Journal of Computational Physics 498 (2024) 112662

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Suppressing instability in a Vlasov–Poisson system by an external electric field through constrained optimization

Lukas Einkemmer [a,*], Qin Li [b], Li Wang [c], Yunan Yang [d]

[a] *Universität Innsbruck, Innsbruck, Austria*
[b] *University of Wisconsin, Madison, WI, United States of America*
[c] *University of Minnesota, Minneapolis, MN, United States of America*
[d] *Cornell University, Ithaca, NY, United States of America*

ARTICLE INFO

Keywords:
Control
Vlasov–Poisson equation
Suppression of plasma instability
Nonlinear dynamics
Discretize-then-optimize
Semi-Lagrangian schemes

ABSTRACT

Maintaining the stability and shape of a plasma is a crucial task in many technological applications ranging from beam shaping to fusion energy. This is often challenging as plasma systems tend to be naturally unstable and kinetic effects can play an important role in the behavior of the instabilities. Due to the large energies involved, the primary way to control plasma systems is the application of external electromagnetic fields.

In this work, we deploy a PDE-constrained optimization formulation that uses a kinetic description for plasma dynamics as the constraint. This is to optimize, over all possible controllable external electric fields, the stability of the plasma dynamics under the condition that the Vlasov–Poisson (VP) equation is satisfied. Mathematically we formulate it as a PDE-constrained optimization. For computing the functional derivative with respect to the external field in the optimization updates, the adjoint equation is derived. Furthermore, in the discrete setting, where we employ the semi-Lagrangian method as the forward solver, we also explicitly formulate the corresponding adjoint solver and the gradient as the discrete analogy to the adjoint equation and the Fréchet derivative. A distinct feature we observed of this constrained optimization is the complex landscape of the objective function and the existence of numerous local minima, largely due to the hyperbolic nature of the VP system. To overcome this issue, we utilize a gradient-accelerated genetic algorithm, leveraging the advantages of the genetic algorithm's exploration feature to cover a broader search of the solution space and the fast local convergence aided by the gradient information. We show that our algorithm obtains good electric fields that are able to maintain a prescribed profile in a beam shaping problem and uses nonlinear effects to suppress plasma instability in a two-stream configuration.

1. Introduction

The stability of plasma systems remains an active area of research. Plasma instabilities, which are responsible for many phenomena observed in astrophysical plasma (see, e.g., [67]), are a key focus of the investigation. However, in engineering applications, the focus is often on shaping the plasma in a specific manner, such as beam shaping in particle accelerators [23], or confining the plasma,

* Corresponding author.
E-mail address: lukas.einkemmer@uibk.ac.at (L. Einkemmer).

https://doi.org/10.1016/j.jcp.2023.112662
Received 30 May 2023; Received in revised form 6 October 2023; Accepted 20 November 2023

Available online 23 November 2023

such as in fusion reactors [62]. In these cases, instabilities can cause potentially dangerous disruptions to the intended operation mode, leading to loss of confinement and the deposition of large amounts of energy at the walls of the device. Magnetohydrodynamics, a fluid model, is commonly used to study stability in the context of magnetic confinement fusion [50].

However, stability in the fluid regime does not necessarily guarantee stability when kinetic effects are taken into account [69,13]. In fact, the system's behavior is much more complex, as even spatially homogeneous equilibrium distributions can exhibit interesting behavior. One such example is the celebrated Landau damping, which was conjectured by Landau [48], experimentally confirmed in [53], and mathematically proven in [54]. It is a physical phenomenon where waves propagating through plasma can be damped due to the interactions between the waves and the charged particles. Specifically, as the wave travels through the plasma, it creates a fluctuating electric field that causes the charged particles to oscillate. These oscillations then interact with the wave, causing it to lose energy and dampen its amplitude. In contrast, the configuration of two spatially homogeneous beams is unstable and leads to an exponential increase in the electric field as well as a drastic change in the distribution function [59,13]. This is the famous two-steam instability.

Mathematically, the problem can be formulated as a PDE-constrained optimization problem. The Vlasov–Poisson equation acts as the constraint, while the optimization process determines the external electric field to achieve maximum plasma stability. The objective functionals we use minimize the mismatch between the Vlasov–Poisson solution and a predetermined state at a fixed time horizon.

For the time and space discretization, we use a Strang splitting-based semi-Lagrangian discontinuous Galerkin approach [21,56,60]. Splitting-based semi-Lagrangian schemes are commonly used for the simulation of kinetic problems as they reduce a (potentially) high-dimensional Vlasov–Poisson or Vlasov–Maxwell problem to a sequence of one-dimensional advection, are free of any CFL (Courant–Friedrichs–Lewy) condition, and do not suffer from the numerical noise inherent in particle methods (see, e.g., [32]). In the case of the semi-Lagrangian discontinuous Galerkin approach, they are also completely local (which has, for example, benefits on high-performance computing and GPU-based systems; see e.g. [29]). However, this property also helps derive the adjoint equations required to compute the gradient in our optimization algorithm. Another popular method to conduct high dimensional simulations for the Vlasov–Poisson or Vlasov–Maxwell equations is the particle-in-cell (PIC) method [18,3,58,40], which approximates the distribution function by superparticles with a weighted representation, and to follow the trajectories of those superparticles.

It is very important that we take the discretized form of the equations into account when deriving the adjoint solver (called the discretize-then-optimize approach) [52]. Otherwise, a careless discretization of the continuous adjoint equation (called the optimize-then-discretize approach) may yield an inconsistent numerical scheme with respect to the discretization for the forward equation, resulting in a low-order or even incorrect gradient computation [38]. Due to the symmetric nature of Strang splitting and the simple characteristics that result from it, the adjoint solver (i.e., the backward problem) takes a form that is very similar to that of the forward problem. Therefore, only minimal modifications to the forward solver are required to produce the adjoint, which would finally feed into the optimization pipeline.

The past decades have seen drastic advances in kinetic-equation-based inverse problems, optimal control, and optimal design, many of which fall within the PDE-constrained optimization framework. In particular, for various applications, the kinetic models employed include the radiative transport equation [57,4,51,14,24], Boltzmann equation [1,61,10], Fokker–Planck equation [15,2, 34], and the Vlasov–Poisson system and its drift-diffusion limit [17,49,9,8]. Many of the problems involve determining unknown parameters in kinetic equations, and the numerical strategy is to deploy the optimization process that looks for the value of the unknown parameter that minimizes the difference between the PDE-simulated data and the true experimental data, hence formulating a PDE-constrained optimization. PDE-constrained optimization is extensively studied for elliptic and hyperbolic PDEs [39], and its specific use for kinetic models is comparatively less heavily investigated, partially due to the high computational cost in the forward simulation: Kinetic models are usually posed on phase domains with spatial and velocity directions both needing to be resolved, making each optimization iteration expensive to compute.

The Vlasov–Poisson (VP) equation is the primary focus of this paper. The equation is widely used in semiconductor studies and fusion energy. In particular, the voltage-to-current map is utilized to reconstruct the doping profile in the VP system in the semiconductor industry [17,49,9,8]. For fusion energy studies, Glass and Han-Kwan investigated the controllability of the system in [35,36], where the external force term serves as the control. In a series of papers [45–47], Knopf and collaborators examined the optimal control problem for this system with the external magnetic field to be tuned. More recently, a mathematical analysis of a PDE-constrained optimization problem for the Vlasov–Poisson system with an external magnetic field and particle-in-cell discretization was conducted in [6], a setting that is relevant for the equilibrium configuration of fusion reactors.

We consider two applications of the proposed optimization algorithm in this work. First, we consider a focusing problem, where the goal of applying an external electric field is to maintain a specific localized shape of the distribution function (this is related to, e.g., beam shaping problems). Second, we show that the optimization algorithm can be used to suppress the onset of the two-stream instability. Since the two-stream instability is unstable in the linear theory, the optimization algorithm has to find a nonlinear effect in order to suppress the unstable modes. This is accomplished by exciting certain higher modes via the external electric field that then mixes nonlinearly with the unstable modes in a beneficial way. In both cases, we observe that the optimization landscape consists of many local minima, even by parametrizing the external electric field. To overcome this, we use a genetic optimization algorithm to produce candidate solutions. The candidate solutions are then polished using the developed gradient-based optimization algorithm. This hybrid approach improves the convergence of the algorithm significantly.

The rest of the paper is organized as follows. In Section 2, we formulate the optimization problem constrained by the Vlasov–Poisson equation introduced below in Section 1.1 and derive the adjoint equation and the gradient formula on the continuous level using first-order optimality conditions. In Section 3, we first present the semi-Lagrangian discretization for the Vlasov–Poisson system.

We then derive the corresponding consistent adjoint system and the gradient formulation through first-order optimality conditions. The relationship between the two adjoint systems in Section 2 and Section 3 is discussed in Remark 3.4. In Sections 4 and 5, we present two concrete control examples, one on maintaining the focusing beams and one on suppressing the two-stream instabilities. The conclusion follows in Section 6.

1.1. The Vlasov–Poisson system

In this paper, we consider the Vlasov–Poisson equation with external electric field

$$\begin{cases}\partial_t f + v\cdot\nabla_x f - E^{\mathrm{t}}_f\cdot\nabla_v f = 0\,,\\ E^{\mathrm{t}}_f = H + \nabla_x V_f\,,\\ \Delta V_f = 1-\rho_f(t,x) = 1-\int f\,\mathrm{d}v\,,\end{cases}\tag{1.1}$$

where $f(t,x,v)$ stands for the density of plasma particles on the phase space $(x,v)\in\mathbb{R}^{2d}$ at a particular time $t\in\mathbb{R}^+$. Here, d is the dimension of both the spatial and velocity spaces. Along the characteristics, plasma moves according to the dynamics

$$\dot{x} = v\,,\quad \dot{v} = -E^{\mathrm{t}} = -(H+\nabla V_f)\,,$$

where the electric field E^{t} is composed of an external component H, and a self-imposed contribution $\nabla_x V_f$. The potential $V_f(t,x)$ satisfies the Poisson equation at every fixed time t, where the source term depends on the plasma's own density function $\rho(t,x)$. This is to model the situation where particles pose repulsion to each other as the point charge potential $1/r$, and thus self-generate potential. We denote the initial condition as:

$$f(t=0,x,v) = f_0(x,v)\,.$$

The global existence of classical solution to the VP system (1.1) is established in one, two and three spatial dimensions respectively in [41,66,5,55].

Without external electric field $H=0$, any initial data that only depends on v is an equilibrium. Indeed, let $f_0(x,v)=f^{\mathrm{eq}}(v)$. Then the solution would be trivially $f(t,x,v)=f^{\mathrm{eq}}(v)$, making that $\nabla_x f=0$ and $E=H+\nabla_x V_f=0$.

2. A PDE-constrained optimization problem

The problem is formulated into a PDE-constrained optimization, with the objective functional J measuring the instability. Suppose the desired property is to force the distribution $f(T,\cdot,\cdot)$ to be as close as possible to a given equilibrium state f^{eq}. Then J is:

$$J(f[H]) = \frac{1}{2}\|f[H](T)-f^{\mathrm{eq}}\|^2_{L^2(x,v)}\,.\tag{2.1}$$

Here we abbreviate $f(T)$ to denote the distribution function over the phase space at time T, and we take the L^2 norm in both the spatial and velocity spaces to measure the difference. The goal is to tune the external electric field H so that $J(f[H])$ is minimized. The notation $f[H]$ reflects f's dependence on H through the VP system (1.1).

In an experimental setup, plasma is confined in a circular domain. Mathematically, this amounts to viewing y,z-domain as constant, and presenting the 3D problem as a pseudo-1D problem with periodic boundary condition, so $x\in\mathbb{T}$ and $v\in\mathbb{R}$. In this setting, the VP system can be simplified, and we arrive at the following formulation:

$$\min_H \quad J(f[H])\tag{2.2a}$$

$$\text{s.t.}\quad \begin{cases}\partial_t f + v\partial_x f - (H+E[f])\partial_v f = 0\\ E[f] = \partial_x G*(1-\rho_f)\\ f(t=0,x,v) = f_0 = f^{\mathrm{eq}}+\tilde{f},\end{cases}\tag{2.2b}$$

where $f[H]$ stands for the PDE solution f given the source H, $E[f]$ is the density-induced electric field generated by the Poisson kernel convolving with the density. Here, $G(\cdot)$ is Green's function to the 1D Poisson equation with periodic boundary condition, i.e., $G''=\delta_0$. Due to the homogeneity of the media in the Poisson equation, the internal field $\partial_x V$ can be explicitly written as a convolution with this Green's function. The quantity ρ_f, by convention, is the density in space:

$$\rho_f(t,x) = \int f(t,x,v)\mathrm{d}v\,.$$

The initial condition f_0 is assumed to be a small perturbation from a given equilibrium f^{eq}, namely $\tilde{f}\ll 1$.

The objective of this optimization problem is to find an external field H that can suppress the perturbation induced by $\tilde{f}$ and drive the system back to f^{eq} within the time frame of $[0,T]$. For the specific form stated in (2.1), our aim is to bring the distribution at time T close to the equilibrium state. It is worth noting that different stability conditions surrounding the various equilibrium states can pose different levels of challenges. Landau damping automatically sends the evolution towards equilibrium, even without the external H, and the introduction of H is expected to accelerate this damping process. On the other hand, two-stream simulation is inherently unstable, and H is crucial in stabilizing the system, making the design substantially more challenging.

2.1. Lagrangian multiplier

A typical approach to execute the optimization (2.2) is to run gradient-based optimization method, which necessitates the computation of the functional derivative $\frac{\mathrm{d}J}{\mathrm{d}H}$. However, like many other PDE-constrained optimization problems, the objective functional J implicitly depends on H through the PDE constraints, making explicit computation challenging. One solution to this challenge is to introduce the Lagrangian multiplier and utilize an adjoint-state solver [7]. This is the approach we will take in this work.

