J Sci Comput (2016) 69:1346–1365
DOI 10.1007/s10915-016-0238-1

An *h*-Adaptive RKDG Method for the Vlasov–Poisson System

Hongqiang Zhu[1] · **Jianxian Qiu**[2] · **Jing-Mei Qiu**[3]

Received: 10 January 2016 / Revised: 8 June 2016 / Accepted: 11 June 2016 /
Published online: 16 June 2016

Abstract In this paper, we propose a new *h*-adaptive indicator for the Runge–Kutta discontinuous Galerkin (RKDG) scheme in simulations of the Vlasov–Poisson (VP) system. This adaptive indicator, tailored for the VP system, is based on the principle that each cell assumes solution variations as equally as possible. Under the framework of the RKDG method, such adaptive indicator is particularly simple and cheap for the computation. Its effectiveness is demonstrated by extensive numerical tests. The detailed adaptive algorithm as well as some important implementation issues, including the grid and data structure, adaptive criteria, data prolongation/projection and mesh projection, is presented.

Keywords Runge–Kutta discontinuous Galerkin · Vlasov–Poisson · *h*-Adaptive

The research is partially supported by NSFC Grants 11201242, 91530107 and 11571290, NSF Grants NSF-DMS-1217008 and NSF-DMS-1522777, Jiangsu Government Scholarship for Overseas Studies, and Air Force Office of Scientific Computing FA9550-12-0318.

✉ Jing-Mei Qiu
jingqiu@math.uh.edu

Hongqiang Zhu
zhuhq@njupt.edu.cn

Jianxian Qiu
jxqiu@xmu.edu.cn

1 School of Natural Sciences, Nanjing University of Posts and Telecommunications, Nanjing 210023, Jiangsu, People's Republic of China

2 School of Mathematical Sciences and Fujian Provincial Key Laboratory of Mathematical Modeling and High-Performance Scientific Computing, Xiamen University, Xiamen 361005, Fujian, People's Republic of China

3 Department of Mathematics, University of Houston, Houston, TX 77204-3008, USA

1 Introduction

The single species Vlasov–Poisson (VP) system

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} f + \mathbf{E}(\mathbf{x}, t) \cdot \nabla_{\mathbf{v}} f = 0, \tag{1.1}$$

$$\mathbf{E}(\mathbf{x}, t) = -\nabla_{\mathbf{x}} \phi(\mathbf{x}, t), \qquad -\triangle_{\mathbf{x}} \phi(\mathbf{x}, t) = \rho(\mathbf{x}, t) \tag{1.2}$$

is a nonlinear kinetic system that models the dynamics of a collisionless plasma consisting of electrons and a uniform background of fixed ions under the effects of a self-consistent electrostatic field and possibly an externally supplied field. $f(\mathbf{x}, \mathbf{v}, t)$ is a probability density function in six-dimensional phase space $(\mathbf{x}, \mathbf{v}) \in \mathbb{R}^3 \times \mathbb{R}^3$, which denotes the probability of finding a particle with velocity $\mathbf{v}$ at position $\mathbf{x}$ at time t. $\mathbf{E}$ is the electric field and ϕ is the self-consistent electrostatic potential. Eq. (1.1) is the Vlasov equation which models the transport of the electrons. The electrons are coupled to the electrostatic potential through Poisson's equation in Eq. (1.2). The probability density function couples to the long range fields via the charge density, $\rho = \int_{\mathbb{R}^3} f(\mathbf{x}, \mathbf{v}, t) d\mathbf{v} - 1$, where we take the limit of uniformly distributed infinitely massive ions in the background.

Computational challenges for kinetic simulations are two folded. The kinetic scale makes the computation very expensive. The Vlasov equation is of six dimensional in phase space plus time. In addition, it is computationally challenging to numerically resolve fine scale filamentation solution structures naturally arise in Vlasov simulations. It is highly desired that numerical methods are high order accurate with low numerical dissipation, are adaptive to resolve multi-scale filamentation solution features, and are robust without artificial numerical oscillations.

Among existing algorithms for VP simulations, the particle-in-cell method [3,14,22] is very popular due to its relatively low computational cost for high dimensional problems. However, it suffers statistical noise $\mathcal{O}(1/\sqrt{N})$ with N being the number of particles. There are highly accurate mesh-based semi-Lagrangian [5,11,17,24–26,28,32,33] and Eulerian [1,8,15,21,23,34,35,37] methods, which have been shown to be advantageous due to their efficiency and effectiveness in resolving rich solution structures. The semi-Lagrangian method is designed by propagating information along characteristic curves. Among existing semi-Lagrangian algorithms for the Vlasov equation, dimensional splitting originally proposed by Cheng and Knorr [7] has been widely used in many different settings, e.g. semi-Lagrangian finite volume [11,17], finite difference [5,24,25,32,33] and finite element discontinuous Galerkin (DG) [26,28] methods. However, they are subject to a second order splitting error. In an Eulerian method, typically the spatial derivatives are discretized, in a truly multi-dimensional setting, by finite difference/volume/element method. Then the spatially discretized ODE systems are evolved with high order numerical time integrator such as the Runge–Kutta (RK) methods [19] via the method-of-line approach. These methods have been well-known for being highly accurate both in space and time, and being very robust as a black-box procedure in a truly multi-dimensional setting (without dimensional splitting) [8,21].

The VP solutions are well-known to exhibit a variety of dynamical phenomena, one of which is the filamentation. It occurs when different characteristics surfaces associated to the nonlinear transport (Vlasov) equation wrap in the phase space, resulting in stiff gradients of the unknown function f. Such phenomenon motivates us to develop *an h-adaptive RK DG approach* for the Vlasov simulations, in order to focus the computational effort where it is most needed. Adaptive methods are widely used to increase spatial and temporal resolution

of numerical simulations beyond the limits imposed by the available hardware and to save the computational cost (both storage and CPU time). Compared with existing adaptive mesh refinement algorithms based on finite volume [2] or finite difference [30] scheme, the h-adaptive finite element DG method is advantageous for its compactness for parallelization and for handling boundary conditions in a complex geometry. Moreover, the DG framework offers great convenience in data projection and data prolongation among different levels of mesh refinement, while preserving the mass conservation. Existing h-adaptive RKDG schemes include the ones developed by Flaherty et al. [4,13,18,27], Zhu and Qiu [38,39] and Dedner et al. [12] for nonlinear time-dependent hyperbolic conservation laws. We also refer to Hartmann and Houston [20], where duality techniques were used for designing the adaptive strategy.

In this paper, we focus on developing an h-adaptive DG scheme for 1D1V Vlasov simulations. We first briefly review the RKDG method for the VP system in Sect. 2. Then we propose an h-adaptive RKDG scheme with mesh-refinement criteria tailored for the VP system in Sect. 3. The effectiveness of the new scheme is demonstrated via several classical test examples for VP simulations in Sect. 4. Finally, concluding remarks are given in Sect. 5.

2 Review

Consider the 1D1V VP system

$$f_t + vf_x + E(x,t)f_v = 0, \tag{2.1}$$

$$E(x,t) = -\phi(x,t)_x, \quad -\phi(x,t)_{xx} = \rho(x,t), \tag{2.2}$$

$$\rho(x,t) = \int_{-v_{max}}^{v_{max}} f(x,v,t)dv - 1, \tag{2.3}$$

on the 2D domain $\Omega = \Omega_x \times \Omega_v$, where Ω_x is a bounded domain with periodic boundary condition and $\Omega_v = [-v_{max}, v_{max}]$ with v_{max} chosen s.t. f vanishes outside Ω_v. Standard RKDG method solves this system in the following way. At each time level, ρ is firstly computed by integrating cell-wise DG polynomials in the v-direction. Then ϕ, hence the electric field E, is obtained by numerically solving the 1D Poisson's equation (2.2). At last, f is updated by applying the DG scheme to Eq. (2.1). Such procedure can be repeated for every RK stage of one time step evolution, until the final time is reached. If uniform mesh is adopted, the 1D Poisson's equation can be solved by a fast Fourier transform (FFT) due to the periodicity in space. Otherwise, a local DG (LDG) scheme is more suitable because of its capability of handling irregular meshes, especially for higher dimensional problems to be considered in our future work.

In what follows, we first briefly review the RKDG scheme for the Vlasov equation. Detailed description of this method can be found in the review paper [10]. Then the LDG scheme [6] for the Poisson's equation is presented.

