac{\rho(\mathbf{x}, t)}{\epsilon_0}, \tag{15}$$

which is customarily written as the PDE

$$\nabla^2\Phi(\mathbf{x},t)+\frac{\rho(\mathbf{x},t)}{\epsilon_0}=0. \tag{16}$$

Coupling equations (11) and (16), the FOM denoted by equations (4)-(10) is simplified to form the surrogate, Vlasov Poisson model:

$$\frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial t}+\mathbf{v}\frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial \mathbf{x}}-\frac{q}{m}\frac{\partial\Phi(\mathbf{x},t)}{\partial\mathbf{x}}\frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial\mathbf{v}}=0, \tag{17}$$

$$-\frac{\partial^2\mathbf{\Phi}(\mathbf{x},t)}{\partial x^2}=\rho(\mathbf{x},t), \tag{18}$$

$$:=q\int_{\mathbf{v}}f(\mathbf{x},\mathbf{v},t)d\mathbf{v}. \tag{19}$$

2.3 The Vlasov-Poisson problem as a Hamiltonian system

The goal of this paper is to demonstrate the viability of symplectic numerical techniques in long-time scale integration of particle dynamics, in terms of the simulation's accuracy, stability, and cost. Symplecticity is a property that is conserved for Hamiltonian systems, a class of ODEs which can also be represented as Hamiltonian PDEs [10]. The reader is directed to [1],[5] for a better understanding of the underlying structures of Hamiltonian systems and how symplectic integrators can be used to preserve these structures. Nevertheless, it is sufficient to begin describing the Hamiltonian system formulation of the Vlasov Poisson problem, then discuss the symplectic numerical technique used to simulate the evolution of the system states, position and velocity.

The Hamiltonian is, in simple terms, a way to describe the total energy of a set of particles (a dynamical system) given their respective positions (X) and velocities (V) [2]. Essentially, the Hamiltonian is found by

$$H(X,V)=\sum_{i=1}^{N_p}\left[\frac{1}{2}V_i^2+\frac{q}{m}\Phi(X_i)\right], \tag{20}$$

where N_p is the number of particles, and $\mathbf{u}=(X,V)$ is the system state [12]. A curious reader would recognize the two terms summed in the Hamiltonian as the kinetic and potential energy of the system. The Hamiltonian is then used to describe the time evolution of the position and velocity through Hamilton's equations:

$$\dot{X}_i=\frac{\partial H(X,V)}{\partial V_i}, \tag{21}$$

$$\dot{V}_i=-\frac{\partial H(X,V)}{\partial X_i}. \tag{22}$$

The dot above the state vectors, X and V, represents the time derivative. In taking the partial derivative of the Hamiltonian as indicated in Hamilton's equations (21) and (22), the time evolution of X and V along the characteristics of the Vlasov Poisson equation is given by:

$$\dot{X}_i=V_i, \tag{23}$$

$$\dot{V}_i=-\frac{q}{m}\frac{\partial\Phi(X_i)}{\partial X_i}. \tag{24}$$

Thereafter, the canonical Hamiltonian system formulation for the state $\mathbf{u}$ is

$$\dot{\mathbf{u}}=\begin{bmatrix}\dot{X}\\ \dot{V}\end{bmatrix}=\mathbb{J}_{2N}\nabla_{\mathbf{u}}H(\mathbf{u})=\begin{bmatrix}\nabla_V H(X,V)\\ -\nabla_X H(X,V)\end{bmatrix}, \tag{25}$$

where $N=N_pd$, $d=1,2,3$ dimensions, and $\mathbb{J}_{2N}$ is known as the canonical symplectic matrix. The canonical symplectic matrix is defined in [12] as the following:

$$(DF_t)\top\mathbb{J}_{2N}DF_t=\mathbb{J}_{2N}, \tag{26}$$

$$\mathbb{J}_{2N}=\begin{pmatrix}0 & I_N\\ -I_N & 0\end{pmatrix}, \tag{27}$$

where F_t represents the flow map of the Hamiltonian system and DF_t is the Jacobi matrix of the flow map, thus the canonical symplectic form of the system is preserved.

2.4 Geometric Integration

Geometric integrators are a popular approach to solving Hamiltonian systems as they can ensure geometric flow properties are conserved over long-time simulations. As depicted in equations (26) and (27), flow described by Hamiltonian systems are *symplectic* or conserve flow areas in phase space [6]. In the context of space weather modeling, long-range simulations are needed to predict the interaction of plasma flow and Earth's atmosphere. Thereafter, the long-time integration of the Vlasov Poisson Hamiltonian system using traditional numerical solvers, e.g. Euler, Maccormack, and Runge Kutta methods, could lead to instabilities and inaccuracies in comparison to symplectic integrators.

2.4.1 Störmer-Verlet Method

The Störmer-Verlet method is chosen primarily due to its popularity as a structure preserving time stepping algorithm. The Störmer-Verlet technique applied to Hamiltonian systems maintains reversibility of the numerical map in phase space; essentially, we may invert the direction of the initial velocity and not change the solution trajectory [6]. The reader is directed to [5] for a brief overview on the method and how it can be applied to Hamiltonian systems to maintain energy conservation in time.

For plasma fluid flow modeled by the Vlasov-Poisson Hamiltonian system, we hypothesize that the symplectic property of the flow can be conserved using the Störmer-Verlet method, sometimes called the "leapfrog" method due to the jumping between states X and V. Adapting the Störmer-Verlet method as proposed in [6] to the Vlasov-Poisson Hamiltonian system in equation (25) results in the following implicit scheme:

$$X_{n+\frac{1}{2}} = X_n - \frac{n}{2}\left[\nabla_V H(X_{n+\frac{1}{2}}, V_n)\right], \tag{28}$$

$$V_{n+1} = V_n + \frac{n}{2}\left[\nabla_X H(X_{n+\frac{1}{2}}, V_n) + \nabla_X H(X_{n+\frac{1}{2}}, V_{n+1})\right], \tag{29}$$

$$X_{n+1} = X_{n+\frac{1}{2}} - \frac{n}{2}\left[\nabla_V H(X_{n+\frac{1}{2}}, V_{n+1})\right]. \tag{30}$$

For high-order models, Hamiltonian system formulations can be very useful as it implies that the Hamiltonian can be separated into its respective kinetic and potential energy parts [3],

$$H(\mathbf{u}) = H(X) + H(V), \tag{31}$$

$$H(\mathbf{u}) = \sum_{i=1}^{N_p}\left[\frac{q}{m}\frac{\partial\Phi(X_i)}{\partial X}\right] + \sum_{i=1}^{N_p}\left[\frac{1}{2}V_i^2\right], \tag{32}$$

decreasing the computational complexity of an implicit scheme. This notion is further supported in [5], which provides an explicit *momentum* formulation of the Störmer-Verlet method. This momentum formulation is adapted to the Vlasov Poisson system as:

$$V^{n+\frac{1}{2}} = V^n - \frac{1}{2}\Delta t\nabla_X\mathcal{H}^X(X^n), \tag{33}$$

$$X^{n+1} = X^n + \Delta t\nabla_V\mathcal{H}^V(V^{n+\frac{1}{2}}), \tag{34}$$

$$V^{n+1} = V^{n+\frac{1}{2}} - \frac{1}{2}\Delta t\nabla_X\mathcal{H}^X(X^{n+1}). \tag{35}$$

3 Simulation of the Vlasov Poisson system

The bulk of this project was spent independently developing the algorithm needed to approximate the solution to the Vlasov Poisson Hamiltonian system using symplectic integrators and the Python programming language. By primarily following the numerical procedures outlined in [12], [7], repeated PIC simulations of the Vlasov Poisson system with different particle amounts, meshgrid sizing, simulation time, and initial conditions are conducted to evaluate the algorithm's robustness. Most research regarding the simulation of the Vlasov Poisson system lacks in explaining fundamental components of their algorithm, e.g. the exact method for solving Poisson's equation, a discussion of the evolution of the Hamiltonian in time, etc. For instance, [12] uses an empirical formulation for the electric field rather than directly solving for the electric potential from the charge density. Additionally, [7] focuses mostly on addressing the computational burden of the PIC simulation using model reduction techniques; being that the Vlasov Poisson problem is a precursor to the Vlasov Maxwell problem, it is understandable to not waste time ensuring the physics of the Vlasov Poisson problem are accurately modeled

prior to model reduction. However, the model developed in this paper can be used to further understand the drawbacks to the numerical procedure used to simulate a Hamiltonian system; a discussion of the findings can be used to simultaneously improve the full order model, then the optimal model reduction technique can be more accurately determined.

