Computer Physics Communications 315 (2025) 109743

Contents lists available at ScienceDirect

Computer Physics Communications

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

Computational Physics

Efficient computation of particle-fluid and particle-particle interactions in compressible flow

Anna Schwarz [a,*,1], Patrick Kopper [a,*,1], Emilian de Staercke [b], Andrea Beck [a]

[a] University of Stuttgart, Institute of Aerodynamics and Gas Dynamics, Wankelstr. 3, Stuttgart, 70563, Germany
[b] École Centrale de Lyon, CNRS, Universite Claude Bernard Lyon 1, INSA Lyon, LMFA, UMR5509, 69130, Ecully, France

ARTICLE INFO

The review of this paper was arranged by Prof. Peter Vincent

Keywords:
High-order
Discontinuous Galerkin
High-performance computing
Large eddy simulation
Lagrangian particle tracking
Particle collisions

ABSTRACT

Particle collisions are the primary mechanism of inter-particle momentum and energy exchange for dense particle-laden flow. Accurate approximation of this collision operator in four-way coupled Euler–Lagrange approaches remains challenging due to the associated computational cost. Adopting a deterministic collision model and a hard-sphere (binary collision) approach eases time step constraints but imposes non-locality on distributed memory architectures, necessitating the inclusion of collision partners from each grid element in the vicinity. Retaining high-order accuracy and parallel efficiency also ties into the correct and compact treatment of the particle-fluid coupling, where adequate kernels are required to effectively project the momentum and thermal energy exchange terms of the particles to the Eulerian grid. In this work, we present an efficient particle collision and projection operator based on an MPI+MPI hybrid approach to enable time-resolved and high-order accurate simulations of compressible, four-way coupled particle-laden flows at dense concentrations. A distinct feature of the proposed particle collision algorithm is the efficient calculation of exact binary inter-particle collisions on arbitrary core counts by facilitating intranode data exchange through direct load/store operations and internode communication using one-sided communication. Combining the particle operator with a hybrid discretization operator based on a high-order discontinuous Galerkin method and a localized low-order finite volume operator allows an accurate treatment of highly compressible particle-laden flows. The approach is extensively validated against a range of benchmark problems. Contrary to literature, the scaling properties are demonstrated on state-of-the-art high performance computing systems, encompassing one-way to four-way coupled simulations. Finally, the proposed algorithm is compatible with unstructured, curved high-order grids which permits the handling of complex geometries as is emphasized by application of the framework to large-scale application cases.

1. Introduction

Compressible flow with suspended solid particles is present in a wide range of technical applications, ranging from spray injection to aerospace engineering [1,2]. Most of these applications are challenging to simulate due to the strongly non-linear, multi-scale, and multi-physics nature of the problems. These challenges have led to the development of numerous modeling approaches over the last years, aimed at enabling high-fidelity simulations of such particle-laden flows. One prominent approach is the Euler–Lagrange or point particle method, in which the particles are described as discrete points. The particles evolve in a Lagrangian manner according to Newton's second law of motion, while the continuous phase is given in Eulerian frame of reference. In contrast to the Euler–Euler method, the Euler–Lagrange method can handle numerous particles with acceptable accuracy while avoiding the enormous computational cost associated with particle-resolved strategies [1].

The modeling of the forces acting on and exerted by the particles in the Euler–Lagrange approach is strongly tied to the volume fraction of the particles in the continuous phase [3]. In the most general sense, particle-laden flow can be classified into three categories: very dilute, dilute, and dense suspensions, ordered by increasing volume fraction. For very dilute flow, the influence of the particulate phase on the continuous phase can be omitted. Moreover, the mean-free path between the particles is large, such that the probability for inter-particle collisions is negligible for most practical applications. The resulting interaction in which only the influence of the fluid forces on the particulate phase are

* Corresponding authors.
E-mail addresses: schwarz@iag.uni-stuttgart.de (A. Schwarz), kopper@iag.uni-stuttgart.de (P. Kopper).
[1] P. Kopper and A. Schwarz share first authorship.

https://doi.org/10.1016/j.cpc.2025.109743
Received 9 January 2025; Received in revised form 10 April 2025; Accepted 2 July 2025
Available online 11 July 2025

considered is called one-way coupling. For a dilute flow regime, it may be necessary to consider the back scattering of the particulate phase onto the continuous flow field, leading to a two-way coupled approach. For a further increase in volume fraction, resulting in dense particle-laden flow, the mean distance of the particles decreases such that inter-particle collisions have to be taken into account. The resulting four-way coupling leads to the highest fidelity Euler–Lagrange approach. However, the identification of particle collisions comes at extensive computational cost. Furthermore, for a particle volume fraction approaching unity, the flow is purely collision dominated. In this case, the point particle method cannot be applied, and more sophisticated computational approaches, such as discrete element methods, must be employed. In this paper, we restrict ourselves to applications with dense suspensions below the granular flow regime with a strong influence of the interstitial fluid. Consequently, these applications necessitate the adoption of a four-way coupled approach. We elect to represent the inter-particle collisions using a deterministic collision model and the hard-sphere approach (binary collisions) to relax the time step restriction compared to its counterpart, the soft-sphere method, where particle collisions are resolved in time [4].

As noted above, high-fidelity simulations of four-way coupled particle-laden flows come at considerable computational cost, much of which is due to the collision operator. The main bottleneck of particle collision algorithms is the search for pairwise collision partners, generally performed via a variant of the nearest-neighbor search approach. The computational effort is linked to the number of eligible collision partners and rises with increasing volume fraction, resulting in numerous collisions per time step and thus a multitude of possible particle pairs. Hence, several authors proposed optimizations of the original nearest-neighbor search algorithm to efficiently detect potential particle collision pairs, with comprehensive overviews given by Sigurgeirsson et al. [5] and Ching and Ihme [6]. These optimized algorithms incorporate approaches to facilitate the early rejection of ineligible particle pairs and may be broadly classified into particle neighbor lists and cell-based/element neighbor lists approaches. Particle neighbor lists approaches attach a distance-sorted list of the surrounding particles to each particle in order to discard particle pairs above a given distance threshold. As these lists contain a considerable amount of redundant information for particles in close proximity, this approach is inherently memory-intensive. Concurrently, there is no intrinsic mechanism to prompt an update of the particle neighbor lists, which may result in inaccurate omission of particle pairs. The cell-based neighbor lists approach addresses these deficiencies by first mapping each particle either to its computational mesh element or to an auxiliary Cartesian grid of bins, also denoted as virtual cell approach or bin neighbor lists [7,8], depending on the cell size chosen. Both methods have in common that only particles in the same and the adjacent grid cells (node sharing) are considered in the nearest neighbor search. Cell sizes for the bin neighbor lists are typically chosen equal to the diameter of the largest particle while the virtual cell approach permits arbitrary sizes. This method has the main drawback that the bins are usually distributed uniformly throughout the domain. The result is an increased computational overhead for applications with strongly differing and non-uniform element sizes such as channel/pipe flows or flows around airfoils, as considered in this paper. This drawback is exacerbated if the particles exhibit velocities across several order of magnitude, rendering this approach typically memory-intensive and computational expensive [6]. In addition, the optimal bin size for a hard-sphere approach remains an open research question as the use of larger cells incurs unwarranted additional inspections, while smaller bins unnecessarily limit the time step size [5]. In comparison, the element-based neighbor list method offers the advantage of being inherently suitable for the efficient collision search on meshes with strongly non-uniform and differently sized element shapes. The omission of a virtual grid implies that in a parallel setting only the computational mesh needs to be efficiently mapped to the processors, whereas in the bin- and virtual-cell approaches the auxiliary Cartesian background grid also has to be mapped to the processors. More recent approaches aim to combine the favorable aspects of the particle neighbor lists and cell-based neighbor lists approaches. Yao et al. [9] accelerated the construction of the neighbor list by combining virtual cells with a particle neighbor lists approach, which, however, can be highly memory-intensive. Breuer and Alletto [10] refined the cell-based neighbor lists and eliminated the necessity to take the surrounding cells into account by employing two virtual Cartesian grids of different size. This comes at the cost of increased computational overhead, especially in a parallel context. Subsequent publications by Krijgsman, Ogarko, and Luding [11,12] aimed to determine the optimal hierarchical cell space for multi-level virtual cell-based neighbor list approaches, focusing mainly on soft-sphere methods. In the context of the bin-based neighbor list approach, an earlier approach by Sigurgeirsson et al. [5] resulted in improved efficiency for hard-sphere collisions, but with the aforementioned drawbacks of the bin-based methods. A more recent approach was proposed by Ching and Ihme [6], who presented a four-way coupled Euler–Lagrange approach using the high-order discontinuous Galerkin (DG) method. The authors aimed to develop an efficient variant of the element-neighbor-list by constructing an element proximity list in reference space. This list is built after mapping the particle positions from physical space to the unit cube in reference space and subsequent truncation of the number of eligible adjacent elements. To increase the efficiency, the number of neighboring elements used for the particle pair search is restricted depending on the position of the particle within the element. This can neglect possible particle collisions, particularly for very distinct particle velocities and trajectories. Finally, it is worth mentioning that recent works speed up the particle neighbor search by using GPU acceleration [8,13]. All in all, this renders the element-based neighbor list method (element-binning approach) as the appropriate choice for this work.

The main objective of this paper is to present computationally efficient algorithms for four-way coupled Euler–Lagrange simulations using an element-based neighbor list approach. As accuracy and compactness of projection kernels for particle-fluid coupling is extensively covered in literature, see [14] for a more comprehensive overview, we place the focus for particle-fluid coupling on addressing parallel efficiency, especially in the context of high performance computing (HPC). The proposed inter-particle collision operators build on this parallelization aspect, enabling highly efficient and accurate calculation of binary inter-particle collisions in combination with particle-fluid coupling on arbitrary core counts. The particular features of this particle collision algorithm compared to the previously presented methods can be summarized as follows. First, the efficiency and HPC suitability of the proposed algorithm is achieved via the combination of the particle operator with an MPI+MPI hybrid approach [15–17]. On multinode computations, the novel collision algorithm includes all particles which are in the halo region of its element in the particle collision pair search without a significant increase in computational work or memory pressure. Potential load imbalances are handled via a dynamic load balancing procedure. The proposed algorithm is code-agnostic, which facilitates integration into any Euler–Lagrange framework. Second, the new inter-particle collision algorithm is implemented in the high-order Euler–Lagrange framework ∃LEXI[2] [18] (so far only two-way coupling). In ∃LEXI, the carrier phase is discretized by a hybrid discretization operator based on a high-order accurate discontinuous Galerkin Spectral Element Method (DGSEM) and a localized low-order finite volume operator, while a Lagrangian approach is employed for the discrete phase. As such, the proposed algorithm ties into the existing capabilities of ∃LEXI of handling complex geometries on unstructured grids featuring boundary conditions, possibly curved elements, and hanging nodes. In combination, these features enable an efficient and highly accurate treatment of inter-particle collisions in a compressible carrier phase on arbitrary core counts which is

[2] https://github.com/flexi-framework/elexi.

demonstrated by its excellent scaling properties and efficient memory utilization on massively parallelized systems. Finally, the ∃LEXI framework is to the author's knowledge the first open-source solver for one- to four-way coupled compressible particle-laden flows in complex geometries using the high-order DG method.

This primary focus of this work is on the implementation, the parallelization challenges and the application of discrete particles in dense suspension within a continuous compressible flow field. In section 2, the underlying equations for both the continuous and discrete phase are presented, including the fluid-particle coupling, particle-wall interactions, and inter-particle collisions. This is followed by a brief outline of the numerical treatment of these equations in section 3. The parallelization strategies for the collision and projection operator are detailed in section 4 and validated in section 5, followed by a demonstration of the parallel performance in section 6. The capabilities of our four-way coupled Euler–Lagrange approach are demonstrated in section 7 using two challenging real-world applications. This paper closes with a brief discussion in section 8.