2.1 RKDG Scheme

Let us describe the RKDG scheme for the Vlasov equation. In our description below, we use the subscript h to denote the numerical discretization of the corresponding continuous function. For example, f_h and E_h are the DG approximations to the unknown functions f and E respectively. Given a mesh discretization $\mathcal{M}_h$ of the phase space domain Ω, we seek the approximate solution $f_h(x,v,t)$ in the finite element space of discontinuous piecewise polynomials

$$V_h^k = \left\{ \psi \in L^2(\Omega) : \psi|_K \in \mathbb{P}^k(K), \forall K \in \mathcal{M}_h \right\},$$

where $\mathbb{P}^k(K)$ denotes the set of polynomials of total degree at most k on the element K. It is known that the dimension of such space is $Q_k + 1$ with $Q_k = k(k+3)/2$. We adopt the following orthogonal basis for V_h^k,

$$w_{\frac{i(i+1)}{2}+j}(x, v) = W_{i-j}(x) W_j(v), \quad i = 0, \ldots, k, \quad j = 0, \ldots, i, \tag{2.4}$$

via the orthogonal 1D basis of Legendre polynomials

$$\begin{cases} W_0(x) = 1, \\ W_l(x) = \dfrac{1}{2^l l!} \dfrac{d^{\,l}(x^2-1)^l}{dx^l}, \quad l > 0. \end{cases} \tag{2.5}$$

Then the local orthogonal basis over cell K is given by

$$w_l^{(K)}(x, v) = w_l\left(\frac{2(x - x_K)}{\Delta x_K}, \frac{2(v - v_K)}{\Delta v_K} \right), \quad l = 0, \ldots, Q_k, \tag{2.6}$$

in which (x_K, v_K) is the center of rectangle K and Δx_K and Δv_K are lengths of K's sides in the direction of x and v respectively. (Note that this approach of basis construction works only for this 1D1V case. For higher dimensional cases other approaches should be considered.) Now the numerical solution $f_h(x, v, t)$ in the space V_h^k can be expressed as

$$f_h(x, v, t)|_K = \sum_{l=0}^{Q_k} f_K^{(l)}(t) w_l^{(K)}(x, v) \tag{2.7}$$

where $f_K^{(l)}(t)$, $l = 0, \ldots, Q_k$ are the degrees of freedom. Particularly, $f_K^{(0)}(t)$ is the cell average of f_h over K.

To obtain the RKDG scheme, we multiply Eq. (2.1) by each of the basis, integrate over each computational cell and perform integration by parts to formulate the following semi-discrete version of the DG scheme for the approximate solution f_h: for all test functions w_l, with $l = 0, \ldots, Q_k$ and all $K \in \mathcal{M}_h$,

$$\frac{d}{dt} \int_K f_h w_l^{(K)} dx dv - \int_K F(f_h) \cdot \nabla w_l^{(K)} dx dv + \sum_{e \in \partial K} \int_e F(f_h) \cdot n_{e,K} w_l^{(K)} ds = 0 \tag{2.8}$$

where $F(f_h) = (v f_h, E_h f_h)$, $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial v})$ and $n_{e,K}$ is the outward unit normal to the edge e. The volume integral term $\int_K F(f_h) \cdot \nabla w_l^{(K)} dx dv$ can be computed either exactly or by a numerical quadrature of sufficiently high order of accuracy. The line integral in Eq. (2.8) is typically discretized by a Gaussian quadrature with sufficient accuracy

$$\int_e F(f_h) \cdot n_{e,K} w_l^{(K)} ds \approx |e| \sum_{l=1}^{m} \omega_l F(f_h(G_l, t)) \cdot n_{e,K} w_l^{(K)}(G_l, t) \tag{2.9}$$

where $F(f_h(G_l, t)) \cdot n_{e,K}$ is replaced by a monotone numerical flux. In this paper, we use the simple Lax–Friedrichs flux

$$\begin{aligned} F(f_h(G_l, t)) \cdot n_{e,K} \approx \frac{1}{2} \big[& (F(f_h^-(G_l, t)) + F(f_h^+(G_l, t))) \cdot n_{e,K} \\ & - \alpha(f_h^+(G_l, t) - f_h^-(G_l, t)) \big] \end{aligned} \tag{2.10}$$

where α is taken as an upper bound for the eigenvalues of the Jacobian in the direction of $n_{e,K}$, and f_h^- and f_h^+ are the values of f_h inside and outside the cell K at the Gaussian point G_l.

Using Eq. (2.7), the first term in Eq. (2.8) can be rewritten by $c_l \frac{\Delta x_K \Delta v_K}{4} \frac{d}{dt} f_K^{(l)}(t)$ where $c_l = \int_{-1}^{1}\int_{-1}^{1} w_l^2(x,v)dxdv$, $l = 0,\ldots,Q_k$ are constants. As a result, the semi-discrete scheme (2.8) is an ODE system for $\{f_K^{(l)}(t) : l = 0,\ldots,Q_k, K \in \mathcal{M}_h\}$. This system coupled with a suitable time discretization scheme, such as the TVD (total variation diminishing) RK method [31], completes the RKDG scheme. In this paper for $k = 1$, we use the second order RK time stepping

$$\begin{aligned} \Phi^{(1)} &= \Phi^n + \Delta t L(\Phi^n), \\ \Phi^{n+1} &= \frac{1}{2}\Phi^n + \frac{1}{2}\Phi^{(1)} + \frac{1}{2}\Delta t L(\Phi^{(1)}) \end{aligned} \tag{2.11}$$

for the ODE system $\Phi_t = L(\Phi)$. For $k = 2$, we adopt the following third order version

$$\begin{aligned} \Phi^{(1)} &= \Phi^n + \Delta t L(\Phi^n), \\ \Phi^{(2)} &= \frac{3}{4}\Phi^n + \frac{1}{4}\Phi^{(1)} + \frac{1}{4}\Delta t L(\Phi^{(1)}), \\ \Phi^{n+1} &= \frac{1}{3}\Phi^n + \frac{2}{3}\Phi^{(2)} + \frac{2}{3}\Delta t L(\Phi^{(2)}). \end{aligned} \tag{2.12}$$

2.2 LDG Scheme

To define the LDG scheme for the 1D Poisson's equation, we rewrite Eq. (2.2) as a system of first-order equations on Ω_x:

$$\phi(x,t)_x = -E(x,t), \quad E(x,t)_x = \rho(x,t). \tag{2.13}$$

Periodic boundary conditions are assumed. Beyond periodicity, we need to enforce some additional conditions to uniquely determine ϕ. In current work we set $\int_{\Omega_x} \phi(x,t)dx = 0$.

Given a partition $\mathcal{M}_h^{(x)} = \bigcup_j I_j$ of the x-domain Ω_x with $I_j = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$ and $x_j = \frac{1}{2}(x_{j-\frac{1}{2}} + x_{j+\frac{1}{2}})$, we make use of the approximation space

$$Z_h^k = \{\xi \in L^2(\Omega_x) : \xi|_{I_j} \in P^k(I_j), \forall I_j \in \mathcal{M}_h^{(x)}\}$$

in which P^k is the space of polynomials in one dimension of degree up to k. The approximate solution (ϕ_h, E_h) is then defined by the following weak formulation: for all $I_j \in \mathcal{M}_h^{(x)}$ and all test functions $(q,r) \in Z_h^k \times Z_h^k$,

$$\hat{\phi}_h\left(x_{j+\frac{1}{2}},t\right) q\left(x_{j+\frac{1}{2}}^-\right) - \hat{\phi}_h\left(x_{j-\frac{1}{2}},t\right) q\left(x_{j-\frac{1}{2}}^+\right) - \int_{I_j} \phi_h q_x dx = -\int_{I_j} E_h q dx, \tag{2.14}$$

$$\hat{E}_h\left(x_{j+\frac{1}{2}},t\right) r\left(x_{j+\frac{1}{2}}^-\right) - \hat{E}_h\left(x_{j-\frac{1}{2}},t\right) r\left(x_{j-\frac{1}{2}}^+\right) - \int_{I_j} E_h r_x dx = \int_{I_j} \rho_h r dx, \tag{2.15}$$

where the superscripts "±" indicate the left and right limits of the corresponding functions with respect to x. As suggested in [6,9], we use the following numerical fluxes

$$\hat{E}_h(x,t) = E_h(x^+,t) + \phi_h(x^+,t) - \phi_h(x^-,t), \tag{2.16}$$

$$\hat{\phi}_h(x,t) = \phi_h(x^-,t). \tag{2.17}$$

The function E_h can be locally solved in terms of ϕ_h by using Eq. (2.14) since $\hat{\phi}_h$ does not depend on E_h. Hence it can be eliminated from Eq. (2.15), leaving ϕ_h as the only unknown. The resulting system is a linear system for ϕ_h which is solved in our numerical tests by using the generalized minimum residual method equipped with the incomplete LU decomposition for the preconditioning.

3 Adaptive RKDG: Algorithm and Implementation Details

In this section, we describe our proposed h-adaptive RKDG-LDG algorithm for the VP system and discuss the related implementation issues. For test examples in the current paper, we only consider one dimensional problems (1D in physical space and 1D in velocity space), hence rectangular meshes are considered. For problems with higher dimension in the physical space, triangular meshes could be preferred depending on the geometry of physical domains. There are many computational advantages that the DG scheme offers in the adaptive setting. For example, compared with finite volume or finite difference adaptive mesh refinement schemes [2,30], it is extremely convenient and computationally efficient to perform data prolongation and data projection between different levels of mesh with preservation of cell averages for mass conservation. Compared with the continuous finite element method, the DG scheme is also well-known for its flexibility to handle the situation when there are hanging nodes in adaptive meshes. We will first present the grid and data structure, followed by the flow chart of our proposed adaptive algorithm. Specific issues, such as refinement criteria and data prolongation/projection will be discussed in details afterwards.