Mesh grid spacing The Vlasov Poisson system of equations is considered to be one-dimensional and one-velocity. Essentially the meshgrid can be considered two-dimensional, but describes the phase space (position and velocity) of the solution. For the Vlasov Poisson Hamiltonian system, consider a spatially discretized grid with the domain $0 \leq \mathbf{x} \leq L$, velocity domain $v_0 \leq \mathbf{v} \leq v_f$, time domain $0 \leq t \leq t_f$ and N_p particles. The corresponding spatial, velocity, and time step sizes are then calculated as follows:

- spatial step size $\Delta \mathbf{x} = \frac{L}{N_x}$
- velocity step size $\Delta \mathbf{v} = \frac{v_f - v_0}{N_v}$
- time step size $\Delta t = \frac{t_f}{N_t + 1}$

where N_x, N_v, and N_t are the number of spatial grid points, velocity grid points, and time steps, respectively. All simulations included in this report are simulated for 40 seconds with time steps $N_t = 2000$. The reader should take careful note to distinguish $(\mathbf{x}, \mathbf{v})$ from $[\mathrm{X},\ \mathrm{V}]\top$, as the latter is the vector containing all particle positions and velocities (of dimension $2N_p \times 1$), which changes with time t. Additionally, it is worth mentioning that, by nature of the symplectic scheme found listed in equations (31) - (33), N_x must be equal to N_p.

Initial and Boundary Conditions Multiple initial conditions can and should be simulated to ensure the algorithm is still able to produce a sufficient and stable solution. In other studies, such as in [11], the model's accuracy varies with different initial conditions. It is also of interest to vary the mesh parameters and explore the sensitivity of the model, as [7] suggests, PIC model stability is heavily dependent on the number of particles and the simulation time. Two different initial conditions for the distribution function, $f(\mathbf{x}, \mathbf{v}, t)$ are considered to test the algorithm's ability to approximate the solution. The boundary conditions are taken to be spatially periodic, or

$$X_0 = X_{i+1}, \tag{36}$$

wbere i denotes the particle index.

The first initial condition considered is a periodic spatial perturbation with a bump-on-tail distribution in velocity, as detailed in [12]:

$$f(\mathbf{x}, \mathbf{v}, t = 0) = (1 + \epsilon \cos(2\pi \mathbf{x}))\left(\frac{1}{1+a} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\mathbf{v}^2} + \frac{a}{1+a} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(\mathbf{v} - v_0)^2} \right), \tag{37}$$

where the parameters $\epsilon = 0.3$, $\alpha = 0.3$, and $v_0 = 4$ are arbitrarily assigned according to [12] and σ is the standard deviation of the function found by python package numpy.variance. The second initial condition considered is, similarly, a periodic spatial perturbation with a Maxwellian distribution in velocity [7]:

$$f(\mathbf{x}, \mathbf{v}, t = 0) = (1 + \alpha \cos(k\mathbf{x}))\left(\frac{1}{\sqrt{2\pi}} e^{(-\frac{\mathbf{v}^2}{2\sigma^2})} \right), \tag{38}$$

where parameter $\alpha = 0.4912$ is the amplitude of the spatial perturbation, $k = 0.5$ is the wave number, and $\sigma = 0.9889$ (standard deviation) are assigned according to [7] to demonstrate one-dimensional landau damping.

The initial distribution functions, equations (37) and (38), are obviously continuous. Due to the continuous nature of the distribution function, it can be difficult to see the correlation between a PIC simulation of the Hamiltonian states $\mathbf{u}(X, V)$ and evolving the distribution function. As seen in equation (17), the goal of simulating the Vlasov Poisson system is to describe the evolution of the distribution function in time. To do so, an ansatz function

$$f(\mathbf{x}, \mathbf{v}, t) = \sum_{i=1}^{N_p} w_i \delta(\mathbf{x} - X_i(t)) \delta(\mathbf{v} - V_i(t)) \tag{39}$$

is used to populate the probability distribution matrix f at every time t, given the evolved particle positions X and velocities V. In the same sense, the initial conditions of the particle positions $X_i(t = 0)$ and $V_i(t = 0)$ are found by rejection sampling of the initial condition. Popular sampling techniques stem from the Markov Chain Monte Carlo (MCMC) methods like Metropolis Hastings as used in [7] and rejection sampling as used in [12]. In order to accurately and sufficiently produce the aggregated distribution function at any time t, the

discrete, initial particle positions X and velocities V must be randomly sub-sampled from the continuous, initial distribution.

A general rejection sampling algorithm is developed by following the guidelines provided in [9]. Firstly, a random, uniform distribution of particles with the same size, minimum, and maximum value of the initial distribution function $f(\mathbf{x}, \mathbf{v}, t = 0)$ is generated and labeled as the '*envelope*'. Then, a while loop is used to generate a list of sampled grid points $(\mathbf{x}, \mathbf{v})$ if the *envelope* $< f(\mathbf{x}, \mathbf{v}, t = 0)$. The grid points are then sub-sampled to be the size of N_p, and preallocated as the initial X and V vectors.

3.1 Implementation

A total of 10 functions are created in attempt to alleviate the use of nested for loops (addressing computational efficiency) and to provide for accessibility throughout the code. Each function is well documented and includes a list of parameters (inputs and outputs), their respective type and dimension.

- **Initial Distribution Functions**: Two functions are provided to approximate the initial, continuous distribution functions given $\mathbf{x}, \mathbf{v}, N_x, N_v$ and N_t. Specifically, the functions implement two for loops to evaluate equations (36) and (37) at every grid point. The output is stored as the initial particle distribution $f(\mathbf{x}, \mathbf{v}, t = 0)$.

- **Update Distribution Function**: Similarly to the initial distribution function, the output of the 'update distribution' function is stored at each time step as $f(\mathbf{x}, \mathbf{v}, t)$. Given the states position X and velocity V at time t, equation (39) is evaluated for each particle. Essentially, if the difference between the gridpoint $(\mathbf{x}, \mathbf{v})$ and the particle state $[X, V]\top$ is less than the spatial step $\Delta\mathbf{x}$, then the particle weight is superimposed to the grid. The weight is taken to be $= \frac{1}{N_p}$ in accordance with [7]

- **Charge Density** The distribution function is integrated over the velocity phase space as in equation (9) to return a spatially dependent charge density, $\rho(\mathbf{x}, t)$ at each time step. Here, q is taken to be -1 as in [7], [12].

- **Electric Potential - Finite Difference** A second order, central finite difference matrix representing the $-\frac{\partial^2}{\partial x^2}$ operator with periodic boundary conditions is used in conjunction with the python package numpy linalg.solve to solve the Poisson equation in (16) for $\mathbf{\Phi}$, given $\rho(\mathbf{x}, t)$.