2. Theory

2.1. Continuous phase

The fluid field is governed by the compressible unsteady Navier–Stokes–Fourier equations, given in vectorial form as

$$\frac{d\mathbf{q}}{dt} + \nabla \cdot \mathbf{F}(\mathbf{q}, \nabla \mathbf{q}) = \mathcal{S}, \tag{1}$$

where $\mathbf{q} = [\rho, \rho u_1, \rho u_2, \rho u_3, \rho e]^T$ is the vector comprising the conservative variables with ρ as the fluid density, u_i the i-th component of the velocity vector and e the total energy per unit mass. The source term $\mathcal{S}$ accounts for the influence of the dispersed phase on the fluid in two- or four-way coupled regimes. The physical flux $\mathbf{F}$ is composed of the inviscid Euler and the viscous fluxes. The equation system is closed by the equation of state of a calorically perfect gas. The dynamic viscosity μ is obtained from Sutherland's law [19], while the heat flux is given by Fourier's law. Following Stokes' hypothesis, the bulk viscosity is set to zero.

2.2. Dispersed phase

According to the Lagrangian point particle approach, particles are treated as discrete points with mass m_p and diameter d_p which move in a Lagrangian manner according to the following system of ordinary differential equations (ODEs)

$$\frac{d\mathbf{x}_p}{dt} = \mathbf{v}_p, \tag{2}$$

$$m_p \frac{d\mathbf{v}_p}{dt} = \mathbf{F} = 3\pi\mu d_p f_D \left(\mathbf{u}_f - \mathbf{v}_p\right), \tag{3}$$

$$I \frac{d\boldsymbol{\omega}_p}{dt} = \mathbf{M} = \rho_f \frac{d_p^5}{64} C_w \boldsymbol{\Omega}|\boldsymbol{\Omega}|, \tag{4}$$

$$m_p c_p \frac{dT_p}{dt} = Q_p = \pi d_p \kappa_c (T_f - T_p)\mathrm{Nu}, \tag{5}$$

with the particle position in physical space $\mathbf{x}_p = [x_{p,1}, x_{p,2}, x_{p,3}]^T$ and the particle velocity $\mathbf{v}_p$ obtained from the integration of the first two ODEs. Equation (3) is approximated by the Maxey–Riley–Gatignol (MRG) equation [20,21], considering only the drag force and applying an empirical correction for higher particle Reynolds numbers. The drag factor f_D is computed according to Loth et al. [22]. The angular particle velocity is $\boldsymbol{\omega}_p = \nabla \times \mathbf{v}_p$, $I = \frac{\pi}{60}\rho_p d_p^5$ is the moment of inertia of a spherical particle, $\boldsymbol{\Omega} = \frac{1}{2}(\nabla \times \mathbf{u}_f) - \boldsymbol{\omega}_p$ is the relative fluid-particle angular velocity, $\mathbf{M}$ is the torque and C_w is a correction factor for higher Reynolds numbers proposed by Dennis et al. [23]. Equation (5) describes the change of the particle temperature T_p, where Q_p is the heat transfer term, Nu the Nusselt number, c_p the specific heat of the particle, and κ_c the thermal conductivity of the continuous phase.

2.3. Fluid-particle coupling

For two- and four-way coupled flow, the influence of the particulate on the continuous phase is modeled using the particle-source-in-cell approach proposed by Crowe et al. [24]. In this approach, the forces acting on the particles and the corresponding work appear as a source term, $\mathcal{S} = [0, \mathcal{S}_{m,1}, \mathcal{S}_{m,2}, \mathcal{S}_{m,3}, \mathcal{S}_e]$ in the momentum equations and the energy equation, respectively. The source terms for the momentum equations $\mathcal{S}_m = [\mathcal{S}_{m,1}, \mathcal{S}_{m,2}, \mathcal{S}_{m,3}]$ and the energy equation $\mathcal{S}_e$ at a point $\mathbf{x}_{ijk},\ i,j,k \in \mathbb{N}_{>0}$ are given by

$$\mathcal{S}_m = -\mathcal{P}\left\{\mathbf{F}, \mathbf{x}_{ijk}\right\}, \tag{6}$$

$$\mathcal{S}_e = -\mathcal{P}\Big\{\mathcal{S}_m \cdot \mathbf{v}_p + Q_p, \mathbf{x}_{ijk}\Big\}, \tag{7}$$

with the projection operator $\mathcal{P}\{\cdot,\cdot\}$, which projects the source term onto the grid. Consequently, this approach is also referred to as particle deposition, a term which should not be confused with the physical deposition of particles. Within this work, the influence of the source term is linearly imposed onto the corner nodes of the host element, as a C_0-continuous version of the inverse distance weighting (IDW) interpolation [25,26]. Given the eight corner nodes of a hexahedral element $\{\boldsymbol{\xi}_n | \boldsymbol{\xi} \in [-1,1]^3\}_{n=1}^8$ and the degrees of freedom (DOF) $\{\boldsymbol{\xi}_{ijk}\}_{i,j,k=1}^{\mathcal{N}}$, the IDW interpolation function for an arbitrary variable $a \in \mathbb{R}$ and a single particle results in

$$\mathcal{P}\left\{a, \mathbf{x}_{ijk}\right\} = \frac{J_{ijk} w_\xi(\boldsymbol{\xi}_{ijk}) w_\xi(\boldsymbol{\xi}_p)\, a}{\sum_{n=1}^8 J_n w_n} \tag{8}$$

with $w_\xi(\boldsymbol{\xi}) = \sum_{n=1}^8 \prod_{d=1}^3 (\xi_n^d - \xi^d)/2$ and the particle position in reference space $\boldsymbol{\xi}_p \in [-1,1]^3$. Here, $w_n = w(\boldsymbol{\xi}_n)$ is the weight defined at the corner node n and J_n is the corresponding Jacobian such that the product of both denotes the volume spanned by node n. Different projection operators such as a Dirac delta function are available in the code should the user prefer another choice for the projection operator $\mathcal{P}$.

2.4. Inter-particle and particle-wall interactions

In the following, we briefly describe the physical model for inter-particle and particle-wall collisions.

2.4.1. Inter-particle collisions

In the hard-sphere approach, only binary collisions between particles are considered and particle deformations are neglected. The jump relations describing the change in momentum are given by Crowe et al. [4] as

$$m_{p,1}(\mathbf{v}_{p,1} - \mathbf{v}_{p,1}^{(0)}) = \mathbf{J}, \tag{9}$$

$$m_{p,2}(\mathbf{v}_{p,2} - \mathbf{v}_{p,2}^{(0)}) = -\mathbf{J}, \tag{10}$$

$$I_{p,1}(\boldsymbol{\omega}_{p,1} - \boldsymbol{\omega}_{p,1}^{(0)}) = d_{p,1}/2(\mathbf{n}_p \times \mathbf{J}), \tag{11}$$

$$I_{p,2}(\boldsymbol{\omega}_{p,2} - \boldsymbol{\omega}_{p,2}^{(0)}) = d_{p,2}/2(\mathbf{n}_p \times \mathbf{J}), \tag{12}$$

where $\mathbf{J}$ is the unknown impulsive force, and $\mathbf{n}_p$ the unit vector from particle 1 to particle 2. Variables with the superscript $(\cdot)^0$ are pre-collision quantities, while no superscript denotes the unknown post-collision quantities. Following [4], the impulsive force, given as

$$\mathbf{J} = J_n \mathbf{n}_p + J_t \mathbf{t}_p, \tag{13}$$

$$J_n = -m_{p,r}(1 + e_n)(\mathbf{v}_{p,r}^{(0)} \cdot \mathbf{n}_p) < 0, \tag{14}$$

$$J_t = \mu_f J_n < 0, \tag{15}$$

is computed based on the relative particle motion, $\mathbf{v}_{p,r} = \mathbf{v}_{p,1} - \mathbf{v}_{p,2}$, the relative mass, $m_{p,r} = \frac{m_{p,1} m_{p,2}}{m_{p,1}+m_{p,2}}$, and two parameters, the normal coefficient of restitution e_n and Coulomb's law of friction with the friction coefficient μ_f. The tangential component of the relative velocity of the contact point,

$$\mathbf{v}_{p,r,t} = \mathbf{v}_{p,c}^{(0)} - (\mathbf{v}_{p,c}^{(0)} \cdot \mathbf{n}_p)\mathbf{n}_p, \tag{16}$$

$$\mathbf{v}_{p,c}^{(0)} = \mathbf{v}_{p,r}^{(0)} + d_{p,1}/2\ \boldsymbol{\omega}_{p,1}^{(0)} \times \mathbf{n}_p + d_{p,2}/2\ \boldsymbol{\omega}_{p,2}^{(0)} \times \mathbf{n}_p, \tag{17}$$

determines if the particles are sliding (first condition) or sticking (second condition) which then leads to

$$J_t = \max\left(\mu_f J_n, -\frac{2}{7}(1+e_n) m_{p,r} \left|\mathbf{v}_{p,r,t}^{(0)}\right|\right) \tag{18}$$

and the tangential vector given as $\mathbf{t}_p = \mathbf{v}_{p,r,t}^{(0)} \left|\mathbf{v}_{p,r,t}^{(0)}\right|^{-1}$.

2.4.2. Intersection of particles with solid walls

Particle-wall collisions are also modeled via the hard-sphere approach, i.e., they are not resolved in time but handled in an a posteriori manner. Thus, the time step dt_{stage} of the current Runge–Kutta stage has to be greater than the time a particle requires to collide with a wall ($dt_{\text{stage}} > dt_{\text{coll}}$). The change in particle momentum and angular particle momentum are modeled according to Crowe et al. [4] as

$$m_{p,2}(\mathbf{v}_{p,2} + 2(\mathbf{v}_{p,1} \cdot \mathbf{n})\mathbf{n}) - m_{p,1}\mathbf{v}_{p,1} = \mathbf{J}, \tag{19}$$

$$I_2\boldsymbol{\omega}_2 - I_1\boldsymbol{\omega}_1 = -\frac{d_{p,1}}{2}\left[\mathbf{n} \times (m_{p,2}\mathbf{v}_{p,2} - m_{p,1}\mathbf{v}_{p,1})\right], \tag{20}$$

where $(\cdot)_2$ are quantities after the impact and $(\cdot)_1$ before it. The change in momentum is designated as $\mathbf{J}$, and $\mathbf{n}$ is the normal vector of the boundary. For a perfectly reflective wall, i.e., an elastic collision, $\mathbf{J} = 0$, while for a plastic deformation $\mathbf{J} \neq 0$. Here, only the former is considered, the reader is referred to, e.g., [27,28] for further details on plastic particle-wall collisions.

3. Numerical methods

In the following, we briefly discuss the numerical treatment of the governing equations for the fluid and dispersed phase.

