SIAM J. SCI. COMPUT.
Vol. 41, No. 5, pp. B1061–B1081

A QUASI-CONSERVATIVE DYNAMICAL LOW-RANK ALGORITHM FOR THE VLASOV EQUATION*

LUKAS EINKEMMER† AND CHRISTIAN LUBICH‡

Abstract. Numerical methods that approximate the solution of the Vlasov–Poisson equation by a low-rank representation have been considered recently. These methods can be extremely effective from a computational point of view, but contrary to most semi-Lagrangian or Eulerian Vlasov solvers, they do not conserve mass and momentum, neither globally nor in respecting the corresponding local conservation laws. This can be a significant limitation for intermediate and long time integration. In this paper we propose a numerical algorithm that overcomes some of these difficulties and demonstrate its utility by presenting numerical simulations.

Key words. low-rank approximation, conservative methods, projector splitting, Vlasov–Poisson equation

AMS subject classifications. 82D10, 15A69, 65N99

DOI. 10.1137/18M1218686

1. Introduction.

Many plasma systems that are of interest in applications (such as magnetic confined fusion or astrophysics) cannot be adequately described by fluid models. Instead kinetic models have to be employed. Since these models are posed in a $2d$-dimensional ($d = 1, 2, 3$) phase space, numerically solving kinetic equations on a grid is extremely expensive from a computational point of view. Thus, traditionally, particle methods have been employed extensively to approximate these types of problems (see, for example, [44]). However, particle methods suffer from excessive noise that, for example, make it difficult to resolve regions with low phase space density.

Due to the improvement in computer performance, methods that directly discretize phase space have recently seen increased interest [43, 18, 7, 42, 39, 40, 6, 16, 2, 11, 3, 12]. Due to their ability to overcome the Courant–Friedrichs–Lewy (CFL) condition, semi-Lagrangian methods, in particular, have been considered extensively in the literature. However, performing these simulations in higher dimensions is still extremely expensive. As a consequence, much effort has been devoted to efficiently implementing these methods on high performance computing systems [41, 1, 11, 25, 32, 10, 4, 13].

More recently, methods that use a low-rank approximation have emerged. In [9, 24] the Vlasov equation is first discretized in time and/or space, and then low-rank algorithms are applied to the discretized system. A different approach is taken in [15], where a low-rank projector-splitting is on top of the procedure; that is, the low-rank algorithm is applied before any time or space discretization is performed. For a chosen rank r, this results in systems of r advection equations in d dimensions (in either the space or the velocity variables, in an alternating fashion) that are then solved by spectral or semi-Lagrangian methods. The advantage of this approach is that the evolution equations that need to be solved numerically are directly posed in terms of

*Submitted to the journal's Computational Methods in Science and Engineering section October 4, 2018; accepted for publication (in revised form) June 24, 2019; published electronically October 8, 2019.

https://doi.org/10.1137/18M1218686

†Department of Mathematics, University of Innsbruck, Innsbruck 6020, Austria (lukas.einkemmer@uibk.ac.at).

‡Mathematisches Institut, Universität Tübingen, Auf der Morgenstelle 10, D-72076 Tübingen, Germany (lubich@na.uni-tuebingen.de).

the degrees of freedom of the low-rank representation. Thus, no intermediate tensors have to be constructed and no tensor truncation algorithms have to be employed. This also leads to increased flexibility in the choice of the time and space discretization methods.

Computing numerical solutions of high-dimensional evolutionary partial differential equations by *dynamical low-rank approximation* has only recently been considered for kinetic problems [15, 14]. However, such algorithms have been investigated extensively in quantum mechanics; see, in particular, [35, 34] for the multiconfiguration time-dependent Hartree (MCTDH) approach to molecular quantum dynamics in the chemical physics literature, and [26, 27] for a computational mathematics point of view of this approach. Some uses of dynamical low-rank approximation in areas outside quantum mechanics are described in [37, 20, 33, 36]. In a general mathematical setting, dynamical low-rank approximation has been studied in [22, 23, 30]. A major algorithmic advance for the time integration was achieved with the projector-splitting methods first proposed in [28] for matrix differential equations and then developed further for various tensor formats in [27, 29, 19, 21, 31]. In contrast to standard time-stepping methods, the projector-splitting methods have been shown to be robust to the typical presence of small singular values in the low-rank approximation [21]. The approach in [15, 14] and in the present paper is based on an adaptation of the projector-splitting method of [28] to kinetic equations.

While low-rank approximations can be very effective from a computational point of view, they destroy much of the physical structure of the problem under consideration. Important physical invariants, such as mass and momentum, are no longer conserved. Perhaps even more problematic is that the low-rank approximation does not take the corresponding local conservation laws into account. This can be a significant issue if these algorithms are to be used for long or even intermediate time integration.

This situation is in stark contrast with the state of the art for semi-Lagrangian or Eulerian Vlasov solvers, where significant research has been conducted to conserve certain physical properties of the exact solution [18, 42, 38, 5, 2, 3, 12]. In particular, methods that conserve mass and momentum are commonly employed. However, to the best of our knowledge, no low-rank algorithms are available that are able to conserve even linear invariants. Furthermore, it has recently been proposed to use low-rank numerical methods to solve fluid problems [14]. Also in this setting, conservation of mass and momentum, a hallmark of traditional fluid solvers, is, of course, of great interest.

In this paper we will consider the Vlasov–Poisson equation (for $x \in \Omega_x \subset \mathbb{R}^d$ and $v \in \Omega_x \subset \mathbb{R}^d$ for $d \leq 3$),

$$\begin{aligned} &\partial_t f(t,x,v) + v \cdot \nabla_x f(t,x,v) - E(f)(x) \cdot \nabla_v f(t,x,v) = 0, \\ &\nabla \cdot E(f)(x) = -\int f(t,x,v)\,\mathrm{d}v + 1, \qquad \nabla \times E(f)(x) = 0, \end{aligned} \tag{1.1}$$

which models the time evolution of a collisionless plasma in the electrostatic regime. This equation has an infinite number of invariants (Casimir invariants). Here we will consider the linear invariants of mass and momentum and the corresponding local conservation laws. In section 2 we will introduce the necessary notation and describe the dynamical low-rank splitting algorithm for the Vlasov equation that was proposed in [15]. We then derive a modification of that numerical method such that a projected version of the continuity and momentum balance equation is satisfied (sec-

tion 3). Subsequently we will discuss the global conservation of mass and momentum in section 4. We will then show (section 5) how this approach can be implemented in the fully discrete setting. Finally, in section 6 we present numerical results for the Vlasov–Poisson equation. In particular, we will demonstrate the efficiency of the proposed algorithms for a two-stream instability, a bump-on-tail instability, and Landau damping. Finally, we conclude in section 7.

2. A low-rank projector-splitting integrator. We remark that in [15] two distinct low-rank algorithms were proposed. The first approach decomposes the kinetic equation into space and velocity, as has been described above. The second approach continues the decomposition in a hierarchical fashion until only one-dimensional equations remain. However, since the former approach can be more generally applied, this has been the main focus of our recent work [14, 17]. In addition, there is some theoretical justification for applying this algorithm to problems with filamentation (see [17]). We will only consider this algorithm in the present paper.

We will start by summarizing the low-rank projector splitting integrator. It should be duly noted that this algorithm neither respects the local conservation laws associated with mass or momentum, nor does it conserve mass or momentum globally (the same is also true for the low-rank algorithms in [9]).

We seek an approximation to the Vlasov–Poisson equation (1.1) in the following form:

$$f(t,x,v) = \sum_{i,j=1}^{r} X_i(t,x) S_{ij}(t) V_j(t,v),$$

with real coefficients $S_{ij}(t) \in \mathbb{R}$, and with functions $X_i(t,x)$ and $V_j(t,v)$ that are orthonormal,

$$\langle X_i, X_k \rangle_x = \delta_{ik} \quad \text{and} \quad \langle V_j, V_l \rangle_v = \delta_{jl},$$

where $\langle \cdot, \cdot \rangle_x$ and $\langle \cdot, \cdot \rangle_v$ are the inner products on $L^2(\Omega_x)$ and $L^2(\Omega_v)$, respectively. The dependence of f on the phase space variables $(x,v) \in \Omega = \Omega_x \times \Omega_v \subset \mathbb{R}^{2d}$ is approximated by the functions $\{X_i : i = 1, \dots, r\}$ and $\{V_j : j = 1, \dots, r\}$, which depend only on the separated variables $x \in \Omega_x$ and $v \in \Omega_v \subset \mathbb{R}^d$, respectively. Such an approach is efficient if the rank r can be chosen much smaller compared to the number of grid points used to discretize X_i and V_j in space.