- **Electric Potential - Generalized Minimal Residual** The same second, order central finite difference matrix is used in conjunction with the python package scipy linalg.gmres to iteratively solve for the electric potential Φ, given $\rho(\mathbf{x}, t)$.

- **Electric Field** Similarly to the electric potential, a first order derivative finite difference matrix representing the $-\frac{\partial}{\partial x}$ operator with periodic boundary conditions is applied to the electric potential $\mathbf{\Phi}$ to obtain the electric field $E(x)$ at any time step.

- **Hamiltonian** This function evaluates the Hamiltonian according to equation (20) given all particles' positions X and velocities V.

- **Hamiltonian in x** The separable Hamiltonian with respect to X is evaluated in accordance with the equation (32). This function is simply evaluating $\sum_{i=1}^{n} \frac{q}{m} \mathbf{\Phi}(X_i)$.

- **Hamiltonian in v** The separable Hamiltonian with respect to V is evaluated in accordance with the (32). This function is simply evaluating $\sum_{i=1}^{n} \frac{1}{2} V_i^2$.

Time-Stepping Loop As detailed in the theoretical background section, the evolution of the Hamiltonian state $\mathbf{u}(X, V)$ is simulated using the symplectic Störmer-Verlet technique outlined in equations (33) - (35). However, the time-stepping for loop involves more than just the evolution of the Hamiltonian states, it requires that the charge density ρ, electric potential Φ, the Hamiltonian H, and the distribution function f also be updated. Pseudocode for the for loop is provided for clarification purposes in **Algorithm 1**.

Repeated PIC simulations of the Vlasov Poisson system with varying initial conditions, amount of particles, simulation times, and mesh sizing are encouraged to determine the robustness of the model. Additionally, it is of interest to document the simulation runtimes to motivate research in symplectic model reduction techniques.

Algorithm 1 Evolving the Distribution Function

Initial distribution function: $f[\mathbf{x}, \mathbf{v}, t = 0]$
Initial charge density: $\rho[\mathbf{x}, t = 0]$
Initial electric potential: $\Phi[\mathbf{x}, t = 0]$
Initial electric field: $\mathbf{E}[\mathbf{x}, t = 0]$
Initial Hamiltonian: $H(t = 0)$
for ii in range(N_t **do**)
$V^{n+\frac{1}{2}} = V - \frac{1}{2} \cdot \Delta t \frac{q}{m} \cdot E[:, ii]$
$X^{n+1} = X + \Delta t \cdot V^{n+\frac{1}{2}}$
- Enforce boundary conditions in X
- Update distribution function $f[:, :]$
- Update charge density $\rho[:]$
- Update electric potential $\Phi[:]$
- Update electric field $\mathbf{E}[:]$
$V = V - \frac{1}{2} \cdot \frac{q}{m} \cdot E[:]$
- Final update to distribution function $f[\mathbf{x}, \mathbf{v}, ii + 1]$
- Final update to charge density $\rho[:, ii + 1]$
- Final update to electric potential $\Phi[:, ii + 1]$
- Final update to electric field $\mathbf{E}[:, ii + 1]$
- Update Hamiltonian $H[ii + 1]$
end for
Return $f[\mathbf{x}, \mathbf{v}, t]$, $\rho[\mathbf{x}, t]$, $\Phi[\mathbf{x}, t]$, $\mathbf{E}[:, t]$, and $H[t]$

4 Numerical Experiments and Results

4.1 Implementation Details and Error Computation

To begin the PIC simulation, the mesh grid sizing and number of particles N_p are required to construct the initial distribution function. The first initial condition in equation (36) is pictured in Figure 1, it is suggested that the default spatial domain be $0 \leq \mathbf{x} \leq 1$ and the velocity domain be $-10 \leq \mathbf{v} \leq 10$. Figure 2 plots the second initial condition (equation (37)), [7] suggests a spatial domain of $0 \leq \mathbf{x} \leq \frac{2\pi}{k}$ and a velocity domain of $-4 \leq \mathbf{v} \leq 4$.

Additionally, the rejection sampling technique discussed in the implementation subsection suggests that a random, uniform distribution of particles occupies the grid. The python package 'numpy' contains a function 'random.uniform' in which the maximum, minimum, and size of the uniform distribution can be set. Figure 3 illustrates the typical random, uniform distribution of particles outputted by the Python function and used as an envelope for rejection sampling. Thereafter, rejection sampling of the initial condition can be conducted and stored as the initial particle positions and velocities.

The initial particle positions and velocities are used to reconstruct the initial distribution function, according to equation (39) and as depicted in Figure 4. The now discrete, initial distribution function is used to begin the time-stepping algorithm described in **Algorithm 1**. The initial conditions for the charge density, electric potential, and electric field are then calculated using their respective functions, as outlined in the **Implementation** subsection. These initial conditions are pictured in Figure 5 for $N_p = 200$ and 1000 particles. It is important to note that the figure depicts the electric potential as calculated using the general minimal residual method (GMRES function in Python), rather than directly solving the linear system. However, the electric potential found by the latter method is almost identical. As an increased number of particles (thus spatial grid points) are included in the initial distribution function, a more smoothed approximation of the electric potential and electric field can be obtained. Additionally, the electric field behavior is consistent with taking the partial derivative of the electric potential with respect to $\mathbf{x}$ and multiplying by a negative, as in equation (14).

In Figure 6, we plot the distribution function (for both initial conditions and particle amounts $N_p = 200, 1000$) at different times in the 40 second simulation. It can be deduced that increasing the number of particles does not necessarily change the flow behavior. Rather, the formation of vortices are apparent in both cases $N_p = 200, 1000$ and are consistent with the physical flow descriptions provided in [7] for the second initial condition of weak landau damping. Revisiting the **Theoretical Background** section, or specifically equation (20), a deeper analysis on the simulation's accuracy is conducted by examining the evolution of the Hamiltonian, or the total energy over time. Specifically, we determine the difference in the relative errors for different particle amounts and initial conditions, where the relative error is computed by:

$$\%rel.error = \frac{|H(t = 0) - H(t = t_f)|}{H(t = 0)} \times 100\%. \tag{40}$$

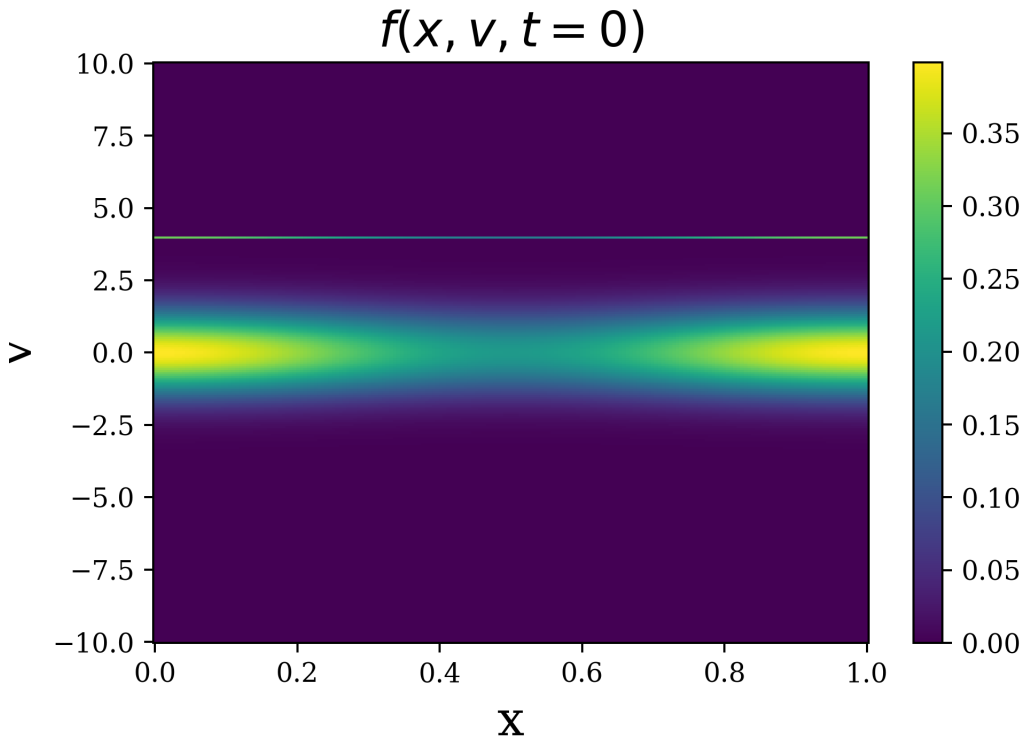


Figure 1: The first initial particle distribution function $f(\mathbf{x}, \mathbf{v})$ at time $t = 0$, as approximated by equation (36).