3.1. Discontinuous Galerkin spectral element method

The Navier–Stokes–Fourier equations are solved via the Discontinuous Galerkin Spectral Element Method (DGSEM), where the computational domain $\Omega \subseteq \mathbb{R}^3$ is discretized by non-overlapping, (non-)conforming hexahedral elements with six possibly curved element faces. Subsequently, the considered governing equations are transformed into the reference coordinate system $\boldsymbol{\xi} = [\xi^1, \xi^2, \xi^3]^T$ of the reference element $E = [-1,1]^3$ via the mapping $\mathbf{x} = \chi(\boldsymbol{\xi}, t)$, $\mathbf{x} \in \Omega$. The weak form is retrieved by a discrete L_2 projection of the governing equation in reference space onto the test space composed of polynomials $\boldsymbol{\phi}(\boldsymbol{\xi})$ up to degree $\mathcal{N}$, followed by an application of Gauss' theorem, yielding

$$\int_E J \frac{\partial \mathbf{q}_h}{\partial t} \boldsymbol{\phi}(\boldsymbol{\xi})\, d\boldsymbol{\xi} + \int_{\partial E} (\mathcal{F} \cdot \mathbf{n})^* \boldsymbol{\phi}(\boldsymbol{\xi})\, d\mathbf{S} - \int_E \mathcal{F}(\mathbf{q}_h, \nabla \mathbf{q}_h) \cdot \nabla_{\boldsymbol{\xi}} \boldsymbol{\phi}(\boldsymbol{\xi})\, d\boldsymbol{\xi} = 0, \tag{21}$$

with the Jacobian J of the mapping, the outward pointing normal vector $\mathbf{n}$, the contravariant flux vector $\mathcal{F}$ and the numerical flux normal to the element face $(\mathcal{F} \cdot \mathbf{n})^*$. To obtain an efficient discretization, the element-local solution $\mathbf{q}_h = \mathbf{q}_h(\boldsymbol{\xi}, t)$ is approximated by a tensor product of one-dimensional nodal Lagrange basis functions l of degree $\mathcal{N}$

$$\mathbf{q}_h(\boldsymbol{\xi}, t) = \sum_{i,j,k=0}^{\mathcal{N}} \hat{\mathbf{q}}_{ijk}(t) l_i(\xi^1) l_j(\xi^2) l_k(\xi^3), \tag{22}$$

with the nodal degrees of freedom $\hat{\mathbf{q}}_{ijk}(t)$. For the numerical integration of eq. (21), the Legendre–Gauss quadrature with $(\mathcal{N}+1)^3$ Legendre–Gauss points is employed. If not stated otherwise, the numerical flux is approximated via the approximate Riemann solver by Roe [29], with the entropy fix by Harten and Hyman [30]. The viscous fluxes are computed with the BR1 scheme [31]. To mitigate aliasing errors, the flux is split according to Pirozzoli [32]. For non-conforming element interfaces with hanging nodes, the flux is calculated with the conservative and high-order accurate mortar technique by Mavriplis [33]. The shock capturing procedure is based on a second-order accurate finite volume (FV) (subcell) scheme with $(\mathcal{N}+1)^3$ integral means per DG element [34]. Following the method of lines approach, the explicit low-storage fourth-order accurate Runge–Kutta (RK) scheme by Carpenter and Kennedy [35] is employed for the integration in time. The reader is referred to [34,36–38] for further details on DGSEM and applications. The methods presented in this work are implemented in the open-source framework ∃LEXI.

3.2. Particle localization and time integration

In contrast to the aforementioned fluid phase, the dispersed particulate phase is tracked in physical space. While tracking approaches in reference space are reported to be more robust due to their natural localization within each element, these methods often do not inherently consider boundary conditions [39]. ∃LEXI elects to follow the approach described in Ortwein et al. [39] by tracking particle-face intersections in physical space following methods known from ray tracing in computer graphics. Neglecting inter-particle collision, the particle motion is thus computed using the following four steps.

3.2.1. Emission and localization

Particles are emitted in parallel at physical locations determined from predefined spatial distribution functions. The Eulerian (computational) grid is fully unstructured and provides no direct relation between the physical position of a grid cell and the corresponding cell index. Therefore, the particle host element is located with the help of a Cartesian background mesh in physical space where each background mesh element contains a mapping to each overlapping computational mesh element [40]. This approach reduces the search space, improving localization performance. For each overlapping element, the location of the particle in reference space is determined by finding the root of

$$\mathbf{x}_p - \chi(\boldsymbol{\xi}_p) = 0 \tag{23}$$

via Newton's method. The particle host element is found once $\boldsymbol{\xi}_p$ satisfies $\boldsymbol{\xi}_p \in [-1,1]^3$.

3.2.2. Field evaluation

The fluid field at the particle's center of mass is calculated from a straightforward evaluation of the DG polynomials, thus

$$\mathbf{q}_h(\boldsymbol{\xi}_p, t) = \sum_{i,j,k=0}^{\mathcal{N}} \hat{\mathbf{q}}_{ijk}(t) l_i(\xi_p^1) l_j(\xi_p^2) l_k(\xi_p^3). \tag{24}$$

The particle interpolation reduces to a linear interpolation of the conserved variables to the particle position for the second-order FV subcells scheme employed for shock capturing, see section 3.1.

3.2.3. Time integration

Particles are integrated in time using linear segments such that the particle path within a single Runge–Kutta (RK) stage, i.e., $t \in [t^n, t^{n+1}]$, is described by

$$\mathbf{x}_p(t;\alpha) = \mathbf{x}_p(t^n) + \alpha \frac{\mathbf{t}}{|\mathbf{t}|}, \qquad \alpha \in [0, |\mathbf{t}|], \tag{25}$$

$$\mathbf{t} = \mathbf{x}_p(t^{n+1}) - \mathbf{x}_p(t^n). \tag{26}$$

Here, $\mathbf{t}$ describes the linear path segment while α is the relative displacement along $\mathbf{t}$.

3.2.4. Face intersections

The linear path segments from eqs. (25) and (26) need to be checked for face intersections to determine updates to the particle host element and/or applications of boundary conditions. For curvilinear sides, the intersection location is performed using a dimension reduction approach based on Bézier clipping following the de Casteljau subdivision approach. Each element side is mapped onto a Bézier polynomial surface while preserving the shape of the element side, yielding a representation as

$$\mathbf{P}(\xi,\eta) = \sum_{m=0}^{\mathcal{N}_{geo}} \sum_{n=0}^{\mathcal{N}_{geo}} \hat{\mathbf{P}}_{mn} \mathcal{B}_m(\xi) \mathcal{B}_n(\eta). \tag{27}$$

Here, $\mathcal{N}_{geo}$ describes the polynomial degree of the mapping, $\hat{\mathbf{P}}$ are the Bézier control points and $(\xi,\eta) \in [-1,1]^2$ the side coordinates in reference space. With this formulation, a face intersection is found if and only if the root of

$$\mathbf{x}_p(t,a) = \mathbf{x}_p(t^n) + \alpha \frac{\mathbf{t}}{|\mathbf{t}|} \overset{!}{=} \mathbf{P}(\xi,\eta) \ni \begin{matrix} \mathrm{t} & \in (t^n, t^{n+1}), \\ (\xi,\eta) & \in [-1,1]^2 \end{matrix} \tag{28}$$

exists. Alternatively, simpler approaches are applied if a face is detected to collapse to a linear or bi-linear side. Intersections with non-conforming element interfaces are always evaluated on the small element faces to ensure watertightness [18]. For more details on the particle tracking approach, see Ortwein et al. [39].

3.3. Particle collisions

To check for the possibility of particle collision between two consecutive Runge–Kutta stages t^n and t^{n+1}, an a posteriori approach is chosen. Here, particle collisions are checked after the particles moved from t^n to t^{n+1}. Hence, it is again assumed that the particle movement in one RK stage is according to a straight line. Two particles collide if there exists a time span $dt_{\mathrm{coll}} > 0$ for which $dt_{\mathrm{coll}} \leq dt_{\mathrm{stage}} \in [0; t^{n+1}]$, i.e., if the following quadratic equation

$$\left|\Delta x_p + dt_{\mathrm{coll}} \Delta v_p\right|^2 = \frac{(d_{p,1} + d_{p,2})^2}{4}, \tag{29}$$

$$\Delta x_p = \mathbf{x}_2(t^n) - \mathbf{x}_1(t^n), \ \Delta v_p = \mathbf{v}_{p,2}(t^n) - \mathbf{v}_{p,1}(t^n) \tag{30}$$

has at least one solution for which $dt_{\mathrm{coll}} \leq dt_{\mathrm{stage}}$. If there exist two positive solutions of eqs. (29) and (30) and the lower dt_{coll} fulfills the condition $dt_{\mathrm{coll}} \leq dt_{\mathrm{stage}}$, then the collision time is given as $t_{\mathrm{coll}} = t^n + dt_{\mathrm{coll}}$, otherwise no collision occurs in the current RK stage. Finally, the collision is only considered valid if the following two conditions hold: First, the collision occurred before an intersection with a domain boundary. Second, the particles approach each other.

After collision pairs are detected, the particle velocities $\mathbf{v}_p(t^{coll})$ after the collision are determined as described in section 2.4.1 and the particle positions are updated accordingly, thus

$$\mathbf{v}_{p,1}(t^{coll}) = \frac{\mathbf{J}}{m_{p,1}} + \mathbf{v}_{p,1}^{(0)}, \tag{31}$$

$$\mathbf{v}_{p,2}(t^{coll}) = -\frac{\mathbf{J}}{m_{p,2}} + \mathbf{v}_{p,2}^{(0)}, \tag{32}$$

$$\mathbf{x}_p(t^{coll}) = \mathbf{x}_p^{(0)} + (dt_{\mathrm{stage}} - dt_{\mathrm{coll}}) \mathbf{v}_p(t^{coll}). \tag{33}$$

4. Implementation details

The following chapter details the implementation choices for the code parallelization of the four-way coupled framework detailing the most critical parts for parallel performance, the particle deposition and collision operators. It concludes with a brief discussion of the load balancing strategy. For further details the reader is referred to Kopper et al. [18].

4.1. Parallelization

ƎLEXI is parallelized using pure Message Passing Interface (MPI) communication [41]. Mesh elements are pre-sorted along a space-filling curve (SFC) by the open-source mesh generator High-Order PreProcessor HOPR[3] [42]. Solution data for the continuous phase is structured to form linear memory segments for communication and is stored using the distributed memory paradigm. As the computational stencil of the DG method used for the continuous phase is element-local, solely the numerical flux is exchanged between individual elements using non-blocking communication. At the same time, intra-element computation is utilized to enable efficient latency hiding.

As the discrete phase is allowed to move arbitrarily between the mesh elements, the processor-local geometry information is enriched by the spatially surrounding elements to create a halo region. An efficient communication-free two-step search algorithm determines the eligible elements [18]. This halo region is stored on each individual compute node using the MPI-3 shared memory (SHM) paradigm. Thus, each processor has direct access to the complete mesh geometry for particle tracking and can determine the final particle position while incorporating boundary conditions. Only once the final particle position is known, the particles are sent to the new host processor using non-blocking MPI communication. ƎLEXI attempts to finish the particle tracking as early as possible to hide the communication latency behind the local computational load from the continuous phase.

4.2. Deposition

If the particle volume fraction exceeds the limit of dilute dispersed flow, the effect of the dispersed particles on the continuous fluid must be considered. This is achieved through deposition of the forces acting on the particles as a corresponding work on the continuous phase with the sign reversed, thus appearing as source terms in the Navier–Stokes–Fourier equations, see eq. (1). As the particle positions generally do not coincide with a fluid DOF, ƎLEXI features several mechanisms to conservatively project the source terms onto the fluid solution. The most straightforward approach assigns the complete source term to the nearest DOF (Dirac delta function). While this results in a highly localized deposition, sharply non-uniform particle concentrations can cause strong oscillations and nonphysical solutions in the DG polynomials. Smoothing approaches based on shape functions can alleviate this issue by extending the influence across multiple solutions points but require large communication stencils and are slightly less accurate [26]. As a consequence, the most efficient approach available in ƎLEXI implements a linear deposition approach based on IDW interpolation, see eq. (8) where the deposition radius is chosen identical to the cell size. Here, each element corner node is assigned a unique node identifier during pre-processing with HOPR while taking periodic boundary conditions into account. Since the high-order DG method utilizes a low number of elements while preserving high numerical accuracy through the arbitrary order polynomial basis, each compute node can allocate an MPI-3 SHM array sufficient to store the deposition source terms for the element corners nodes of the complete computational domain. Deposition on the corner nodes is performed locally on each compute using atomic `MPI_ACCUMULATE` calls. The source terms are summed up across the compute nodes using non-blocking `MPI_IALLREDUCE` operations once the compute node-local deposition is complete. This procedure results in a conservative deposition approach which is numerically stable, computationally efficient, and features reasonable computational accuracy.

[3] https://github.com/hopr-framework/hopr.