The dynamics of the Vlasov–Poisson equation is constrained to the corresponding low-rank manifold by replacing (1.1) with an evolution equation

$$\partial_t f = -P(f) \left(v \cdot \nabla_x f - E(f) \cdot \nabla_v f \right),$$

where $P(f)$ is the orthogonal projector onto the manifold. The projector can be written as

$$P(f)g = P_{\overline{V}} g - P_{\overline{V}} P_{\overline{X}} g + P_{\overline{X}} g, \tag{2.1}$$

where $P_{\overline{X}}$ is the orthogonal projector onto the vector space $\overline{X} = \operatorname{span}\{X_i : i = 1, \dots, r\}$, and $P_{\overline{V}}$ is the orthogonal projector onto the vector space $\overline{V} = \operatorname{span}\{V_j : j = 1, \dots, r\}$. Then, as first suggested in [28], the dynamics is split into the three terms of (2.1). In the simplest case, the first-order Lie–Trotter splitting, we solve

$$\partial_t f = -P_{\overline{V}} (v \cdot \nabla_x f - E(f) \cdot \nabla_v f), \tag{2.2}$$

$$\partial_t f = +P_{\overline{V}} P_{\overline{X}} (v \cdot \nabla_x f - E(f) \cdot \nabla_v f), \tag{2.3}$$

$$\partial_t f = -P_{\overline{X}} (v \cdot \nabla_x f - E(f) \cdot \nabla_v f) \tag{2.4}$$

one after the other. Now, let us define

$$K_j(t,x) = \sum_i X_i(t,x) S_{ij}(t), \qquad L_i(t,v) = \sum_j S_{ij}(t) V_j(t,v).$$

The advantage of the splitting scheme then becomes that (2.2) only updates K_j (the V_j stay constant during that step), (2.3) only updates S_{ij} (the X_i and V_j stay constant during that step), and (2.4) only updates L_i (the X_i stay constant during that step). The corresponding evolution equations are derived in [15] and are of the following form:

$$\partial_t K_j(t,x) = -\sum_l c^1_{jl} \cdot \nabla_x K_l(t,x) + \sum_l c^2_{jl} \cdot E(K)(t,x) K_l(t,x), \tag{2.5}$$

$$\partial_t S_{ij}(t) = \sum_{k,l} (c^1_{jl} \cdot d^2_{ik} - c^2_{jl} \cdot d^1_{ik}[E(S(t))]) S_{kl}(t), \tag{2.6}$$

$$\partial_t L_i(t,v) = \sum_k d^1_{ik}[E(L(t,\cdot))] \cdot \nabla_v L_k(t,v) - \sum_k (d^2_{ik} \cdot v) L_k(t,v). \tag{2.7}$$

The coefficients c^1_{jl}, c^2_{jl} and d^1_{ik}, d^2_{ik} are vector-valued but independent of x and v and, with the exception of d^1_{ik}, also constant in time. They are given by integrals over Ω_v and Ω_x, respectively; see [15, section 2] for details.

Assuming that the initial value is represented as $f^0(x,v) = \sum_{i,j} X^0_i(x) S^0_{ij} V^0_j(v)$, the algorithm with time step size τ then proceeds in the following three steps. We describe this algorithm for the time step from t_0 to t_1, but it is understood that the same procedure is then used from t_n to t_{n+1} for arbitrary n.

Step 1. Solve (2.5) with initial value $K_j(0,x) = K^0_j = \sum_i X^0_i(x) S^0_{ij}$. Then perform a QR decomposition of $K^1 = [K_1(\tau,\cdot), \dots, K_r(\tau,\cdot)]$ to obtain X^1_i and $\widehat{S}^1_{ij}$.

Step 2. Solve (2.6) with initial value $S_{ij}(0) = \widehat{S}^1_{ij}$ to obtain $\widetilde{S}^0_{ij} = S_{ij}(\tau)$.

Step 3. Solve (2.7) with initial value $L_i(0,v) = L^0_i = \sum_j \widetilde{S}^0_{ij} V^0_j$. Then perform a QR decomposition of $L^1 = [L_1(\tau,\cdot), \dots, L_r(\tau,\cdot)]$ to obtain V^1_j and S^1_{ij}.

The output of the algorithm is then the low-rank representation

$$f(\tau,x,v) \approx f^1(x,v) = \sum_{i,j} X^1_i S^1_{ij} V^1_i.$$

For a detailed derivation of this algorithm the reader is referred to [15]. We note that the extension to second order Strang splitting is immediate.

3. Local conservation. The Vlasov–Poisson equation (1.1) satisfies the continuity equation

$$\partial_t \rho(t,x) + \nabla \cdot (\rho(t,x) u(t,x)) = 0 \tag{3.1}$$

and the momentum balance equation

$$\partial_t \big(\rho(t,x) u(t,x)\big) + \nabla \cdot (\rho(t,x) u(t,x) \otimes u(t,x)) = -E(t,x)\rho(t,x), \tag{3.2}$$

where

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

From these equations, global conservation of mass and momentum is easily obtained by integrating in x. Without the projection operators, (2.2)–(2.4) would satisfy the

continuity equation (3.1) and the momentum balance equation (3.2). Overall this would ensure that the splitting scheme (without projection operators) respects the local conservation laws for mass and momentum. However, it can easily be seen that the projection operators destroy this property. In addition, as has already been pointed out in [15], global conservation of mass and momentum is lost as well.

A crucial observation that supports the following numerical method is that the conserved quantities only depend on x. While we cannot modify the algorithm such that the conservation laws are satisfied exactly (while keeping V_j constant in step 1 and X_i constant in step 3, and both X_i and V_j constant in step 2), our goal is to derive a numerical method that satisfies the *projected* conservation laws for mass and momentum,

$$P_{\overline{X}}\left(\partial_t \rho + \nabla \cdot (\rho u)\right) = 0, \qquad P_{\overline{X}}(\partial_t \rho + \nabla \cdot (\rho u \otimes u) + E\rho) = 0. \tag{3.3}$$

The idea is to add to (2.2)–(2.4) corrections of the form

$$\sum_{i,j} \lambda_{ij} X_i V_j, \tag{3.4}$$

where the coefficients λ_{ij} are determined such that the projected continuity equation and the projected momentum balance hold true. This results in an overdetermined system for the λ_{ij}, for which we seek the smallest solution in the Euclidean norm.

One might object at this point and argue that such a correction is unnecessarily restrictive. Certainly, one could envisage that for (2.5) and (2.7) an arbitrary function of x and v, respectively, could be used as the correction. Unfortunately, as we will describe in more detail in Remark 3.2, this would introduce, for example, nonzero values in the density function at high velocities. This, clearly unphysical, artifact then pollutes the numerical solution. Thus, the benefit of the ansatz given in (3.4) is that the X_i and V_j, which are already used to represent the numerical solution, are also used for the correction. Since the algorithm adapts the functions X_i and V_j in accordance with the solution, the artifact described above is avoided. This behavior is confirmed by numerical simulation.

In the following, the correction given in (3.4) will be made precise for the three steps of the splitting algorithm.

Step 1. We replace the evolution equation (2.5) by

$$\begin{aligned} \partial_t K_j &= \left\langle V_j^0, F(f) + \sum_{k,l} \lambda_{kl} X_k^0 V_l^0 \right\rangle_v \\ &= \langle V_j^0, F(f)\rangle_v + \sum_k \lambda_{kj} X_k^0 \end{aligned} \tag{3.5}$$

with $F(f) = -v \cdot \nabla_x f + E(f) \cdot \nabla_v f$ for $f(t,x,v) = \sum_l K_l(t,x) V_l^0(v)$, and λ_{kl} is yet to be determined. Note that $\langle V_j^0, F(f)\rangle_v$ equals the right-hand side of (2.5), so that (3.5) differs from (2.5) only by the extra term $\sum_k \lambda_{kj} X_k^0$, which is the inner product of (3.4) with V_j^0. Now, we impose

$$\begin{aligned} 0 &= P_{\overline{X}^0}(\partial_t \rho + \nabla \cdot (\rho u)) \\ &= \sum_i X_i^0 \left[\sum_j \lambda_{ij} \alpha_j + \sum_j \langle X_i^0 V_j^0, F(f)\rangle_{xv} \alpha_j + \langle X_i^0, \nabla \cdot (\rho u)\rangle_x \right], \end{aligned} \tag{3.6}$$

where $\alpha_j = \int V_j^0 \, \mathrm{d}v$, and

(3.7)

$$\begin{aligned} 0 &= P_{\overline{X}^0} \left(\partial_t(\rho u) + \nabla \cdot (\rho u \otimes u) + E\rho \right) \\ &= \sum_i X_i^0 \left[\sum_j \lambda_{ij} \beta_j + \sum_j \langle X_i^0 V_j^0, F(f) \rangle_{xv} \beta_j + \langle X_i^0, \nabla \cdot (\rho u \otimes u) \rangle_x + \langle X_i^0, E(f)\rho \rangle_x \right], \end{aligned}$$

where $\beta_j = \int v V_j^0 \, \mathrm{d}v \in \mathbb{R}^d$. Together, (3.6) and (3.7) yield $(1+d)r$ linear equations for the r^2 unknowns λ_{ij}. (We suppose $r > 1+d$ in what follows.) Since the equations for different i decouple, this allows us to put this into matrix form as follows: with the row vector $\alpha = (\alpha_1, \dots, \alpha_r)$ and with the $d \times r$ matrix $\beta = (\beta_1, \dots, \beta_r)$ we obtain

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} \lambda_{i(\cdot)} = \begin{bmatrix} b_i \\ d_i \end{bmatrix} \tag{3.8}$$

with

$$\begin{aligned} b_i &= -\sum_j \langle X_i^0 V_j^0, F(f) \rangle_{xv} \alpha_j - \langle X_i^0, \nabla \cdot (\rho u) \rangle_x, \\ d_i &= -\sum_j \langle X_i^0 V_j^0, F(f) \rangle_{xv} \beta_j - \langle X_i^0, \nabla \cdot (\rho u \otimes u) \rangle_x - \langle X_i^0, E(f)\rho \rangle_x. \end{aligned}$$

These systems of equations have multiple solutions. In order to minimize the magnitude of the correction that is applied, we seek the solution with the smallest Euclidean norm. This can be done easily and at negligible cost as the matrix is only of size $(1+d) \times r$.

It is still necessary to compute the right-hand side. We have

$$\nabla \cdot (\rho u) = \sum_j \nabla K_j \cdot \beta_j, \qquad \nabla \cdot (\rho u \otimes u) = \sum_j \nabla K_j \cdot \gamma_j,$$

where $\gamma_j = \int (v \otimes v) V_j^0 \, \mathrm{d}v$. Since E and ρ have to be computed in any case, and α, β, γ on modern computer architectures can be computed alongside the coefficients c^1 and c^2 at (almost) no extra cost, only the projections in x are of any concern from a computational point of view. These require the computation of r integrals and consequently $\mathcal{O}\left(rn^d\right)$ arithmetic operations when n quadrature points are used in each coordinate direction.