3.1 Grid and Data Structure

The data structures for h-adaptive meshes can be generally classified into two categories, block-based structure and cell-based structure. For block-based structure, there is a coarse base grid covering the entire computational domain. Each block, constituted by finer cells of same size, covers a rectangular sub-domain where more resolution is required. These structured blocks are nested on different mesh-levels, laying over the base grid. For cell-based structure, each cell is refined or coarsened separately from the others, as needed. Both block-based and cell-based structures can be organized as a logical hierarchical tree structure. The nodes of the tree represent the blocks for block-based structure and individual cells for cell-based structure. It is hard to say which approach is better in practice. Both structures have their own advantages and disadvantages. For block-based structure, the computational effort to maintain the tree structure information is smaller since the resulting tree structure is lighter, compared to the cell-based structure. Further more, the original solver can be directly applied to the blocks without any modification. As to the defects, the first is that block-based structure is less flexible. It is hard to cover complex features in the solution with only a few blocks. The second is that clustering algorithm is required to organize individual cells into blocks. Usually, such algorithm is sophisticated and needs to be performed periodically when time is involved. For cell-based structure, the main advantages are the high flexibility

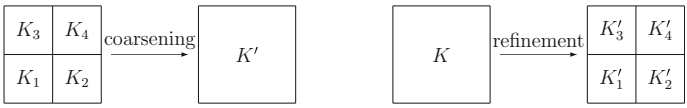

Fig. 1 Sketches of local mesh coarsening (*left*) and refinement (*right*)

in refinement and coarsening, as well as the efficiency in using cells. Since the RKDG and LDG methods used in this paper are friendly to irregular meshes, we choose the cell-based structure in this work.

Each cell in the initial partition of the computational domain is considered as the *root* of a tree. Following our previous work in [39], we adopt a simple way to perform local mesh refinement and coarsening. More specifically, mesh refinement is achieved by dividing a cell into four new cells (*children*) of equal size (see the right sketch in Fig. 1). The four new cells, which are generated simultaneously in a single division, are called a 'GROUP'. To coarsen the mesh, we can only merge the cells that form a GROUP (see the left sketch in Fig. 1). A cell is called a *leaf* cell if it does not have any children. All the leaf cells constitute the computational mesh. Each leaf cell has a variable L denoting its mesh-level which is defined by the number of divisions needed to obtain this cell. L is set to zero for roots, and is increased by one after a division and decreased by one after a merger. We also need a maximal mesh-level, denoted by LEV, to restrict fineness of the adaptive mesh. Finally, we describe the data structure associated with each leaf cell and non-leaf cell. For the non-leaf cells, the associated data include cell coordinates, mesh-level and the pointers to its father and children in the tree. For the leaf cells, additional data such as degrees of freedom for solutions, pointers for neighboring cells in four directions, indicator for mesh refinement and coarsening (value $-1/1$ for mesh coarsening/refinement respectively), and indicator for boundary cells (1 for boundary cells and 0 otherwise) are stored.

3.2 Algorithm Flow Chart

Below, we first describe the flow chart of the algorithm. In our descriptions, the superscript n stands for the current time level t^n unless otherwise specified. For each of the computational cell K (a leaf on the tree), associated information include the DG solution with $\{f_K^{(l)} : l = 0, \ldots, Q_k\}$ as its degrees of freedom and its mesh level L_K. Here and below, argument t in $f_K^{(l)}(t)$ is omitted for simplicity.

Algorithm 3.1 (h-adaptive scheme for the VP system)

- *The initial set up.* The algorithm starts from an uniform rectangular mesh $\mathcal{M}_h^0$ as the root grid. We perform the L^2 projection of the initial data $f(x, v, t = 0)$ on $\mathcal{M}_h^0$. The numerical solution is a piecewise polynomial of degree k living on $\mathcal{M}_h^0$. Associated with each root cell $K \in \mathcal{M}_h^0$, we have the following

$$\{f_K^{0,(l)} : l = 0, \ldots, Q_k\}, \qquad L_K^0 = 0.$$

- *Solution evolution from t^n to t^{n+1}, for $n = 0, 1, \ldots$.*

  1. *Mesh refinement and coarsening.* Each cell in the current mesh will be marked to be refined, coarsened, or kept unchanged via the "adaptive indicator" discussed in Sect. 3.3. We take a cell $K \in \mathcal{M}_h^n$ as an example to demonstrate the idea.
     - The cell will be quartered if it is marked to be refined and its mesh-level $L_K < LEV$. There are four newly generated cells $K_1, \ldots, K_4$ (children), each of

which will have its mesh-level increase by one from that of its parent cell. The corresponding polynomial on new cells will be obtained from data prolongation mechanism discussed in Sect. 3.4.

– Each GROUP of cells (four children) will be removed if all of them are marked to be merged. Mesh-level is decreased by one for the new cell. The corresponding polynomial on the new leaf cell will be obtained from data projection discussed in Sect. 3.4.

2. *Poisson solver in physical space.* Obtain the one-dimensional spatial mesh by projecting the two-dimensional mesh $\mathcal{M}_h^{n+1}$ along the v-direction (see Sect. 3.5 for details). Compute ρ_h through Eq. (2.3) and solve Eq. (2.2) by LDG method to get ϕ_h and E_h.
3. *Evolve solution.* Evolve the solution on the current mesh from t_n to t_{n+1} by applying the RKDG procedure to Eq. (2.1) to update $\{f_K^{(l),n+1} : l = 0, \ldots, Q_k, \forall K \in \mathcal{M}_h^{n+1}\}$.

3.3 Refinement Criteria

For h-adaptive methods, one key issue lies in where the mesh should be refined or coarsened. We name the procedure 'adaptive indicator' which is used to mark each cell to be refined, coarsened or kept unchanged. For nonlinear hyperbolic problems that developing discontinuities, meshes near the discontinuities should be refined in order to catch sharp transitions of discontinuities and coarse meshes should be used in other regions to save the computational cost. Several troubled-cell indicators are used as adaptive indicators to detect the discontinuous regions with success in [38,39]. However, the VP solutions are known to remain smooth given smooth initial conditions [29], yet develop filamentation solution structures with large gradients in the phase space over time. Therefore, troubled-cell indicators for nonlinear hyperbolic system are no longer suitable. We propose a new "adaptive indicator" tailored for the VP system based on the principle that each cell assumes solution variations as equally as possible. If the solution variation on cell K is 'too big', cell K is marked to be refined. If the solution variation on cell K is small enough, cell K is marked to be coarsened. Otherwise, K is kept unchanged.

We measure solution variations on cell K in the x and v directions by $|f_K^{(1)}|$ and $|f_K^{(2)}|$ in Eq. (2.7) respectively. A cell will be marked to be refined (or merged) if its variations are much bigger (or smaller) than the average values. Specifically, we define the adaptive indicator as follows.

Algorithm 3.2 (Adaptive indicator) For every cell $K \in \mathcal{M}_h$, let $\theta_x^K = |f_K^{(1)}|$, $\theta_v^K = |f_K^{(2)}|$. We compute their average values in the corresponding directions

$$\bar{\theta}_x = \frac{\Sigma_K \theta_x^K}{N}, \quad \bar{\theta}_v = \frac{\Sigma_K \theta_v^K}{N}, \tag{3.1}$$

with N being the total number of cells. Let $\bar{\theta}_{max} = \max(\bar{\theta}_x, \bar{\theta}_v)$, then a cell K is marked to be

$$\begin{cases} \text{coarsened}, & \text{if } \theta_x^K < \gamma_1 \bar{\theta}_{max} \quad \text{and } \theta_v^K < \gamma_1 \bar{\theta}_{max}, \\ \text{refined}, & \text{if } \theta_x^K > \gamma_2 \bar{\theta}_{max} \text{ or } \theta_v^K > \gamma_2 \bar{\theta}_{max}, \\ \text{kept unchanged}, & \text{otherwise.} \end{cases} \tag{3.2}$$

In our numerical tests, we set $\gamma_1 = 1/2$ and $\gamma_2 = 2$.

3.4 Data Prolongation and Data Projection

We propose to perform L^2 projection for data prolongation and projection between different levels of meshes. Suppose we have already known f_h on mesh $\mathcal{M}_h$, and we need to determine its L^2 projection on a new cell K' represented by a new polynomial function f_h'. In particular, f_h' would satisfy the following equations,

$$\int_{K'} f_h' w_l^{(K')}(x,v)dxdv = \int_{K'} f_h w_l^{(K')}(x,v)dxdv, \quad l = 0,\ldots,Q_k. \tag{3.3}$$

Let the degrees of freedom associated with f_h' on cell K' be $f_{K'}^{(l)}$, $l = 0,\ldots,Q_k$, as in Eq. (2.7), then

$$f_{K'}^{(l)} = \frac{4}{c_l \Delta x_{K'} \Delta v_{K'}} \int_{K'} f_h w_l^{(K')}(x,v)dxdv. \tag{3.4}$$

Note that the formula with $l = 0$ implies the mass conservation. Since f_h is a piecewise polynomial, the integral in Eq. (3.4) can be computed exactly. Now we are ready to provide the formulas of data projection and data prolongation between fine and coarse meshes.