To be specific, denoting by $g(t,x,v)$ the multiplier for the VP equation and $\eta(x,v)$ the multiplier for the initial condition, we define the Lagrangian:

$$L(f,H,g,\eta) = J(f) - \langle \partial_t f + v\partial_x f - (H + \partial_x G * (1-\rho))\partial_v f\,, g\rangle_{x,v,t} - \langle f(t=0) - f_0, \eta\rangle_{x,v}\,. \tag{2.3}$$

The original problem (2.2) then translates to an unconstrained formulation with f separated from its dependence on H:

$$\min_{f,H,g,\eta} L(f,H,g,\eta)\,.$$

Note that the argument to be minimized is now changed to the whole set of (f,H,g,η). The first-order optimality condition requires the following:

$$\partial_g L = 0\,, \quad \partial_f L = 0\,, \quad \partial_H L = 0\,, \quad \text{and} \quad \partial_\eta L = 0\,.$$

The first equation corresponds directly to the requirement that the PDE constraints in (2.2b) must hold, and the second equation leads to the adjoint equation that g needs to satisfy. More precisely, we perform integration by parts of (2.3), to move all the (partial) derivatives from f to g. That is, upon a straightforward calculation:

$$\begin{aligned} &L(f,H,g,\eta) \\ &= J(f) - \langle f(t=T), g(t=T)\rangle_{x,v} + \langle f(t=0), g(t=0)\rangle_{x,v} \\ &\quad + \langle f\,, \partial_t g + v\partial_x g - H\partial_v g\rangle_{x,v,t} - \langle f(t=0) - f_0, \eta\rangle_{x,v} \\ &\quad - \langle \partial_x G * (1-\rho_f) f, \partial_v g\rangle_{x,v,t}\,, \end{aligned} \tag{2.4}$$

where we note that

$$\langle \partial_x G * \rho_f f,\ \partial_v g\rangle_{x,v,t} = \langle f(t,x,v),\ \langle \partial_x G(x-y) f(t,y,w)\partial_w g(t,y,w)\rangle_{y,w}\rangle_{x,v,t}\,. \tag{2.5}$$

Since the optimality condition requires

$$\frac{\partial L}{\partial f}(t,x,v) = 0\,, \quad 0 \le t < T\,, \quad \text{and} \quad \frac{\partial L}{\partial f(T)}(x,v) = 0,$$

the former, utilizing (2.4) and (2.5), yields the following adjoint equation after some calculations:

$$\partial_t g + v\partial_x g - H\partial_v g + [G' * (\rho_f - 1)]\partial_v g + G' * \langle f, \partial_v g\rangle_v = -\frac{\partial J}{\partial f}(t,x,v)\,. \tag{2.6}$$

The latter provides the final condition for the adjoint state g:

$$g(T,x,v) = \frac{\partial J}{\partial f(T)}(x,v) = f(T) - f^{\mathrm{eq}}\,. \tag{2.7}$$

The differential equation (2.6) together with the final condition (2.7) form the adjoint equations that g satisfies. Note that this requires us to solve g backward in time (starting from the final condition and integrating until we reach time $t=0$).

To compute the gradient, first note that for the Lagrangian L defined in (2.3), we have

$$\partial_H L = \langle \partial_v f\,, g\rangle_{v,t}\,.$$

Now if we restrict f to the solution manifold of (2.2b), then the last two terms in L vanish, and $L \equiv J$ holds on this manifold, i.e., $L(f[H]) = J(f[H])$. Consequently,

$$\partial_H J = \partial_H L = \langle \partial_v f\,, g\rangle_{v,t}\,, \tag{2.8}$$

where $f = f[H]$ solves the PDE constraint (2.2b) and $g = g[H]$ solves the adjoint equations (2.6) and (2.7).

We remark that different choices of J change (2.6) only through the source term on the right-hand side, and they determine the final-time condition for $g(T,x,v)$ in (2.7). We also note that due to the nonlinearity of the last term in (2.4), the adjoint equation (2.6) does not have the same form as the forward problem—specifically, the last term on the left-hand side results from the quadratic nonlinearity. In sum, (2.6)-(2.7) provide the adjoint equation that g needs to satisfy. The term adjoint variable is commonly used to refer to g, while the state variable is represented by the VP solution f.

2.2. Gradient-based method

To sum up, the derivative $\frac{\mathrm{d}J}{\mathrm{d}H}$ is computed in (2.8) where f satisfies the constraints in (2.2b) while g satisfies (2.6) with the final condition given by (2.7). The two equations are solved forward and backward in time, respectively.

With the functional gradient in hand, executing the gradient-based optimization method is straightforward. We summarize the gradient descent applied in this particular setting in Algorithm 1.

Algorithm 1 GD for simulating (2.2).

```
Given $f_0$, $f^{\mathrm{eq}}$, number of iterations $it$, and the initial guess $H^0$
for $n = 0, 1, \dots, it-1$ do
    Compute $f[H^n](t,x,v)$ according to (2.2b);
    Compute $g[H^n](t,x,v)$ according to (2.6) and (2.7);
    Assemble $\frac{\mathrm{d}J}{\mathrm{d}H}\Big|_{H^n} = \langle \partial_v f[H^n], g[H^n]\rangle_{v,t}$ using (2.8);
    $H^{n+1} = H^n - h\,\frac{\mathrm{d}J}{\mathrm{d}H}\Big|_{H^n}$, where $h$ is determined by line search;
    Evaluate $J(f[H^{n+1}])$ using (2.1).
end for
```

It is worth noting that for a crude estimate, one can proceed by designing numerical solvers for (2.2b) and (2.6) individually and assembling them in (2.8). This procedure is usually termed Optimize-Then-Discretize (OTD). This approach, however, may easily introduce incompatibility between forward and adjoint solvers, degrading the accuracy of the gradient computation and leading to slow convergence in optimization. See discussion in [39, Section 3.2.3] and [51].

One numerical strategy to overcome such incompatibility is, to begin with discretization and translate the PDE constraints into an algebraic system on which the optimization is performed. This approach is commonly referred to as the Discretize-Then-Optimize (DTO) approach. This approach considers only the discrete system, and the adjoint is naturally presented in the discrete setting. As a result, the compatibility is automatic, and accuracy is maintained during the final assembly of the functional gradient. This approach forms the basis of our discussion in Section 3.

3. Discretize-then-optimize formulation

In this section, we derive the discrete counterpart of the problem (2.2) and deploy the DTO approach to design the associated algorithm as a discrete analog of Algorithm 1. This approach calls for a preset discrete system to represent the PDE and the objective function, which we summarize in Section 3.1, and the adjoint derivation is provided in Section 3.2.

3.1. Discretization of the VP system and the objective function

There are many ways to discretize the VP system in time and space. The Eulerian methods, such as [32] and Lagrangian methods, such as [68], have both been widely used. We focus here on the semi-Lagrangian methods, which, for the VP system, date back to the seminal paper by Cheng & Knorr [16]. A main advantage of semi-Lagrangian methods is that they are fully explicit but still unconditionally stable, i.e., they do not suffer from a CFL condition.

The main idea of the semi-Lagrangian method is to trace back the characteristics exactly and perform the interpolation/projection approximately when the translated solution does not necessarily coincide with the grid points/approximation space. This can be done in a variety of ways. Both spline based [16,63,32] and Fourier based methods [43,44] have been used heavily. This work uses the more recently developed semi-Lagrangian discontinuous Galerkin approach [21,56,60,27]. Those methods are local, which has many advantages when implementing such problems on high-performance computing systems [29,25] and GPU based [26,28] systems. In the present context, however, the approach is convenient as we can write the scheme as an explicit matrix-vector product, which helps in deriving the adjoint equation (there is, e.g., no tridiagonal solve as is the case for spline interpolation).

To further tame the high dimensionality that may appear in the problem, the semi-Lagrangian methods are often combined with a (Strang) splitting procedure in order to reduce the problem to one-dimensional advection equations (i.e., the characteristics become extremely simple in this case). The splitting that we describe here is Hamiltonian and thus can be shown to give good long-time results with respect to energy conservation. It can also be generalized to higher-order (see [12]) and more complicated models (for the Vlasov–Maxwell system see [20]).

In the following, we detail our discretization procedure and provide a practical formula for gradient computation.

3.1.1. Time discretization with splitting

We perform the splitting in time to isolate the computation in space and velocity separately. Consider the uniform time step Δt and denote $t_n := n\Delta t$. Then within each time step, to update the solution from time t^n to time t^{n+1}, we split the PDE operator into

$$\partial_t f + v\partial_x f = 0\,, \quad \text{and} \quad \partial_t f + (E+H)\partial_v f = 0\,.$$

We define the operators $A = -v\partial_x$, and $B(E) = (E+H)\partial_v$. We also use f^n as the short-hand notation for $f(t^n, x, v) = f(n\Delta t, x, v)$. Then the solution writes, deploying the second-order Strang splitting scheme [30]:

$$f^{n+1} = e^{\frac{\Delta t}{2}A} e^{\Delta t B(E^{n+1/2})} e^{\frac{\Delta t}{2}A} f^n \,, \tag{3.1}$$

where $E^{n+1/2}(x) = E\left[e^{\frac{\Delta t}{2}A} f^n\right]$ is regarded as constant-in-time when deployed. Since each of these two operations can be executed exactly, we can analytically write down the action of these operations in the following updating procedure:

(1) Compute $f^\star(x,v) = f(t^n, x - v\Delta t/2, v)$.
(2) Compute $E^{n+1/2}(x)$ by solving $\partial_x E^{n+1/2} = 1 - \rho_{f^\star}$.
(3) Compute $f^{\star\star}(x,v) = f^\star(x, v + (E^{n+1/2} + H)\Delta t)$. Note that here $E^{n+1/2} + H$ depends on space x but **not** on time t.
(4) Compute $f(t^{n+1}, x, v) = f^{\star\star}(x - v\Delta t/2, v)$.

Remark 3.1. To show this method provides second-order accuracy, as expected by the Strang splitting, one needs to verify that the $E^{n+1/2}$ computation is, in fact, a first-order approximation of $E(f^{n+1/2})$. Such error analysis amounts to comparing $\rho_{f^\star}$ with $\rho_{f^{n+1/2}}$. This can be done because, within each Δt,

$$\rho_{f^\star} = \int e^{\frac{\Delta t}{2}A} f^n \, \mathrm{d}v = \int e^{\frac{\Delta t}{2}B(E^n)} e^{\frac{\Delta t}{2}A} f^n \, \mathrm{d}v = \rho_{f(t=(n+1/2)\Delta t)} + \mathcal{O}(\Delta t) \,,$$

where we used the fact that $e^{\frac{\Delta t}{2}B(E^n)}$ is a translation in velocity space and thus does not change the velocity integration.

Another popular time-splitting choice is the Lie splitting

$$f^{n+1} = e^{\Delta t B(E^0)} e^{\Delta t A} f^n \,.$$

Not only does this splitting provides only the first-order convergence, but also induces complication in the adjoint solver (see subsection 3.2). Instead, Strang splitting requires a symmetric application of the two operators, and the adjoint solver is more convenient:

$$\left(e^{\frac{\Delta t}{2}A} e^{\Delta t B(E^{n+1/2})} e^{\frac{\Delta t}{2}A}\right)^* = e^{-\frac{\Delta t}{2}A} \left(e^{\Delta t B(E^{n+1/2})}\right)^* e^{-\frac{\Delta t}{2}A} \,, \tag{3.2}$$

where we used $*$ to denote the adjoint. Thus, for the linear case, i.e., where $E^{n+1/2}$ is independent of time and considered an external input, we get the original scheme (3.1) except that Δt is replaced with $-\Delta t$.

Then similar to the notation above, it is tempting to define

$$g^{n,\star\star} = e^{-\frac{\Delta t}{2}A} g^n \,, \quad g^{n,\star} = \left(e^{\Delta t B(E^{n+1/2})}\right)^* \quad \text{and} \quad g^{n-1} = e^{-\frac{\Delta t}{2}A} g^{n,\star} \,. \tag{3.3}$$

3.1.2. *Semi-Lagrangian scheme in space*

To proceed, we also divide the phase space into cells $C_{ij} = [x_{i-1/2}, x_{i+1/2}] \times [v_{j-1/2}, v_{j+1/2}]$ of size $\Delta x \times \Delta v$, with $(i,j) \in [1, n_x] \times [1, n_v]$. In each cell, we approximate $f^n(x,v)$ by a constant value that is denoted by f^n_{ij}. Denoting χ as the characteristic function, our approximation is

$$f^n(x,v) \approx \sum_{ij} \mathsf{f}^n_{ij} \, \chi_{C_{ij}}(x,v) \,. \tag{3.4}$$

To simplify the notation we concatenate f^n for:

$$\mathsf{f}^n = [\mathsf{f}^n_{ij}]_{i=1,j=1}^{n_x,n_v} \in \mathbb{R}^{n_x \times n_v} \,. \tag{3.5}$$

What we describe here and in the following is the simplest case of a semi-Lagrangian discontinuous Galerkin scheme. In general, the assumption of having a piecewise constant function can be replaced by piecewise polynomials in order to obtain a numerical method of higher order (for more details see [21,56,60,27]). This has the advantage that, in many applications, we require fewer cells and that numerical diffusion is reduced. While in the following, for the sake of simplicity, we only consider the piecewise constant case (which gives a second-order scheme in space), let us emphasize that the same can be done for the higher-order variants. This is possible since the general structure of the update, which can be written as the linear combination of the degrees of freedom in two adjacent cells, also holds true for higher-order semi-Lagrangian discontinuous Galerkin schemes.

In this fully discrete setting, we now need to translate the operators ($e^{\frac{\Delta t}{2}A}$ in Steps 1 and 4 and $e^{\Delta t B(E^{n+1/2})}$ in Step 3) to the corresponding matrices, which we discuss below.