Step 2. We replace the evolution equation (2.6) by

$$\begin{aligned} \partial_t S_{ij} &= -\left\langle X_i^1 V_j^0, F(f) + \sum_{k,l} \lambda_{kl} X_k^1 V_l^0 \right\rangle_{xv} \\ &= -\langle X_i^1 V_j^0, F(f) \rangle_{xv} - \lambda_{ij} \end{aligned} \tag{3.9}$$

for $f(t,x,v) = \sum_{k,l} X_k^0(x) \, S_{kl}(t) \, V_l^0(v)$, so that $F(f)$ depends only on the S_{kl}, and where λ_{ij} is yet to be determined. Then we impose the constraints

$$\begin{aligned} 0 &= P_{\overline{X}^1} \left(\partial_t \rho - \nabla \cdot (\rho u) \right) \\ &= -\sum_i X_i^1 \left[\sum_j \lambda_{ij} \alpha_j + \sum_j \langle X_i^1 V_j^0, F(f) \rangle_{xv} \alpha_j + \langle X_i^1, \nabla \cdot (\rho u) \rangle_x \right] \end{aligned} \tag{3.10}$$

with $\alpha_j = \int V_j^0 \,\mathrm{d}v$ and

$$0 = P_{\overline{X}^1}\left(\partial_t(\rho u) - \nabla\cdot(\rho u\otimes u) - E\rho\right)$$

(3.11)

$$= -\sum_i X_i^1 \left[\sum_j \lambda_{ij}\beta_j + \sum_j \langle X_i^1 V_j^0, F(f)\rangle_{xv}\beta_j + \langle X_i^1, \nabla\cdot(\rho u\otimes u)\rangle_x + \langle X_i^1, E\rho\rangle_x\right]$$

with $\beta_j = \int v V_j^0 \,\mathrm{d}v$. Equations (3.10) and (3.11) yield $(1+d)r$ linear equations for the r^2 unknowns λ_{ij}. Since the equations for different i decouple, we once again can put this into the form given by (3.8). The only difference lies in the right-hand side, which is computed as follows:

$$b_i = -\sum_j \langle X_i^1 V_j^0, F(f)\rangle_{xv}\alpha_j - \langle X_i^1, \nabla\cdot(\rho u)\rangle_x,$$
$$d_i = -\sum_j \langle X_i^1 V_j^0, F(f)\rangle_{xv}\beta_j - \langle X_i^1, \nabla\cdot(\rho u\otimes u)\rangle_x - \langle X_i^1, E\rho\rangle_x.$$

As before, we seek the solution that minimizes the Euclidean norm of the λ_{ij}. This can be done efficiently as we only have to solve r systems of size $(1+d)\times r$. Computing the right-hand side requires $\langle X_i^1 V_j^0, F(f)\rangle_{xv}$, which needs to be computed to conduct this splitting step in any case. Thus, only the projections in x remain. As noted above, they can be computed in $\mathcal{O}\left(rn^d\right)$ arithmetic operations when n quadrature points are used in each coordinate direction.

Step 3. We replace the evolution equation (2.5) by

$$\begin{aligned}\partial_t L_i &= \left\langle X_i^1, F(f) + \sum_{kl}\lambda_{kl} X_k^1 V_l^0\right\rangle_x \\ &= \langle X_i^1, F(f)\rangle_x + \sum_l \lambda_{il} V_l^0\end{aligned} \tag{3.12}$$

for $f(t,x,v) = \sum_{k,l} X_k^1(x)\, L_k(t,v)$, so that $F(f)$ depends only on the functions L_k, and where λ_{ij} is yet to be determined. Then we impose the constraints

(3.13)

$$0 = P_{\overline{X}^1}\left(\partial_t\rho + \nabla\cdot(\rho u)\right) = \sum_i X_i^1 \left[\sum_l \lambda_{il}\alpha_l + \langle X_i^1, F(f)\rangle_{xv} + \langle X_i^1, \nabla\cdot(\rho u)\rangle_x\right]$$

with $\alpha_l = \int V_l^0 \,\mathrm{d}v$ and

$$0 = P_{\overline{X}^1}\left(\partial_t(\rho u) + \nabla\cdot(\rho u\otimes u) + E\rho\right)$$

$$= \sum_i X_i^1 \left[\sum_l \lambda_{il}\beta_l + \langle X_i^1, vF(f)\rangle_{xv} + \langle X_i^1, \nabla\cdot(\rho u\otimes u)\rangle_x - \langle X_i^1, E\rho\rangle_x\right] \tag{3.14}$$

with $\beta_l = \int v V_l^0 \,\mathrm{d}v$. As before, (3.13) and (3.14) yield $(1+d)r$ linear equations for the r^2 unknowns λ_{ij}. We can once again put this into the form given by (3.8), with the right-hand side

$$b_i = -\langle X_i^1, F(f)\rangle_{xv} - \langle X_i^1, \nabla\cdot(\rho u)\rangle_x,$$
$$d_i = -\langle X_i^1, vF(f)\rangle_{xv} - \langle X_i^1, \nabla\cdot(\rho u\otimes u)\rangle_x + \langle X_i^1, E\rho\rangle_x.$$

As before, this can be done efficiently as the matrix involved is small and the right-hand side can be efficiently computed alongside the coefficients that are needed for the low-rank splitting algorithm.

Note that in the third step we have

$$b_i = \sum_k \langle X_i^1, \nabla X_k^1 \rangle_x \cdot \int v L_k \,\mathrm{d}v - \sum_k \langle X_i^1, E X_k^1 \rangle_x \int \nabla_v L_k \,\mathrm{d}v - \sum_k \langle X_i^1, \nabla X_k^1 \rangle_x \cdot \int v L_k \,\mathrm{d}v = 0,$$

where we have assumed that the L_k go to zero as $|v| \to \infty$. Thus, step 3 already satisfies the continuity equation.

Remark 3.1. At first it looks more natural to use K_j and L_i instead of X_j^0 and V_i^0 in steps 1 and 3. These are the quantities that are updated in that step of the algorithm. The correction would then also reflect the corresponding changes that occur as the subflows are advanced in time. However, note that in actual numerical simulations S can be very ill-conditioned. Now, since $K_j = \sum_i X_i S_{ij}$, the smallest singular value of $K = (K_1, \ldots, K_r)$ is equal to that of S. Specifically, this is a problem for momentum conservation, as many problems start with zero or very small momentum. This then changes over time as the algorithm selects appropriate basis functions which carry a nonzero momentum. However, since initially the contribution of these functions to K (contrary to X) is very small, the coefficients in the correction have to become large. This implies that the correction overall becomes quite large. Choosing X_j^0 instead of K_j, as we have done here, solves this issue. The situation is analogous for V_i^0 and L_i.

Remark 3.2. Let us now consider a correction $R_i(v)$ with an arbitrary L^2 function for (2.5),

$$\partial_t L_i = \langle X_i^1, F(f) \rangle_x + R_i(v).$$

This correction is more general than the ansatz we made in (3.4), which only allows linear combination of the V_j^0, which are already used to represent the numerical solution and thus decay to zero. In fact, any property of the V_j^0 that is invariant upon taking linear combinations is preserved by our approach.

As we illustrate now, taking instead a general correction $R_i(v)$ of smallest L^2 norm such that the local conservation laws are satisfied leads to undesirable behavior. To abide by the continuity equation the correction has to satisfy

$$\int R_i(v) \,\mathrm{d}v = -\langle X_i^1, F(f) \rangle_{xv} - \langle X_i^1, \nabla \cdot (\rho u) \rangle_x.$$

Minimizing the correction in the L^2 norm immediately yields

$$R_i = \frac{-1}{|\Omega_v|} \left[\langle X_i^1, F(f) \rangle_{xv} + \langle X_i^1, \nabla \cdot (\rho u) \rangle_x \right],$$

where $|\Omega_v|$ is the volume of the domain in the v-direction. Note, in particular, that R_i is independent of v. Thus, the L^2-optimal correction equally distributes the defect in velocity space and thus introduces nonzero densities for large velocities, which is an unphysical artifact that does not arise with the correction (3.4) considered in this paper.

4. Global conservation. The algorithm developed above satisfies a projected version of the local conservation law. For the local mass conservation law this is stated as

$$P_{\overline{X}}\left(\partial_t \rho + \nabla \cdot (\rho u)\right) = 0.$$

However, contrary to the continuous formulation, conservation of the total mass $\int \rho \, \mathrm{d}x$ cannot be deduced from this expression by simply integrating in x, since the projection $P_{\overline{X}}$ does not commute with integration. In fact, conservation of mass, in general, is violated for the scheme described in the previous section. The situation for momentum is similar.

Since we have an underdetermined system of equations it is possible, in principle, to add an equation that enforces global conservation of mass and momentum. This has to be done for each step in the splitting algorithm.

Step 1. We impose

$$0 = \partial_t \int \rho \, \mathrm{d}x = \sum_{ij} \kappa_i \lambda_{ij} \alpha_j + \sum_j \langle V_j^0, F(f) \rangle_{xv}, \tag{4.1}$$

where $\alpha_j = \int V_j^0 \, \mathrm{d}v$ and $\kappa_i = \int X_i^0 \, \mathrm{d}x$, and

$$0 = \partial_t \int \rho u \, \mathrm{d}x = \sum_{ij} \kappa_i \lambda_{ij} \beta_j + \sum_j \langle V_j^0, F(f) \rangle_{xv} \beta_j, \tag{4.2}$$

where $\beta_j = \int v V_j^0 \, \mathrm{d}v$. This adds $1+d$ linear equations to the $(1+d)r$ linear equations (3.8) required for the local conservation laws. Note that in contrast to those equations, here all the λ_{ij} are coupled to one another. Thus, we have to solve a single system of size $(1+d)(r+1) \times r^2$. We will discuss the computational ramifications later in this section.