4.3. Collision operator

The distinct feature of the proposed collision operator is the combination of an element-based binning method with an MPI+MPI hybrid approach. The computation of the collision operator is the most expensive computational operation. As ∃LEXI calculates exact (binary) hard-sphere collisions, this requires comparisons of all particle trajectories within a given sphere around each particle position. An early reduction of the amount of potential collision partners which need to be checked is thus crucial for acceptable code performance. At the same time, accurate results require that no eligible particles are omitted, which poses a challenge during parallel runs. Defining the barycenters of two elements as $\boldsymbol{x}_{B,c}$ and $\boldsymbol{x}_{B',c}$ with the corresponding convex hull radii r and r', respectively, a particle needs to be considered if its host element is within

$$|\boldsymbol{x}_{B',c} - \boldsymbol{x}_{B,c}| = |\boldsymbol{d}| \leq d_{\max} + r + r', \tag{34}$$

see Fig. 2. Here, $d_{\max}$ is the maximum distance a particle can travel during one time step and can be calculated from the velocity of the fastest overall particle and the time increment of the explicit Runge–Kutta scheme.

Although element-based binning reduces memory requirements, retaining complete particle data on a per-task basis nevertheless results in excessive memory pressure. In order to reduce memory pressure, intranode information is stored uniquely, while a ghost layer, or halo region, is created to avoid blocking on inter-node information exchange. This halo region contains for each mesh element the list of other elements with potential collision partners, which are chosen depending on the maximum possible distance a particle can travel in one computational time step (here RK stage). ∃LEXI utilizes its MPI-3 shared memory paradigm to perform this element-based binning approach while simultaneously minimizing the amount of particle data which needs to be exchanged. During code initialization, an element-based mapping array based on the physical distance is constructed on each compute node, comprising for each mesh element the list of other elements close enough for potential collision partners. Periodic boundaries are incorporated through virtual element shifts during the distance calculation. Each list is sorted according to the global element index to ensure determinism. As the mapping is build on the global mesh information, the list automatically contains elements inside the halo region which are residing on other threads. The union of the individual lists contain all elements required for particle collisions on a given compute node.

During code execution, prior to entering a particle collision operator step, the particles are sorted according to their host element along the space-filling curve using a linked-list approach. From this sorting, both the number of particles in each element and their global index is obtained. This total number of particles on each compute node is broadcast using an `MPI_EXSCAN` operation while simultaneously starting an MPI Remote Memory Access (RMA) epoch on the particle lists and data. Fig. 3 illustrates the positioning of these operations within the time-stepping algorithm. The communication overhead is hidden behind the calculation of the remaining particle paths which iterates until each particle encounters a computational boundary. Once the communication finishes and the MPI call returns, each process can obtain the required particle data. Fig. 4 gives an overview of the performed memory operations and communication. Each compute node fetches the indices of particles inside eligible elements using non-blocking `MPI_GET` calls. With this information, the position of the particle data on each thread can be computed which is again fetched into compute node-local shared memory using non-blocking RMA operations. The data is enriched with the time of flight for each particle, measured from its current position until it reaches its next boundary intersection. This information is required for the collision step as collisions occurring outside the computational domain are invalid solutions. Once these operations are finished, the position and trajectories of each particle within the halo distance around a compute node are stored in MPI-3 shared memory on each compute node.

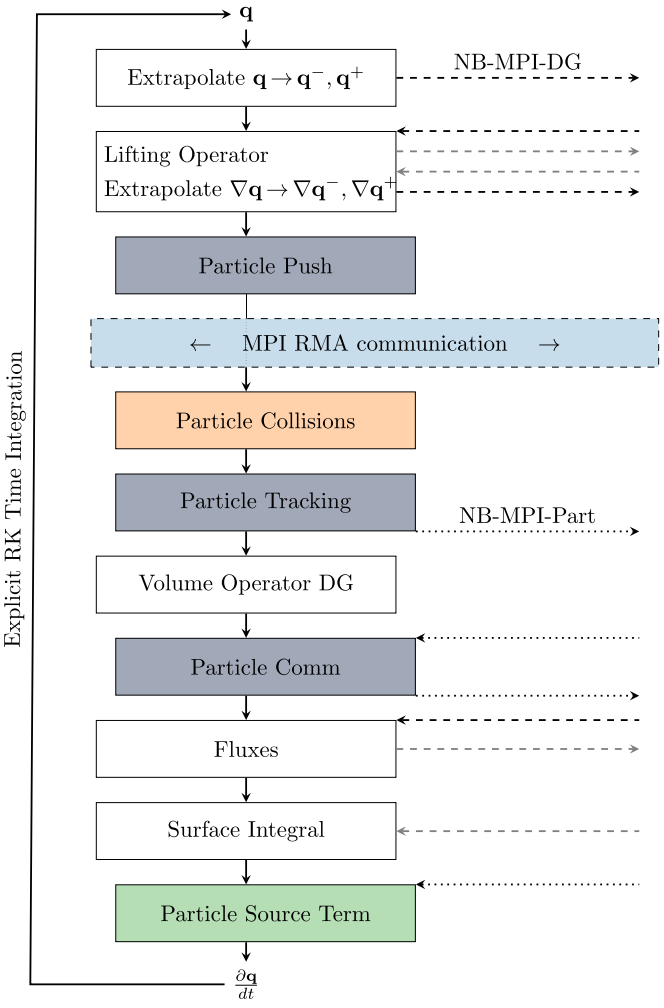

Fig. 1. Flow chart of the discontinuous Galerkin operator for 4-way coupled particle-laden flow. Dashed gray line indicate non-blocking MPI communication for the DG operator (NB-MPI-DG), dotted black lines MPI communication for the particle operator (NB-MPI-Part). DG operations are shaded in white. Particle operations are shaded in gray, with the particle collision operator highlighted in orange and the particle deposition operator in green. See Kopper et al. [18] for a detailed breakdown. The blue block represents MPI RMA operations introduced for 4-way coupling. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

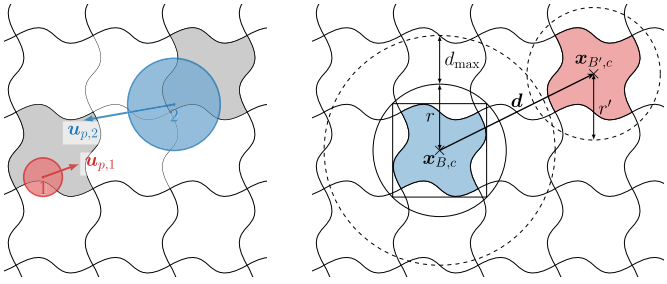

Fig. 2. Sketch of halo region determination for particle collisions. Left: Theoretical extent of the Lagrangian particles and their host elements (shaded gray). Right: Spheres of influence for possible particle-particle collisions for the same elements.

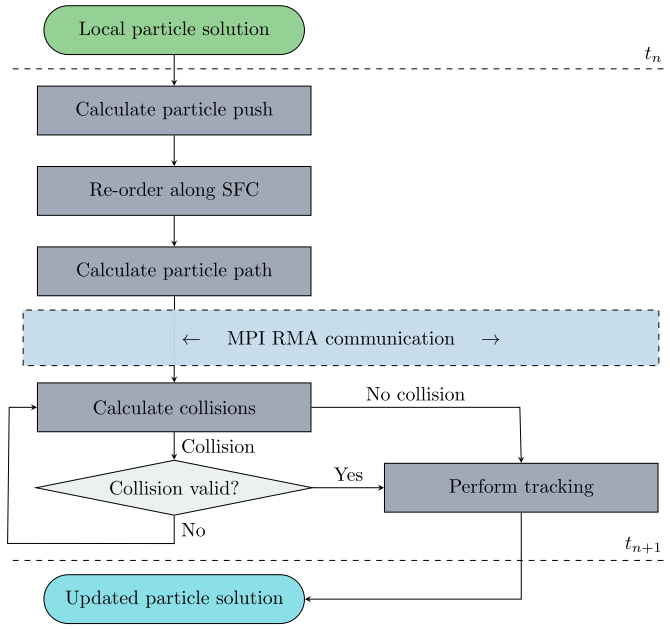

Fig. 3. Algorithm for parallel exact hard-sphere collision detection.

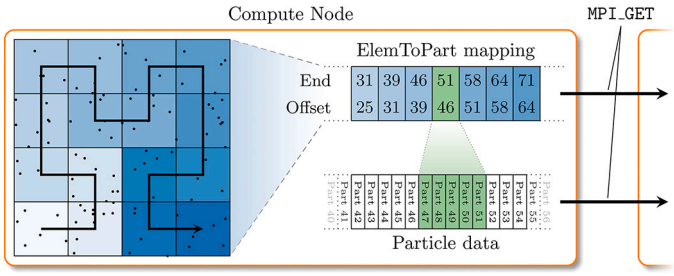

Fig. 4. Data structure and inter-node communication for the element-particle mapping and the actual particle data.

The actual collision detection is performed in parallel on each processor. Every thread loops over its local elements to obtain the list of elements containing possible collision partners. As this list contains elements in the halo region and is uniquely sorted along the space-filling curve, each thread is ensured to detect collisions in deterministic order, thus ensuring kinetic energy preservation. Each particle pair is checked for exact hard-sphere collisions using eqs. (29) and (30). The obtained time of collision is checked against the time of flight until encountering a boundary condition to eliminate physically incompatible solutions. Once a valid collision solution is obtained, the particles are moved to the collision location and their momentum is updated using eqs. (31) to (33). Since only one collision is permitted per particle within each time step due to the hard-sphere approach, particles with confirmed collisions are removed from the list of eligible collision partners and transferred to the main tracking algorithm to determine their final location and host element.

4.4. Load balance

While the continuous Euler phase exerts a fixed computational load per grid element, the computational effort of the dispersed Lagrangian phase is dependent on the particle positions and only weakly correlated with the underlying grid. As the particle distribution generally cannot be computed a priori, ∃LEXI relies on dynamic load balancing to alleviate most of the disparities in computational load [18]. Each stage of the time stepping algorithm in Fig. 1 is equipped with conditional high-precision timers to compute the total time spent in each stage. Prior to triggering the actual load balancing, these timers are activated together with counters for the particle number and collision partners in each cell. After initiating load balancing, the recorded computation time is distributed among the cells in proportion to their contribution to the total particle processing time. This contribution is determined using the relative fractions derived from the particle and collision counters for each cell.

5. Validation

Before turning to the application cases, we validate the various building blocks for particle-laden flow, from the projection function in the particle source term to the particle collision operator. The reader is referred to Kopper et al. [18] for details on the validation of the particulate phase such as convergence properties and the correctness of the one- and two-way coupling. An extensive validation of the continuous phase is given in Krais et al. [38]. In the following, only viscous effects such as drag and heating are considered for the particulate phase. Moreover, particle collisions are assumed to be purely elastic, i.e., $\mu_f = 0$ and $e_n = 1$, such that the tangential component of the impulse is zero and the normal component reduces to $J_n = -2m_{p,r}(\mathbf{v}_{p,r}^{(0)} \cdot \mathbf{n}_p)$. Both phases are integrated in time by a fourth-order accurate explicit Runge–Kutta scheme [35] with a relative Courant–Friedrichs–Lewy (CFL) number of CFL = 0.9. The Riemann solver by Harten–Lax–Van Leer–Einfeldt (HLLE) [43] is employed for the approximation of the numerical flux function. The high-order DGSEM is used for the spatial discretization of the continuous phase, which is assumed to be inviscid, i.e., only the Euler equations are considered in this section, if not stated otherwise. It has to be noted that all quantities are non-dimensional.