Computation of $e^{\frac{\Delta t}{2}A} = e^{-\frac{\Delta t}{2}v\partial_x}$. Noting that a direct application of this operator on a function leads to

$$e^{\frac{\Delta t}{2}A} f^n(x,v) = f^n(x - v\Delta t/2, v) = \sum_{ij} \mathsf{f}^n_{ij} \, \chi_{C_{ij}}(x - v\Delta t/2, v).$$

Since $v\Delta t/2$, in general, is not a multiple of Δx, the resulting function does not lie in our approximation space. Therefore we have to perform a projection (see Fig. 1 for an illustration). That is, we look for an approximation $\mathsf{f}^{n,\star}_{ij}$ such that

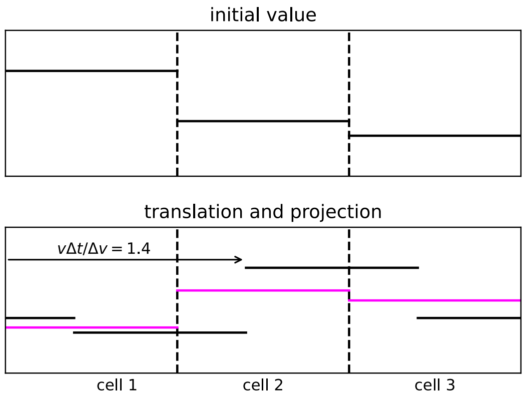

Fig. 1. Illustration of the semi-Lagrangian discontinuous Galerkin scheme (note that for the first scheme described here we have a constant, i.e., a polynomial of degree 0, in each cell).

$$\sum_{ij} \mathsf{f}_{ij}^{n,\star}\, \chi_{C_{ij}}(x,v) \approx \sum_{ij} \mathsf{f}_{ij}^{n}\, \chi_{C_{ij}}(x - v\Delta t/2, v)\,.$$

We choose linear interpolation using the two nearby cells. Decompose $-v\Delta t/(2\Delta x)$ into its integer component and the remainder by setting

$$\frac{-v\Delta t}{2\Delta x} = \mathrm{n}(v) + \alpha(v) := \lfloor -v\Delta t/(2\Delta x)\rfloor + (-v\Delta t/(2\Delta x) - \lfloor -v\Delta t/(2\Delta x)\rfloor)\,, \tag{3.6}$$

where n is the closest lower bound integer, and $\alpha \in [0,1)$. Then the linear interpolation is:

$$\mathsf{f}_{ij}^{n,\star} = (1-\alpha(v_j))\mathsf{f}^n_{i+\mathrm{n}(v_j),j} + \alpha(v_j)\mathsf{f}^n_{i+\mathrm{n}(v_j)+1,j}\,.$$

Due to the definition of $\mathrm{n}(v)$, we have a nice property:

$$\frac{v\Delta t}{2\Delta x} = -\mathrm{n}(v) - \alpha(v) = \underbrace{-\mathrm{n}(v) - 1}_{\mathrm{n}(-v)} + \underbrace{1-\alpha(v)}_{\alpha(-v)} \quad \Rightarrow \mathrm{n}(-v) = -\mathrm{n}(v) - 1\,, \alpha(-v) = 1-\alpha(v)\,. \tag{3.7}$$

Remark 3.2. It is instructive to consider some special cases:

- For $\mathrm{n}(v) = -1$, we have a translation from left to right with CFL < 1, $\alpha = 1 - v$ and our scheme simply becomes an upwind scheme. Here, CFL denotes the constant in the Courant–Friedrichs–Lewy condition [19].
- For $\mathrm{n}(v) = 0$ we have a translation from right to left with CFL < 1, $\alpha = |v|$ and again we obtain the upwind scheme.

With this formulation, it is straightforward to translate the update formula into a matrix form:

$$\mathsf{f}^{n,\star}_{:,j} = \mathsf{A}^{v_j}\mathsf{f}^n_{:,j}\,, \quad \text{with} \quad \mathsf{A}^v_{kl} = \begin{cases} 1-\alpha(v)\,, & l = k + \mathrm{n}(v)\,, \\ \alpha(v)\,, & l = k+\mathrm{n}(v)+1\,, \\ 0\,, & \text{otherwise}\,. \end{cases} \tag{3.8}$$

Here, $\mathsf{A}^v \in \mathbb{R}_{n_x \times n_x}$ is a sparse matrix with only two diagonals being non-zero.

It is worth noting that $(\mathsf{A}^v)^\top = \mathsf{A}^{-v}$, namely, the transpose is precisely the matrix we would get if we replace v by $-v$. To see this, we recall (3.7) that produces $\mathsf{A}^{-v}_{kl} = \mathsf{A}^v_{lk}$. This nice property reflects the fact that the operator is anti-self-adjoint, and the computation suggests the semi-Lagrangian formulation preserves such property on the discrete level.

Computation of $E^{n+1/2}(x)$. In this fully discrete setting, we follow (3.4) to write, within n-th step,

$$\mathsf{E}^{n,\star} \approx E^{n+1/2} \quad \text{with} \quad \mathsf{E}^{n,\star} = [\mathsf{E}^{n,\star}_i] = \mathsf{E}(\mathsf{f}^{n,\star})\,, \tag{3.9}$$

where $\mathsf{E} : \mathbb{R}^{n_x \times n_v} \to \mathbb{R}^{n_x}$ with:

$$\mathsf{E}(\mathsf{f}) = \mathsf{C}\left(1-\rho_{\mathsf{f}}\right)\,, \quad \text{where} \quad \mathsf{C}_{ij} = \partial_x G(x_i - x_j) \quad \text{and} \quad \left(\rho_{\mathsf{f}}\right)_i = \frac{1}{n_v}\sum_j \mathsf{f}_{ij}\,, \tag{3.10}$$

where G is the Green's function for the Poisson equation. In practice, we do not form G explicitly, but only compute its action to a vector on the fly via the Fast Fourier Transform. More precisely, we have

$$\mathsf{E}(\mathsf{f}) = \mathcal{F}^{-1}\left[\mathrm{diag}\left(0, \frac{\mathrm{i}}{1}, \cdots, \frac{\mathrm{i}}{n_x - 1}\right)\mathcal{F}(1-\rho_f)\right],$$

where i is the imaginary unit, $\mathcal{F}$ and $\mathcal{F}^{-1}$ represent the Discrete and Inverse Discrete Fourier transforms, respectively.

Computation of $e^{\Delta t B(E^{n+1/2})}$. The computation of transporting in the velocity domain is exactly the same as before, except the velocity term v is now replaced by the acceleration term $\mathsf{E}^{n,\star}$, and the action of the operator is taken on the velocity space. Namely, we work on the vector of

$$\mathsf{f}^{n,\star\star}_{i,:} = \mathsf{f}^{n,\star}_{i,:}\mathsf{B}^{\mathsf{E}^{n,\star}_i+\mathsf{H}_i}, \quad \text{with} \quad \mathsf{B}^a_{lk} = \begin{cases} 1-\alpha(2a), & l = k + \mathrm{n}(2a), \\ \alpha(2a), & l = k+\mathrm{n}(2a)+1, \\ 0, & \text{otherwise}, \end{cases} \tag{3.11}$$

where $\mathsf{B} \in \mathbb{R}^{n_v \times n_v}$, and H is a vector of length n_x with $\mathsf{H}_i = H(x_i)$. Notice that α and n contain an extra factor of 2 here because this step is run on a time interval of Δt instead of $\Delta t/2$, as described in (3.6).

To summarize the calculations above, we describe the fully discrete update formula in Algorithm 2.

Algorithm 2 Semi-Lagrangian scheme for solving (2.2b) for a given H.

```
Parameter: $\mathsf{H}$
Input: initial condition $\mathsf{f}^0$, discretization set $(x_i, v_j)$ for $(i,j) \in [1,n_x]\times[1,n_v]$, $\Delta t$, and $N$ so that $T = N\Delta t$, the semi-Lagrangian matrices $\mathsf{A}^{v_j}$ for all $j$, and $\mathsf{H}$
Output: $\mathsf{f}^N$
for $n < N$ do
    Compute $\mathsf{f}^{n,\star}$ according to (3.8);
    Compute $\mathsf{E}^{n,\star}$ using (3.9);
    Compute $\mathsf{f}^{n,\star\star}$ according to (3.11) with the updated $\mathsf{B}^{\mathsf{E}^{n,\star}_i+\mathsf{H}_i}$;
    Compute $\mathsf{f}^{n+1}$ according to (3.8).
end for
```

Reformulation of the optimization The optimization problem (2.2) was originally presented as a PDE-constrained problem on the continuous level. However, when we discretize the Vlasov–Poisson equation, the problem becomes an algebraic one and the constraints are also transformed into algebraic forms. Therefore, we need to translate the optimization problem into a discrete form that matches the new algebraic constraints as follows:

$$\min_{\mathsf{H}} \quad \mathsf{J}(\mathsf{f}[\mathsf{H}]) = \frac{1}{2}\sum_{ij} |\mathsf{f}^N_{ij} - \mathsf{f}^{\mathrm{eq}}_{ij}|^2 \Delta x \Delta v \tag{3.12a}$$

$$\text{s.t.} \quad \mathsf{f}^N \text{solves Algorithm 2 for a given H}. \tag{3.12b}$$

Note that though the definition of J only calls for f^N at the final time, its computation goes through iterations from $n = 1$ to $n = N$, so we concatenate everything into one $\mathsf{f} = [\mathsf{f}^0, \cdots, \mathsf{f}^N]$ to keep a record of all relevant computations.

3.2. *Adjoint state solver for the discrete system*

On the discrete setting, the adjoint equation needs to be designed accordingly. We describe the process of computing the adjoint equation for computing (3.12). To start, we expand out Algorithm 2 to obtain the following Lagrangian:

$$\mathsf{L}(\mathsf{H}, \mathsf{f}, \mathsf{f}^\star, \mathsf{f}^{\star\star}, \mathsf{g}, \mathsf{g}^\star, \mathsf{g}^{\star\star}) = \mathsf{J}(\mathsf{f}) + \text{Term I} + \text{Term II} + \text{Term III} \tag{3.13}$$

with

$$\frac{\text{Term I}}{\Delta x \Delta v} = \sum_{n=0}^{N-1}\sum_{ij}\left[-\mathsf{f}^{n,\star}_{ij} + \left((1-\alpha(v_j))\mathsf{f}^n_{i+\mathrm{n}(v_j),j} + \alpha(v_j)\mathsf{f}^n_{i+\mathrm{n}(v_j)+1,j}\right)\right]\mathsf{g}^{n,\star}_{ij}, \tag{3.14a}$$

$$\frac{\text{Term II}}{\Delta x \Delta v} = \sum_{n=0}^{N-1}\sum_{ij}\left[-\mathsf{f}^{n,\star\star}_{ij} + \left((1-\alpha(2\mathsf{E}^{n,\star}_i + 2\mathsf{H}_i))\mathsf{f}^{n,\star}_{i,j+\mathrm{n}(\mathsf{E}^{n,\star}_i+\mathsf{H}_i)} + \alpha(2\mathsf{E}^{n,\star}_i + 2\mathsf{H}_i)\mathsf{f}^{n,\star}_{i,j+\mathrm{n}(\mathsf{E}^{n,\star}_i+\mathsf{H}_i)+1}\right)\right]\mathsf{g}^{n,\star\star}_{ij}, \tag{3.14b}$$

$$\frac{\text{Term III}}{\Delta x \Delta v} = \sum_{n=0}^{N-1}\sum_{ij}\left[-\mathsf{f}^{n+1}_{ij} + \left((1-\alpha(v_j))\mathsf{f}^{n,\star\star}_{i+\mathrm{n}(v_j),j} + \alpha(v_j)\mathsf{f}^{n,\star\star}_{i+\mathrm{n}(v_j)+1,j}\right)\right]\mathsf{g}^{n+1}_{ij}, \tag{3.14c}$$

where as usual, we concatenate all terms, for example: $\mathsf{f}^* = [\mathsf{f}^{0,*}, \cdots, \mathsf{f}^{N,*}]$, and $\mathsf{g}^{n,\star}$, $\mathsf{g}^{n,\star\star}$, and g^n are the Lagrange multipliers, also known as the adjoint states.

Similar to the continuous setting, suppose we run Algorithm 2 to obtain f, then f is naturally a function of H. On this solution manifold, according to the definition (3.13), Term I, Term II and Term III are all satisfied and thus can be removed, implying $\mathsf{L}(\mathsf{H}, \mathsf{f}[\mathsf{H}]\cdots) = \mathsf{J}(\mathsf{f}[\mathsf{H}])$, and thus, according to the chain rule, we have

$$\nabla_{\mathsf{H}}\mathsf{J} = \nabla_{\mathsf{H}}\mathsf{L} + \nabla_{\mathsf{H}}\mathsf{f}\cdot\nabla_{\mathsf{f}}\mathsf{L}\,.$$

As a result, if we can find the values for g so that $\nabla_{\mathsf{f}}\mathsf{L} = 0$, the constrained derivative in (3.12) is simply $\nabla_{\mathsf{H}}\mathsf{L}$. Noticing that the H information comes in only through Term II, we immediately have:

$$\frac{1}{\Delta x\Delta v}\partial_{\mathsf{H}_i}\mathsf{J} = \frac{1}{\Delta x\Delta v}\frac{\partial\mathsf{L}}{\partial\mathsf{H}_i} = \sum_{n=0}^{N-1}\frac{\Delta t}{\Delta v}\sum_j\left[\mathsf{f}^{n,\star}_{i,j+\mathrm{n}(2\mathsf{E}^{n,\star}_i+2\mathsf{H}_i)+1} - \mathsf{f}^{n,\star}_{i,j+\mathrm{n}(2\mathsf{E}^{n,\star}_i+2\mathsf{H}_i)}\right]\mathsf{g}^{n,\star\star}_{ij}\,, \tag{3.15}$$

where we used $\alpha' = \frac{\Delta t}{2\Delta v}$. This formula holds on account of g that ensures $\nabla_{\mathsf{f}}\mathsf{L} = 0$. To do so, we set up the final time solution and propagate it backwards in time for all values of g. We describe the process below.

Remark 3.3. It is worth noting that since we are utilizing first order semi-Lagrangian DG method, the H dependence is linear and the derivative is rather straightforward. When higher-order DG methods are employed, derivatives take on more complicated forms.

3.2.1. *Setting the final data*

Similar to the continuous setting, to specify the final condition we set $\nabla_{\mathsf{f}}\mathsf{L} = 0$ and get

$$0 = \frac{\partial\mathsf{L}}{\partial\mathsf{f}^N_{ij}} = \underbrace{\Delta x\Delta v(\mathsf{f}^N_{ij} - \mathsf{f}^{\mathrm{eq}}_{ij})}_{\partial\mathsf{J}/\partial\mathsf{f}^N_{ij}} \underbrace{-\ \Delta x\Delta v\mathsf{g}^N_{ij}}_{\text{Term III contribution}}\,, \tag{3.16}$$

from which we derive the final condition for the adjoint state g^N

$$\mathsf{g}^N_{ij} = \mathsf{f}^N_{ij} - \mathsf{f}^{\mathrm{eq}}_{ij}\,,\quad \text{or equivalently}\quad \mathsf{g}^N = \mathsf{f}^N - \mathsf{f}^{\mathrm{eq}}\,. \tag{3.17}$$

We remark that (3.17) is the discretized version of (2.7) for the objective function (2.1).