Step 2. We impose

$$0 = -\partial_t \int \rho \, \mathrm{d}x = \sum_{ij} \kappa_i \lambda_{ij} \alpha_j + \sum_{ij} \kappa_i \langle X_i^1 V_j^0, F(f) \rangle_{xv} \alpha_j,$$

where $\alpha_j = \int V_j^0 \, \mathrm{d}v$ and $\kappa_i = \int X_i^1 \, \mathrm{d}x$, and

$$0 = -\partial_t \int \rho u \, \mathrm{d}x = \sum_{ij} \gamma_i \lambda_{ij} \beta_j + \sum_{ij} \kappa_i \langle X_i^1 V_j^0, F(f) \rangle_{xv} \beta_j,$$

where $\beta_j = \int v V_j^0 \, \mathrm{d}v$.

Step 3. We impose

$$0 = \partial_t \int \rho \, \mathrm{d}x = \sum_{ij} \kappa_i \lambda_{ij} \alpha_j + \sum_i \kappa_i \langle X_i^1, F(f) \rangle_{xv},$$

where $\alpha_j = \int V_j^0 \, \mathrm{d}v$ with $\kappa_i = \int X_i^1 \, \mathrm{d}x$, and

$$0 = \partial_t \int \rho u \, \mathrm{d}x = \sum_{ij} \kappa_i \lambda_{ij} \beta_j + \sum_i \kappa_i \langle v X_i^1, F(f) \rangle_{xv},$$

where $\beta_j = \int v V_j^0 \, \mathrm{d}v$.

The problem with this approach is that there is no guarantee that the resulting linear system even has a solution. This is most easily demonstrated by considering step 3 in our algorithm. In this case $b_i = 0$ (see section 3). Now, let us consider the rank 2 function on the domain $[0, 2\pi] \times \mathbb{R}$ given by

$$X_1^1(x) = \frac{2}{\sqrt{3\pi}} \cos^2 x, \quad X_2^1(x) = \frac{1}{\sqrt{\pi}} \sin(2x), \quad V_1^0(v) = \frac{\mathrm{e}^{-v^2}}{(\pi/2)^{1/4}}, \quad V_2^0(v) = \frac{2v\mathrm{e}^{-v^2}}{(\pi/2)^{1/4}}.$$

This gives $\alpha = (\sqrt[4]{2\pi}, 0)$ and $\kappa = (2\sqrt{\pi/3}, 0)$. Thus,

$$\lambda_{11} = 0.$$

Since $\beta_1 = 0$, we have

$$\begin{aligned} \langle X_1^1, F(f) \rangle_{xv} &= \langle X_1^1, \nabla X_2^1 \rangle \cdot \beta_2 \\ &\propto \int \cos^2(x) \cos(2x) \, \mathrm{d}x \\ &\neq 0. \end{aligned}$$

This is in contradiction to the condition of global mass conservation. Thus, it is not possible to both satisfy the continuity equation and obtain global conservation of mass. Let us duly note that this is not just a theoretical exercise. We do observe the corresponding behavior in the numerical simulations conducted. We have only considered conservation of mass here, but the same behavior is observed for the momentum as well. We now have the following options.

Local: We enforce only the local conservation laws, while minimizing the Euclidean norm of the correction.

Global: We enforce only the global conservation laws, while minimizing the Euclidean norm of the correction.

Combined: We try to find the best approximation to both the local conservation laws and the global conservation of mass and momentum. This results in a linear least squares problem for the correction. The different equations can be weighted to focus on either the local conservation laws or the global conservation of mass and momentum. This is done as follows: instead of (3.8) we consider

$$w \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \lambda_{i(\cdot)} = w \begin{bmatrix} b_i \\ d_i \end{bmatrix}. \tag{4.3}$$

The introduction of the weight $w \in \mathbb{R}_{\geq 0}$ does not change the solution of the linear system. However, in the combined approach there is no guarantee that such a solution exists; see the description above. In the latter case, we then solve the incompatible equations (4.1)–(4.3), in a least squares sense, by minimizing the Euclidean norm of the (now weighted) residuals. The problem is then to find λ_{ij} such that

$$w^2 \sum_i \left\| \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \lambda_{i(\cdot)} - \begin{bmatrix} b_i \\ d_i \end{bmatrix} \right\|_2^2 + \left(\sum_{ij} \kappa_i \lambda_{ij} \alpha_j + \sum_j \langle V_j^0, F(f) \rangle_{xv} \right)^2 \tag{4.4}$$

$$+ \left\| \sum_{ij} \kappa_i \lambda_{ij} \beta_j + \sum_j \langle V_j^0, F(f) \rangle_{xv} \beta_j \right\|_2^2 \to \min. \tag{4.5}$$

Thus, the weight w has the effect of determining the relative importance of the error committed with respect to the local conservation law, the first term in (4.4), compared

to the global conservation, the second and third term in (4.4). For $w \to \infty$ only the local conservation law is enforced, and for $w = 0$ only the global conservation is enforced.

All of these configurations will be considered in section 6. However, before proceeding let us discuss the computational cost of the combined approach. We have to compute an underdetermined (but incompatible) linear least squares problem with r^2 unknowns λ_{ij} and $(1+d)(r+1)$ data. This problem can be solved by computing the Moore–Penrose pseudo-inverse which requires a QR decomposition of A^T. Thus, it requires at most $\mathcal{O}\left(r^4\right)$ arithmetic operations, which is typically less expensive compared to the cost of the low-rank algorithm itself or even to adding the correction terms for a known λ_{ij} in (3.5) and (3.12).

5. Conservative low-rank algorithm. In the proposed algorithm correction terms are added to the three evolution equations. This implies that our correction is a continuous function of time for the respective subflows. However, in order to increase performance it is often of interest to use a specifically tailored numerical method for solving these subflows. For example, methods based on fast Fourier techniques (FFT) and semi-Lagrangian schemes have been proposed in [15]. To employ these algorithms while still maintaining the conservation laws for mass and momentum is not necessarily straightforward. Thus, we will now introduce a procedure that allows us to apply our correction independent of the specific time integration strategy that is chosen for solving the evolution equations (3.5), (3.9), and (3.12). The approach outlined here is similar to the projection schemes described in [8].

We start with the evolution equation for K_j, which is given as follows:

$$\partial_t K_j = \langle V_j^0, F(f)\rangle_v + \sum_k \lambda_{kj} X_k^0 \qquad \text{for} \quad f = \sum_l K_l V_l^0.$$

Now, we split this equation into

$$\partial_t K_j = \langle V_j^0, F(f)\rangle_v \tag{5.1}$$

and

$$\partial_t K_j = \sum_k \lambda_{kj} X_k^0. \tag{5.2}$$

Equation (5.1) is identical to what has to be solved in the case of the original low-rank algorithm described in section 3 (i.e., the algorithm without correction). Thus, starting from an appropriate initial value K_j^0 we compute an approximation at time τ, where τ is the time step size. This value is henceforth denoted by $K_j^\star$. Now, instead of solving (5.2) we consider the following approximation:

$$\frac{K_j^1 - K_j^\star}{\tau} = \sum_k \lambda_{kj} X_k^0.$$

It remains to derive the conditions on λ_{kj} under which the (discretized) conservation laws are satisfied. We have

$$\begin{aligned}\rho^1 - \rho^0 + \tau\nabla\cdot(\rho^0 u^0) &= \sum_j (K_j^1 - K_j^0)\alpha_j + \tau\sum_j (\nabla K_j^0)\cdot\beta_j \\ &= \sum_j \left(K_j^\star - K_j^0 + \tau\sum_k \lambda_{kj} X_k^0\right)\alpha_j + \tau\sum_j (\nabla K_j^0)\cdot\beta_j,\end{aligned}$$

where $\alpha_j = \int V_j^0 \,\mathrm{dv}$ and $\beta_j = \int v V_j^0 \,\mathrm{d}v$. Now, we apply the projection onto $\overline{X}^0$ to obtain

$$\begin{aligned}0 &= P_{\overline{X}^0}(\rho^1 - \rho^0 + \tau\nabla\cdot(\rho^0 u^0)) \\ &= \sum_k X_k^0 \left[\tau \sum_k \lambda_{kj}\alpha_j + \sum_j \langle X_k^0, K_j^\star - K_j^0\rangle_x \alpha_j + \tau \sum_j \langle X_k^0, \nabla K_j^0\rangle_x \cdot \beta_j\right].\end{aligned}$$

This is the analogue to (3.6).

For the momentum balance equation we have

$$\begin{aligned}&\rho^1 u^1 - \rho^0 u^0 + \tau\nabla\cdot(\rho^0 u^0 \otimes u^0) + \tau E^0 \rho^0 \\ &\qquad = \sum_j (K_j^1 - K_j^0)\beta_j + \tau \sum_j (\nabla K_j^0)\cdot\gamma_j + \tau E^0\rho^0 \\ &\qquad = \sum_j (K_j^\star - K_j^0 + \tau\sum_k \lambda_{kj} X_k^0)\beta_j + \tau\sum_j (\nabla K_j^0)\cdot\gamma_j + \tau E^0\rho^0,\end{aligned}$$

where $\gamma_j = \int (v\otimes v) V_j^0 \,\mathrm{d}v$, and we have used E^0 to denote the electric field at the beginning of the time step. Applying the projection onto $\overline{X}^0$ we obtain

$$\begin{aligned}0 &= P_{\overline{X}^0}(\rho^1 u^1 - \rho^0 u^0 + \tau\nabla\cdot(\rho^0 u^0\otimes u^0) + \tau E^0\rho^0) \\ &= \sum_k X_k^0 \Bigg[\tau\sum_j \lambda_{kj}\beta_j + \sum_j \langle X_k^0, K_j^\star - K_j^0\rangle_x \beta_j \\ &\qquad + \tau\sum_j \langle X_k^0, \nabla K_j^0\rangle_x\cdot\gamma_j + \tau\langle X_k^0, E^0\rho^0\rangle_x\Bigg],\end{aligned}$$

which is the analogue to (3.7).