- *Data projection.* When four cells K_1, K_2, K_3, K_4 are merged to a new cell K' (see the left sketch in Fig. 1), the new degrees of freedom computed by Eq. (3.4) are as follows for the DG scheme with P^2 polynomial space,

$$\begin{aligned}
f_{K'}^{(0)} =& \frac{1}{4}\left(f_{K_1}^{(0)} + f_{K_2}^{(0)} + f_{K_3}^{(0)} + f_{K_4}^{(0)}\right),\\
f_{K'}^{(1)} =& \frac{3}{8}\left(-f_{K_1}^{(0)} + f_{K_2}^{(0)} - f_{K_3}^{(0)} + f_{K_4}^{(0)}\right) + \frac{1}{8}\left(f_{K_1}^{(1)} + f_{K_2}^{(1)} + f_{K_3}^{(1)} + f_{K_4}^{(1)}\right),\\
f_{K'}^{(2)} =& \frac{3}{8}\left(-f_{K_1}^{(0)} - f_{K_2}^{(0)} + f_{K_3}^{(0)} + f_{K_4}^{(0)}\right) + \frac{1}{8}\left(f_{K_1}^{(2)} + f_{K_2}^{(2)} + f_{K_3}^{(2)} + f_{K_4}^{(2)}\right),\\
f_{K'}^{(3)} =& \frac{5}{16}\left(-f_{K_1}^{(1)} + f_{K_2}^{(1)} - f_{K_3}^{(1)} + f_{K_4}^{(1)}\right) + \frac{1}{16}\left(f_{K_1}^{(3)} + f_{K_2}^{(3)} + f_{K_3}^{(3)} + f_{K_4}^{(3)}\right),\\
f_{K'}^{(4)} =& \frac{9}{16}\left(f_{K_1}^{(0)} - f_{K_2}^{(0)} - f_{K_3}^{(0)} + f_{K_4}^{(0)}\right) + \frac{3}{16}\left(-f_{K_1}^{(1)} - f_{K_2}^{(1)} + f_{K_3}^{(1)} + f_{K_4}^{(1)}\right)\\
&+ \frac{3}{16}\left(-f_{K_1}^{(2)} + f_{K_2}^{(2)} - f_{K_3}^{(2)} + f_{K_4}^{(2)}\right) + \frac{1}{16}\left(f_{K_1}^{(4)} + f_{K_2}^{(4)} + f_{K_3}^{(4)} + f_{K_4}^{(4)}\right),\\
f_{K'}^{(5)} =& \frac{5}{16}\left(-f_{K_1}^{(2)} - f_{K_2}^{(2)} + f_{K_3}^{(2)} + f_{K_4}^{(2)}\right) + \frac{1}{16}\left(f_{K_1}^{(5)} + f_{K_2}^{(5)} + f_{K_3}^{(5)} + f_{K_4}^{(5)}\right).
\end{aligned} \tag{3.5}$$

  For the DG scheme with P^1 polynomial space, only the first three formulas are needed.
- *Data prolongation.* When a cell K is divided into four subcells K_1', K_2', K_3', K_4' (see the right sketch in Fig. 1), the new degrees of freedom for $k = 2$ can be computed by the following formulas with $l = 1, 2, 3, 4$,

$$\begin{aligned}
f_{K_l'}^{(0)} =& f_K^{(0)} + 2\lambda_x^{(l)} f_K^{(1)} + 2\lambda_v^{(l)} f_K^{(2)} + 4\lambda_x^{(l)}\lambda_v^{(l)} f_K^{(4)},\\
f_{K_l'}^{(1)} =& \frac{1}{2} f_K^{(1)} + 3\lambda_x^{(l)} f_K^{(3)} + \lambda_v^{(l)} f_K^{(4)}, \qquad f_{K_l'}^{(2)} = \frac{1}{2} f_K^{(2)} + \lambda_x^{(l)} f_K^{(4)} + 3\lambda_v^{(l)} f_K^{(5)},\\
f_{K_l'}^{(3)} =& \frac{1}{4} f_K^{(3)}, \qquad f_{K_l'}^{(4)} = \frac{1}{4} f_K^{(4)}, \qquad f_{K_l'}^{(5)} = \frac{1}{4} f_K^{(5)}
\end{aligned} \tag{3.6}$$

  where $\lambda_x^{(l)} = \frac{(-1)^l}{4}$ for $l = 1,\ldots,4$, $\lambda_v^{(1)} = \lambda_v^{(2)} = -\frac{1}{4}$ and $\lambda_v^{(3)} = \lambda_v^{(4)} = \frac{1}{4}$. For $k = 1$, one can use the same formulas, but dropping the higher moment terms $f_K^{(3)}$, $f_K^{(4)}$, and $f_K^{(5)}$.

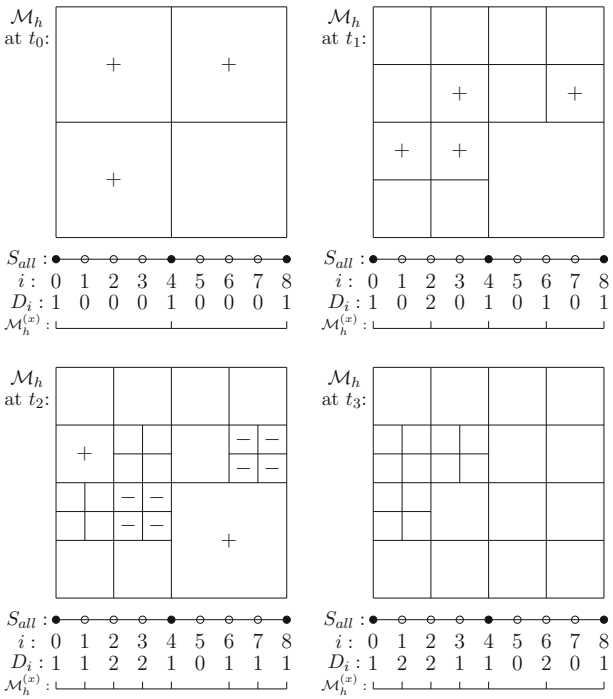

Fig. 2 A simple example to generate the 1D spatial mesh projected from the 2D mesh

3.5 Mesh Projection

The 1D mesh for the Poisson's equation is projected from the 2D adaptive mesh. The mesh projection process is designed in the following cost-effective fashion.

1. Initially, we let $S_{all} = \{x_i, i = 0, \ldots N_x 2^{LEV}\}$, where N_x denotes the number of cells along the x-axis at $t = 0$, be the set of grid points in the x-direction of the fully refined mesh. See S_{all} in Fig. 2 as an example of having a 2×2 root grid with $LEV = 2$.
2. We set a counter D_i associated with each of the grid point in S_{all}. Initially, $D_i = 0$, except that at the grid points for the 1D projection of 2D root grid, we let $D_i = 1$. *The projected 1D mesh $\mathcal{M}_h^{(x)}$ is constructed from the grid points with nonzero D_i*'s. See Fig. 2.
3. We update D_i's as the 2D mesh is dynamically refined or coarsened. When a cell is refined or coarsened, the corresponding D_i will be increased or decreased by 1 respectively. See four plots in Fig. 2 for an example of updating D_i in a dynamic refinement/coarsening process. In the figure, $+/-$ sign is used to mark the cells that will be refined/coarsened at the next time-level. Again, $\mathcal{M}_h^{(x)}$ is constructed from the grid points with nonzero D_i's.

4 Numerical Tests

In this section, we perform a detailed study of the proposed h-adaptive RKDG scheme and evaluate its performance in simulating several classical test examples by comparing with the fixed-mesh RKDG scheme. For convenience, we refer fixed-mesh (standard) RKDG scheme and h-adaptive RKDG scheme as nonadaptive scheme and adaptive scheme respectively.

Attention has not been paid to the issue of time discretization efficiency, so global time steps are used in the RK method, which depend on the smallest cell size at each time-level. Study of local time stepping schemes is the subject of future work. In order to make the scheme more robust during the numerical simulations, the maximum-principle-satisfying technique proposed in [36] is applied.

Below, we recall some analytically conserved quantities in the VP system which can be used as diagnostics in a numerical scheme.

1. L^p norm for $1 \leq p < \infty$:

$$\|f\|_p = \left(\int_v \int_x |f(x,v,t)|^p dx dv \right)^{\frac{1}{p}}. \tag{4.1}$$

2. Energy:

$$\text{Energy} = \int_v \int_x f(x,v,t) v^2 dx dv + \int_x E^2(x,t) dx. \tag{4.2}$$

3. Entropy:

$$\text{Entropy} = \int_v \int_x f(x,v,t) \log(f(x,v,t)) dx dv. \tag{4.3}$$

Tracking relative deviations of these quantities numerically will be a good measure of the quality of numerical schemes. The relative deviation is defined to be the deviation away from the corresponding initial value divided by the magnitude of the initial value. Periodic boundary conditions are imposed in the x-direction and zero boundary conditions are imposed in the v-direction for all of our test problems.