3.2.2. *To compute $\mathsf{g}^{n,\star\star}$ from g^{n+1}*

This involves backward propagation for half a time step and can be achieved mathematically by taking derivatives with respect to $\mathsf{f}^{n,\star\star}$ and setting the derivative to zero. We first note that, for Term III, the right-hand side of (3.14c) can be written as:

$$\begin{aligned}&\sum_{ij}\left[(1-\alpha(v_j))\mathsf{f}^{n,\star\star}_{i+\mathrm{n}(v_j),j} + \alpha(v_j)\mathsf{f}^{n,\star\star}_{i+\mathrm{n}(v_j)+1,j}\right]\mathsf{g}^{n+1}_{ij}\\ &= \sum_{ij}\mathsf{f}^{n,\star\star}_{ij}\left[(1-\alpha(v_j))\mathsf{g}^{n+1}_{i-\mathrm{n}(v_j),j} + \alpha(v_j)\mathsf{g}^{n+1}_{i-\mathrm{n}(v_j)-1,j}\right].\end{aligned} \tag{3.18}$$

Then, we differentiate (3.13) with respect to $\mathsf{f}^{n,\star\star}_{ij}$ for all i,j,n and set the derivative to be zero:

$$0 = \frac{1}{\Delta x\Delta v}\frac{\partial L}{\partial\mathsf{f}^{n,\star\star}_{ij}} = \underbrace{\left[(1-\alpha(v_j))\mathsf{g}^{n+1}_{i-\mathrm{n}(v_j),j} + \alpha(v_j)\mathsf{g}^{n+1}_{i-\mathrm{n}(v_j)-1,j}\right]}_{\text{Term III contribution}}\underbrace{-\mathsf{g}^{n,\star\star}_{ij}}_{\text{Term II}}\,.$$

Therefore, we obtain the formula

$$\mathsf{g}^{n,\star\star}_{ij} = (1-\alpha(v_j))\mathsf{g}^{n+1}_{i-\mathrm{n}(v_j),j} + \alpha(v_j)\mathsf{g}^{n+1}_{i-\mathrm{n}(v_j)-1,j}\,. \tag{3.19}$$

3.2.3. *To compute $\mathsf{g}^{n,\star}$ from $\mathsf{g}^{n,\star\star}$*

We essentially need to take the variation of L with respect to $\mathsf{f}^{n,\star}_{ij}$ and set it to zero. The contribution from Term I is rather straightforward. To deal with Term II, we note the identity for the right-hand side of (3.14b):

$$\begin{aligned}&\sum_{ij}\left[(1-\alpha(2(\mathsf{E}^{n,\star}_i+\mathsf{H}_i)))\,\mathsf{f}^{n,\star}_{i,j+\mathrm{n}(\mathsf{E}^{n,\star}_i+\mathsf{H}_i)} + \alpha(2(\mathsf{E}^{n,\star}_i+\mathsf{H}_i))\,\mathsf{f}^{n,\star}_{i,j+\mathrm{n}(\mathsf{E}^{n,\star}_i+\mathsf{H}_i)+1}\right]\mathsf{g}^{n,\star\star}_{ij}\\ &= \sum_{ij}\mathsf{f}^{n,\star}_{ij}\left[(1-\alpha(2(\mathsf{E}^{n,\star}_i+\mathsf{H}_i)))\,\mathsf{g}^{n,\star\star}_{i,j-\mathrm{n}(\mathsf{E}^{n,\star}_i+\mathsf{H}_i)} + \alpha(2(\mathsf{E}^{n,\star}_i+\mathsf{H}_i))\,\mathsf{g}^{n,\star\star}_{i,j-\mathrm{n}(\mathsf{E}^{n,\star}_i+\mathsf{H}_i)-1}\right].\end{aligned} \tag{3.20}$$

It is important to note that $\mathsf{E}^{n,\star}$ is also dependent on $\mathsf{f}^{n,\star}$, and therefore, taking the variation of the above formula with respect to $\mathsf{f}^{n,*}$ involves two product rules. The first one composes α' and $\nabla_f E$, while the second one composes n' and $\nabla_f E$. Since

$$\alpha' = \frac{\Delta t}{2\Delta v},\quad \mathrm{n}' = 0\,, \tag{3.21}$$

it amounts to finding $\nabla_f E$. Recall (3.9)-(3.10) in which E depends on f linearly. Therefore, we have

$$\mathsf{E}(\mathsf{f}^{n,\star}+\epsilon\psi) - \mathsf{E}(\mathsf{f}^{n,\star}) = \epsilon\mathsf{E}(\psi)\quad\Rightarrow\quad\langle\nabla_{\mathsf{f}}\mathsf{E},\psi\rangle = \lim_{\epsilon\to0}\frac{\mathsf{E}(\mathsf{f}^{n,\star}+\epsilon\psi) - \mathsf{E}(\mathsf{f}^{n,\star})}{\epsilon} = \mathsf{E}(\psi)\,. \tag{3.22}$$

Combining it to the full formula of (3.13), we have the directional derivative with respect to ψ as:

$$\begin{aligned}&\frac{1}{\Delta x \Delta v}\langle \nabla_{\mathsf{f}} \mathsf{L}|_{\mathsf{f}^{n,\star}}, \psi\rangle = \frac{1}{\Delta x \Delta v}\sum_{ij}\frac{\partial \mathsf{L}}{\partial \mathsf{f}_{ij}^{n,\star}}\psi_{ij}\\ &= \underbrace{-\langle \mathsf{g}^{n,\star}, \psi\rangle}_{\text{term I}}\\ &\underbrace{+\sum_{ij}\psi_{ij}\left(\left(1-\alpha(2(\mathsf{E}_i^{n,\star}+\mathsf{H}_i))\right)\mathsf{g}^{n,\star\star}_{i,j-\mathsf{n}(2\mathsf{E}_i^{n,\star}+2\mathsf{H}_i)} + \alpha(2(\mathsf{E}_i^{n,\star}+\mathsf{H}_i))\,\mathsf{g}^{n,\star\star}_{i,j-\mathsf{n}(2\mathsf{E}_i^{n,\star}+2\mathsf{H}_i)-1}\right)}\\ &\underbrace{+\sum_{ij}\mathsf{f}_{ij}^{n,\star}\frac{\Delta t}{\Delta v}\left[-\mathsf{E}(\psi)_i\,\mathsf{g}^{n,\star\star}_{i,j-\mathsf{n}(2\mathsf{E}_i^{n,\star}+2\mathsf{H}_i)} + \mathsf{E}(\psi)_i\,\mathsf{g}^{n,\star\star}_{i,j-\mathsf{n}(2\mathsf{E}_i^{n,\star}+2\mathsf{H}_i)-1}\right]}_{\text{term II, contribution from }\alpha},\end{aligned} \tag{3.23}$$

where we used (3.21).

Recalling the definition of $\mathsf{E}(\mathsf{f})$ in (3.10), it is straightforward to see that

$$\sum_{ij}\mathsf{E}(\psi)_i\phi_{ij} = -\sum_{ij}\psi_{ij}\mathsf{E}(\phi)_i\,.$$

Then the last term of (3.23) becomes

$$\frac{\Delta t}{\Delta v}\sum_{ij}\mathsf{E}(\phi)_i\psi_{ij}\,,\quad \text{for}\quad \phi=[\phi_{ij}]\quad\text{with}\quad \phi_{ij}=\mathsf{f}_{ij}^{n,\star}\left(\mathsf{g}^{n,\star\star}_{i,j-\mathsf{n}(2\mathsf{E}_i^{n,\star}+2\mathsf{H}_i)} - \mathsf{g}^{n,\star\star}_{i,j-\mathsf{n}(2\mathsf{E}_i^{n,\star}+2\mathsf{H}_i)-1}\right). \tag{3.24}$$

By setting (3.23) to zero, and noting that this holds for any ψ, we obtain the updating formula from $\mathsf{g}^{n,\star\star}$ to $\mathsf{g}^{n,\star}$:

$$\mathsf{g}_{ij}^{n,\star} = \left(1-\alpha(2(\mathsf{E}_i^{n,\star}+\mathsf{H}_i))\right)\mathsf{g}^{n,\star\star}_{i,j-\mathsf{n}(2\mathsf{E}_i^{n,\star}+2\mathsf{H}_i)} + \alpha(2(\mathsf{E}_i^{n,\star}+\mathsf{H}_i))\,\mathsf{g}^{n,\star\star}_{i,j-\mathsf{n}(2\mathsf{E}_i^{n,\star}+2\mathsf{H}_i)-1} + \frac{\Delta t}{\Delta v}\mathsf{E}(\phi)_i\,, \tag{3.25}$$

where ϕ is defined in (3.24) for every time step n.

3.2.4. To compute g^{n-1} from $\mathsf{g}^{n,\star}$

This is achieved by differentiating (3.13) with respect to f^n and set the derivative to be zero. Noticing in (3.14), only Term I and Term II depend on f^n. Setting $\partial_{\mathsf{f}_{ij}^n}\mathsf{L}=0$ gives:

$$\mathsf{g}_{ij}^n = (1-\alpha(v_j))\mathsf{g}^{n,\star}_{i-\mathsf{n}(v_j),j} + \alpha(v_j)\mathsf{g}^{n,\star}_{i-\mathsf{n}(v_j)-1,j}\,. \tag{3.26}$$

In the derivation, we called on the identity (3.18) once again.

The collection of (3.17), (3.19), (3.25) and (3.26) together gives the update procedure to compute g^n, $\mathsf{g}^{n,\star}$ and $\mathsf{g}^{n,\star\star}$. These updates are then combined with (3.15) to calculate the gradient for running the gradient descent algorithm, as summarized in Algorithm 3.

Algorithm 3 GD for simulating (3.12).

```
Given $\mathsf{f}_0$, $\mathsf{f}^{\mathrm{eq}}$, number of iterations $it$, and the initial guess $\mathsf{H}^0$
for $k = 0, 1, \dots, it-1$ do
    Compute $\mathsf{f}[\mathsf{H}^k]$ by running Algorithm 2;
    Compute $\mathsf{g}[\mathsf{H}^k]$: set $\mathsf{g}^N$ according (3.17);
    for $n = N-1, N-2, \cdots, 1$ do
        Update $\mathsf{g}^{n,\star\star}$ from $\mathsf{g}^{n+1}$ using (3.19);
        Update $\mathsf{g}^{n,\star}$ from $\mathsf{g}^{n,\star\star}$ using (3.25);
        Update $\mathsf{g}^n$ from $\mathsf{g}^{n,\star}$ using (3.26);
    end for
    Assemble $\nabla_{\mathsf{H}}\mathsf{J}|_{\mathsf{H}^k}$ using (3.15);
    $H^{k+1} = H^k - h\,\nabla_{\mathsf{H}}\mathsf{J}|_{\mathsf{H}^k}$, where $h$ is determined by line search;
    Evaluate $\mathsf{J}(\mathsf{f}[\mathsf{H}^{k+1}])$ using (3.12a).
end for
```

Remark 3.4. It is easy to see the direct one-to-one correspondence between continuous and discrete settings by comparing Algorithm 1 and Algorithm 3. In particular,

- Evaluation of J and J: Equation (3.12a) is the Riemann sum approximation to (2.2a);
- Fréchet derivative $\frac{\delta J}{\delta H}$ and gradient $\nabla_{\mathsf{H}}\mathsf{J}$: Equation (3.15) is also a Riemann sum approximation to (2.8) in (v,t)-integral and finite differencing in v.

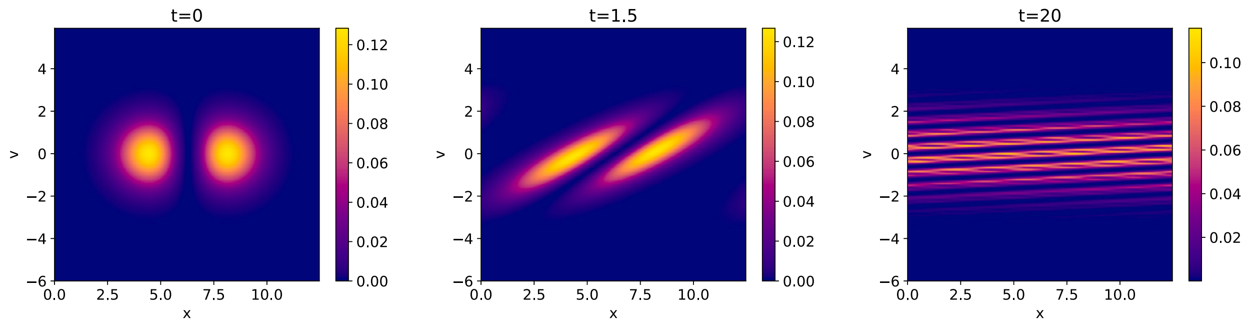

Fig. 2. The distribution function $f(t,x,v)$ is shown at time $t=0$, $t=1.5$, and $t=20$ with the external electric field set to be $H=0$ (i.e., no control imposed). As expected, the beams spread out quickly forming filaments. For this simulation, 128 grid points in both the space and velocity direction and a time step size of $\Delta t=0.5$ has been used. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

- Equations (3.19), (3.25) and (3.26) form the whole semi-Lagrangian scheme for the backward propagation equation shown in (2.6). In particular,
  - Recalling (3.7), Equation (3.19) is equivalent to:

$$\mathsf{g}_{ij}^{n,\star\star}=(1-\alpha(-v_j))\mathsf{g}^{n+1}_{i+\mathsf{n}(-v_j),j}+\alpha(-v_j)\mathsf{g}^{n+1}_{i+\mathsf{n}(-v_j)+1,j}\,,$$

    which is exactly the semi-Lagrangian scheme backward in time for $\partial_t g+v\partial_x g=0$ for $\Delta t/2$ stepsize with flipped velocity $-v_j$, resonating $g^{n,\star\star}=e^{-\frac{\Delta t A}{2}}g$ defined in (3.3).
  - Similarly, Equation (3.26) also represents the semi-Lagrangian scheme backward in time for $\Delta t/2$ for $\partial_t g+v\partial_x g=0$ with velocity $-v_j$.
  - Equation (3.25) is the semi-Lagrangian numerical scheme for

$$\partial_t g-H\partial_v g+[G'*(\rho_f-1)]\partial_v g+G'*\langle g\partial_v f\rangle=0\,,$$

    the acceleration part of (2.6), where $\mathsf{E}^{n,\star}+\mathsf{H}$ serves as the approximation to $G'*(1-\rho_f)+H$ at all grid points, and $\mathsf{E}(\phi)/\Delta v$ numerically presents $G'*\langle g\partial_v f\rangle$ with $\phi/\Delta v\approx -f\partial_v g$, recalling E is a linear operator. To fully see the equivalence, we need to use the fact that $\langle g,\partial_v f\rangle_v=-\langle\partial_v g,f\rangle_v$.