In fact, these equations are precisely in the form of (3.8). Only the right-hand side

$$\begin{aligned}b_i &= -\sum_j \left\langle X_k^0, \frac{K_j^\star - K_j^0}{\tau}\right\rangle_x \alpha_j - \sum_j \langle X_k^0, \nabla K_j^0\rangle_x\cdot\beta_j, \\ d_i &= -\sum_j \left\langle X_k^0, \frac{K_j^\star - K_j^0}{\tau}\right\rangle_x \beta_j - \sum_j \langle X_k^0, \nabla K_j^0\rangle_x\cdot\gamma_j - \sum_j \langle X_k^0, E^0 K_j^0\rangle_x \alpha_j\end{aligned}$$

is modified. Thus, there is no additional difficulty in implementing this approach.

A similar procedure can be applied to steps 2 and 3 of the splitting algorithm. For step 2 we obtain (3.8) with

$$\begin{aligned}b_i &= \sum_j \frac{S_{ij}^\star - S_{ij}^1}{\tau}\alpha_j - \sum_j \langle X_i^1, \nabla X_j^1\rangle_x S_{ij}^1 \beta_j, \\ d_i &= \sum_j \frac{S_{ij}^\star - S_{ij}^1}{\tau}\beta_j - \sum_j \langle X_j^1, \nabla X_i^1\rangle_x S_{ij}^1\gamma_j - \sum_j \langle E^0, X_j^1\rangle_x S_{ij}^1\alpha_j,\end{aligned}$$

and for step 3 we obtain (3.8) with (denoting the integral $\langle g \rangle_v = \int g \, \mathrm{d}v$)

$$b_i = -\left\langle \frac{L_i^\star - L_i^0}{\tau} \right\rangle_v - \sum_j \langle X_i^1, \nabla X_j^1 \rangle_x \cdot \langle v L_j^0 \rangle_v,$$

$$d_i = -\left\langle \frac{L_i^\star - L_i^0}{\tau} \right\rangle_v - \sum_j \langle X_i^1, \nabla X_j^1 \rangle_x \cdot \langle (v \otimes v) L_j^0 \rangle_v - \sum_j \langle X_i^1, E X_j^1 \rangle_x \langle L_j^0 \rangle_v.$$

Thus, we are able to apply the procedure introduced in section 3, independent of the specific numerical discretization. This has the added benefit that the correction and the associated coefficients only need to be computed once for each step of the splitting algorithm. The only downside here is that we have traded the continuous version of the conservation laws for a discretized version.

6. Numerical results. Although the algorithm described in section 5 is independent of the time and space discretization that is used to solve the evolution equations for K and L, respectively, for the numerical experiments in this section we have to specify an appropriate scheme. Since our main goal is to investigate the error in the conserved quantities due to the dynamic low-rank algorithm, we follow [15] and employ a spectral space discretization based on FFT. This has the advantage that no error in the conserved quantities is introduced by the space discretization. In addition, we employ sufficiently small time steps for the time integrator in order to render the resulting error negligible. Thus, only the error due to the projector splitting and the error in the low-rank representation remain. We should, however, draw attention to the fact that the proposed numerical scheme is also able to correct errors that originate from the space or time discretization of the evolution equations.

In the following we will present numerical results for a range of test problems that are commonly considered in the literature.

6.1. Two-stream instability. In this section we will present numerical results for a two-stream instability. Specifically, we consider the domain $[0, 10\pi] \times [-9, 9]$ and impose the initial value

$$f_0(x, v) = \frac{1}{2\sqrt{2\pi}} \left(\mathrm{e}^{-(v-v_0)^2/2} + \mathrm{e}^{-(v+v_0)^2/2} \right) (1 + \alpha \cos(kx)),$$

where $\alpha = 10^{-3}$, $k = \frac{1}{5}$, and $v_0 = 2.4$. All simulations are conducted using 128 grid points in both the x- and v-directions. In both directions periodic boundary conditions are employed. In all simulations the second order Strang splitting scheme with time step size $\tau = 0.025$ is used.

This setup models two beams propagating in opposite directions and is an unstable equilibrium. Small perturbations in the initial particle-density function eventually force the electric energy to increase exponentially. This is called the linear regime. At some later time saturation sets in (the nonlinear regime). This phase is characterized by nearly constant electric energy and significant filamentation of the phase space. This test problem has been considered in [9, 24, 15] in the context of low-rank approximations. It has been established there that low-rank approximations of relatively small rank are sufficient in order to resolve the linear regime. However, once saturation sets in, the reference solution (computed using a full grid simulation) shows only small oscillations in the electric field. For the low-rank approximation, however, oscillations with significant amplitude can be observed. Since filamentation makes it very difficult to efficiently resolve the small structures in this regime (the L^∞ error

will be large for any numerical method), we consider it a good test example for the conservative method developed in this work.

In Figure 6.1 numerical simulations of the two-stream instability for rank $r = 10$ are shown for the algorithm without correction (labeled low-rank), the correction that exactly satisfies the local projected continuity equations described in section 3 (labeled local), the algorithm of section 4 that combines both local and global corrections (labeled combined), and the algorithm that conserves mass and momentum exactly but does not satisfy the local continuity equations (labeled global). In addition, the full grid simulation is shown (labeled full grid). We observe that all methods show excellent agreement in the linear regime. In the nonlinear regime the local correction shows the best performance (the least amount of oscillations) out of all the low-rank methods. The performance of the combined approach is also significantly better compared to the uncorrected algorithm and the global correction. The uncorrected algorithm clearly performs worst.

Figure 6.1 also shows the error in mass, momentum, energy, and the L^2 norm. We see that although the local correction results in a significant improvement with respect to the qualitative behavior of the electric field, the errors in mass and momentum are still comparable to the uncorrected algorithm. As has been discussed in section 4, in general, satisfying both the local continuity equations and the global invariants is not possible. We clearly see this in the numerical simulation. Nevertheless, the combined approach results in a significant reduction in the error in mass and momentum (by approximately two orders of magnitude).

Now, we increase the rank to $r = 15$ and consider a longer time interval (up to $t = 300$). The numerical results are shown in Figure 6.2. It can be observed very clearly that the uncorrected algorithm as well as the global correction result in qualitatively wrong results (the electric energy decreases by more than two orders of magnitude). On the other hand, the local correction and the combined approach keep the electric energy stable until the final time of the simulation. With respect to the conservation of the invariants the same conclusion as above can be drawn.

As has been mentioned in section 4, the combined approach can be adjusted to be closer to either the local correction or the global correction. The results in Figure 6.3 show how we can trade off the error in mass and momentum and the error in the local conservation laws. We clearly see that the solution deteriorates as the error in the conservation laws increases.

6.2. Bump-on-tail instability. In this section we will present numerical results for a bump-on-tail instability. Specifically, we consider the domain $(0, 20\pi) \times (-9, 9)$ and impose the initial value

$$f(0,x,v) = \frac{1}{\sqrt{2\pi}} \left(\alpha \mathrm{e}^{-v_1^2/2} + \beta \mathrm{e}^{-2(v_1-4.5)^2} \left(1 + \gamma \cos(k x_1)\right) \right),$$

where we have chosen $\alpha = 9/10$, $\beta = 2/10$, $\gamma = 0.03$, and $k = 0.3$. All simulations are conducted using 128 grid points in both the x- and v-directions. In both directions periodic boundary conditions are employed. In all simulations the second order Strang splitting scheme with time step size $\tau = 0.025$ is used.

For the bump-on-tail instability we initially see an increase in the electric energy, followed by saturation, and later on a fully nonlinear regime, where the electric energy slowly oscillates in time. The uncorrected low-rank algorithm captures the behavior up to and including saturation very well, but shows a significant deviation from the true solution in the fully nonlinear regime. Even at the beginning of saturation the