5.1. Particle deposition

First, the validity of the employed particle deposition procedure is demonstrated using the interaction of a shock wave with a spanwise-inhomogeneous particle cloud, following [44,45,14]. Thus, only a two-way coupling is considered. Since ∃LEXI is a pure three-dimensional code, a quasi two-dimensional setting is chosen where the computational domain $\Omega = [0, 5.5] \times [-1.1, 1.1] \times [0, 0.001]$ is discretized by $500 \times 250 \times 1$ elements with $N = 4$. The initial solution is a right moving shock wave with a Mach number of $M = 3$, located at $x = 0.315$, and a pre-shock state of $(\rho, \mathbf{u}_f, p) = (1.2, \mathbf{0}, 1)$. Non-reflecting boundary conditions are prescribed at the left and right boundary conditions. Symmetry boundary conditions are utilized at the upper and lower boundaries, while periodic conditions are employed in the remaining direction. A cubic particle cloud with $d_p = 10^{-4}$ and $\rho_p = 8900$ is initialized in the rectangular region $[0.315, 0.634] \times [-0.16, 0.16] \times [0, 0.001]$ with a volume fraction of around 4%, resulting in a total number of 5×10^5 computational particles. Each computational particle represents 16 real particles. A fluid with a constant dynamic viscosity of $\mu = 1.7144 \times 10^{-8}$ is assumed. As illustrated in Fig. 5, the resulting centerline pressure profiles are comparable to [44,14]. Slices of the corresponding temperature and pseudo-Schlieren ($|\nabla\rho|$) profile at non-dimensional simulation times $t = 0.8, 1.86$ are depicted in Fig. 6.

5.2. Simple particle collision

To assess the accuracy and robustness of the particle collision algorithm, the numerical particle collision time between two colliding particles is compared to its analytical solution. For the robustness check, the particles are located on different processors separated by a periodic boundary condition, see Fig. 7. A computational domain of size $\Omega = [-1.1, 1.1]^3$ is discretized with 4^3 elements with periodic boundary conditions. A uniform initial solution is filled with two particles with $d_p = 0.04$. The first particle is located at $\mathbf{x}_{p,1}(t_0) = [-0.02, -0.5, 0]$ with $\mathbf{v}_{p,p,1}(t_0) = [0, -15, 0]$, and the second particle is placed at $\mathbf{x}_{p,2}(t_0) =$

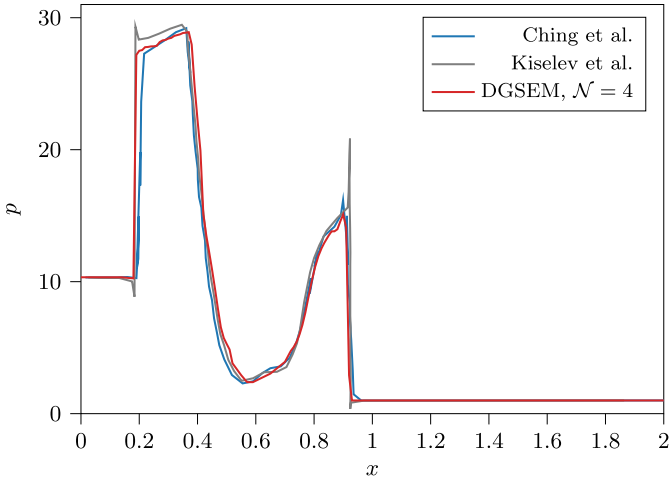

Fig. 5. Pressure distribution along the centerline for the interaction of a shock wave with a spanwise-inhomogeneous particle cloud at a non-dimensional time of $t = 0.24$. Results are compared to the numerical results by Kiselev et al. [44] (third-order finite difference (FD) method) and Ching and Ihme [14] (DG, $\mathcal{N} = 3$) at $t = 0.75$.

$[0.017, 0.5, 0]$ with $\mathbf{v}_{p,p,2}(t_0) = [0, 15, 0]$. The exact solution can be computed analytically by solving eqs. (29) and (30), resulting in $t_{\text{coll}}^{\text{ex}} = 9.962\,37 \times 10^{-2}$. The relative error between the analytically and numerically predicted time of the first particle collision is around machine precision at $O(10^{-16})$.

5.3. Kinetic energy analogy and preservation

This validation case aims to further test the robustness and efficacy of the proposed framework by checking the kinetic energy analogy and the conservation of the total kinetic energies of the particles. The setup is chosen such that it mimics the motion of molecules in an ideal gas under thermodynamic equilibrium, following the principles of kinetic theory, similar to [46,6]. The equilibrium properties of the system are determined entirely by the particle mass, denoted by m_p, the number density of particles, n_p, and the energy content in the system, i.e., the mean square speed $|\overline{\mathbf{v}}_p|^2$. The mean square speed is analogous to the specification of a temperature for the particles (granular temperature), given as

$$\overline{\mathbf{v}}_p = \sqrt{\frac{3k_B T}{m_p}}, \tag{35}$$

for an ideal gas law, with the Boltzmann constant k_B and $T = 4 \times 10^9$. In order for the particles to converge to an equilibrium state, the following two major assumptions must hold true [47]. First, particles are assumed to have no acceleration and negligible interaction with the fluid phase, thus only binary particle collisions are considered (dual limit of small particle volume fractions and large Stokes numbers). Second, the number of particles is large enough to justify a statistical treatment of the particulate phase. It can be shown that in this case, the particle velocity distribution eventually converges to an equilibrium velocity distribution which follows a 3D Maxwell–Boltzmann distribution, expressed as

$$f_M\left(\left|\overline{\mathbf{v}}_p\right|\right) = 4\pi\left|\overline{\mathbf{v}}_p\right|^2 \left[\frac{2\pi k_B T}{m}\right]^{-3/2} \exp\left(-\frac{1}{2}\frac{m\left|\overline{\mathbf{v}}_p\right|^2}{k_B T}\right). \tag{36}$$

The computational domain is of size $\Omega = [0, 1]^3$ with periodic boundaries and is initially filled with an inviscid, quiescent fluid. The domain is equipped with 10 000 randomly distributed particles which are initialized with $d_p = 0.02$, $\rho_p = 10^{-10}$, leading to a number density of particles of $n_p \approx 10^5$, defined as the ratio of the particle volume fraction to the particle volume. The initial particle velocities are random in each direction with a constant velocity magnitude of one. This setup results in a volume fraction of around 0.0418 and about 10^5 particle collisions over 1000 time steps. As illustrated in Fig. 8, the particle velocities converge towards the three-dimensional Maxwell speed distribution. It is reasonable to conclude that under these conditions the aforementioned assumptions of kinetic theory are valid. Moreover, the total kinetic energy production of the particles is less than machine precision, independent of the number of processors used, such that the total kinetic energy is preserved over time.

6. Parallel performance

ƎLEXI is designed as massively parallel code aimed towards modern HPC systems. As such, retaining scalability on large core counts is an inherent design goal. For the current work, the scaling performance is evaluated via simulations performed on the HPE Apollo *Hawk* system at the High Performance Computing Center (HLRS) in Stuttgart, Germany, and on the *LUMI* (Large Unified Modern Infrastructure) system at the CSC–IT Center for Science in Kajaani, Finland. *Hawk* utilizes dual-socket AMD EPYC™ 7742 nodes (128 cores per node) combined with 256 GB RAM and an InfiniBand HDR200 interconnect in an enhanced 9D-hypercube topology. The code was compiled with the GNU compiler version 9.2.0 with the libraries mpt 2.23, hdf5 1.10.5 and aocl 3.0. The *LUMI*-C hardware partition utilizes newer AMD EPYC™ 7763 nodes (128 cores per node) combined with 256 GB RAM and an HPE Cray Slingshot-11 200 Gbit s^{-1} interconnect with a dragonfly network topology. Here, the code was compiled with the GNU compiler version 13.2.1 with the libraries Cray MPICH 8.1, hdf5 1.12.2 and Cray LibSci 24.03. The scaling performance is evaluated for the time spent to advance the computational simulation, i.e., without initialization effort

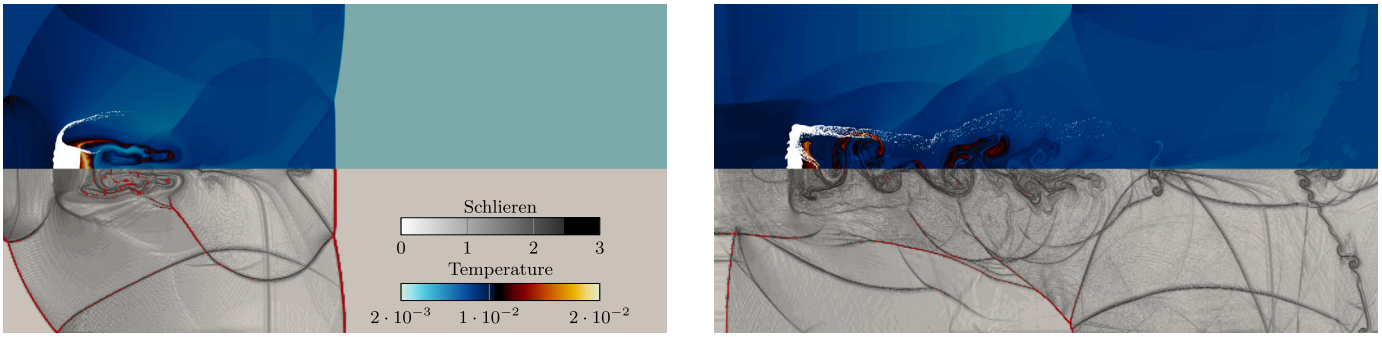

Fig. 6. Top: Temperature distribution for the interaction of a shock wave with a spanwise-inhomogeneous particle cloud at non-dimensional times $t = 0.8$ (left) and $t = 1.86$ (right) with particles. Bottom: Pseudo-Schlieren ($|\nabla\rho|$) profile with the FV subcell distribution denoting the shock capturing highlighted in red.

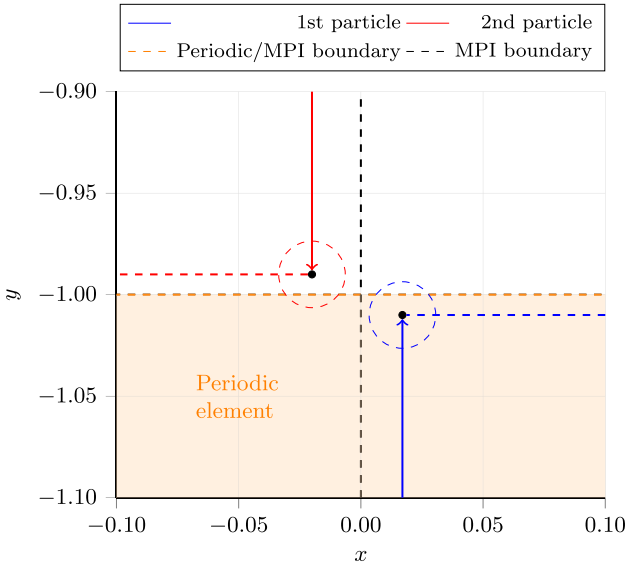

Fig. 7. Sketch of the simple particle collision test case with the numerical and analytical particle trajectories including multiple process boundaries and periodic boundary conditions.