For clarity, the variance of the RKDG schemes will be denoted as `InitialResolution-LEVj-Order` where `InitialResolution` is expressed by the product of N_x (number of cells along the x-axis) and N_v (number of cells along the v-axis) at the initial time, `LEVj` indicates that $LEV = j$ is being used, and `Order` is either `P1` (P^1 case, i.e. $k = 1$) or `P2` (P^2 case, i.e. $k = 2$). Note that the special case `LEV0` stands for the nonadaptive RKDG schemes using fixed uniform meshes in this work. For example, `32*64-LEV4-P1` denotes the h-adaptive RKDG scheme with initial resolution $32 * 64$ and $k = 1$, and `64*128-LEV0-P2` denotes the nonadaptive RKDG scheme with (initial) resolution $64 * 128$ and $k = 2$. Since there are no analytical solutions for the test examples used in this paper, the results computed by scheme `256*512-LEV0-P2` are used as reference results. In addition, to save space, only the P^2 results are reported for they are sufficient to illustrate the capability of the adaptive scheme.

Example 4.1 (Strong Landau damping). Consider the example of strong Landau damping for the VP system. The initial condition is

$$f(x,v,t=0) = \frac{1}{\sqrt{2\pi}} (1 + \alpha \cos(\beta x)) \exp\left(-\frac{v^2}{2}\right) \tag{4.4}$$

with $\alpha = 0.5$, $\beta = 0.5$ on the domain $(x,v) \in [0, 4\pi] \times [-5, 5]$. The final time is $T = 60$. Time evolution of the distribution function for scheme `16*32-LEV4-P2`, along with the corresponding meshes, is shown in Fig. 3. We can see clearly that the adaptive meshes resolve the fine solution structures and filamentation very well. Fine meshes are generated at the regions with fine structures while coarse meshes are used elsewhere, and this works dynamically. In order to compare with the nonadaptive results, we compute the average cell number of an adaptive scheme defined by $\bar{N} = (\sum_{n=0}^{TOT} N_n)/TOT$ where N_n is the number of cells at the nth time-level and TOT is the total number of time-levels, and then use the same

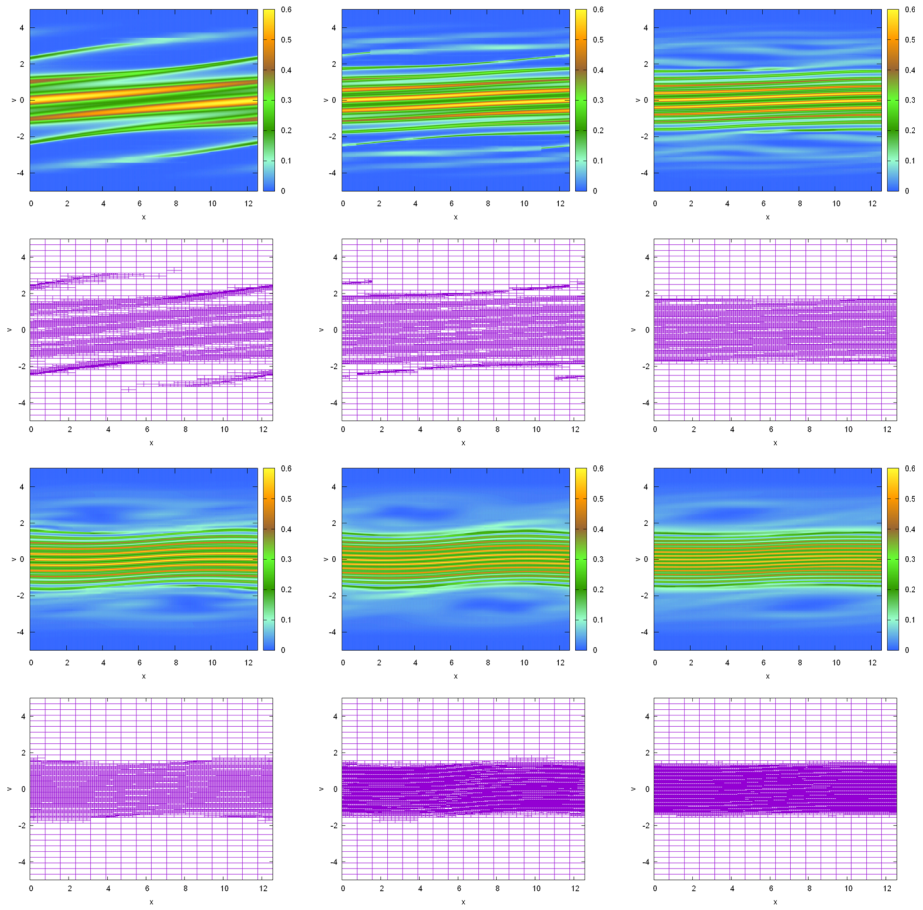

Fig. 3 Strong Landau damping, time evolution of f and the adaptive meshes for scheme `16*32-LEV4-P2`. *Upper two rows from left to right* $t = 10, 20, 30$; *lower two rows from left to right* $t = 40, 50, 60$

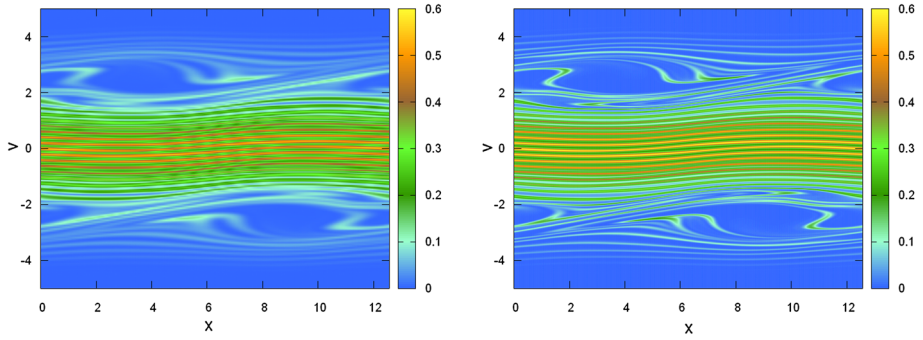

Fig. 4 Strong Landau damping, solution contours for schemes `81*162-LEV0-P2` (*left*) and `256*512-LEV0-P2` (*right*) at $t = 60$

number of cells to obtain the nonadaptive results. For this adaptive scheme, $\bar{N} = 12{,}913.4 \approx 81 * 162$. We plot the solution contour for the nonadaptive scheme `81*162-LEV0-P2` in Fig. 4, together with the contour for `256*512-LEV0-P2` as a reference solution. Comparing

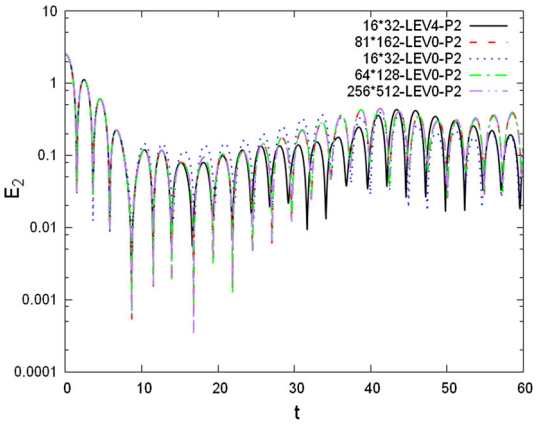

Fig. 5 Strong Landau damping, time evolution of L^2 norm of the electric field

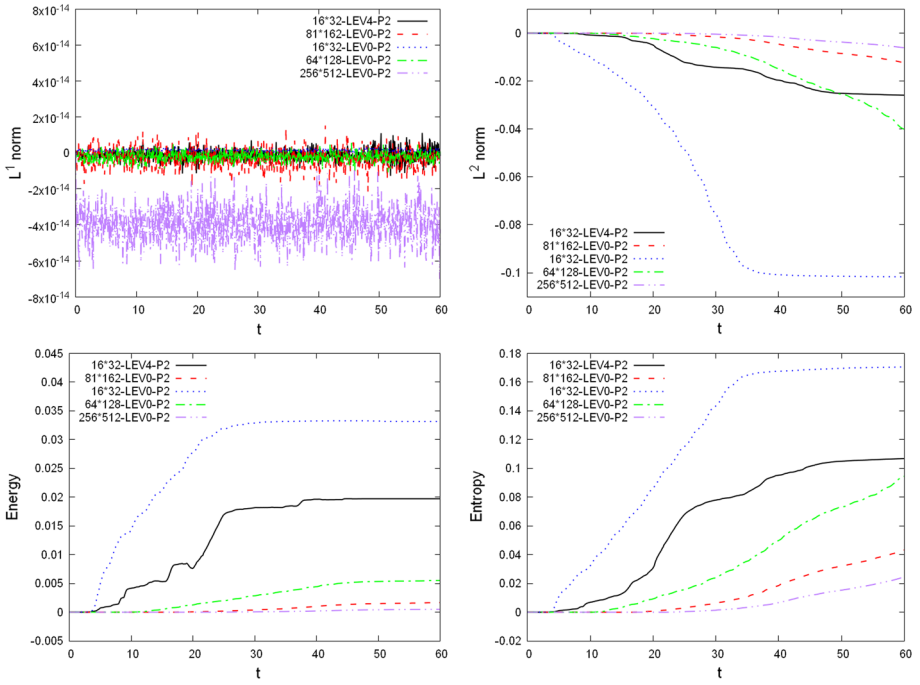

Fig. 6 Strong Landau damping, time evolution of relative deviations of the discrete L^1 (*upper left*) and L^2 (*upper right*) norms as well as the kinetic energy (*lower left*) and entropy (*lower right*)

these figures with the last contour figure in Fig. 3, we observe that the adaptive solution fails to capture the gentle structure outside the filamentation area since coarse meshes are always used there. However, the adaptive solution shows clearer filamentation than scheme `81*162-LEV0-P2`. In Fig. 5, time evolution of L^2 norm of the electric field is provided. We find that the nonadaptive and adaptive schemes produce almost the same initial linear decay rate and the growth rate due to particle trapping, regardless of the different resolutions. At last we report the time evolution of discrete L^1 norm, L^2 norm, kinetic energy and entropy in Fig. 6. In general, the adaptive scheme has better performance than the nonadaptive scheme with the coarse mesh in preserving norms.