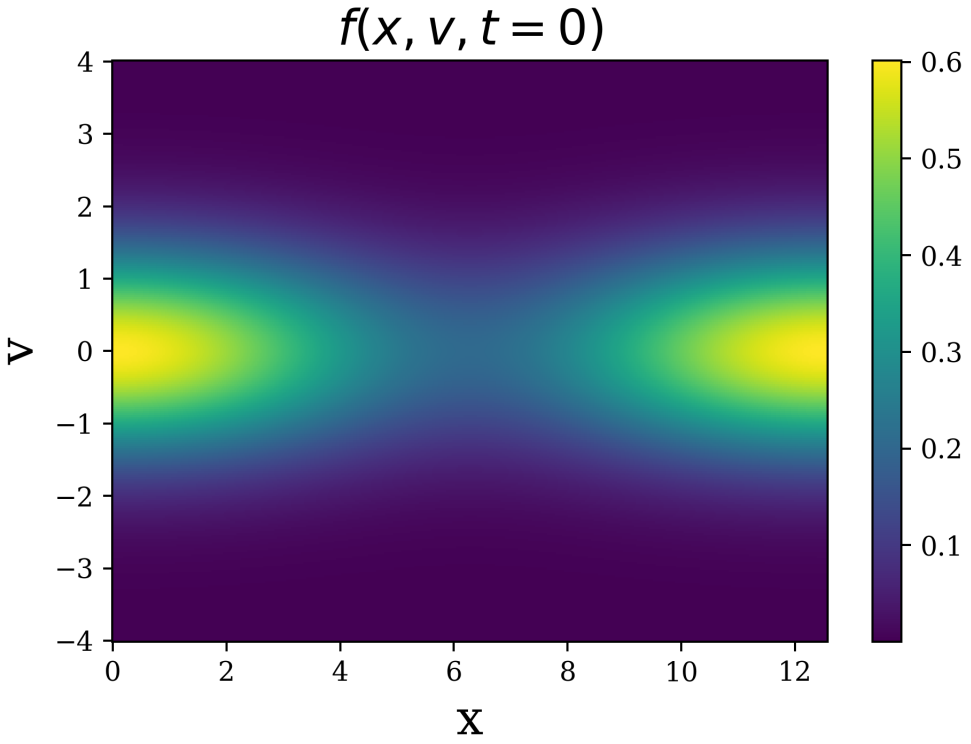


Figure 2: The second initial particle distribution function $f(\mathbf{x}, \mathbf{v})$ at time $t = 0$, as approximated by equation (37).

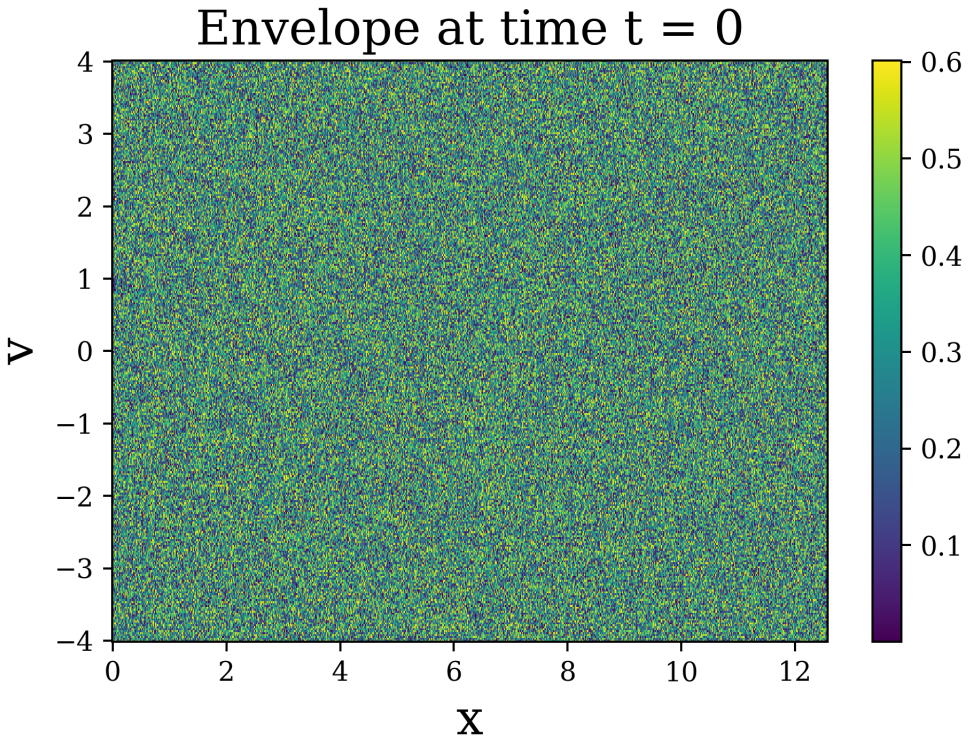
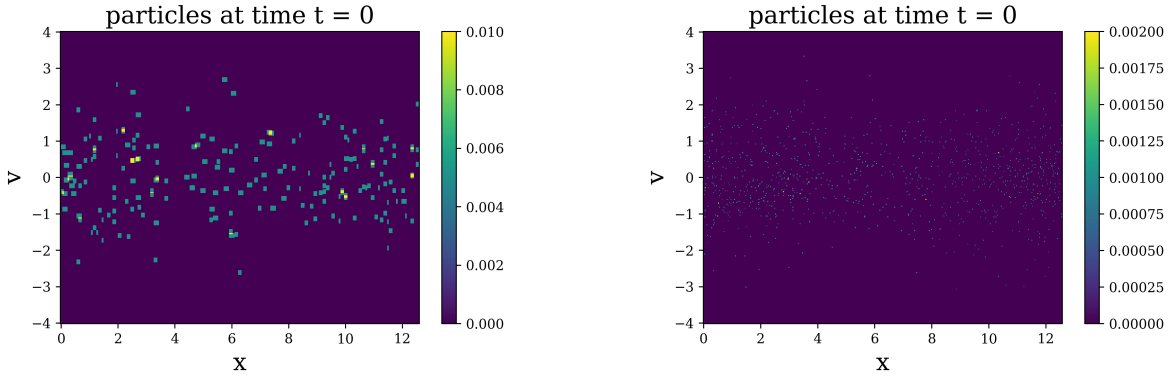