4. Example 1: maintaining focusing beams

This section is dedicated to the first scenario of our study, which is to design an external field H to maintain the focusing feature of the plasma beam, i.e., the solution to the Vlasov–Poisson system (1.1).

To be specific, we start with the following initial value

$$f(0,x,v)=\chi(x)\frac{\exp(-v^2/2)}{2\pi},\qquad \chi(x)=\exp(-a(x-b)^2)\sin(\tfrac{x}{2})^2$$

with (x,v) on the domain $[0,4\pi]\times[-6,6]$ and $a=0.2$, $b=2\pi$. This initial value corresponds to two spatially concentrated beams with the velocity distributed according to the Maxwellian. Without an external electric field (setting $H=0$), this localization in space is lost almost immediately. Particles with higher velocity travel faster, and by deploying the periodic boundary condition, they loop back to the domain, forming a filament-type dynamical pattern (see Fig. 2). Our goal is now to apply an external electric field in such a way as to keep the distribution function as close as possible to the initial value and thus maintain its focusing beam feature. More specifically, we set the final time to be $t=20$ and look for H that minimizes the following objective functional:

$$J(f)=\frac{1}{2}\|f(20,\cdot,\cdot)-f(0,\cdot,\cdot)\|_2^2\,.$$

This $1+1$ dimensional problem considered here is a simplification to the multi-dimensional beam focusing/beam shaping problem at large, whose overarching goal is to apply external electric fields in such a way that the plasma stays confined (i.e., localized) and as close as possible to a prescribed shape in two dimensions as it propagates along the third. In such problems, the time t is a pseudo time that points into the direction where the beam is propagating along; see [22,33,37] for more details.

To discretize the problem, we use 128 grid points in both the space and velocity direction, and the time step size is set to be $\Delta t=0.5$.

We set the admissible set for the external field in the form of Fourier expansion. This is to parameterize H as:

$$H(x)=\sum_{k\in K}a_k\sin\left(\tfrac{1}{2}kx\right),$$

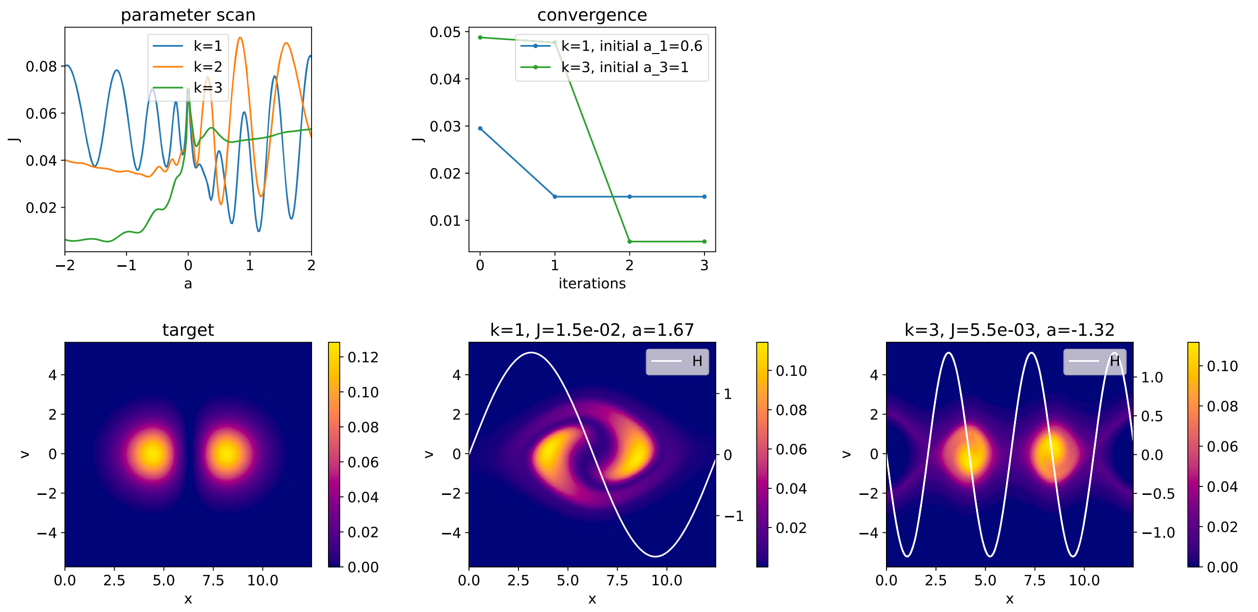

Fig. 3. The top left plot shows the parameter scan for $k = 1$, $k = 2$, and $k = 3$. The convergence of the optimization algorithm for two runs using $k = 1$ and $k = 3$ respectively is shown on the top middle and the corresponding distribution function at time $t = 20$ is shown at the bottom.

where $K \subset \mathbb{N}$ is a subset that might be confined for instance by experimental constraints. The optimization then is translated from finding H as a function to finding its Fourier coefficients $\{a_k\}$. Since the target is even (with respect to the physical space x), it is reasonable to assume that the external electric field (and thus the force) is an odd function. Hence, only sinusoidal functions are included.

Our first goal is to study the optimization landscape of the objective function J with respect to coefficients $\{a_k\}$ in the simplest setting. We first consider a single mode situation and thus only change the values for the coefficient a_1 or a_2 or a_3, respectively. For each value of a_i, we solve the Vlasov–Poisson system and compute the value of the objective functional J. The numerical results are presented in Fig. 3. It is immediately clear that even in this simple configuration (only a single Fourier mode), the landscape of J exhibits several local minima. Further, we observe that the proposed optimization algorithm converges very quickly to the corresponding (local) minima. The optimized solution stays localized in the physical space. However, the shape of the bump is significantly deformed and no longer preserves the original smooth bell shape. This indicates that using only a single Fourier mode to parameterize the external electric field is insufficient.

To potentially improve the objective function, we repeat the parameter scan procedure with a larger set for K: $K = \{1, 2\}$ or $K = \{2, 3\}$, i.e., H is described using two parameters. The results are shown in Fig. 4. The first two subplots are for the parameter scan, and they show the landscape of the objective function. We draw similar conclusions to the previous case, i.e., there are many local minima scattered on the parameter space, and the gradient-based methods should only be able to find local minima. Locally in time, fast convergence of the gradient-based method is observed. Due to the larger parameter space, the obtained objective function indeed has lower values, and the produced solution matches the target much better (see the bottom three plots in Fig. 4; note that a zoomed-in part of the distribution function is shown to make the comparison with the target easier).

Let us also duly note that performing the parameter scan (i.e., generating the pictures on the top of Fig. 4) gives us a basic understanding of the problem and the potential landscape of the objective function. The results are presented here only for illustrative purposes. When the number of parameters is large, the scanning becomes impractical since it requires significant computational resources. In situations like this, employing an efficient optimization algorithm is necessary. We now present the case where $K = \{1, \dots, 10\}$, and the optimization problem is posed on a ten-dimensional space. In Fig. 5, we apply the gradient descent algorithm (Algorithm 3) starting from three different configurations of the parameters. In all three cases, our algorithm converges to a solution in merely a few iterations where the objective function plateaus at a value around 0.005 and the produced local optimal external electric field H roughly preserves the concentrated beam structure. The bottom row of Fig. 5 also presents the mechanism of the beam-shaping: Within the area of each bump, the force directs the plasma towards the center of the bump – positive force for the particles sitting in the left region of the bump, and negative force for the particles sitting in the right region of the bump. This is to counter the natural tendency of the plasma to lose confinement and become less localized in physical space.

In this non-convex optimization, the obtained minimizers are typically local minima, and the quality of the beam-shaping significantly depends on the chosen initial guess. A global optimization strategy is then desirable. We propose deploying the gradient computation in Algorithm 3 to accelerate and improve the off-the-shelf global optimization strategy.

To illustrate this, we employ the genetic algorithm named differential evolution [65] for optimization that is already implemented in the Python package SciPy. The brute-force implementation leads to slow convergence (see the results in Fig. 6). When the gradient computation is integrated into the polishing process, the computational cost is dramatically reduced; see also Fig. 6 for

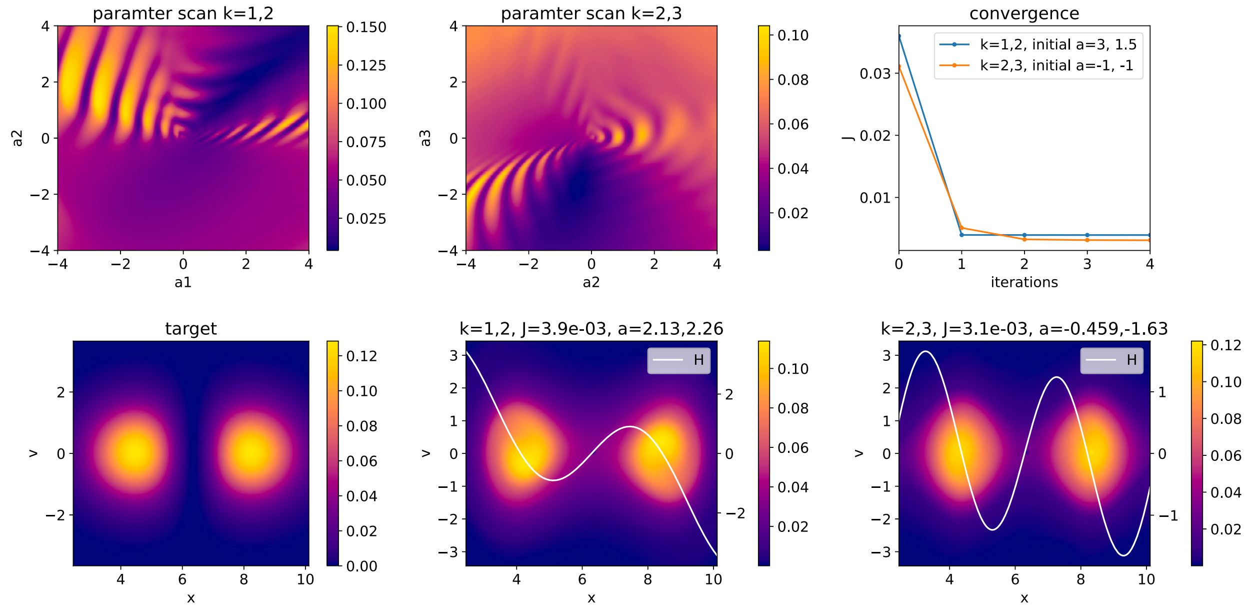

Fig. 4. A parameter scan for $K = \{1,2\}$ and $K = \{2,3\}$ is shown on the top. The convergence of the optimization algorithm for two different initial values is shown on the top right and the corresponding distribution function at time $t = 20$ is shown at the bottom. Note that we have restricted both the x and v axis in the plots of the distribution function in order to make the comparison easier.

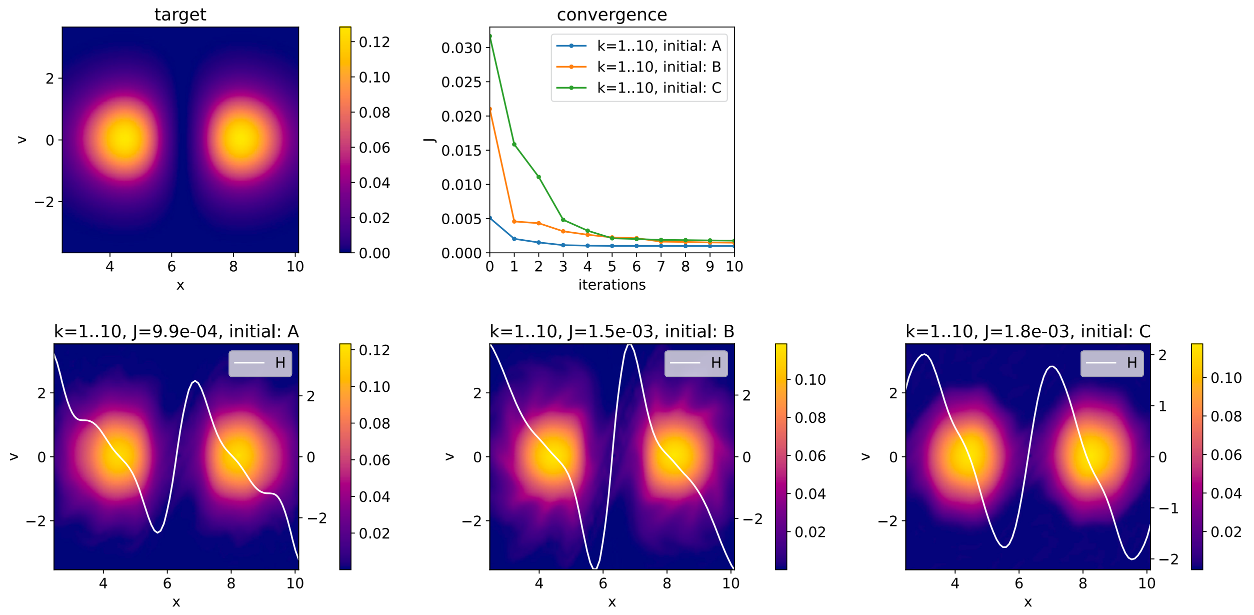

Fig. 5. We show the convergence of the optimization algorithm for three different runs with different initial guesses. The initial guesses for the three runs are listed in Table 1. Note that we have restricted both the x and v axes in the plots of the distribution function in order to make the comparison easier.

the computational cost comparison. In particular, the genetic algorithm generates a large number of candidate solutions, and these candidates are "genetically mutated" (hence the name of the algorithm) in search of better solutions. The integration of the gradient computation is to apply the gradient descent to the best candidate solution (i.e., the solution with the smallest J) as well as $n_p - 1$ other randomly chosen candidates. The total n_p candidate solutions are updated through the gradient descent for *it* steps within each outer iteration. In Fig. 6, we also compared the performance of the algorithm using different choices of n_p and *it*: polishing only a few candidates with a small number for *it* seems to generate rather fast convergence. In particular, to reduce the objective functional J to approximately $1.2 \cdot 10^{-3}$, setting $n_p = 1$ and $it = 3$ calls for 700 forward and adjoint solvers, while the brute-force genetic algorithm incurs 5 times more of the cost.