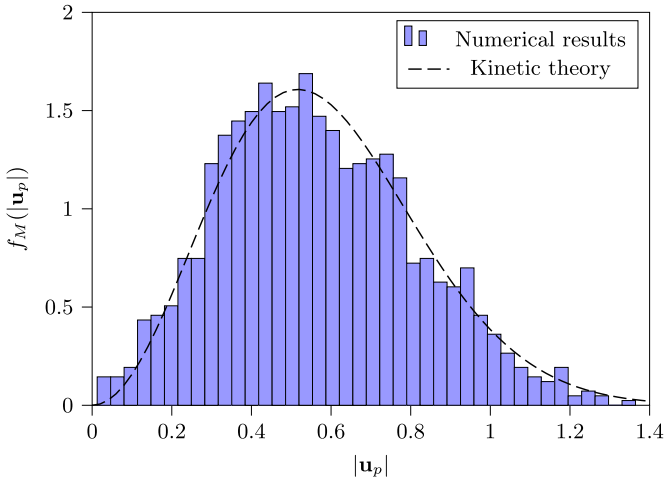

Fig. 8. Distribution of the particle velocity magnitude predicted by the numerical simulation and given by the kinetic theory.

and input/output times. Each run was repeated 5 times to eliminate fluctuations in overall machine load. Scaling is investigated using an unstructured grid in the form of a Cartesian periodic box with elementary dimensions $2L \times L \times L,\ L \in \mathbb{R}_{>0}$ and 16 hexagonal elements/L. For weak scaling runs on multiple compute nodes, this box is extended in x_1-direction through multiplication with the number of nodes to maintain identical grid spacing. The fluid field is initialized to a uniform flow with the velocity vector $\mathbf{u}_f = [1, 1, 1]^T$. A sponge zone using an exponential temporal filter following Pruett et al. [48] is set up in the final x_1-decile to dampen numerical oscillations before reaching the outflow boundary while periodicity constraints are imposed on the other two directions. Particles are emitted once initially with a concentration of 125 000 particles per unit cube L^3 with their initial velocity set equal to the local fluid velocity. First, the memory consumption and time-to-solution are discussed, followed by the parallel efficiency of ∃LEXI. To the author's best knowledge, this represents the first time that comprehensive performance data for one-way, two-way, and four-way coupled Euler–Lagrange simulations is published on these state-of-the-art HPC clusters.

Table 1
CPU time and memory consumption for 8.85×10^5 degrees of freedom and 2.5×10^5 particles for different degrees of phase coupling on one compute node on *LUMI*.

Coupling	CPU Time [s]	Memory [MB]
0-way (pure fluid)	5.50	902.9
1-way	11.52	5819.6
2-way	18.12	6438.7
4-way	150.55	6517.8

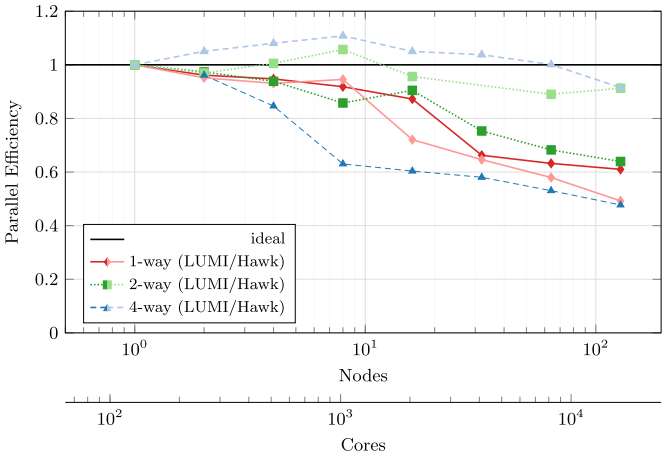

Fig. 9. Weak scaling of ∃LEXI with the split-form DG scheme and $\mathcal{N} = 5$ plotted as the parallel efficiency over the number of CPUs for fixed loads, i.e., DOF/Particles per CPU. The parallel efficiency is computed based on the performance on a compute single node, i.e., on 128 CPUs.

6.1. Memory consumption and time-to-solution

The MPI+MPI hybrid implementation means that data required on a single compute node is stored uniquely in memory. The resulting approach is particularly memory-efficient on Simultaneous Multi-Threading (SMT) systems with large core counts, such as common in the aforementioned HPC systems. Table 1 depicts the CPU time for 10 Runge–Kutta loops, summed about over all cores, and the traceable resource total memory usage of all tasks in the job[4] for a single compute node calculation on *LUMI*. For the given setup, the chosen polynomials degree of $\mathcal{N} = 5$ results in $\approx 8.85 \times 10^5$ DOFs with about 1.02 kB per DOF, closely matching other codes of the FLEXI family [49]. The inclusion of a 1-way coupled Lagrangian phase significantly increases memory consumption. Note that only a minor portion of this allocated memory is devoted to storing the actual particle data with about 350 bytes per particle [18]. Most of the additional memory is dedicated towards providing the geometric information necessary for particle tracking. Enabling of particle-to-fluid interaction increases the memory further, as information for each unique deposition node must be available. From this point on, our proposed MPI+MPI hybrid approach for particle collision only adds minor memory consumption. However, the exact calculation of the particle interactions results in a runtime increase with a factor of ≈ 8.3.

6.2. Parallel efficiency

Weak scaling efficiency of ∃LEXI for one-way, two-way, and four-way coupling between the phases is presented in Fig. 9. As the MPI-3 shared memory parallelization is performed on a compute node level,

[4] The total memory usage is recorded by `SLURM` using the `TresUsageInTot` metric.

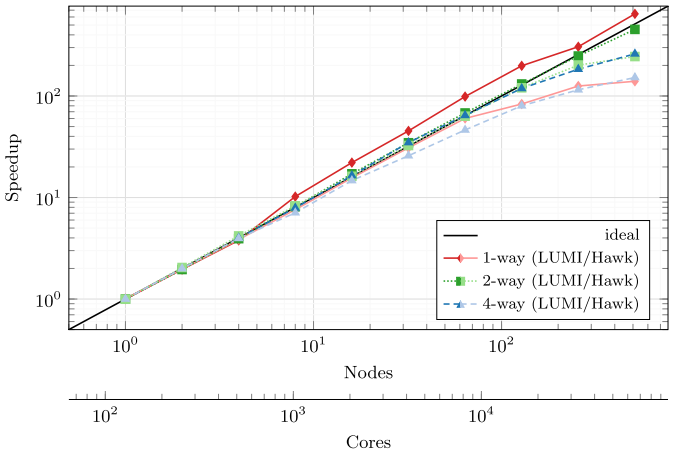

Fig. 10. Strong scaling of ∃LEXI with the split-form DG scheme and $N = 5$ plotted as the parallel efficiency over the number of CPUs for fixed case size, i.e., total number of DOF/Particles. The speed-up is computed based on the performance on a compute single node, i.e., on 128 CPUs.

the performance is normalized relative to the computational capability of a single compute node, consisting of 128 CPUs. The only exception is the four-way coupled simulation on *LUMI*, where the additional MPI windows exhaust the hardware message queue of the Slingshot-11 interconnect. This exhaustion occurs for all four-way coupled simulations using two or more *LUMI* nodes, requiring a transition to slower software message matching. These cases were normalized to align with the one- and two-way coupled results for two nodes on *LUMI* to ensure consistency.

One-way coupled simulations have the lowest compute-to-communication ratio, rendering them the most sensitive to bandwidth and latency limitations imposed by the interconnect. Weak scaling efficiency consistently exceeds 61 %, demonstrating reliable performance even at the scale of 128 nodes (16 384 cores) on *LUMI*. As the Slingshot-11 interconnect in dragonfly topology on *LUMI* features a lower diameter compared to the InfiniBand network with a hypercube topology on *Hawk*, one-way coupled performance results on *Hawk* fall below similar simulations on *LUMI* in terms of efficiency for high node counts. In contrast, weak scaling results for two- and four-way coupling on *Hawk* exhibit a performance increase up to 8 nodes, which was confirmed to originate from the energy-optimized runtime environment provided by the PowerSched framework [50]. At higher node counts, simulation performance on *Hawk* is again limited by the interconnect which results in a drop of weak scaling efficiency. However, the increased computational load compared to the one-way coupled approach result in 91.5 % weak scaling efficiency for the four-way coupled simulation on 128 *Hawk* nodes. The greater computational capacity of the *LUMI* nodes, combined with the challenges associated with MPI message matching, results in reduced weak scaling efficiency at high node counts for the same test case on *LUMI*. Higher node counts exceed the 2^{31}-limit of 4-byte signed integers and are not considered for the weak scaling performance.

Strong scaling results in form of the computational speed-up for the $128L \times L \times L$ case, thus approx 5.66×10^7 DOFs and 1.60×10^7 particles, are depicted in Fig. 10. Similar to the weak scaling tests, all tests were performed for one-way, two-way, and four-way coupling between the phases. Exhaustion of the Slingshot-11 hardware message queue occurs at 4 *LUMI* nodes which was thus chosen as a reference for normalization. For the one-way and two-way coupled simulations on *Hawk*, nearly ideal speed-up is observed up to 64 compute nodes (8192 Cores). At higher core counts, computational jobs at *Hawk* are transferred from a dedicated partition to distributed racks using best-effort topology-aware scheduling. This leads to a gradual decrease in strong scaling efficiency as jobs with ≥ 128 compute nodes operate with higher interconnect latency due to the hypercube topology. The effect is more pronounced for the one-way coupled case as the deposition of the particulate force on the fluid domain is performed with an expensive atomic operation in the time-stepping loop. While this atomic operation could be circumvented by allocating shadow arrays, we prioritized optimizing ∃LEXI towards lower memory consumption in this space-time/time-memory trade-off. Strong scaling efficiency for four-way coupled simulations initially remains below those neglecting collisions as there is limited computational load available to hide the MPI RMA operations. While this reduces the speed-up at low core counts, the one-sided nature of these operations translates into improved scalability compared to one/two-way coupled simulations at high node numbers, indicating potential for further scaling. Results on *LUMI* generally outperform the corresponding simulations on *Hawk*, with the one-way coupled simulation even demonstrating superscalar scaling due to reduced memory pressure. Both the one-way and the two-way coupled simulation on *LUMI* show slopes parallel to ideal scaling up to the maximum of 512 compute nodes (65 536 cores) tested while the four-way coupling approach enters the area of diminishing returns for > 256 compute nodes. Overall, the results on both *LUMI* and *Hawk* affirm the scalability and adaptability of ∃LEXI across diverse hardware environments, enabling efficient calculation of increasingly complex simulations at scale.

7. Applications

In this section, the applicability of ∃LEXI to more challenging large-scale test cases is demonstrated. For this, a plane particle-laden jet impinging on a cavity and the particle-laden flow around a transonic NACA0012 airfoil under atmospheric conditions comparable to the Martian environment are considered. In the following, only viscous forces, i.e., drag and heat, are taken into account. In all cases, four-way coupling between the fluid and the dispersed phase is assumed. Finally, we want to emphasize that the current applications only serve as numerical examples. Thus, detailed physical investigations are out of the scope of this paper.