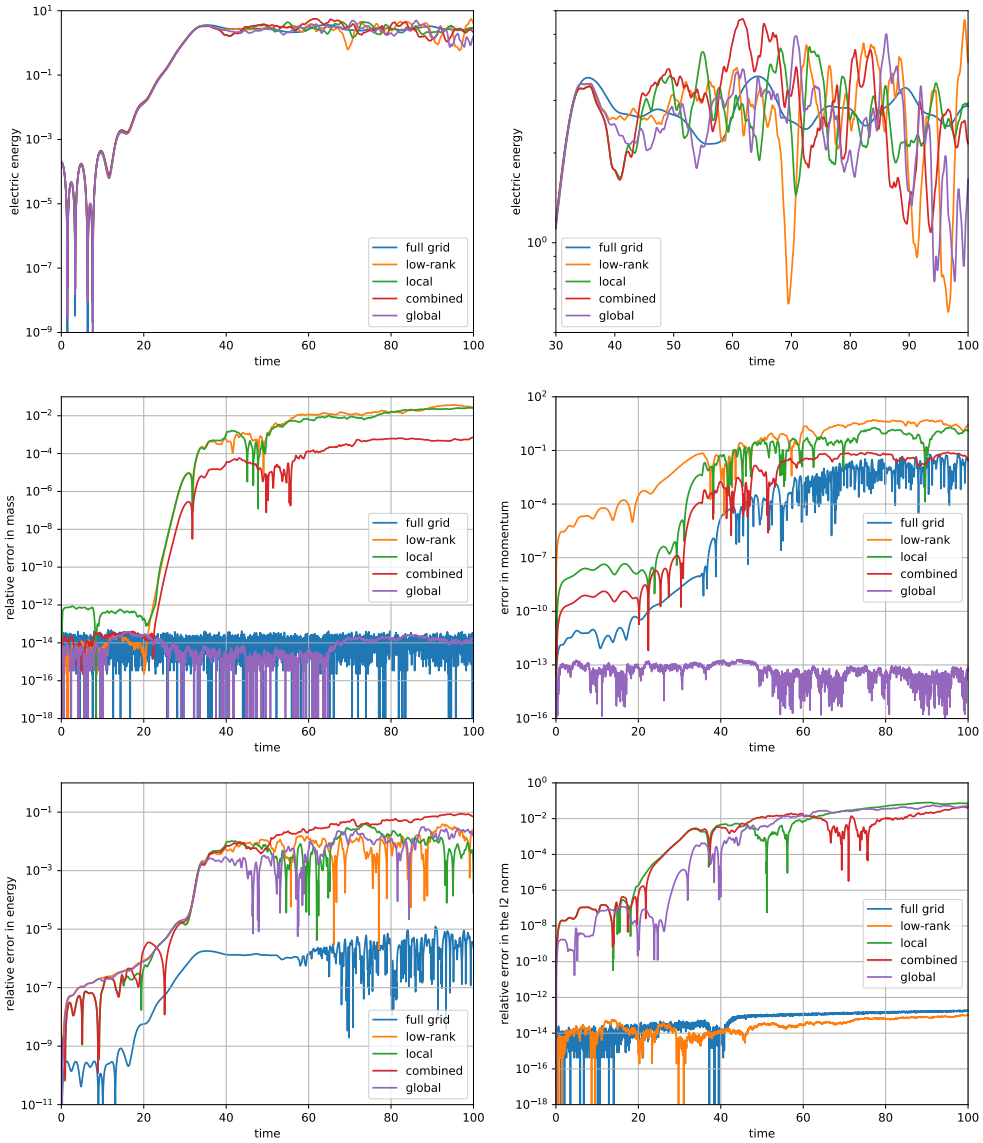

FIG. 6.1. *Numerical simulations of the* $1+1$ *dimensional (one dimension in space and one dimension in velocity) two-stream instability with rank* $r = 10$ *are shown. The combined approach uses* $w = 1$ *as the weight. For comparison, a direct simulation, based on a spectral method, is also shown.*

error in momentum is relatively large (above 10^{-1}). Mass conservation is significantly better, but the corresponding error still grows to above 10^{-4}. Employing the local correction somewhat improves conservation of mass and momentum. However, the dynamics are largely indistinguishable from the uncorrected algorithm. The largest improvement can be seen with the global correction. In this case we obtain mass and momentum conservation up to machine precision, as expected. Even a small improvement in the error in energy can be observed in this case.

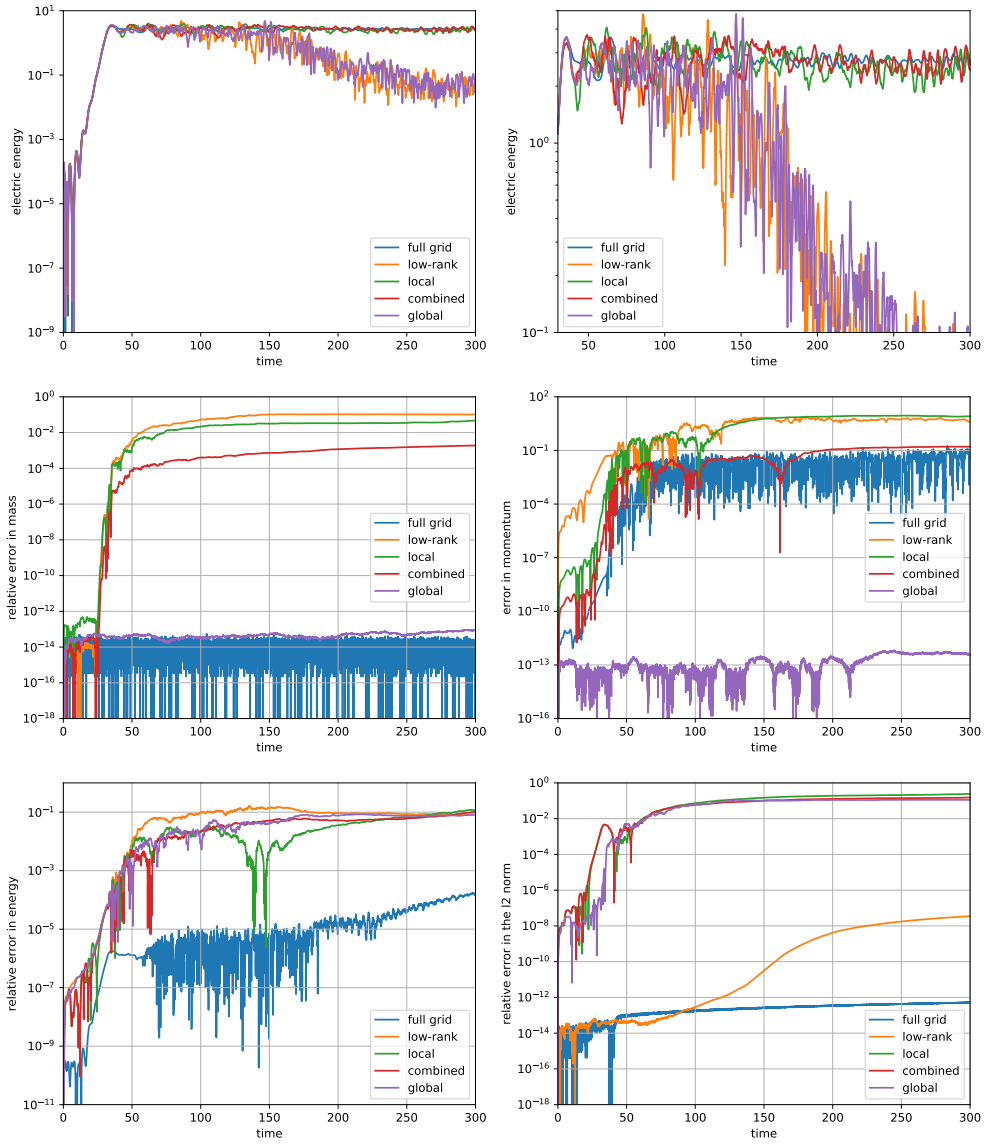

FIG. 6.2. *Numerical simulations of the* $1+1$ *dimensional (one dimension in space and one dimension in velocity) two-stream instability with rank* $r = 15$ *are shown. The combined approach uses* $w = 1$ *as the weight. For comparison, a direct simulation, based on a spectral method, is also shown.*

6.3. Landau damping. In this section we will present numerical results for the classic linear Landau damping. Specifically, we consider the domain $(0, 4\pi) \times (-6, 6)$ and impose the initial value

$$f_0(x, v) = \frac{1}{\sqrt{2\pi}} \mathrm{e}^{-v^2/2}(1 + \alpha \cos(kx)),$$

where we have chosen $\alpha = 10^{-2}$ and $k = \frac{1}{2}$. All simulations are conducted using 64 grid points in the x-direction and 256 grid points in the v-direction. In both directions

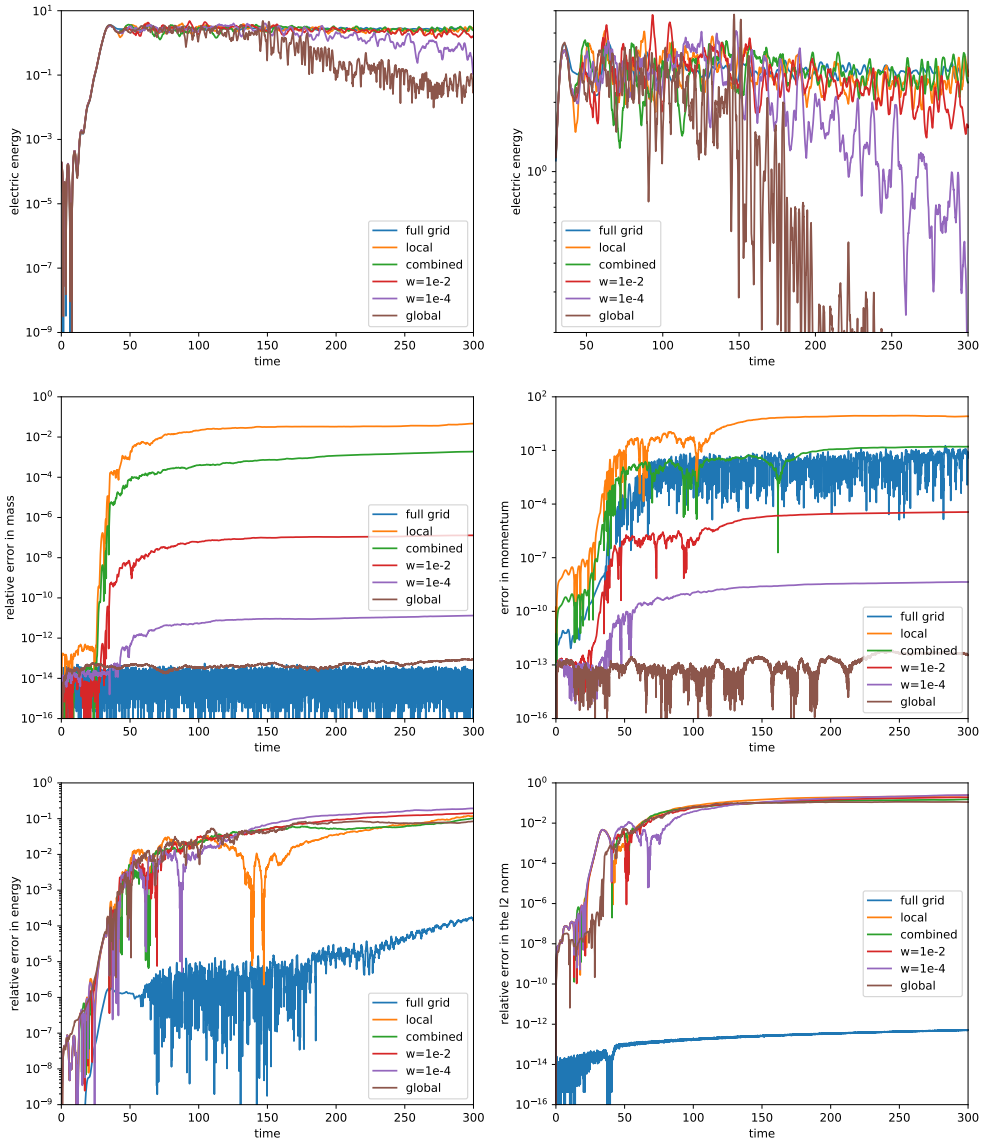

FIG. 6.3. *Numerical simulations of the* 1+1 *dimensional (one dimension in space and one dimension in velocity) two-stream instability with rank* $r = 15$ *are shown. For the combined approach we also show numerical results for* $w = 10^{-2}$ *and* $w = 10^{-4}$. *A weight of* $w = 0$ *corresponds to the global correction, and a weight of* $w = 1$ *to the combined correction described in section* 4. *For comparison, a direct simulation, based on a spectral method, is also shown.*

periodic boundary conditions are employed. In all simulations the second order Strang splitting scheme with time step size $\tau = 0.025$ is used.