Table 1
Initial guesses for Algorithm 3 for problem shown in Fig. 5.

init. config.	A	B	C
a_1	−0.69531099	0.94504888	−0.69531099
a_2	−1.7011901	1.14103725	−1.7011901
a_3	−3.70236071	−1.55007537	−3.1878159
a_4	−1.049485	0.8429296	0.97433649
a_5	−0.45695289	−0.08029718	1.82681106
a_6	1.87686503	2.40461788	1.68046644
a_7	1.91960996	0.9644806	2.31895602
a_8	1.69153168	2.25242665	1.69153168
a_9	0.42096132	−0.12171427	1.33032262
a_{10}	−0.40649424	−0.60917031	−0.89049334

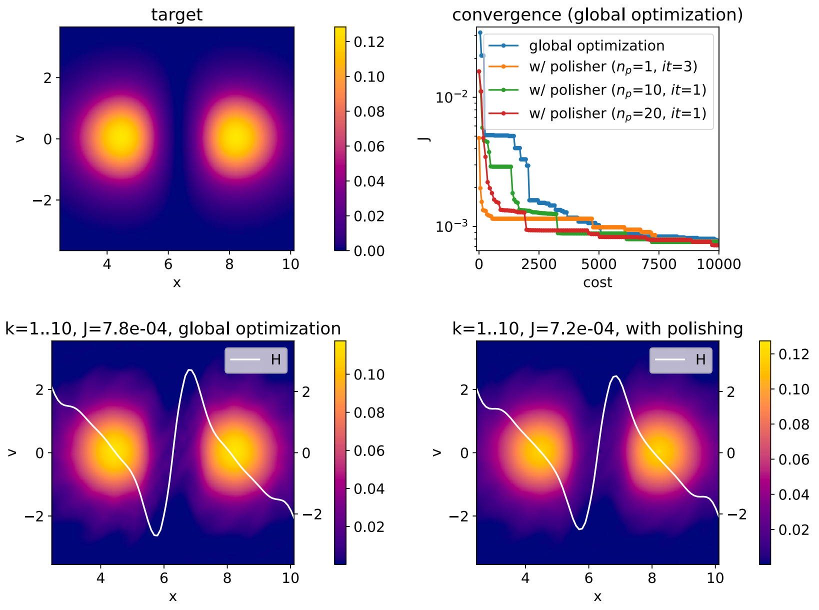

Fig. 6. The plot presents the convergence performance of the genetic optimization algorithm with gradient based polishing for different values of n_p (the number of candidate solutions in each generation that are polished) and it (the number of gradient-descent sub-iterations during the polishing). We use 50 candidate solutions in the genetic algorithm. For the computational cost we made a crude assumption that each forward and backward problem solve has a unit cost, and thus the computational cost per iteration is $50 + 2 \cdot n_p \cdot it$. Including gradient-based optimization as part of a global optimization algorithm significantly reduces the computational cost. On the bottom of the plot, the distribution function for the best solutions, found with and without the gradient-based polishing algorithm, are shown. The optimization results are similar while gradient-polishing achieves the convergence with much less cost. Note that we have restricted both the x and v axes in the plots of the distribution function in order to make the comparison easier.

It is worth noting that in the simulations conducted here, we have kept the n_p and it constant for the entire run, but a more delicate choice of parameter is needed to further improve the computation. Like many other genetic programming algorithms, it is crucial to balance "global exploration" and "local exploitation" to find the global minimum. We expect physical heuristics can be very useful in parameter tuning for this global optimization search, but we do not further explore this direction.

Another interesting aspect of the problem considered in this section is that even though we perform the optimization over a finite time interval. That is, we constrain the solution only at time $t = 20$. The obtained result extrapolates well for beam-shaping in the larger time horizon. This is presented in Fig. 7 where we employ the parameter configuration that achieves $J = 7.2 \cdot 10^{-4}$ from Fig. 6 and runs the simulation up to $t = 80$. It can be seen that the plasma is gradually leaking out from the confined region, but the concentration around the original two physical domains is still clear.

5. Example 2: suppressing a two-stream instability

The two-stream instability is a classic problem in plasma physics and a toy model that presents some shared features of many other plasma instabilities. Two beams propagating in opposite directions form an unstable equilibrium: A perturbation to the equilibrium leads to an exponential growth of the electric field and an entangled beam structure in phase space. Many other unstable equilibria show similar instabilities. For example, the so-called bump-on-tail instability, which models the propagation of a beam into a stationary plasma, can be used to heat the plasma in the context of a fusion reactor. For more details, we refer to [13].

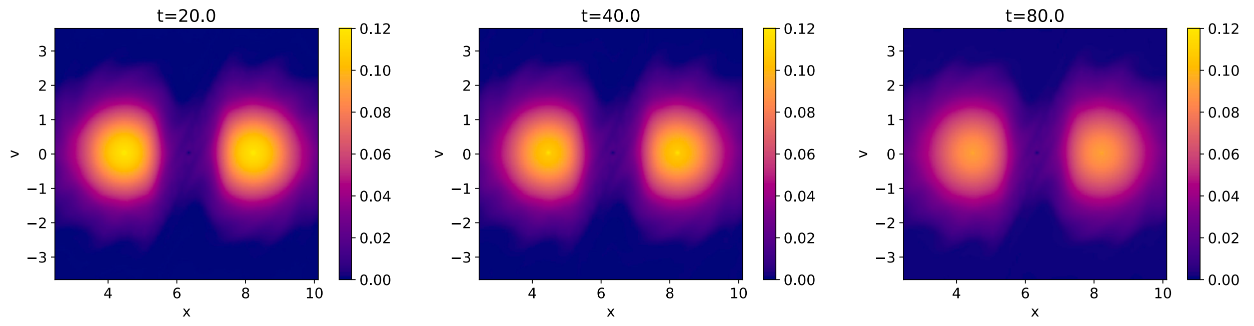

Fig. 7. The parameter found in the genetic algorithm with gradient-polishing that achieves $J = 7.2 \cdot 10^{-4}$ at $t = 20$ is applied as the external field to the VP system, which is then ran to $t = 40$ and $t = 80$. The plasma particles start to leak and show blurring compared to the target distribution, but the beam shape is relatively preserved well. Note that we have restricted both the x and v axes in the plots of the distribution function in order to make the comparison easier.

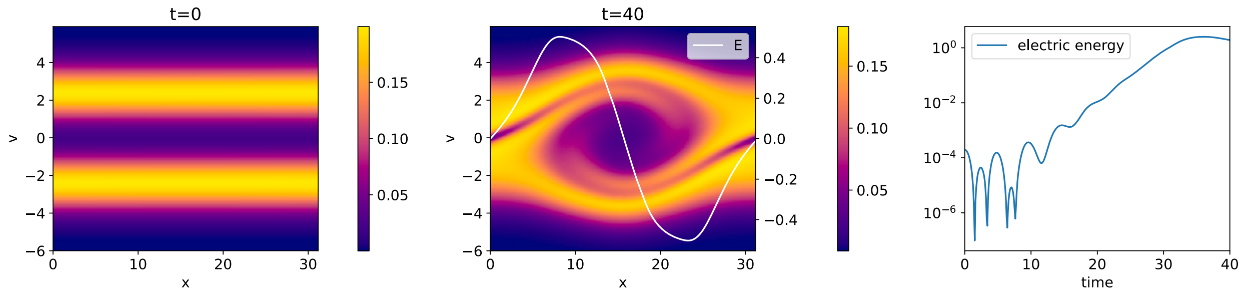

Fig. 8. Time evolution of the two-stream instability when the external field is set to be $H = 0$: A small perturbation at $t = 0$ leads to an exponential increase of the electric energy until nonlinear effects cap it off with a saturation. The amplitude of E is marked on the right side of the plot in the middle panel.

Here, we will consider the 1+1 dimensional two-stream instability given by the following initial value

$$f(0,x,v) = (1+\alpha\cos(\beta x))f^{\mathrm{eq}}(v), \quad (x,v) \in [0, 2\pi/\beta] \times [-6,6]$$

with

$$f^{\mathrm{eq}}(v) = \frac{1}{2\sqrt{2\pi}}\left(\exp\left(-\frac{(v-\overline{v})^2}{2}\right) + \exp\left(-\frac{(v+\overline{v})^2}{2}\right)\right).$$

The parameters are chosen as $\alpha = 10^{-3}$, $\beta = 0.2$, and $\overline{v} = 2.4$. We use 128 grid points in both the space and velocity direction to discretize the problem. The time step size is chosen as $\Delta t = 0.1$. The need for a smaller time step size makes this problem more computationally demanding than the focus problem considered in the previous section.

If no external electric field is applied (i.e., $H = 0$), we observe an exponential increase in the electric energy. This behavior continues until the electric field is large enough such that strongly nonlinear effects take over. This leads to saturation of the electric field and a filamented vortex in phase space (see Fig. 8).

Since plasma instabilities are often undesired and can lead to dangerous disruptions in many applications (such as fusion reactors), our goal here is to add an external electric field in order to suppress (or delay) the instability. Thus, our objective functional is set to

$$J(f) = \frac{1}{2}\|f(40,\cdot_x,\cdot_v) - f^{\mathrm{eq}}(\cdot_v)\|_2^2.$$

As in the previous example, we parametrize the external electric field using Fourier modes, i.e.,

$$H(x) = \sum_{k\in K} a_k \cos\left(\tfrac{1}{2}kx\right),$$

where $K \subset \mathbb{N}$. Note that here we choose only cosine modes due to the form of the initial perturbation.

To build some basic understanding of the landscape of the objective function, we first present a parameter scan for a small number of Fourier modes, as was done in the previous section. For a single Fourier mode, we can reduce the objective functional somewhat, from approx $J \approx 0.92$ for $H = 0$ to $J \approx 0.4$ for the optimized solution. However, we require a relatively large electric field ($a \approx 0.15$) compared to the initial perturbation ($\alpha = 10^{-3}$) to do it (see Fig. 9).

At first sight, for two modes the situation looks similar (see the middle top of Fig. 9). However, when we zoom in to consider the case when the parameters take on small values, an interesting landscape for J appears; see the first two plots of Fig. 10. When the initial guess starts in this region, the objective functional decreases significantly along the gradient descent direction, up to approximately 0.2. This provides a stark contrast to the one-mode situation where the parameter needs to be one magnitude bigger

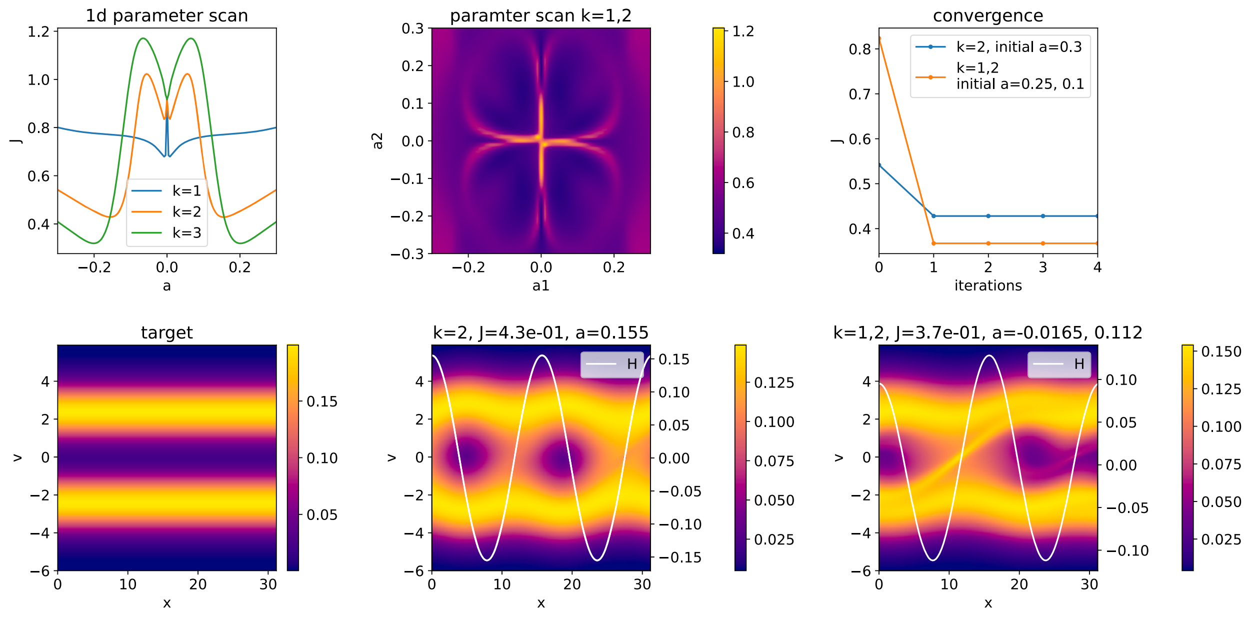

Fig. 9. A one-dimensional parameter scan for $K = \{1\}$, $K = \{2\}$, and $K = \{3\}$ is shown on the top left and a two-dimensional parameter scan for $K = \{1, 2\}$ is shown on the top middle. The convergence of Algorithm 3 for two different initial guesses/two configurations is shown on the top right. The corresponding distribution function at time $t = 40$ is shown along with the target at the bottom.