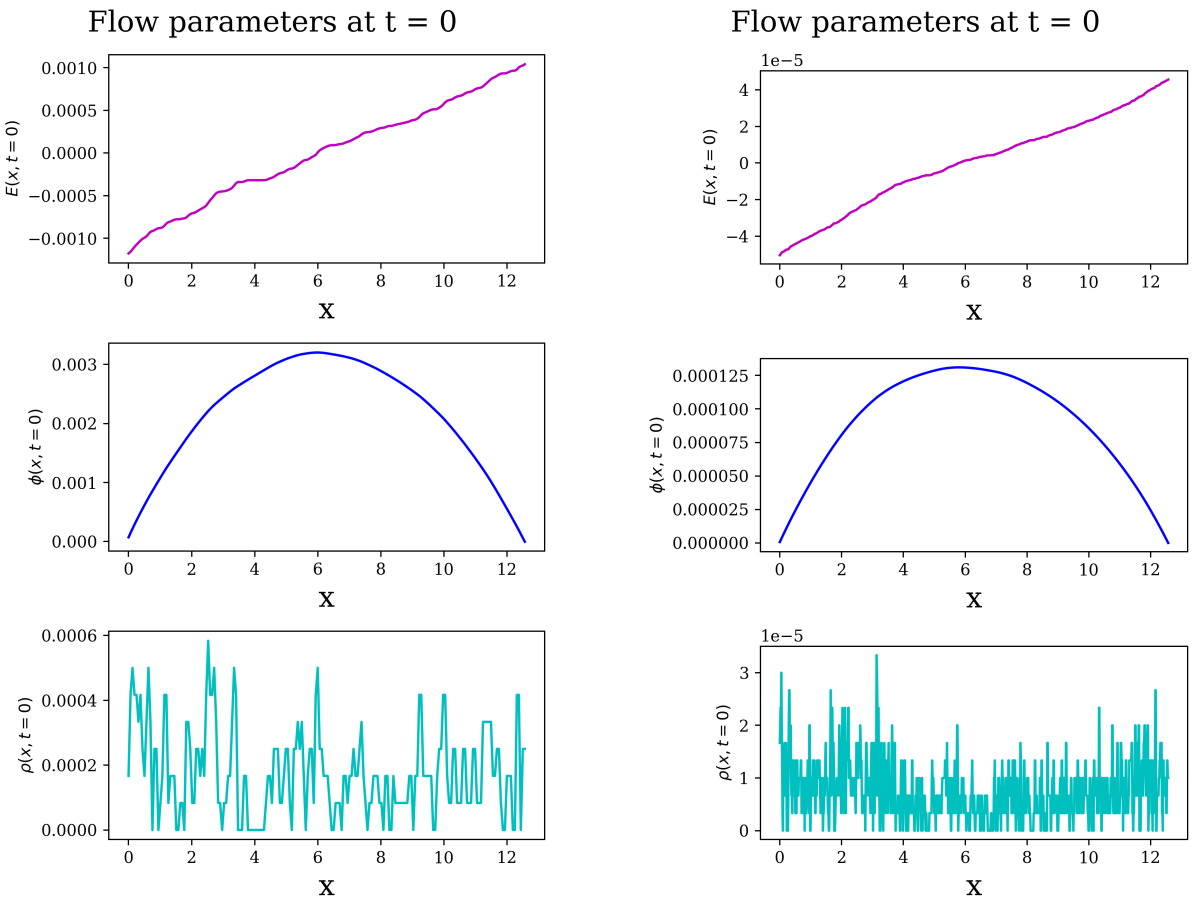
Figure 3: The *envelope* that is used to implement rejection sampling of the initial distribution function.



(a) $N_p = 200$ particles.

(b) $N_p = 1000$ particles.

Figure 4: Rejection sampling of the initial condition in equation (35) for two different sets of particles.



(a) $N_p = 200$ particles.

(b) $N_p = 1000$ particles.

Figure 5: Initial conditions for the charge density $\rho(\mathbf{x}, t = 0)$, electric potential $\Phi(\mathbf{x}, t = 0)$, and the electric field $\mathbf{E}(\mathbf{x}, t = 0)$ as approximated by the initial distribution function in Figure 4.

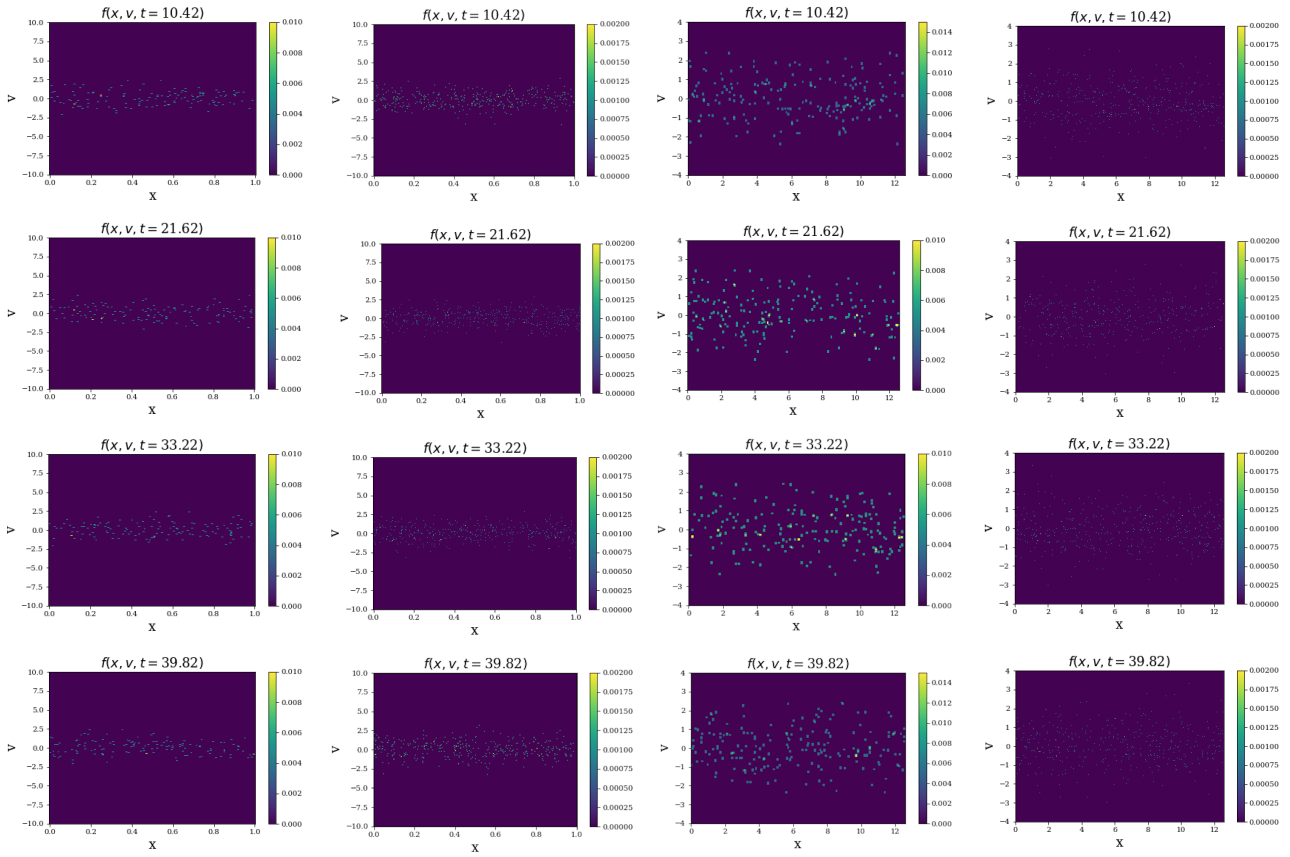
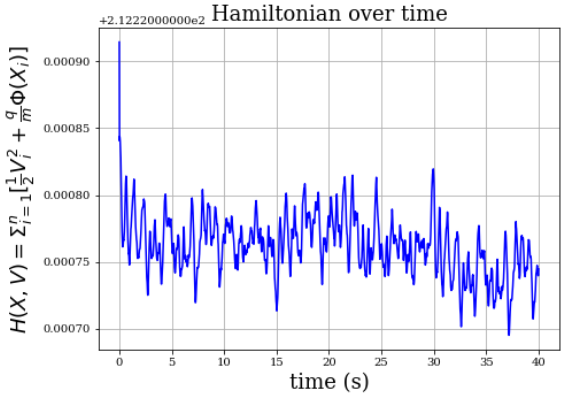
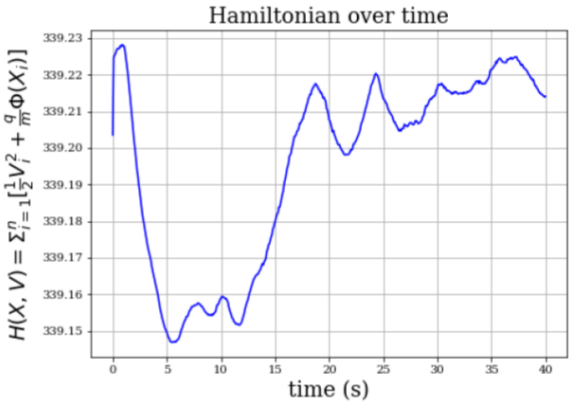