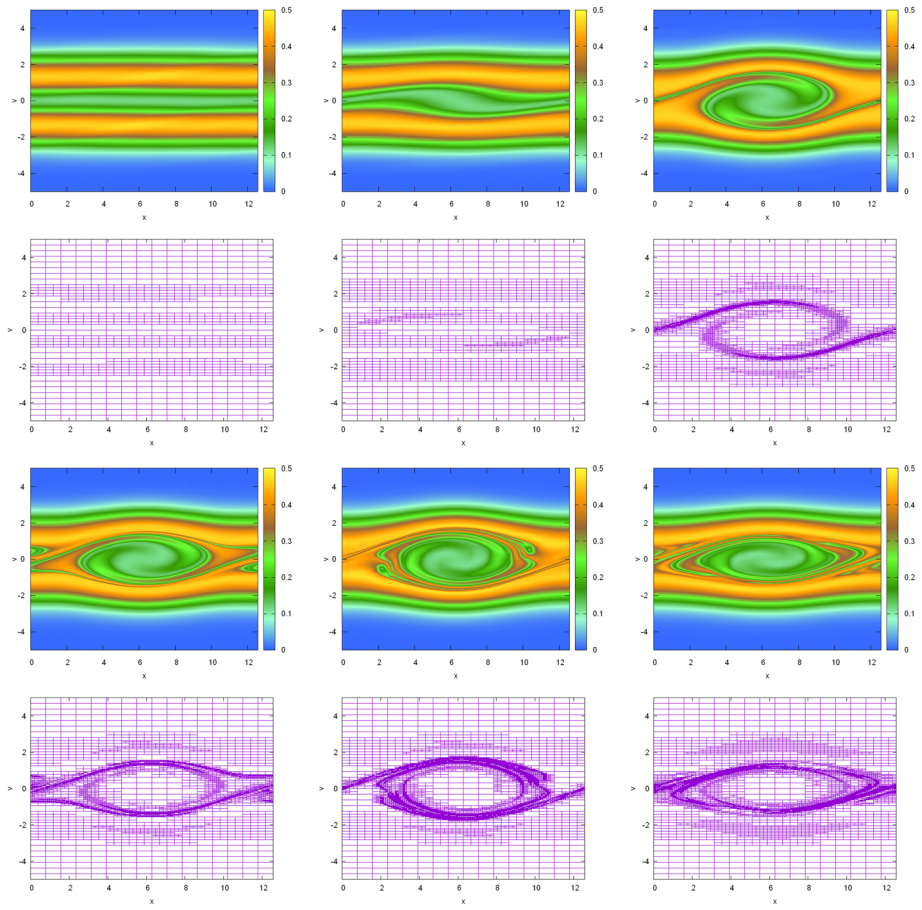

Fig. 7 Two-stream instability [16], time evolution of f and the adaptive meshes for scheme `16*32-LEV4-P2`. *Upper two rows from left to right* $t = 5, 15, 25$; *lower two rows from left to right* $t = 35, 45, 53$

Example 4.2 (Two-stream instability [16]). The next example we consider is the case of two-stream instability, with an unstable initial distribution function:

$$f(x, v, t = 0) = \frac{2}{7\sqrt{2\pi}}(1 + 5v^2)\exp\left(-\frac{v^2}{2}\right) [1 + \alpha((\cos(2\beta x) + \cos(3\beta x))/1.2 + \cos(\beta x))] \tag{4.5}$$

where $\alpha = 0.01$, $\beta = 0.5$. The simulation is up to $T = 53$ with $v_{max} = 5$. The length of the domain in the x-direction is $L = 2\pi/\beta$ and the background ion distribution function is fixed, uniform and chosen so that the total net charge density for the system is zero. Time evolution of the distribution function for scheme `16*32-LEV4-P2`, together with the corresponding meshes, is plot in Fig. 7. It is shown that the generated adaptive meshes are in accord with development of the solution and fine solution structures are well resolved. In this case, $\bar{N} = 5837.8 \approx 55 * 110$. We show the solution contours for nonadaptive schemes `55*110-LEV0-P2` and `256*512-LEV0-P2` in Fig. 8. The comparison between

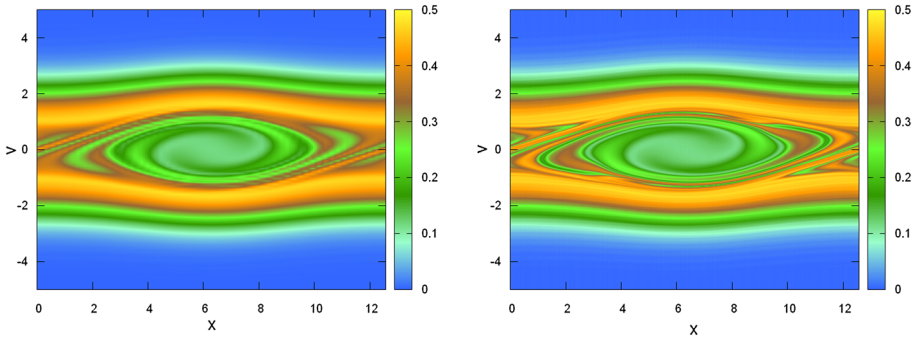

Fig. 8 Two-stream instability [16], solution contours for schemes `55*110-LEV0-P2` (*left*) and `256*512-LEV0-P2` (*right*) at $t = 53$

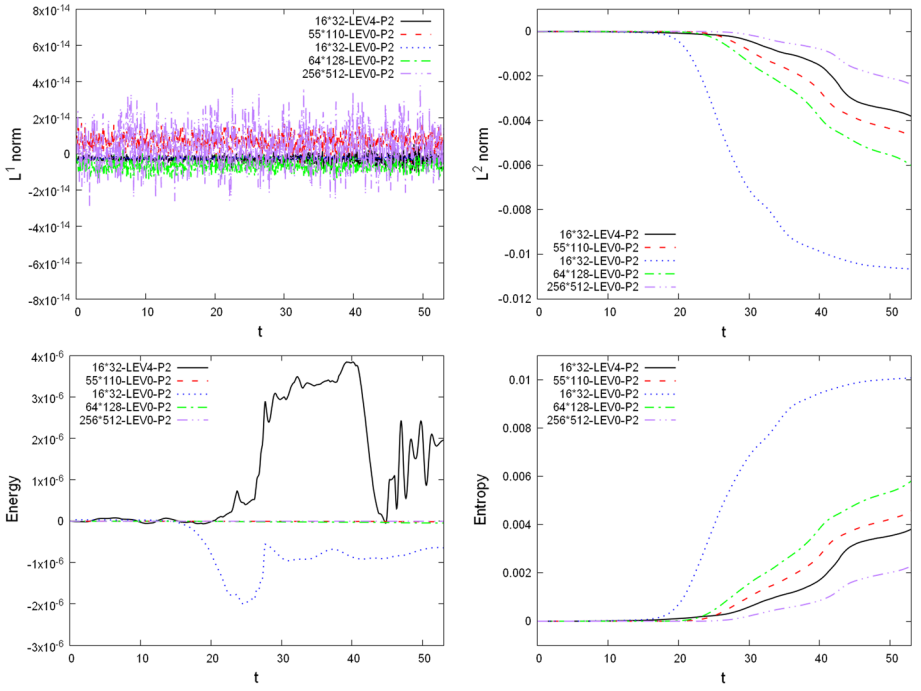

Fig. 9 Two-stream instability [16], time evolution of relative deviations of the discrete L^1 (*upper left*) and L^2 (*upper right*) norms as well as the kinetic energy (*lower left*) and entropy (*lower right*)

these figures and the last contour figure in Fig. 7 demonstrates the better performance of scheme `16*32-LEV4-P2` over scheme `55*110-LEV0-P2` in resolving the fine solution structures. Lastly, we report the time evolution of discrete L^1 norm, L^2 norm, kinetic energy and entropy in Fig. 9.