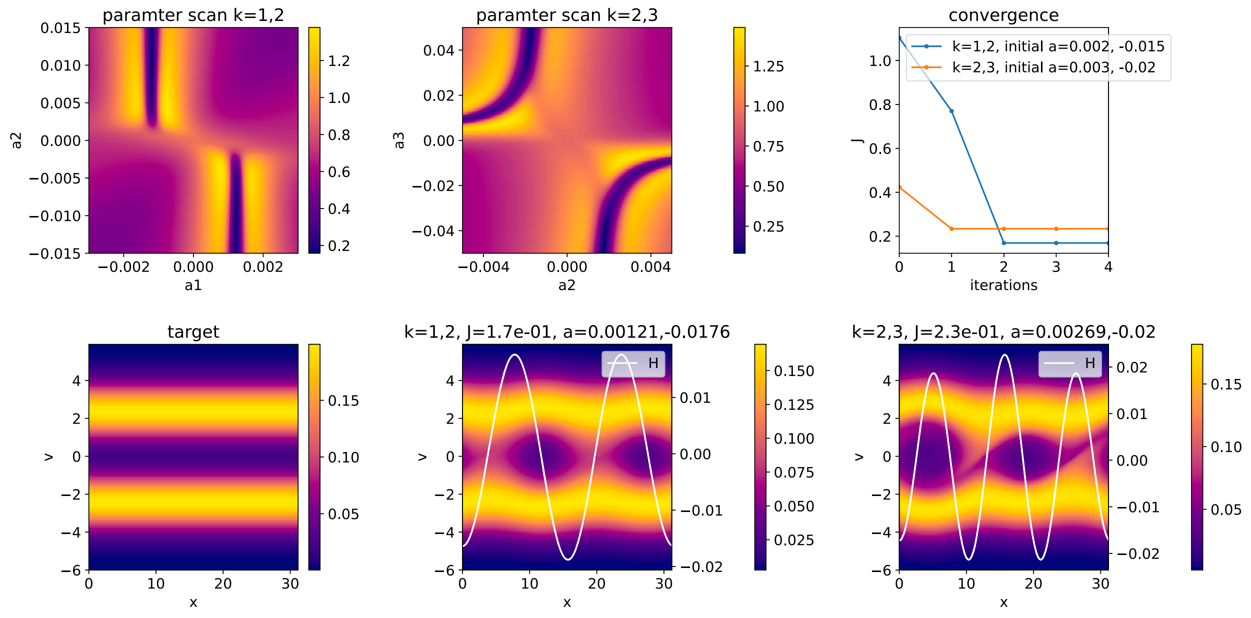

Fig. 10. A parameter scan for $K = \{1, 2\}$ and $K = \{2, 3\}$ is shown on the top. The convergence of Algorithm 3 for two different initial guesses is shown on the top right, and the corresponding distribution function at time $t = 40$ is shown at the bottom.

to achieve even a J value of approximately 0.4. That is, the size of the external electric field required is reduced by approximately an order of magnitude. We also note that, as for the focus beam problem, convergence of the proposed optimization algorithm is rapid, at most two iterations are required for this example. However, as seen in the later three plots of Fig. 10, even in this situation, the instability in the phase space is still only partially suppressed.

Like the focus beam example, we expect increasing the number of tuning Fourier modes would bring better stability. To do so, we use $K = \{1, \ldots, 5\}$ and thus run the optimization in this five-dimensional problem. Once again, we employ a hybrid approach that uses a global genetic optimization algorithm combined with gradient-descent polishing of chosen candidates per iteration. In Fig. 11, we show three genetic algorithm runs using three different initial configurations. In each run, three candidate solutions are deployed. Initial configuration A presents the best candidate that we have found that reduces the objective functional to $J \approx 2.4 \cdot 10^{-3}$, and

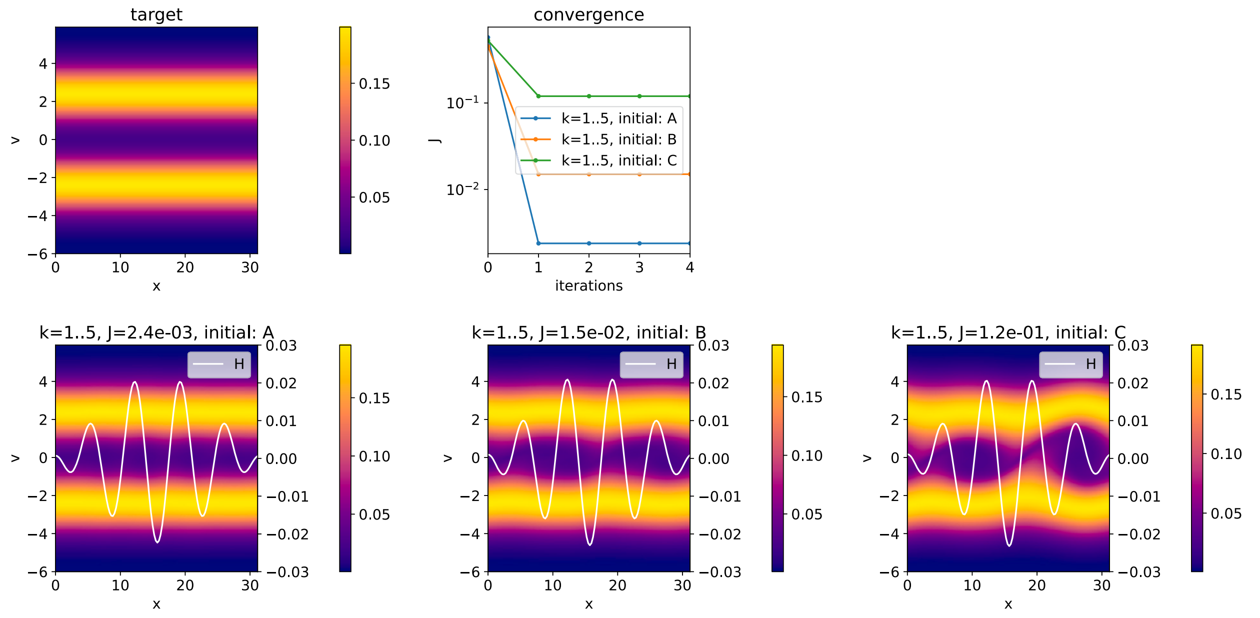

Fig. 11. Algorithm 3 is ran for $K = \{1, \dots, 5\}$ with three different initial guesses. The convergence in iteration is shown on the top right and the obtained distribution functions at $t = 40$ are shown on the bottom. For initial configuration A, B and C are given in Table 2. The best obtained solution has $J \approx 2.4 \cdot 10^{-3}$ generated by $[a_1, \dots, a_5] = [0.00000591, -0.00003512, 0.00134810, -0.01075167, 0.01016702]$.

Table 2
Initial guesses for Algorithm 3 for problem shown in Fig. 11.

init. config.	A	B	C
a_1	−0.00016439	0.00015670	−0.00018648
a_2	−0.00003536	−0.00016387	−0.00043187
a_3	0.00135148	0.00113154	0.00172712
a_4	−0.01075463	−0.01082209	−0.01063006
a_5	0.01016917	0.01086655	0.01045662

correspondingly, the solution on the phase space even up to $t = 40$ is almost indistinguishable from the target, well preserving the equilibrium. We also see that while convergence is rapid, the value of J obtained depends significantly on the initial configuration.

It is worth noting that the optimal external field is rather small but already performs well, suppressing the instability effectively. To better understand the mechanism, we examine the magnitude of Fourier modes of the self-consistent electric field E and plot their evolution in time. As seen in Fig. 12, in the case of $H = 0$, all Fourier modes are unstable, achieving high amplitude at the final time $t = 40$. In particular, $k = 1$ and $k = 2$ take on high values and have exponential growth very quickly. This observation is consistent with the linear theoretical analysis that predicts exponential growth rate at approximately 0.226 for $k = 1$ and 0.15 for $k = 2$ using the parameters for this particular example; see, e.g., [64]. The higher modes initially are stable according to the linear theory, but due to the nonlinear coupling, the linear theory eventually fades off, and the instability sets in. The instability continues growing in time until the electric field is large enough ($E \approx 0.5$) such that strongly nonlinear effects take over and lead to saturation.

We now examine the stability brought to these Fourier modes when the small external electric field is added. We impose the optimal H we found from Run-A in Fig. 11 to the plasma dynamics, and we see that this external field is able to suppress the growth of the unstable modes. Considering that the initial perturbation to the phase space distribution is relatively small, the external field only needs to control this perturbation, providing an intuitive explanation that a small field can already preserve the beam structure. We should note that based on the linear theory, all modes are decoupled. Thus, the linear theory cannot provide a mechanism for explaining the suppression of higher modes using H that contains only lower frequencies. This instability-suppressing effect represented here is a fully nonlinear mechanism that mixes all modes of information, as seen in the bottom plot of Fig. 12.

It should, however, be noted that, in this case, the instability is only suppressed over the time interval for which the optimization is done. This is illustrated in Fig. 13: the optimization solution is achieved by setting the objective function evaluated at $t = 40$, but we use the same configuration to run further into the future time horizon. It can be seen that eventually, the tendency of the most unstable $k = 1$ mode to grow exponentially leads to an onset of the instability at $t = 50$ and saturation around $t = 70$. This strongly indicates that the imposition of an external field can only delay the two-stream instability but not completely eliminate it.

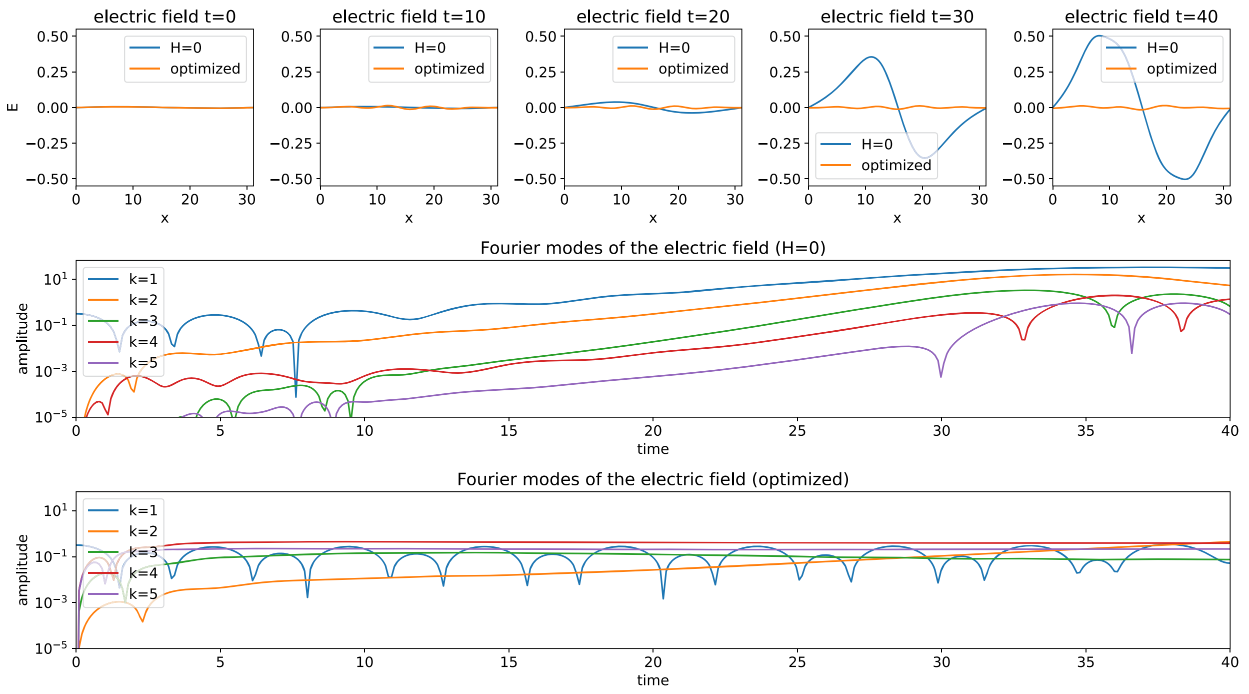

Fig. 12. Snapshots of the electric field E using $H = 0$ and the optimal H from results displayed in Fig. 11 are shown on the top. The plot in the second row shows the first five Fourier modes of E with $H = 0$: They clearly demonstrate the exponential growth that suggests instability. The plot on the bottom shows the evolution of the electric field E when the optimal external field (found by the optimization algorithm) is applied. The external field invokes all Fourier modes to be excited but none present exponential growth. These modes nonlinearly interact, providing overall stability.

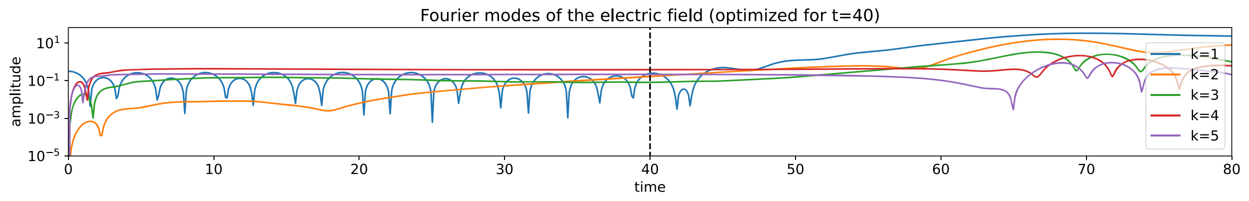

Fig. 13. The first five Fourier modes of the self-generated electric field E are shown as a function of time when the VP system is imposed by the found optimized external field H, found through running optimization up to $t = 40$ (as indicated by the black dashed vertical line). Over this time interval, the mixing of the modes suppresses the instability. As one continues running the VP system, in the long time horizon, the instability of the $k = 1$ mode manifests itself, leading to an exponential growth of the electric energy. The nonlinearity eventually kicks in, capping off the growth at saturation.

6. Conclusion

Fusion energy holds the promise of clean, safe, and virtually limitless energy generation, and to a large extent, many fusion energy engineering problems boil down to plasma control, with prominent examples being the tokamak and stellarator design. The current paper initiates a line of study that looks into the mathematical formulation and optimization strategies for controlling the behavior of plasma on the kinetic level using the Vlasov–Poisson equation as the forward model, with a semi-Lagrangian discretization. Our current formulation has yet to mature to apply to state-of-the-art engineering problems, and advancements are required on multiple fronts. These include exploring different optimization strategies, employing various plasma models, and considering alternative forward solvers. All of these choices have an impact on the final output of the optimization algorithm. Nevertheless, our findings reveal a universal challenge inherent in all formulations: the hyperbolic nature of plasma dynamics, which leads to filamentation in solutions resembling wave-type instabilities in the associated PDE-constrained optimization problem.