The characteristic feature of the Landau damping problem is the exponential decay in the electric energy. It can be shown by a linear analysis that the decay rate is given by $\gamma \approx -0.153$. This value has been verified by a number of numerical simulations in the literature. It has been shown in [15] that the corresponding behavior is captured well by the low-rank approximation. From the standpoint of conserved

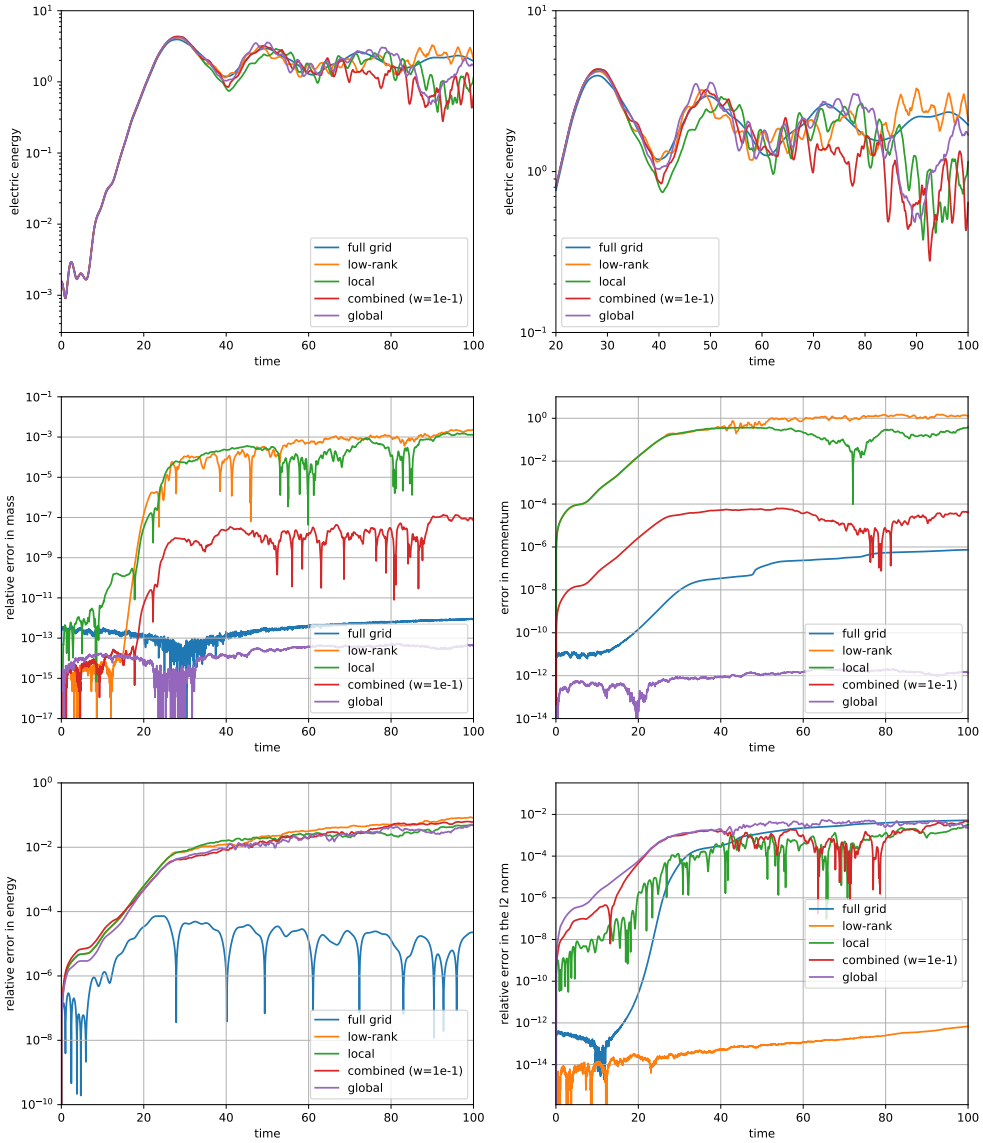

FIG. 6.4. *Numerical simulations of a bump-on-tail instability with rank $r = 15$ are shown. For comparison, a direct simulation, based on a semi-Lagrangian discontinuous Galerkin method, is also shown.*

quantities, Landau damping is less challenging compared to the two-stream instability. Mass is conserved almost up to machine precision even without performing any correction. The error in momentum is on the order of 10^{-9}. There is thus no need to perform any local correction. We can, however, use the global correction in order to reduce the error in momentum to machine precision. The corresponding numerical results are shown in Figure 6.5. For the corrected low-rank algorithm both mass and momentum are conserved up to machine precision. We observe no difference in the qualitative behavior of the numerical simulation.

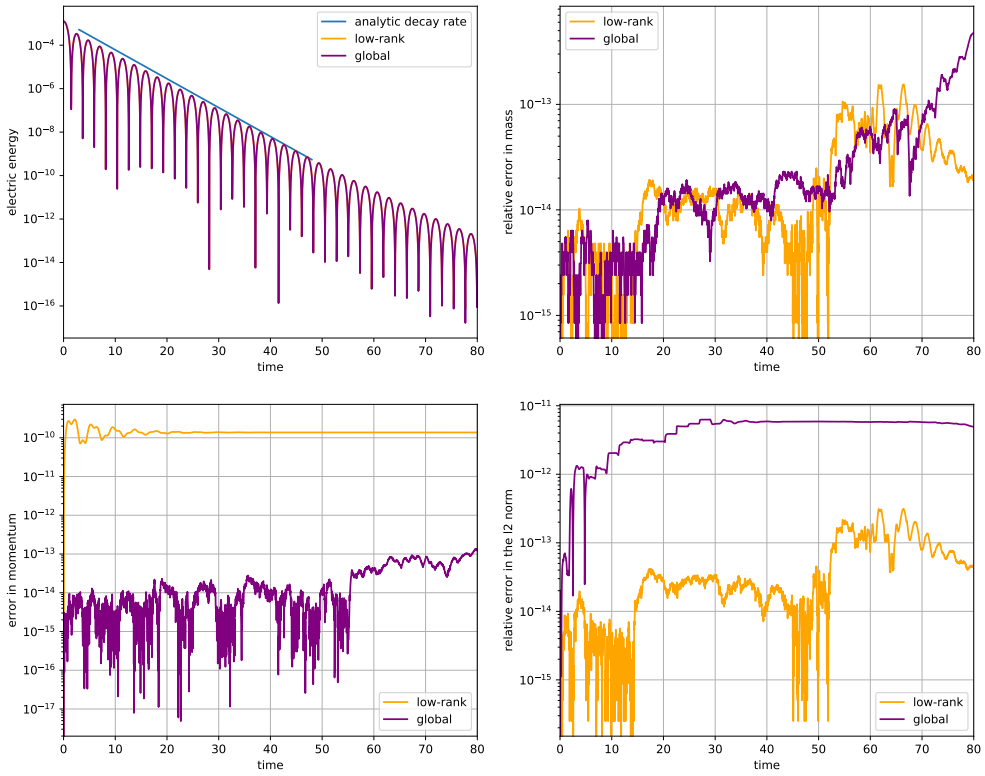

FIG. 6.5. *Numerical simulations of linear Landau damping with rank $r = 10$ are shown. The numerical solution is compared to the analytically determined decay rate $\gamma = -0.153$.*

7. Conclusion and outlook. We have proposed a numerical algorithm for a dynamical low-rank Vlasov–Poisson solver that is able to correct the error in specified invariants. More specifically, we are concerned with global conservation of mass and momentum, and a projected version of the local conservation laws that are associated with these two quantities. The algorithm, in particular the local correction approach, can significantly improve the qualitative behavior in long time integration (see the numerical results for the two-stream instability). Moreover, in many simulations, such as the bump-on-tail instability, the error in momentum is quite large without correction (on the order of unity and, in particular, significantly larger than the error in energy and in the L^2 norm). The global correction or the combined approach can then be used to significantly reduce this error. Applying the correction procedure does *not* negatively effect conservation of energy. We usually do observe some increase in the L^2 norm. However, the resulting behavior is comparable to full grid solvers (see Figure 6.4) and the L^2 error is still much smaller than the original error in mass and momentum, thus resulting in an overall improvement of the scheme. Let us also note that the numerical method introduced here can be generalized to other invariants as well.