7.1. Round jet impinging on a cavity

In this section, we investigate the particle behavior for a plane particle-laden jet impinging on a cavity by comparing a simulation without particles to a simulation with a four-way coupled dispersed phase. The primary focus of this application is to demonstrate the capabilities of ∃LEXI as well as the effectiveness of the load balancing. The setup is chosen to mimic a dry-ice blasting procedure, where a high-velocity jet laden with frozen carbon dioxide particles is utilized to clean solid surfaces. Due to the various physical phenomena that occur during the process, this test case can be considered particularly challenging [51]. For the chosen setup, the continuous phase is air with a dynamic viscosity of $\mu = 2.71 \times 10^{-5}\,\mathrm{kg\,m^{-1}\,s^{-1}}$. The jet enters the domain with a Mach number of $M = 0.6$ and a Reynolds number of $Re = 89605$ based on the jet radius $r_{\mathrm{jet}} = 0.05\,\mathrm{m}$ and the characteristic flow velocity of the jet $u_{\mathrm{jet}} = 275.0679\,\mathrm{m\,s^{-1}}$. The computational domain is discretized using 392 630 hexahedral elements with $N = 4$. The domain is defined as $\Omega = \Omega_{\mathrm{box}} \cup \Omega_{\mathrm{cavity}}$ with $\Omega_{\mathrm{box}} \in [0, 0.0333] \times [0, 0.06] \times [0, 0.0008]\,\mathrm{m}$ and a cavity $\Omega_{\mathrm{cavity}} = [0.0333, 0.03665] \times [0, 0.015] \times [0, 0.0008]\,\mathrm{m}$. Adiabatic no-slip boundary conditions are prescribed on the upper domain, while pressure outflow conditions ($p = 1.2 \times 10^5\,\mathrm{Pa}$) are set at the left and lower domain. Inflow conditions are defined based on the total pressure and total temperature of the jet, $p_t = 144\,097\,\mathrm{Pa}$ and $T_t = 331.8\,\mathrm{K}$, respectively, at the lower right domain. Periodic boundary conditions are prescribed in the third direction. The fluid is initially at rest with $p = 1.2 \times 10^5\,\mathrm{Pa}$ and $\rho = 1.2597\,\mathrm{kg\,m^{-3}}$. Following Liu et al. [52], particles are emitted with a size of $d_p = \{20, 28, 40, 57, 80\}\,\mu\mathrm{m}$, a density of $\rho_p = 1560\,\mathrm{kg\,m^{-3}}$, an initial temperature of $T_p = 195\,\mathrm{K}$, and a specific heat at constant pressure of $c_p = 519.16\,\mathrm{J\,K^{-1}\,kg^{-1}}$. The emission rate of the particles is chosen such that a volume fraction of $\approx 1 \times 10^{-3}$ is reached in the jet. The simulation is carried out without particles until

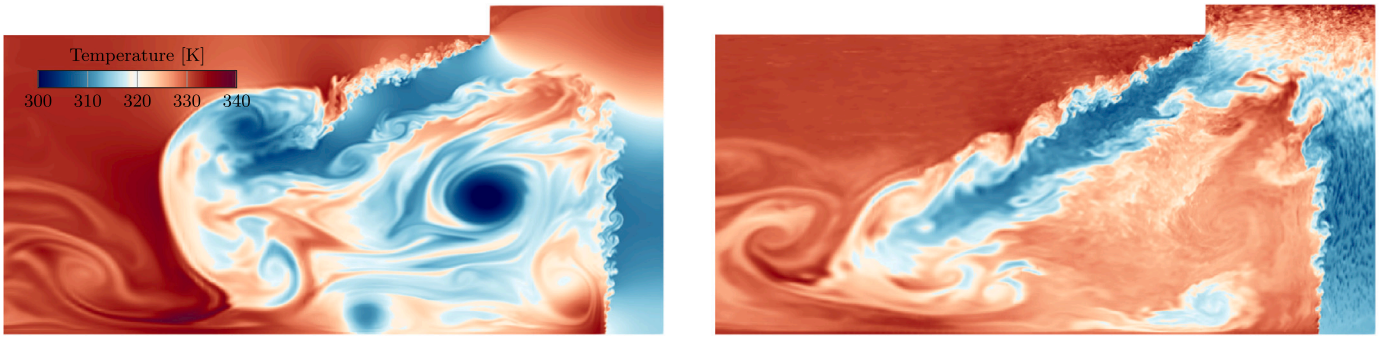

Fig. 11. Instantaneous temperature distribution of a round jet impinging on a cavity at $4T^*$ without (left) and at $8T^*$ with 4-way coupled particles (right).

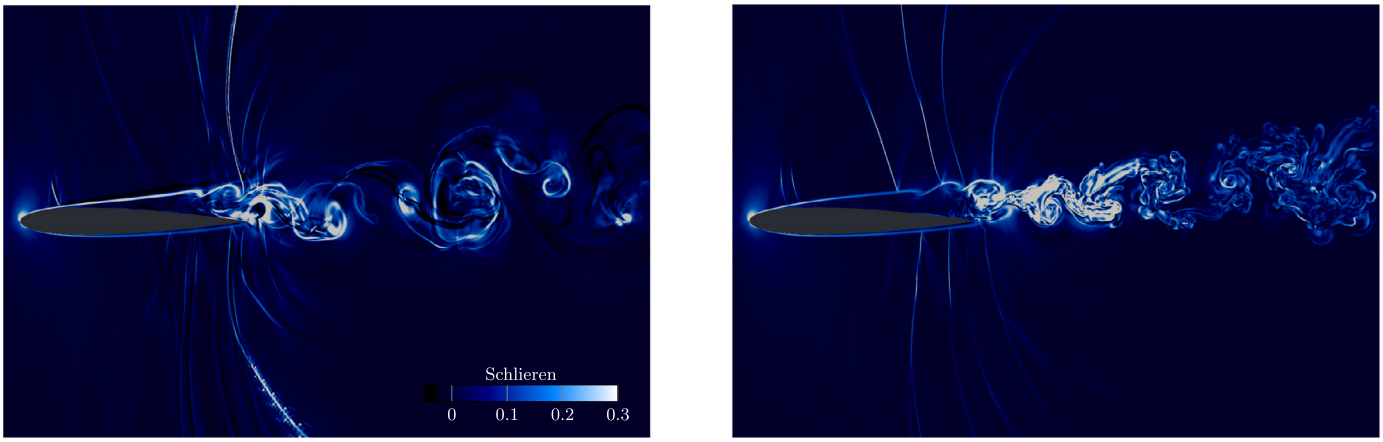

Fig. 12. Comparison of pseudo-Schlieren as a representation of density gradients for the unladen (left) and particle-laden (right) flow around a NACA 0012 airfoil at $Ma = 0.8$ and $Re = 50\,000$.

$t = 4T^*$, with the characteristic time $T^* = L_{\text{jet}}/u_{\text{jet}}$ defined as the ratio of the length of the jet $L_{\text{jet}} = 0.036\,65$ m to the characteristic velocity of the jet. Without load balancing, the simulation achieved a time-to-solution of 1.1×10^{-3} s per degree of freedom and Runge–Kutta stage, which was nearly halved to 5.6×10^{-4} s through dynamic workload rebalancing.

Slices of the corresponding temperature profile without and with four-way coupled particles at simulation times of $t = 4, 8\,T^*$ are depicted in Fig. 11. In the simulation without particles, a second jet emerges due to the chosen size and depth of the cavity, which, combined with the main jet, forms a strong recirculation area comprised of a primary vortex and various small vortices. As illustrated on the right of Fig. 11, the presence of a four-way coupled particulate phase strongly modifies the flow field. First, the influence of the dry-ice particles is clearly visible in the locally decreasing temperature field. Second, the mean characteristics of the complex flow are slightly stabilized due to the viscous particle forces imposed on the continuous phase. Simultaneously, the high number of particle collisions causes local perturbations. Thus, neglecting particle collisions in dense particle-laden flows can lead to vastly different flow structures and in turn particle motions.

7.2. Airfoil flow under Martian atmospheric conditions

This test case was chosen to demonstrate the performance of ∃LEXI at scales representative of practical applications. The flow around a NACA0012 airfoil under atmospheric conditions similar to the Martian environment was investigated experimentally [53] and numerically [54]. ∃LEXI was validated against the experimental results with no suspended particles at $M = 0.6$ and $Re = 50\,000$ with the results summarized in Table 2. Atmospheric parameters were selected according to Rafkin and Banfield [55] and the angle of attack was kept stationary at $\alpha = 5°$. For Mach numbers exceeding 0.7, URANS simulations with

Table 2
Flow field parameters and comparison of the LES results of ∃LEXI against experiments [53] for the flow without suspended particles.

Parameter	Variable	Experiment	∃LEXI
Spec. gas cons.	R [J kg^{-1} K^{-1}]	-	191.14
Heat cap. ratio	γ [-]	-	1.351
Temperature	T [K]	-	213
Lift coeff.	C_l [-]	0.4205	0.3972

the transition SST model and a coupled discrete phase model performed by Liu et al. [54] found that sand particles contained in the Martian atmosphere break up the stable shock structures observed in the unloaded flow. Consequently, the current study employs the large-eddy simulation (LES) approach with an inflow Mach number of $Ma = 0.8$ and Reynolds number of $Re = 50\,000$. Adiabatic no-slip boundary conditions are set on the airfoil surface with the far field prescribed by Dirichlet boundary conditions and a periodicity constraint applied in the spanwise direction. The domain is discretized with 68 000 hexahedral elements and a polynomial degree of $\mathcal{N} = 4$, resulting in 8.5×10^6 degrees of freedom. The simulation was performed on 16 *Hawk* nodes with a simulation efficiency (simulated physical time per utilized core hour) of 5.76×10^{-8} s/CPUh and allowed to run until the integral forces on the airfoil were statistically stationary.

A comparison of a pseudo-Schlieren visualization of the resulting instantaneous LES flow field for the simulation without particles and the case with a sand particle concentration of 225 mg m^{-3} is shown in Fig. 12. The instantaneous snapshots of both simulations exhibit a shock on the forward suction side, followed by flow separation and a second shock system near the recirculation region at the trailing edge. While the main shock front on the pressure side is located near the trailing

edge for the unladen flow, this shock is pushed forward in the presence of particles. A similar shock is also observed in the particle-laden RANS simulations conducted by Liu et al. [54], but their time-averaged results do not predict a shock formation on the pressure side for the unladen case. Moving downstream, the recirculation flow develops into an irregular vortex street. Here, the augmentation of turbulent fluctuations, introduced by the inertial particles, is clearly evident. At the same time, the downstream momentum introduced by the particulate phase re-energizes the wake, resulting in a reduction in wake width. These results highlight the importance of time-accurate simulations, as the presence of inertial particles significantly alters the instantaneous flow structures and reveals that the stationary flow solution neglects significant parts of the result.

8. Conclusion

The efficient and accurate numerical treatment of dense particle-laden flows is challenging, especially on HPC systems. Focusing on literature for Euler–Lagrange particle tracking in a compressible carrier fluid, the primary focus is placed more on the time-accurate coupling of both phases and the adequate collision treatment than on the efficiency on highly parallel systems. This work aimed to alleviate this deficiency by proposing a four-way coupled Euler–Lagrange approach based on the combination of the particle operator with an MPI+MPI hybrid approach. The proposed algorithm enables a highly efficient and accurate calculation of binary inter-particle collisions including the effective treatment of the particle-fluid coupling in a compressible carrier phase on arbitrary core counts. Special focus was placed on the detection of the particle collisions on parallel systems with possibly curved element faces. Built on pure MPI following the MPI-everywhere strategy, the approach offers flexibility and portability across various systems, requiring no modifications for varying CPU counts per compute node. The implementation is thoroughly validated and the excellent scaling properties on massively parallel systems are demonstrated. This work concluded with two more challenging test cases to demonstrate its applicability to large-scale applications. Finally, the work is open-source available and welcomes external contributions.

CRediT authorship contribution statement

Anna Schwarz: Validation, Methodology, Visualization, Software, Conceptualization, Writing – original draft. **Patrick Kopper:** Visualization, Software, Investigation, Writing – original draft, Validation, Methodology, Conceptualization. **Emilian de Staercke:** Software, Investigation. **Andrea Beck:** Supervision, Funding acquisition, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The research presented in this paper was funded in parts by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2075 - 390740016 and by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) and Sweden, Germany, Spain, Greece, and Denmark under grant agreement No 101093393. We acknowledge the support by the Stuttgart Center for Simulation Science (SimTech). The authors gratefully acknowledge the support and the computing time on *Hawk* provided by the HLRS through the project "hpcdg". We acknowledge the EuroHPC Joint Undertaking for awarding this project access to the EuroHPC supercomputer *LUMI*, hosted by CSC (Finland) and the LUMI consortium through a EuroHPC Development Access call.

Data availability

Data will be made available on request.