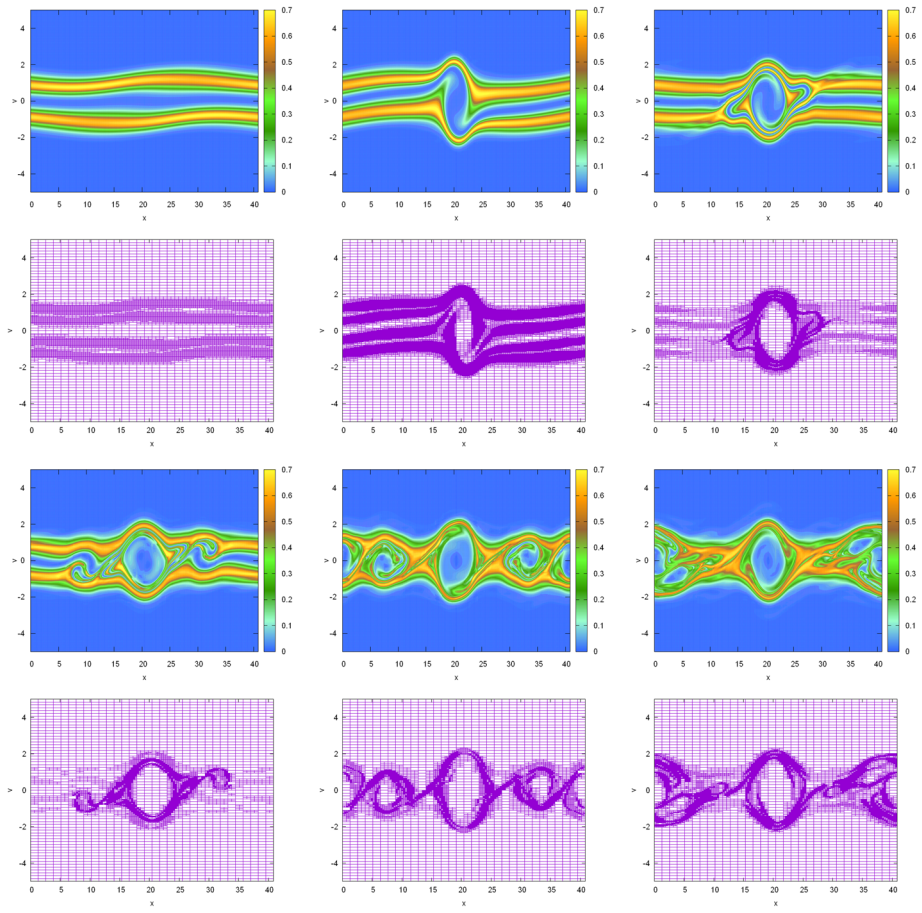

Fig. 10 Two-stream instability [11], time evolution of f and the adaptive meshes for scheme `32*64-LEV4-P2`. *Upper two rows from left to right* $t = 10, 20, 30$; *lower two rows from left to right* $t = 40, 55, 70$

Example 4.3 (Two-stream instability [11]). We consider the symmetric two stream instability with the initial condition

$$f(x, v, t = 0) = \frac{1}{2v_{th}\sqrt{2\pi}} (1 + 0.05\cos(\beta x)) \left[\exp\left(-\frac{(v-u)^2}{2v_{th}^2} \right) + \exp\left(-\frac{(v+u)^2}{2v_{th}^2} \right) \right] \tag{4.6}$$

where $u = 0.99$, $v_{th} = 0.3$, $\beta = 2/13$. The computational domain is $(x, v) \in [0, 13\pi] \times [-5, 5]$ and $t \in (0, 70]$. The background ion distribution function is fixed, uniform and chosen so that the total net charge density for the system is zero. Time evolution of the distribution function and the corresponding meshes for scheme `32*64-LEV4-P2` are reported in Fig. 10. We again observe that the adaptive meshes are generated dynamically according to the development of the solution structure. As a result, this adaptive scheme is able to capture the fine solution structures. For this scheme, $\bar{N} = 24{,}941.3 \approx 112 * 224$. We also plot the solution contours for nonadaptive schemes `112*224-LEV0-P2` and `256*512-LEV0-P2` in Fig. 11, and compare with the last contour figure in Fig. 10. Many more solution details can

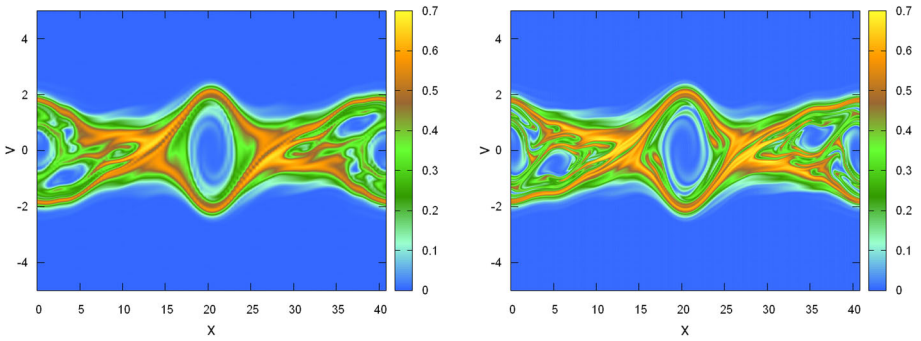

Fig. 11 Two-stream instability [11], solution contours for schemes `112*224-LEV0-P2` (*left*) and `256*512-LEV0-P2` (*right*) at $t = 70$

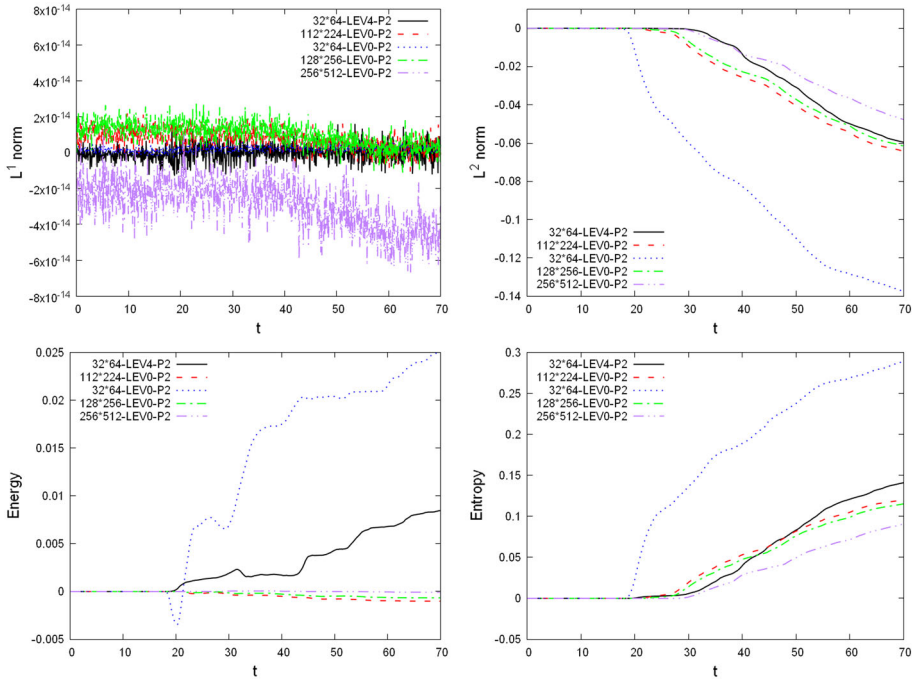

Fig. 12 Two-stream instability [11], time evolution of relative deviations of the discrete L^1 (*upper left*) and L^2 (*upper right*) norms as well as the kinetic energy (*lower left*) and entropy (*lower right*)

be spotted for scheme `32*64-LEV4-P2` than `112*224-LEV0-P2`, which shows advantage of the adaptive scheme over the nonadaptive scheme when they use the same degrees of freedom (in the mean sense). Finally, we report the time evolution of discrete L^1 norm, L^2 norm, kinetic energy and entropy in Fig. 12.

To gain a better understanding of effectiveness of the adaptive strategy, for all the examples above we first show the related data in Table 1, including (a) TND: total number of divisions; (b) N_T: number of cells at the final time; (c) $\bar{N}$; and (d) PR: the percentage ratio of $\bar{N}$ to

Table 1 Related data

Example	Scheme	TND	N_T	$\bar{N}$	PR
Example 4.1	`16*32-LEV4-P2`	7.1E+4	29,468	12,913.4	9.85
Example 4.2	`16*32-LEV4-P2`	1.9E+4	10,220	5837.8	4.45
Example 4.3	`32*64-LEV4-P2`	2.5E+5	37,136	24,941.3	4.76

Table 2 Comparison of CPU time

Example	Scheme	CPU time
Example 4.1	`16*32-LEV4-P2`	2.48
	`81*162-LEV0-P2`	0.44
	`128*256-LEV0-P2`	1.97
	`256*512-LEV0-P2`	17.18
Example 4.2	`16*32-LEV4-P2`	1.09
	`55*110-LEV0-P2`	0.13
	`128*256-LEV0-P2`	2.63
	`256*512-LEV0-P2`	17.06
Example 4.3	`32*64-LEV4-P2`	5.90
	`112*224-LEV0-P2`	1.00
	`128*256-LEV0-P2`	1.69
	`256*512-LEV0-P2`	11.05

the number of cells of a fully refined mesh, i.e. $PR = 100\bar{N}/(2^{LEV}N_0)$. In the table we can see that all the values of PR are far less than 100, which indicates that our adaptive algorithm needs much less cells than the one adopting fixed mesh providing that they produce comparable solutions.