Figure 6: The distribution function $f(\mathbf{x}, \mathbf{v}, t)$ as simulated with two different initial conditions. The first two columns are PIC simulations with the initial condition approximated by (36), and the second two columns depict the simulations with initial condition approximated by (37). Additionally, the first and third column are simulations ran with 200 particles and the second/fourth columns are those ran with 1000 particles.

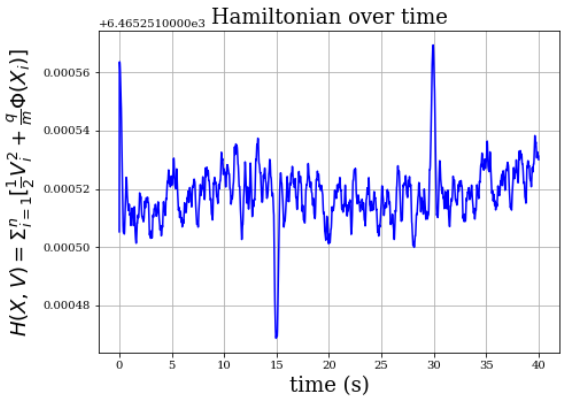


(a) Relative error = 3.99×10^{-03}%.

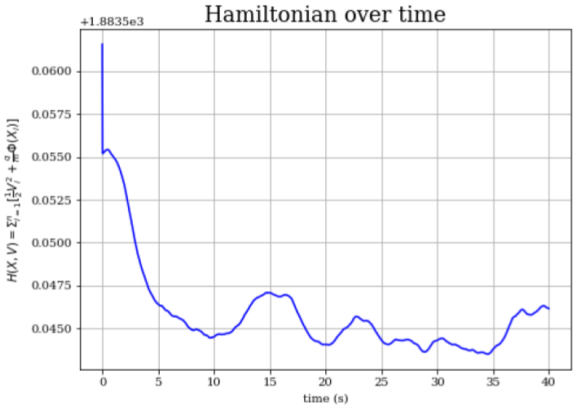


(b) Relative error = 3.61×10^{-03}%.

Figure 7: The Hamiltonian is plotted as a function of time over the 40 second simulation for both initial conditions and number of particles $N_p = 200$. The plot on the left corresponds to the simulation of the first initial condition (equation (36)) and the right plot corresponds to the second initial condition (equation (37)).



(a) Relative error = 3.83×10^{-07}%.



(b) Relative error = 1.45×10^{-04}%

Figure 8: The Hamiltonian is plotted as a function of time over the 40 second simulation for both initial conditions and particle amount $N_p = 1000$. The plot on the left corresponds to the simulation of the first initial condition (equation (34)) and the right plot corresponds to the second initial condition (equation (35)).

Figures 7 and 8 plot the total energy (the Hamiltonian) of the system as a function of time, initial condition, and number of particles in the PIC simulation.

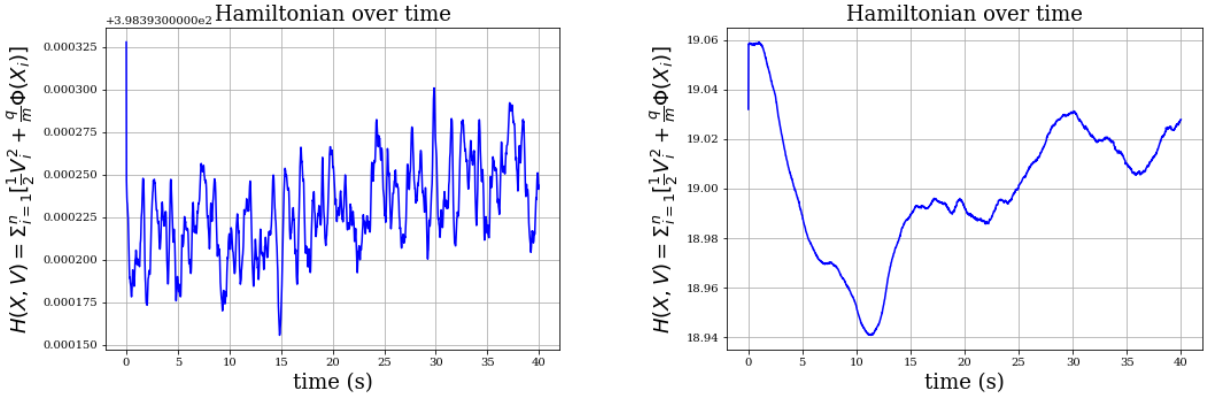
Additionally, the method chosen to solve the Poisson equation given in equation (18) is either by generalized minimal residual method (which iteratively solves a linear system of equations) or by directly solving the linear system. Thereafter, it is important to discuss the potential loss of accuracy using one method over the other. Figures 7 and 8 demonstrate the Hamiltonian's evolution for the direct Poisson solver, whereas Figure 9 demonstrates the Hamiltonian's evolution when using the iterative solver (for only the case of $N_p = 200$ particles due to time restrictions).

Lastly, the run times for each simulation are provided to demonstrate the computational inefficiency of PIC simulations and motivate the eventual reduced order modeling of the Vlasov Poisson system.

Computer Processing Unit (CPU)	$N_p = 200$ runtime	$N_p = 10^3$ runtime
i5 quadcore	3m30s	1h25m21s
i5 dualcore	6m6s	2h5m30s

5 Discussion & Conclusions

Before delving into the numerical results, let us reiterate the objective of modeling the Vlasov Poisson system. Essentially, the goal is to demonstrate the accuracy of symplectic numerical techniques in modeling Hamiltonian systems and, in specific, Hamiltonian systems which approximate collisionless, charged particle (plasma) flow, i.e. the Vlasov Maxwell system of equations (the FOM). Prior to developing the FOM, we take advantage of



(a) Relative error $= 2.12 \times 10^{-05}\%$.

(b) Relative error $= 2.17 \times 10^{-02}\%$

Figure 9: The Hamiltonian is plotted as a function of time over the 40 second simulation for both initial conditions and particle amount $N_p = 200$. The plot on the left corresponds to the simulation of the first initial condition (equation (36)) and the right plot corresponds to the second initial condition (equation (37)). These simulations are conducted by using the GMRES function in Python to iteratively solve for the electric potential Φ rather than directly solving the linear system of equations, as in Figures 7 and 8.

a simpler formulation known as the Vlasov Poisson problem and its Hamiltonian system formulation is used to evolve the particle positions and velocities in time. As a result, an algorithm which approximates the time-evolved particle distribution function is independently developed from start to finish. The algorithm can employ two different initial conditions for any mesh grid sizing, number of particles, and simulation time. The proposed algorithm also outputs multiple animations, figures, and values which are used to discuss the model's robustness and the future improvements that can be made. The proposed model is the first part of a series of projects that investigate geometric model reduction techniques to alleviate the computational burden of long-time PIC simulations.