A further application of the correction approach presented here would be in automatically choosing a suitable rank for the simulation. While adaptively choosing the rank is, in principle, straightforward, it is often not clear how to best measure the error. In particular, for the nonlinear regime the error in the particle density

function is always large. Nevertheless, good qualitative agreement in the observables of interest can often still be obtained (this is true for both semi-Lagrangian and low-rank Vlasov solvers; see, for example, [13]). From the simulations conducted, we conclude that the size of the correction would be a good candidate for such an error estimate. This measures whether a physically viable solution can still be obtained on the low-rank manifold (i.e., whether there is a point on the low-rank manifold that gives an approximation of the desired tolerance while still satisfying mass and momentum conservation).

REFERENCES

[1] J. Bigot, V. Grandgirard, G. Latu, C. Passeron, F. Rozar, and O. Thomine, *Scaling GYSELA code beyond 32K-cores on bluegene/Q*, ESAIM: Proc., 43 (2013), pp. 117–135.

[2] N. Crouseilles, L. Einkemmer, and E. Faou, *Hamiltonian splitting for the Vlasov–Maxwell equations*, J. Comput. Phys., 283 (2015), pp. 224–240.

[3] N. Crouseilles, L. Einkemmer, and E. Faou, *An asymptotic preserving scheme for the relativistic Vlasov–Maxwell equations in the classical limit*, Comput. Phys. Commun., 209 (2016), pp. 13–26.

[4] N. Crouseilles, G. Latu, and E. Sonnendrücker, *A parallel Vlasov solver based on local cubic spline interpolation on patches*, J. Comput. Phys., 228 (2009), pp. 1429–1446.

[5] N. Crouseilles, M. Mehrenberger, and E. Sonnendrücker, *Conservative semi-Lagrangian schemes for Vlasov equations*, J. Comput. Phys., 229 (2010), pp. 1927–1953.

[6] N. Crouseilles, M. Mehrenberger, and F. Vecil, *Discontinuous Galerkin semi-Lagrangian method for Vlasov-Poisson*, in ESAIM: Proc., 32 (2011), pp. 211–230.

[7] N. Crouseilles, T. Respaud, and E. Sonnendrücker, *A forward semi-Lagrangian method for the numerical solution of the Vlasov equation*, Comput. Phys. Commun., 180 (2009), pp. 1730–1745.

[8] A. Dedner, F. Kemm, D. Kröner, C. Munz, T. Schnitzer, and M. Wesenberg, *Hyperbolic divergence cleaning for the MHD equations*, J. Comput. Phys., 175 (2002), pp. 645–673.

[9] V. Ehrlacher and D. Lombardi, *A dynamical adaptive tensor method for the Vlasov-Poisson system*, J. Comput. Phys., 339 (2017), pp. 285–306.

[10] L. Einkemmer, *A mixed precision semi-Lagrangian algorithm and its performance on accelerators*, in Proceedings of the 2016 International Conference on High Performance Computing & Simulation (HPCS), 2016, pp. 74–80.

[11] L. Einkemmer, *High performance computing aspects of a dimension independent semi-Lagrangian discontinuous Galerkin code*, Comput. Phys. Commun., 202 (2016), pp. 326–336.

[12] L. Einkemmer, *A study on conserving invariants of the Vlasov equation in semi-Lagrangian computer simulations*, J. Plasma Phys., 83 (2017), 705830203.

[13] L. Einkemmer, *A performance comparison of semi-Lagrangian discontinuous Galerkin and spline based Vlasov solvers in four dimensions*, J. Comput. Phys. 376 (2019), pp. 937–951.

[14] L. Einkemmer, *A low-rank algorithm for weakly compressible flow*, SIAM J. Sci. Comput, to appear, preprint, https://arxiv.org/abs/1804.04561, 2018.

[15] L. Einkemmer and C. Lubich, *A low-rank projector-splitting integrator for the Vlasov–Poisson equation*, SIAM J. Sci. Comput., 40 (2018), pp. B1330–B1360, https://epubs.siam.org/doi/abs/10.1137/18M116383X.

[16] L. Einkemmer and A. Ostermann, *A strategy to suppress recurrence in grid-based Vlasov solvers*, Eur. Phys. J. D, 68 (2014), 197.

[17] L. Einkemmer, A. Ostermann, and C. Piazzola, *A Low-rank Projector-splitting Integrator for the Vlasov–Maxwell Equations with Divergence Correction*, preprint, https://arxiv.org/abs/1902.00424, 2019.

[18] F. Filbet and E. Sonnendrücker, *Comparison of Eulerian Vlasov solvers*, Comput. Phys. Commun., 150 (2003), pp. 247–266.

[19] J. Haegeman, C. Lubich, I. Oseledets, B. Vandereycken, and F. Verstraete, *Unifying time evolution and optimization with matrix product states*, Phys. Rev. B, 94 (2016), 165116.

[20] T. Jahnke and W. Huisinga, *A dynamical low-rank approach to the chemical master equation*, Bull. Math. Biol., 70 (2008), pp. 2283–2302.

[21] E. Kieri, C. Lubich, and H. Walach, *Discretized dynamical low-rank approximation in the presence of small singular values*, SIAM J. Numer. Anal., 54 (2016), pp. 1020–1038, https://doi.org/10.1137/15M1026791.

[22] O. Koch and C. Lubich, *Dynamical low-rank approximation*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 434–454, https://doi.org/10.1137/050639703.

[23] O. Koch and C. Lubich, *Dynamical tensor approximation*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2360–2375, https://doi.org/10.1137/09076578X.

[24] K. Kormann, *A semi-Lagrangian Vlasov solver in tensor train format*, SIAM J. Sci. Comput., 37 (2015), pp. 613–632, https://doi.org/10.1137/140971270.

[25] G. Latu, N. Crouseilles, V. Grandgirard, and E. Sonnendrücker, *Gyrokinetic semi-lagrangian parallel simulation using a hybrid OpenMP/MPI programming*, Recent Advances in Parallel Virtual Machine and Message Passing Interface, EuroPVM/MPI 2007, Lecture Notes in Comput. Sci. 4757, Springer, Berlin, Heidelberg 2007, pp. 356–364.

[26] C. Lubich, *From Quantum to Classical Molecular Dynamics: Reduced Models and Numerical Analysis*, European Mathematical Society, Zurich, 2008.

[27] C. Lubich, *Time integration in the multiconfiguration time-dependent Hartree method of molecular quantum dynamics*, Applied Mathematics Research eXpress, 2015 (2015), pp. 311–328.

[28] C. Lubich and I. Oseledets, *A projector-splitting integrator for dynamical low-rank approximation*, BIT Numer. Math., 54 (2014), pp. 171–188.

[29] C. Lubich, I. V. Oseledets, and B. Vandereycken, *Time integration of tensor trains*, SIAM J. Numer. Anal., 53 (2015), pp. 917–941, https://doi.org/10.1137/140976546.

[30] C. Lubich, T. Rohwedder, R. Schneider, and B. Vandereycken, *Dynamical approximation by hierarchical Tucker and tensor-train tensors*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 470–494, https://doi.org/10.1137/120885723.

[31] C. Lubich, B. Vandereycken, and H. Walach, *Time integration of rank-constrained Tucker tensors*, SIAM J. Numer. Anal., 56 (2018), pp. 1273–1290, .

[32] M. Mehrenberger, C. Steiner, L. Marradi, N. Crouseilles, E. Sonnendrucker, and B. Afeyan, *Vlasov on GPU (VOG project)*, preprint, https://arxiv.org/abs/1301.5892, 2013.

[33] H. Mena, A. Ostermann, L. Pfurtscheller, and C. Piazzola, *Numerical low-rank approximation of matrix differential equations*, J. Comput. Appl. Math., 340 (2018), pp. 602–614, https://doi.org/10.1016/j.cam.2018.01.035.

[34] H.-D. Meyer, F. Gatti, and G. A. Worth, *Multidimensional Quantum Dynamics*, John Wiley & Sons, Weinheim, Germany, 2009.

[35] H.-D. Meyer, U. Manthe, and L. S. Cederbaum, *The multi-configurational time-dependent Hartree approach*, Chem. Phys. Lett., 165 (1990), pp. 73–78.

[36] E. Musharbash and F. Nobile, *Dual Dynamically Orthogonal approximation of incompressible Navier Stokes equations with random boundary conditions*, J. Comput. Phys., 354 (2018), pp. 135–162.

[37] A. Nonnenmacher and C. Lubich, *Dynamical low-rank approximation: applications and numerical experiments*, Math. Comput. Simulation, 79 (2008), pp. 1346–1357.

[38] J. Qiu and A. Christlieb, *A conservative high order semi-Lagrangian WENO method for the Vlasov equation*, J. Comput. Phys., 229 (2010), pp. 1130–1149.

[39] J. Qiu and C. Shu, *Positivity preserving semi-Lagrangian discontinuous Galerkin formulation: theoretical analysis and application to the Vlasov–Poisson system*, J. Comput. Phys., 230 (2011), pp. 8386–8409.

[40] J. Rossmanith and D. Seal, *A positivity-preserving high-order semi-Lagrangian discontinuous Galerkin scheme for the Vlasov–Poisson equations*, J. Comput. Phys., 230 (2011), pp. 6203–6232.

[41] F. Rozar, G. Latu, and J. Roman, *Achieving memory scalability in the GYSELA code to fit exascale constraints*, in Parallel Processing and Applied Mathematics, Lecture Notes in Comput. Sci. 8385, Springer, Berlin, Heidelberg, 2013, pp. 185–195.

[42] N. J. Sircombe and T. D. Arber, *VALIS: A split-conservative scheme for the relativistic 2D Vlasov–Maxwell system*, J. Comput. Phys., 228 (2009), pp. 4773–4788.

[43] E. Sonnendrücker, J. Roche, P. Bertrand, and A. Ghizzo, *The semi-Lagrangian method for the numerical resolution of the Vlasov equation*, J. Comput. Phys., 149 (1999), pp. 201–220.

[44] J. P. Verboncoeur, *Particle simulation of plasmas: Review and advances*, Plasma Phys. Control. Fusion, 47 (2005), A231.