Secondly, we compare the CPU time (in hours) between adaptive and nonadaptive schemes in Table 2. The data shows that all the adaptive schemes cost CPU time much less than the corresponding nonadaptive schemes with fully refined uniform meshes. (Note that for Example 4.3, the corresponding nonadaptive scheme with fully refined mesh should be scheme `512*1024-LEV0-P2`, which is not shown in the table because it is highly time-consuming to run this scheme.) It seems that the CPU time of the adaptive schemes is close to the nonadaptive schemes using one-level coarser mesh than the fully refined one. As a result, our adaptive algorithm has the advantage of saving the computational cost and improving the solution quality.

Lastly, we remark that, in practice it is not easy to predict a sufficiently refined uniform mesh for solution evolution of the VP system. An adaptive algorithm is in great need.

5 Concluding Remarks

In this paper, we propose an h-adaptive RKDG scheme for solving the VP system. The adaptive indicator is designed based on the principle that each cell assumes solution variations as equally as possible. Under the framework of the RKDG method with rectangular cells

being used, this adaptive indicator is easy to code and cheap to run. Our numerical tests have validated its effectiveness in saving the computational cost and improving the solution quality.

Our future research includes the following: (1) Apply the methodology to higher dimensional VP systems. (2) Extend the methodology to triangular meshes so that complicated geometries can be handled. (3) Design anisotropic h-adaptive schemes for the VP system. Adaptive isotropic meshes often tend to use too many cells in the region of large solution error. This is especially true when problems have an anisotropic feature that the solution exhibit a strong directional behavior. For example, solution of the strong Landau damping problem in the filamentation region changes more significantly in the velocity space than in the physical space (see Fig. 3). More benefits of mesh adaptation can be taken via an anisotropic adaptive approach.

References

1. Ayuso, B., Carrillo, J.A., Shu, C.-W.: Discontinuous Galerkin methods for the one-dimensional Vlasov–Poisson system. Kinet. Relat. Model. **4**, 955–989 (2011)
2. Berger, M.J., Oliger, J.: Adaptive mesh refinement for hyperbolic partial differential equations. J. Comput. Phys. **53**(3), 484–512 (1984)
3. Birdsall, C.K., Langdon, A.B.: Plasma Physics Via Computer Simulation. CRC Press, Boca Raton (2004)
4. Biswas, R., Devine, K., Flaherty, J.: Parallel, adaptive finite element methods for conservation laws. Appl. Numer. Math. **14**, 255–283 (1994)
5. Carrillo, J.A., Vecil, F.: Nonoscillatory interpolation methods applied to Vlasov-based models. SIAM J. Sci. Comput. **29**, 1179–1206 (2007)
6. Castillo, P., Cockburn, B., Perugia, I., Schötzau, D.: An a priori error analysis of the local discontinuous Galerkin method for elliptic problems. SIAM J. Numer. Anal. **38**(5), 1676–1706 (2000)
7. Cheng, C.-Z., Knorr, G.: The integration of the Vlasov equation in configuration space. J. Comput. Phys. **22**, 330–351 (1976)
8. Cheng, Y., Gamba, I.M., Morrison, P.J.: Study of conservation and recurrence of Runge–Kutta discontinuous Galerkin schemes for Vlasov–Poisson systems. J. Sci. Comput. **56**(2), 319–349 (2013)
9. Cockburn, B., Kanschat, G., Perugia, I., Schötzau, D.: Superconvergence of the local discontinuous Galerkin method for elliptic problems on Cartesian grids. SIAM J. Numer. Anal. **39**, 264–285 (2001)
10. Cockburn, B., Shu, C.-W.: Runge–Kutta discontinuous Galerkin methods for convection-dominated problems. J. Sci. Comput. **16**, 173–261 (2001)
11. Crouseilles, N., Mehrenberger, M., Sonnendrücker, E.: Conservative semi-Lagrangian schemes for Vlasov equations. J. Comput. Phys. **229**(6), 1927–1953 (2010)
12. Dedner, A., Makridakis, C., Ohlberger, M.: Error control for a class of Runge–Kutta discontinuous Galerkin methods for nonlinear conservation laws. SIAM J. Numer. Anal. **45**, 514–538 (2007)
13. Devine, K., Flaherty, J.: Parallel adaptive hp-refinement techniques for conservation laws. Appl. Numer. Math. **20**, 367–386 (1996)
14. Eastwood, J.W.: Particle simulation methods in plasma physics. Comput. Phys. Commun. **43**, 89–106 (1986)
15. Fijalkow, E.: A numerical solution to the Vlasov equation. Comput. Phys. Commun. **116**(2–3), 319–328 (1999)
16. Filbet, F., Sonnendrücker, E.: Comparison of Eulerian Vlasov solvers. Comput. Phys. Commun. **150**(3), 247–266 (2003)
17. Filbet, F., Sonnendrucker, E., Bertrand, P.: Conservative numerical schemes for the Vlasov equation. J. Comput. Phys. **172**(1), 166–187 (2001)
18. Flaherty, J., Loy, R., Shephard, M., Szymanski, B., Teresco, J., Ziantz, L.: Adaptive local refinement with octree load-balancing for the parallel solution of three-dimensional conservation laws. J. Parallel Distrib. Comput. **47**, 139–152 (1997)
19. Gottlieb, S., Shu, C.W., Tadmor, E.: Strong stability-preserving high-order time discretization methods. SIAM Rev. **43**(1), 89–112 (2001)
20. Hartmann, R., Houston, P.: Adaptive discontinuous Galerkin finite element methods for nonlinear hyperbolic conservation laws. SIAM J. Sci. Comput. **24**, 979–1004 (2002)

21. Heath, R., Gamba, I., Morrison, P., Michler, C.: A discontinuous Galerkin method for the Vlasov–Poisson system. J. Comput. Phys. **231**, 1140–1174 (2012)
22. Hockney, R.W., Eastwood, J.W.: Computer Simulation Using Particles. CRC Press, Boca Raton (2010)
23. Klimas, A., Farrell, W.: A splitting algorithm for Vlasov simulation with filamentation filtration. J. Comput. Phys. **110**, 150–163 (1994)
24. Qiu, J.-M., Christlieb, A.: A conservative high order semi-Lagrangian WENO method for the Vlasov equation. J. Comput. Phys. **229**(4), 1130–1149 (2010)
25. Qiu, J.-M., Shu, C.-W.: Conservative semi-Lagrangian finite difference WENO formulations with applications to the Vlasov equation. Commun. Comput. Phys. **10**(4), 979–1000 (2011)
26. Qiu, J.-M., Shu, C.-W.: Positivity preserving semi-Lagrangian discontinuous Galerkin formulation: theoretical analysis and application to the Vlasov–Poisson system. J. Comput. Phys. **230**(23), 8386–8409 (2011)
27. Remacle, J.-F., Flaherty, J., Shephard, M.: An adaptive discontinuous Galerkin technique with an orthogonal basis applied to compressible flow problems. SIAM Rev. **45**, 53–72 (2003)
28. Rossmanith, J.A., Seal, D.C.: A positivity-preserving high-order semi-Lagrangian discontinuous Galerkin scheme for the Vlasov–Poisson equations. J. Comput. Phys. **230**, 6203–6232 (2011)
29. Schaeffer, J.: Global existence of smooth solutions to the Vlasov Poisson system in three dimensions. Commun. Part. Differ. Equ. **16**(8–9), 1313–1335 (1991)
30. Shen, C., Qiu, J.-M., Christlieb, A.: Adaptive mesh refinement based on high order finite difference WENO scheme for multi-scale simulations. J. Comput. Phys. **230**(10), 3780–3802 (2011)
31. Shu, C.-W., Osher, S.: Efficient implementation of essentially non-oscillatory shock-capturing schemes. J. Comput. Phys. **77**, 439–471 (1988)
32. Sonnendrücker, E., Roche, J., Bertrand, P., Ghizzo, A.: The semi-Lagrangian method for the numerical resolution of the Vlasov equation. J. Comput. Phys. **149**(2), 201–220 (1999)
33. Xiong, T., Qiu, J.-M., Xu, Z., Christlieb, A.: High order maximum principle preserving semi-Lagrangian finite difference WENO schemes for the Vlasov equation. J. Comput. Phys. **273**, 618–639 (2014)
34. Zaki, S., Boyd, T., Gardner, L.: A finite element code for the simulation of one-dimensional Vlasov plasmas. II. Applications. J. Comput. Phys. **79**, 200–208 (1988)
35. Zaki, S., Gardner, L., Boyd, T.: A finite element code for the simulation of one-dimensional Vlasov plasmas. I. Theory. J. Comput. Phys. **79**, 184–199 (1988)
36. Zhang, X., Shu, C.-W.: On maximum-principle-satisfying high order schemes for scalar conservation laws. J. Comput. Phys. **229**, 3091–3120 (2010)
37. Zhou, T., Guo, Y., Shu, C.-W.: Numerical study on Landau damping. Phys. D **157**(4), 322–333 (2001)
38. Zhu, H., Qiu, J.: Adaptive Runge–Kutta discontinuous Galerkin methods using different indicators: one-dimensional case. J. Comput. Phys. **228**, 6957–6976 (2009)
39. Zhu, H., Qiu, J.: An h-adaptive RKDG method with troubled-cell indicator for two-dimensional hyperbolic conservation laws. Adv. Comput. Math. **39**, 445–463 (2013)