The particle distribution function is repeatedly simulated for 40 seconds and depicted in Figure 6; despite increasing the number of particles, the particle behavior is similar between initial conditions and at each time step. No formal verification and validation procedure was conducted to ensure the physics of the model are accurate, rather we rely on simply comparing our results to those depicted in [7]. Our model's validity is limited as it has not been simulated with > 5000 particles, and most literature ([7], [12]) suggest simulating with over 10^4 or even 10^5 particles. However, there is a clear indication that the relative error in the Hamiltonian decreases significantly as the number of particles simulated increases. It should also be noted that simulation with the least relative error corresponds to the first initial condition (equation (36)) and $N_p = 1000$ particles, depicted in Figure 8(a). This is most likely due to the fact that the algorithm was originally developed following the work in [12], and the second initial condition was just recently simulated. Still, it is important that more simulations are ran with the second initial condition to test the convergence of the solution and compare with the work in [7]. Additionally, there is generally more of an obvious, bounded energy error for the first initial condition. The fluctuations seen in Figures 7 and 8 (a) are typical for time-stepping schemes, however, the large drop in energy after the first time step is still problematic. In an attempt to understand this discrepancy, the Poisson equation is solved using the generalized minimal residual method (GMRES) instead, as seen in Figure 9.

The tolerance of the GMRES solver is set to the default $1e-05$ and converges at each time step. Despite the incorporation of an iterative solver, the immediate drop in energy after the first time step is still apparent and shown in Figure 9(a). However, there is a large decrease in the relative error in comparison to the simulations ran with the direct Poisson solver. Therefore, more simulations should be ran with the GMRES solver and an increased number of particles for the first initial condition. As for the second initial condition, the GMRES solver seemingly increases the relative error and the overall behavior of the Hamiltonian is less oscillatory than the first condition. Unfortunately [7] does not explicitly mention the way they've solved Poisson's equation, but it could be of interest to rather follow their numerical procedure more closely to potentially understand the discrepancy. Overall, we've presented an algorithm which can solve the Vlasov Poisson system of equations by advancing the Hamiltonian states X and V in time. Potentially the biggest drawback to the algorithm is the computational burden of each PIC simulation. However, this is expected as PIC methods are notoriously expensive, and increasing the number of particles directly increases the accuracy and expense of each simulation.

For the reader's curiosity, an in depth introduction on symplectic model reduction techniques is included in the appendix. These techniques will serve as a jumping point in addressing the computational cost of the full order Vlasov Poisson model, allowing for further symplectic preserving or machine learning techniques to be researched.

References

[1] Babak Maboudi Afkham and Jan S Hesthaven. Structure preserving model reduction of parametric hamiltonian systems. *SIAM Journal on Scientific Computing*, 39(6):A2616–A2644, 2017.

[2] Thomas J Bridges and Sebastian Reich. Numerical methods for hamiltonian pdes. *Journal of Physics A: Mathematical and General*, 39(19):5287, apr 2006.

[3] Fernando Casas, Nicolas Crouseilles, Erwan Faou, and Michel Mehrenberger. High-order hamiltonian splitting for the vlasov–poisson equations. *Numerische Mathematik*, 135:769–801, 2017.

[4] Hernán Cendra, Darryl D. Holm, Mark J. W. Hoyle, J. Sur, Ar, Los Alamos National Laboratory, UC Santa Cruz, and Caltech. The maxwell–vlasov equations in euler–poincaré form. *Journal of Mathematical Physics*, 39:3138–3157, 1998.

[5] Matthew Dixon and Sebastian Reich. Symplectic time-stepping for particle methods. *GAMM-Mitteilungen*, 27(1):9–24, 2004.

[6] Ernst Hairer, Christian Lubich, and Gerhard Wanner. Geometric numerical integration illustrated by the stormer verlet method. *Acta Numerica*, 12:399–450, 2003.

[7] Jan S. Hesthaven, Cecilia Pagliantini, and Nicolo Ripamonti. Adaptive symplectic model order reduction of parametric particle-based vlasov-poisson equation, 2022.

[8] Yingzhe Li, Yang He, Yajuan Sun, Jitse Niesen, Hong Qin, and Jian Liu. Solving the vlasov–maxwell equations using hamiltonian splitting. *Journal of Computational Physics*, 396:381–399, 2019.

[9] Jun Liu. Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, 6, 08 2000.

[10] Harsh Sharma, Zhu Wang, and Boris Kramer. Hamiltonian operator inference: Physics-preserving learning of reduced-order models for hamiltonian systems. *Physica D: Nonlinear Phenomena*, 431:133122, 2022.

[11] Guillaume Steimer, Emmanuel Franck, Vincent Vigon, Laurent Navoret, and Nicolas Crouseilles. Internship report data driven reduced modelling of the vlasov-poisson equation, 2021.

[12] Tomasz M Tyranowski and Michael Kraus. Symplectic model reduction methods for the vlasov equation. *arXiv preprint arXiv:1910.06026*, 2019.

6 Appendix

6.1 Introduction to Model Reduction

After the full order model of the Vlasov Poisson problem is simulated using symplectic, numerical integration (Störmer-Verlet method), it is deduced that the large number of particles needed for a sufficient simulation renders the model inefficient and inaccessible. Determining the optimal numerical techniques for plasma particles modeled as a Hamiltonian system ultimately translates to investigating the techniques which preserve their symplectic structure and addresses the computational burden of high order PIC simulations. Therefore, in approaching the problem of reducing the high order model to a reduced order model (ROM), still capable of replicating Hamiltonian system flow dynamics within a specified accuracy and uncertainty, symplectic model reduction techniques are investigated and proposed as further research.

In surveying model reduction techniques, we must take into account the nature of a Hamiltonian system. As briefly discussed, Hamiltonian systems rely on a principle of least energy and are formed by the position and velocity of particles in the plasma fluid.

Classical reduction techniques like the proper orthogonal decomposition (POD) are troublesome for Hamiltonian systems as they do not imply structure preservation or energy conservation and can lead to an unstable solution. Additionally, the classical reduction techniques would require a very expensive offline stage to run the full order model and drive the POD reduced basis [1]. Thereafter, symplectic reduction techniques which adapt the classical reduced basis methods to preserve symplectic geometry are needed to improve the computational costs of simulating the Vlasov Poisson problem.

Eventually, uncertainty quantification of the ROM compared to the FOM can be made, and the ROM will provide leeway for efficient time-dependent predictions of plasma flow behavior. Additionally, the symplectic nature of the model reduction techniques develops our understanding of preserving a reduced Hamiltonian structure and maintaining the stability of the reduced solution over long time integration [1]. Therefore, it goes a step further than using Hamiltonian mechanics and symplectic methods to simulate a problem, symplectic preserving ROMs offer accessibility and efficiency by reducing the computational cost to run a simulation.

6.1.1 Proper Symplectic Decomposition

Highly analogous to proper orthogonal decomposition (POD), which is customarily used in combination with Galerkin projection to reduce high fidelity models by projecting it to a lower dimensional subspace, the proper symplectic decomposition (PSD) does the same while preserving the symplectic structure of Hamiltonian systems. Rather than relying on the Petrov-Galerkin projection as the POD does, which often can lead to instability in the reduced Hamiltonian subspace because energy and flow volume are not conserved, the PSD algorithm preserves energy and maintains stability by constructing a symplectic basis to approximate the low dimensional symplectic subspace solution [1] .