This non-convexity in the objective function landscape, reminiscent of the cycle-skipping behavior observed in Helmholtz-type inverse problems [42], highlights the significant challenge at hand. The wave-type inverse problem has garnered a lot of research interest and has triggered the development of many techniques, such as modifications to the full-waveform inversion (FWI), qualitative analysis [11], and landscape reshape [31]. We expect these results to be useful for handling the non-convexity in plasma control. In this work, we tackled the non-convexity challenge by using a combination of global and local optimization algorithms to exploit and explore the optimization landscape, respectively. This opens the door for further investigation along the direction of optimization algorithms employment for a practical tool to mitigate this challenge.

CRediT authorship contribution statement

Lukas Einkemmer: Conceptualization, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Qin Li:** Conceptualization, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Li Wang:** Conceptualization, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Yunan Yang:** Conceptualization, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing.

Declaration of competing interest

There are no competing interests.

Data availability

Data will be made available on request.

Acknowledgement

Q. Li is supported in part by ONR-N00014-21-1-214, NSF-1750488 and Office of the Vice Chancellor for Research and Graduate Education at the University of Wisconsin Madison with funding from the Wisconsin Alumni Research Foundation. She thanks Dr. Antoine Cerfon for the insightful discussion. L. Wang is partially supported by NSF grant DMS-1846854. Y. Yang acknowledges support from Dr. Max Rössler, the Walter Haefner Foundation, and the ETH Zürich Foundation. All four authors have participated in "Frontiers in kinetic theory: connecting microscopic to macroscopic scale" partially supported by Simons Foundation and held at the Isaac Newton Institute (UK) in Spring 2022, where the work was initiated.

References

[1] Giacomo Albi, Lorenzo Pareschi, Mattia Zanella, Boltzmann-type control of opinion consensus through leaders, Philos. Trans. R. Soc. A, Math. Phys. Eng. Sci. 372 (2028) (2014) 20140138.
[2] Mario Annunziato, Alfio Borzì, A Fokker–Planck control framework for multidimensional stochastic processes, J. Comput. Appl. Math. 237 (1) (2013) 487–507.
[3] T.D. Arber, Keith Bennett, C.S. Brady, A. Lawrence-Douglas, M.G. Ramsay, N.J. Sircombe, P. Gillies, R.G. Evans, Holger Schmitz, A.R. Bell, et al., Contemporary particle-in-cell approach to laser-plasma modelling, Plasma Phys. Control. Fusion 57 (11) (2015) 113001.
[4] Guillaume Bal, Inverse transport theory and applications, Inverse Probl. 25 (5) (2009) 053001.
[5] Claude Bardos, Pierre Degond, Global Existence for the Vlasov-Poisson Equation in 3 Space Variables with Small Initial Data, Ann. Inst. Henri Poincaré, Anal. Non Linéaire 2 (1985) 101–118, Elsevier.
[6] Jan Bartsch, Patrik Knopf, Stefania Scheurer, Jörg Weber, Controlling a Vlasov-Poisson plasma by a particle-in-cell method based on a Monte Carlo framework, preprint, arXiv:2304.02083, 2023.
[7] Lorenz T. Biegler, Omar Ghattas, Matthias Heinkenschloss, Bart van Bloemen Waanders, Large-Scale PDE-Constrained Optimization: an Introduction, Springer, 2003.
[8] Martin Burger, Heinz W. Engl, Peter A. Markowich, Paola Pietra, Identification of doping profiles in semiconductor devices, Inverse Probl. 17 (6) (nov 2001) 1765.
[9] Martin Burger, René Pinnau, Fast optimal design of semiconductor devices, SIAM J. Appl. Math. 64 (1) (2003) 108–126.
[10] Russel Caflisch, Denis Silantyev, Yunan Yang, Adjoint DSMC for nonlinear Boltzmann equation constrained optimization, J. Comput. Phys. 439 (2021) 110404.
[11] F. Cakoni, D. Colton, Qualitative Methods in Inverse Scattering Theory: An Introduction, in: Interaction of Mechanics and Mathematics, Springer Berlin Heidelberg, 2005.
[12] F. Casas, N. Crouseilles, E. Faou, M. Mehrenberger, High-order Hamiltonian splitting for the Vlasov–Poisson equations, Numer. Math. 135 (3) (2017) 769–801.
[13] Francis F. Chen, Introduction to Plasma Physics and Controlled Fusion, vol. 1, Springer, 2016.
[14] Ke Chen, Qin Li, Jian-Guo Liu, Online learning in optical tomography: a stochastic approach, Inverse Probl. 34 (7) (may 2018) 075010.
[15] Ke Chen, Qin Li, Li Wang, Stability of inverse transport equation in diffusion scaling and Fokker–Planck limit, SIAM J. Appl. Math. 78 (5) (2018) 2626–2647.
[16] C.Z. Cheng, G. Knorr, The integration of the Vlasov equation in configuration space, J. Comput. Phys. 22 (3) (1976) 330–351.
[17] Yingda Cheng, Irene M. Gamba, Kui Ren, Recovering doping profiles in semiconductor devices with the Boltzmann–Poisson model, J. Comput. Phys. 230 (9) (2011) 3391–3412.
[18] Georges-Henri Cottet, Petros D. Koumoutsakos, et al., Vortex Methods: Theory and Practice, vol. 8, Cambridge University Press, Cambridge, 2000.
[19] Richard Courant, Kurt Friedrichs, Hans Lewy, On the partial difference equations of mathematical physics, IBM J. Res. Dev. 11 (2) (1967) 215–234.
[20] N. Crouseilles, L. Einkemmer, E. Faou, Hamiltonian splitting for the Vlasov–Maxwell equations, J. Comput. Phys. 283 (2015) 224–240.
[21] N. Crouseilles, M. Mehrenberger, F. Vecil, Discontinuous Galerkin Semi-Lagrangian Method for Vlasov-Poisson, ESAIM: Proceedings, vol. 32, EDP Sciences, 2011, pp. 211–230.
[22] Ronald C. Davidson, Qin Hong, Physics of Intense Charged Particle Beams in High Energy Accelerators, World Scientific, 2001.
[23] Ronald C. Davidson, Edward A. Startsev, Self-consistent Vlasov-Maxwell description of the longitudinal dynamics of intense charged particle beams, Phys. Rev. Spec. Top., Accel. Beams 7 (2) (2004) 024401.
[24] Herbert Egger, Matthias Schlottbom, Numerical methods for parameter identification in stationary radiative transfer, Comput. Optim. Appl. 62 (2015) 67–83.
[25] L. Einkemmer, High performance computing aspects of a dimension independent semi-Lagrangian discontinuous Galerkin code, Comput. Phys. Commun. 202 (2016) 326–336.
[26] L. Einkemmer, Semi-Lagrangian Vlasov simulation on GPUs, Comput. Phys. Commun. 254 (2020) 107351.
[27] Lukas Einkemmer, A performance comparison of semi-Lagrangian discontinuous Galerkin and spline based Vlasov solvers in four dimensions, J. Comput. Phys. 376 (2019) 937–951.
[28] Lukas Einkemmer, Alexander Moriggl, A semi-Lagrangian discontinuous Galerkin method for drift-kinetic simulations on massively parallel systems, preprint, arXiv:2212.03036, 2022.

[29] Lukas Einkemmer, Alexander Moriggl, Semi-Lagrangian 4d, 5d, and 6d kinetic plasma simulation on large-scale GPU-equipped supercomputers, Int. J. High Perform. Comput. Appl. (2022) 10943420221137599.
[30] Lukas Einkemmer, Alexander Ostermann, An almost symmetric strang splitting scheme for nonlinear evolution equations, Comput. Math. Appl. 67 (12) (2014) 2144–2157.
[31] Björn Engquist, Yunan Yang, Optimal transport based seismic inversion: beyond cycle skipping, Commun. Pure Appl. Math. 75 (10) (2022) 2201–2244.
[32] F. Filbet, E. Sonnendrücker, Comparison of Eulerian Vlasov solvers, Comput. Phys. Commun. 150 (3) (2003) 247–266.
[33] Francis Filbet, Eric Sonnendrücker, Modeling and numerical simulation of space charge dominated beams in the paraxial approximation, Math. Models Methods Appl. Sci. 16 (05) (2006) 763–791.
[34] Arthur Fleig, Roberto Guglielmi, Optimal control of the Fokker–Planck equation with space-dependent controls, J. Optim. Theory Appl. 174 (2017) 408–427.
[35] Olivier Glass, On the controllability of the Vlasov–Poisson system, J. Differ. Equ. 195 (2) (2003) 332–379.
[36] Olivier Glass, Daniel Han-Kwan, On the controllability of the Vlasov–Poisson system in the presence of external force fields, J. Differ. Equ. 252 (10) (2012) 5453–5491.
[37] Michaël Gutnic, Guillaume Latu, Eric Sonnendrücker, Adaptive two-dimensional Vlasov simulation of heavy ion beams, Nucl. Instrum. Methods Phys. Res., Sect. A, Accel. Spectrom. Detect. Assoc. Equip. 577 (1–2) (2007) 125–128.
[38] William W. Hager, Runge-Kutta methods in optimal control and the transformed adjoint system, Numer. Math. 87 (2000) 247–282.
[39] Michael Hinze, René Pinnau, Michael Ulbrich, Stefan Ulbrich, Optimization with PDE Constraints, vol. 23, Springer Science & Business Media, 2008.
[40] Roger W. Hockney, James W. Eastwood, Computer Simulation Using Particles, CRC Press, 2021.
[41] Sergei Viktorovich Iordanskii, The cauchy problem for the kinetic equation of plasma, Tr. Mat. Inst. Steklova 60 (1961) 181–194.
[42] A. Kirsch, An Introduction to the Mathematical Theory of Inverse Problems, Applied Mathematical Sciences, Springer, New York, 2011.
[43] A.J. Klimas, W.M. Farrell, A splitting algorithm for Vlasov simulation with filamentation filtration, J. Comput. Phys. 110 (1) (1994) 150–163.
[44] A.J. Klimas, A.F. Viñas, Absence of recurrence in Fourier–Fourier transformed Vlasov–Poisson simulations, J. Plasma Phys. 84 (4) (2018).
[45] Patrik Knopf, Optimal control of a Vlasov–Poisson plasma by an external magnetic field, Calc. Var. Partial Differ. Equ. 57 (2018) 1–37.
[46] Patrik Knopf, Confined steady states of a Vlasov-Poisson plasma in an infinitely long cylinder, Math. Methods Appl. Sci. 42 (18) (2019) 6369–6384.
[47] Patrik Knopf, Jörg Weber, Optimal control of a Vlasov-Poisson plasma by fixed magnetic field coils, Appl. Math. Optim. 81 (2020) 961–988.
[48] Lev Davidovich Landau, On the vibrations of the electronic plasma, Usp. Fiz. Nauk 93 (3) (1967) 527–540.
[49] A. Leitao, P.A. Markowich, J.P. Zubelli, On inverse doping profile problems for the stationary voltage–current map, Inverse Probl. 22 (3) (may 2006) 1071.
[50] Frank L. Lewis, Draguna Vrabie, Vassilis L. Syrmos, Optimal Control, John Wiley & Sons, 2012.
[51] Qin Li, Li Wang, Yunan Yang, Monte Carlo gradient in optimization constrained by radiative transport equation, preprint, arXiv:2209.12114, 2022.
[52] Jun Liu, Zhu Wang, Non-commutative discretize-then-optimize algorithms for elliptic PDE-constrained optimal control problems, J. Comput. Appl. Math. 362 (2019) 596–613.
[53] J.H. Malmberg, C.B. Wharton, Collisionless damping of electrostatic plasma waves, Phys. Rev. Lett. 13 (Aug 1964) 184–186.
[54] Clément Mouhot, Cédric Villani, On Landau damping, Acta Math. 207 (1) (2011) 29–201.
[55] Klaus Pfaffelmoser, Global classical solutions of the Vlasov-Poisson system in three dimensions for general initial data, J. Differ. Equ. 95 (2) (1992) 281–303.
[56] J.M. Qiu, C.W. Shu, Positivity preserving semi-Lagrangian discontinuous Galerkin formulation: theoretical analysis and application to the Vlasov–Poisson system, J. Comput. Phys. 230 (23) (2011) 8386–8409.
[57] Kui Ren, Guillaume Bal, Andreas H. Hielscher, Frequency domain optical tomography based on the equation of radiative transfer, SIAM J. Sci. Comput. 28 (4) (2006) 1463–1489.
[58] Lee F. Ricketson, Antoine J. Cerfon, Sparse grid techniques for particle-in-cell schemes, Plasma Phys. Control. Fusion 59 (2) (2016) 024002.
[59] K.V. Roberts, Herbert L. Berk, Nonlinear evolution of a two-stream instability, Phys. Rev. Lett. 19 (6) (1967) 297.
[60] James A. Rossmanith, David C. Seal, A positivity-preserving high-order semi-Lagrangian discontinuous Galerkin scheme for the Vlasov–Poisson equations, J. Comput. Phys. 230 (16) (2011) 6203–6232.
[61] A. Sato, T. Yamada, K. Izui, S. Nishiwaki, S. Takata, A topology optimization method in rarefied gas flow problems using the Boltzmann equation, J. Comput. Phys. 395 (2019) 60–84.
[62] L.S. Solov'ev, V.D. Shafranov, Plasma confinement in closed magnetic systems, Rev. Plasma Phys. 5 (1995) 1–247.
[63] E. Sonnendrücker, J. Roche, P. Bertrand, A. Ghizzo, The semi-Lagrangian method for the numerical resolution of the Vlasov equation, J. Comput. Phys. 149 (2) (1999) 201–220.
[64] Eric Sonnendrücker, Numerical Methods for the Vlasov-Maxwell Equations, 2023, in preparation.
[65] Rainer Storn, Kenneth Price, Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces, J. Glob. Optim. 11 (4) (1997) 341.
[66] Seiji Ukai, Takayoshi Okabe, On Classical Solutions in the Large in Time of Two-Dimensional Vlasov's Equation, 1978.
[67] V.N. Ursov, V.V. Usov, Plasma flow nonstationarity in pulsar magnetospheres and two-stream instability, Astrophys. Space Sci. 140 (1988) 325–336.
[68] J.P. Verboncoeur, Particle simulation of plasmas: review and advances, Plasma Phys. Control. Fusion 47 (5A) (2005) A231.
[69] G.V. Vogman, J.H. Hammer, U. Shumlak, W.A. Farmer, Two-fluid and kinetic transport physics of Kelvin–Helmholtz instabilities in nonuniform low-beta plasmas, Phys. Plasmas 27 (10) (2020) 102109.