The PSD of a Hamiltonian system constructs a symplectic matrix A_{2Nx2K} such that the Hamiltonian system $H(u)$ is best approximated on a lower-dimensional subspace of dimension K and is as follows:

$$A^\top \mathbb{J}_{2N} A = \mathbb{J}_{2K}$$

The lower dimensional time derivative approximation of the solution to the Hamiltonian system, $\dot{\zeta}$, $\tilde{H}_d(\zeta) = H_(A\zeta)$, is written in a similar manner to the original system given in equation (32):

$$\dot{\zeta} = A^+ \mathbb{J}_{2N} \nabla_u H(u) = \mathbb{J}_{2K} \nabla_\zeta H_(A\zeta) \tag{41}$$

where A^+ is the symplectic inverse of the symplectic matrix A. PSD algorithms for Cotangent Lift and complex Singular Value Decomposition (SVD) are included due to their popularity as symplectic model reduction techniques. Each algorithm maintains the form of the original Hamiltonian system, thus the symplectic structure and provides a procedure for constructing the symplectic matrix. Thereafter, the reduced order model and the corresponding reduced Hamiltonian $\tilde{H}_d : \mathbb{R}^{2K} -> \mathbb{R}$.

Algorithm 2 Cotangent Lift

Require: An empirical data ensemble $\{q(t_i), p(t_i)\}_{i=1}^N$
Ensure: A symplectic matrix A_1 in block-diagonal form.
1. Construct an extended snapshot matrix M_1.
2. Compute the SVD of M_1 to obtain a POD basis matrix ϕ.
3. Construct the symplectic matrix $A_1 = diag(\phi, \phi)$

Algorithm 3 Complex SVD

Require: : An empirical data ensemble $\{q(t_i), p(t_i)\}_{i=1}^{N}$
Ensure: : A symplectic matrix A_2 in block form.
1. Construct a complex snapshot matrix M_2
2. Compute the SVD of M_2 to obtain a basis matrix $\phi + i\phi$
3. Construct the symplectic matrix $A_2 = [\phi, -\psi; \psi, \phi]$.

Algorithm 4 Greedy Algorithm to construct an index vector, β

Require: A basis matrix $\psi = [\psi_1, ..., \psi_m] \in \mathcal{R}^{n\times m}$
Ensure: An index vector $\beta = [\beta_1; ...; \beta_m] \in \mathcal{R}^m$
1. Select the first interpolation index $[\rho, \beta_1] = max\{|\psi_1|\}$
2. Initialize $U = [\psi_1], \beta = \beta_1$
for i = 2 to m do **do**
3. Solve the coefficient vector τ to match ψ_i, $U(\beta, :)\tau = \psi_i(\beta)$
4. Calculate the residual $r = \psi_i - U\tau$
5. Select the interpolation index corresponding to the largest magnitude of the residual $r, [\rho, \beta_i] = max\{|r|\}$
6. Update $U = [U, \psi_i], \beta = [\beta; \beta_i]$.
end for

6.1.2 Dynamic Mode Decomposition and Discrete Empirical Interpolation Method

PSD algorithms can also be considered computationally expensive due to the *symplectic lift* of the entire ensemble data, which can be proportional to the computational expense of modeling the high dimensional, full order model. Thereafter, adaptive time stepping algorithms are proposed extract accurate and prevalent dynamics of the flow whilst addressing the computational bottleneck of a large number of particles to form a sufficient and stable solution. This proposed approach is known as a *hyper reduction* technique and first proposed in [7]. The combination of Dynamic Mode Decomposition (DMD) applied for the hyper-reduction of the electric potential $\mathbf{\Phi}(x, t)$ and Discrete Empirical Interpolation Methods (DEIM) for the particle-to-grid mapping of the reduced Hamiltonian allows for a more computationally efficient approximation to the nonlinear Hamiltonian system [7].

Dynamic Mode Decomposition

Dynamic mode decomposition is primarily used to *drive* a reduced model using data measurements from the full order dynamical system. In the case of complex flows, DMD attempts to preserve coherent spatial structures (or the *DMD modes*) and match the resulting modes to temporal variations in the flow [7].

In application to the Vlasov-Poisson equation, the DMD method will be adapted to incorporate a "sliding-window" or sampling window where the local linear system can be approximated using only the most recent data. Thereafter, the scheme becomes *adaptive* in nature and addresses the computational burden that most offline stages have in traditional PSD schemes. Essentially, DMD both reduces our computational efforts by needing only a snapshot of one temporal interval to form the ROM and could potentially be considered more accurate as it recognizes time variation of the data measurements.

DMD methods are used to hyper-reduce the electric potential $\mathbf{\Phi}$, dependent on the reduced position state $X_r^i(t)$, and the resulting reduced basis $\mathbf{\Phi}_r$ is used in the computation of the reduced Hamiltonian system. Essentially, we create a time series of these reduced $\mathbf{\Phi}$s from each temporal interval, or time-discrete approximations of the electric potential for a time window length $T + 1$ and fixed parameter, i.

$$\mathbf{G}_i = [\phi^i_{\tau-T-1}\phi^i_{\tau-T}...\phi^i_{\tau-2}] \tag{42}$$

$$\mathbf{G'}_i = [\phi^i_{\tau-T}\phi^i_{\tau-T+1}...\phi^i_{\tau-1}] \tag{43}$$

With the assumption that the electric potential does not have explicit dependence on the parameter indexed, we can drop the superscript i and concatenate column-wise the resulting $\mathbf{G}_i$, $\mathbf{G'}_i$ data sets to form the generalized global data sets $\mathbf{G}$, $\mathbf{G'}$. Thereafter, DMD is used to approximate the projection of the mapping, symplectic matrix A. *Discrete Empirical Interpolation Method (DEIM)*

In connection to the DMD of the electric potential, the DEIM is used to hyper-reduce the approximate reduced Hamiltonian as particle-to-grid mapping is still a major computational burden. Essentially, the DEIM finds a way to accurately replace the nonlinear Hamiltonian operator with a projection of only the most important components of the function into a lower-dimensional space [1].

The reduced dynamical system is thus approximated with the following equations, noting that $g_H^{DD}(U, Z_i)$

is the DMD-DEIM approximation of the gradient of the reduced hamiltonian.

$$\dot{Z}(t) = \mathbb{J}_{2N} g_H^{DD}(U, Z, t), \tag{44}$$

$$\dot{U}(t) = (\mathbb{I}_{2N} - UU^T)(\mathbb{J}_{2N} G_H^{p*}(U, Z) Z^T - G_H^{p*}(U, Z) Z^T \mathbb{J}_{2N}^T) S(Z)^{-1}, \tag{45}$$

6.1.3 Hamiltonian Neural Networks

A Hamiltonian Neural Network (HNN) is a data driven process which aims to reduce the dimension of a model when used in combination with an autoencoder, an unsupervised machine learning process. The HNN learns the data from the "compressed" data, the first stage of the autoencoder, then reconstructs the solution in the original dimensional subspace through "decompression", the second stage of the autoencoder [11].

The HNN aims to reduce the model of plasma flow given by the Vlasov-Poisson equation by *learning* the low dimensional Hamiltonian received from the compression stage; as investigated in [11]. The error associated with the first stage of the autoencoder and the resulting HNN are of the same order, meaning the HNN is able to accurately learn the model given the compressed data.

A possible drawback to this method is its' invalidity when used in combination with periodic data, which begs the question: can an autencoder be used with another data-driven reduction technique? Nevertheless, machine learning techniques like HNN are continuously being researched and developed for time dependent predictive models that exploit the Hamiltonian formulation of a system.