References

[1] S. Balachandar, J.K. Eaton, Turbulent dispersed multiphase flow, Annu. Rev. Fluid Mech. 42 (2010) 111–133, https://doi.org/10.1146/annurev.fluid.010908.165243.
[2] A.Y. Varaksin, S.V. Ryzhkov, Particle-laden and droplet-laden two-phase flows past bodies (a review), Symmetry 15 (2023) 388, https://doi.org/10.3390/sym15020388.
[3] S. Elghobashi, On predicting particle-laden turbulent flows, Appl. Sci. Res. 52 (1994) 309–329, https://doi.org/10.1007/BF00936835.
[4] C.T. Crowe, J.D. Schwarzkopf, M. Sommerfeld, Y. Tsuji, Multiphase Flows with Droplets and Particles, CRC Press, 2011.
[5] H. Sigurgeirsson, A. Stuart, W.-L. Wan, Algorithms for particle-field simulations with collisions, J. Comput. Phys. 172 (2001) 766–807, https://doi.org/10.1006/jcph.2001.6858.
[6] E.J. Ching, M. Ihme, Development of a particle collision algorithm for discontinuous Galerkin simulations of compressible multiphase flows, J. Comput. Phys. 436 (2021) 110319, https://doi.org/10.1016/j.jcp.2021.110319.
[7] H. Norouzi, R. Zarghami, N. Mostoufi, New hybrid CPU-GPU solver for CFD-DEM simulation of fluidized beds, Powder Technol. 316 (2017) 233–244, https://doi.org/10.1016/j.powtec.2016.11.061.
[8] H. Liu, H. Ma, Q. Liu, X. Tang, J. Fish, An efficient and robust GPGPU-parallelized contact algorithm for the combined finite-discrete element method, Comput. Methods Appl. Mech. Eng. 395 (2022) 114981, https://doi.org/10.1016/j.cma.2022.114981.
[9] Z. Yao, J.-S. Wang, G.-R. Liu, M. Cheng, Improved neighbor list algorithm in molecular simulations using cell decomposition and data sorting method, Comput. Phys. Commun. 161 (2004) 27–35, https://doi.org/10.1016/j.cpc.2004.04.004.
[10] M. Breuer, M. Alletto, Efficient simulation of particle-laden turbulent flows with high mass loadings using LES, Int. J. Heat Fluid Flow 35 (2012) 2–12, https://doi.org/10.1016/j.ijheatfluidflow.2012.01.001.
[11] V. Ogarko, S. Luding, A fast multilevel algorithm for contact detection of arbitrarily polydisperse objects, Comput. Phys. Commun. 183 (2012) 931–936, https://doi.org/10.1016/j.cpc.2011.12.019.
[12] D. Krijgsman, V. Ogarko, S. Luding, Optimal parameters for a hierarchical grid data structure for contact detection in arbitrarily polydisperse particle systems, Comput. Part. Mech. 1 (2014) 357–372, https://doi.org/10.1007/s40571-014-0020-9.
[13] Z.-m. Liao, L.-b. Chen, Z.-h. Wan, N.-s. Liu, X.-y. Lu, GPU acceleration of four-way coupled PP-DNS for compressible particle-laden wall turbulence, Int. J. Multiph. Flow 176 (2024) 104840, https://doi.org/10.1016/j.ijmultiphaseflow.2024.104840.
[14] E.J. Ching, M. Ihme, Efficient projection kernels for discontinuous Galerkin simulations of disperse multiphase flows on arbitrary curved elements, J. Comput. Phys. 435 (2021) 110266, https://doi.org/10.1016/j.jcp.2021.110266.
[15] T. Hoefler, J. Dinan, D. Buntinas, P. Balaji, B. Barrett, R. Brightwell, W. Gropp, V. Kale, R. Thakur, MPI + MPI: a new hybrid approach to parallel programming with MPI plus shared memory, Computing 95 (2013) 1121–1136, https://doi.org/10.1007/s00607-013-0324-2.
[16] H. Zhou, J. Gracia, R. Schneider, MPI collectives for multi-core clusters: optimized performance of the hybrid MPI+MPI parallel codes, in: Workshop Proceedings of the 48th International Conference on Parallel Processing, ICPP 2019, ACM, 2019, pp. 1–10.
[17] L. Quaranta, L. Maddegedara, A novel MPI+MPI hybrid approach combining MPI-3 shared memory windows and C11/C++11 memory model, J. Parallel Distrib. Comput. 157 (2021) 125–144, https://doi.org/10.1016/j.jpdc.2021.06.008.
[18] P. Kopper, A. Schwarz, S.M. Copplestone, P. Ortwein, S. Staudacher, A. Beck, A framework for high-fidelity particle tracking on massively parallel systems, Comput. Phys. Commun. 289 (2023) 108762, https://doi.org/10.1016/j.cpc.2023.108762.
[19] W. Sutherland, LII. The viscosity of gases and molecular force, Lond. Edinb. Dublin Philos. Mag. J. Sci. 36 (1893) 507–531, https://doi.org/10.1080/14786449308620508.
[20] M.R. Maxey, J.J. Riley, Equation of motion for a small rigid sphere in a nonuniform flow, Phys. Fluids 26 (1983) 883–889, https://doi.org/10.1063/1.864230.
[21] R. Gatignol, The Faxén formulae for a rigid particle in an unsteady non-uniform Stokes flow, J. Méc. Théor. Appl. 2 (1983) 143–160.
[22] E. Loth, J.T. Daspit, M. Jeong, T. Nagata, T. Nonomura, Supersonic and hypersonic drag coefficients for a sphere, AIAA J. 59 (2021) 3261–3274, https://doi.org/10.2514/1.J060153.
[23] S.C.R. Dennis, S.N. Singh, D.B. Ingham, The steady flow due to a rotating sphere at low and moderate Reynolds numbers, J. Fluid Mech. 101 (1980) 257–279, https://doi.org/10.1017/S0022112080001656.

[24] C.T. Crowe, M.P. Sharma, D.E. Stock, The particle-source-in cell (PSI-CELL) model for gas-droplet flows, J. Fluids Eng. 99 (1977) 325–332, https://doi.org/10.1115/1.3448756.
[25] D. Shepard, A two-dimensional interpolation function for irregularly-spaced data, in: Proceedings of the 1968 23rd ACM National Conference, ACM Press, 1968.
[26] T. Stindl, Entwicklung und Untersuchung eines Partikelverfahrens zur Simulation elektromagnetischer Wechselwirkungen in verdünnten Plasmaströmungen, Ph.D. thesis, University of Stuttgart, 2015, https://elib.uni-stuttgart.de/handle/11682/3980.
[27] J.P. Bons, R. Prenter, S. Whitaker, A simple physics-based model for particle rebound and deposition in turbomachinery, J. Turbomach. 139 (2017) 081009, https://doi.org/10.1115/1.4035921.
[28] A. Schwarz, P. Kopper, J. Keim, H. Sommerfeld, C. Koch, A. Beck, A neural network based framework to model particle rebound and fracture, Wear (2022) 204476, https://doi.org/10.1016/j.wear.2022.204476.
[29] P.L. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, J. Comput. Phys. 43 (1981) 357–372, https://doi.org/10.1016/0021-9991(81)90128-5.
[30] A. Harten, J.M. Hyman, Self adjusting grid methods for one-dimensional hyperbolic conservation laws, J. Comput. Phys. 50 (1983) 235–269, https://doi.org/10.1016/0021-9991(83)90066-9.
[31] F. Bassi, S. Rebay, A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations, J. Comput. Phys. 131 (1997) 267–279, https://doi.org/10.1006/jcph.1996.5572.
[32] S. Pirozzoli, Numerical methods for high-speed flows, Annu. Rev. Fluid Mech. 43 (2011) 163–194, https://doi.org/10.1146/annurev-fluid-122109-160718.
[33] C. Mavriplis, Nonconforming discretizations and a posteriori error estimators for adaptive spectral element techniques, Ph.D. thesis, Massachusetts Institute of Technology, 1989.
[34] M. Sonntag, Shape derivatives and shock capturing for the Navier-Stokes equations in discontinuous Galerkin methods, Ph.D. thesis, Universität Stuttgart, 2017.
[35] M.H. Carpenter, A. Kennedy, Fourth-order 2N-storage Runge–Kutta schemes, NASA Tech. Memo. 109112 (1994) 1–26.
[36] A.D. Beck, T. Bolemann, D. Flad, H. Frank, G.J. Gassner, F. Hindenlang, C.-D. Munz, High-order discontinuous Galerkin spectral element methods for transitional and turbulent flow simulations, Int. J. Numer. Methods Fluids 76 (2014) 522–548, https://doi.org/10.1002/fld.3943.
[37] F. Hindenlang, G.J. Gassner, C. Altmann, A. Beck, M. Staudenmaier, C.-D. Munz, Explicit discontinuous Galerkin methods for unsteady problems, Comput. Fluids 61 (2012) 86–93, https://doi.org/10.1016/j.compfluid.2012.03.006.
[38] N. Krais, A. Beck, T. Bolemann, H. Frank, D. Flad, G. Gassner, F. Hindenlang, M. Hoffmann, T. Kuhn, M. Sonntag, C.-D. Munz, FLEXI: A high order discontinuous Galerkin framework for hyperbolic-parabolic conservation laws, Comput. Math. Appl. 81 (2021) 186–219, https://doi.org/10.1016/j.camwa.2020.05.004, arXiv:1910.02858.
[39] P. Ortwein, S.M. Copplestone, C.-D. Munz, T. Binder, W. Reschke, S. Fasoulas, A particle localization algorithm on unstructured curvilinear polynomial meshes, Comput. Phys. Commun. 235 (2019) 63–74, https://doi.org/10.1016/j.cpc.2018.09.024.
[40] P. Kopper, S.M. Copplestone, M. Pfeiffer, C. Koch, S. Fasoulas, A. Beck, Hybrid parallelization of Euler–Lagrange simulations based on MPI-3 shared memory, Adv. Eng. Softw. 174 (2022) 103291, https://doi.org/10.1016/j.advengsoft.2022.103291.
[41] Message Passing Interface Forum, MPI: A Message-Passing Interface Standard Version 4.0, 2021, https://www.mpi-forum.org/docs/mpi-4.0/mpi40-report.pdf.
[42] F. Hindenlang, T. Bolemann, C.D. Munz, Mesh Curving Techniques for High Order Discontinuous Galerkin Simulations, Springer International Publishing, Cham, 2015, pp. 133–152.
[43] B. Einfeldt, C. Munz, P. Roe, B. Sjögreen, On Godunov-type methods near low densities, J. Comput. Phys. 92 (1991) 273–295, https://doi.org/10.1016/0021-9991(91)90211-3.
[44] V.P. Kiselev, S.P. Kiselev, E.V. Vorozhtsov, Interaction of a shock wave with a particle cloud of finite size, Shock Waves 16 (2006) 53–64, https://doi.org/10.1007/s00193-006-0043-0.
[45] G.B. Jacobs, W.S. Don, T. Dittmann, High-order resolution Eulerian–Lagrangian simulations of particle dispersion in the accelerated flow behind a moving shock, Theor. Comput. Fluid Dyn. 26 (2012) 37–50, https://doi.org/10.1007/s00162-010-0214-6.
[46] S. Sundaram, L.R. Collins, Collision statistics in an isotropic particle-laden turbulent suspension. Part 1. Direct numerical simulations, J. Fluid Mech. 335 (1997) 75–109, https://doi.org/10.1017/S0022112096004454.
[47] D.L. Koch, Kinetic theory for a monodisperse gas-solid suspension, Phys. Fluids A 2 (1990) 1711–1723, https://doi.org/10.1063/1.857698.
[48] C.D. Pruett, T.B. Gatski, C.E. Grosch, W.D. Thacker, The temporally filtered Navier–Stokes equations: properties of the residual stress, Phys. Fluids 15 (2003) 2127–2140, https://doi.org/10.1063/1.1582858.
[49] M. Kurz, D. Kempf, M.P. Blind, P. Kopper, P. Offenhäuser, A. Schwarz, S. Starr, J. Keim, A. Beck, GALÆXI: solving complex compressible flows with high-order discontinuous Galerkin methods on accelerator-based systems, Comput. Phys. Commun. 306 (2025) 109388, https://doi.org/10.1016/j.cpc.2024.109388.
[50] C. Simmendinger, M. Marquardt, J. Mäder, R. Schneider, PowerSched - managing power consumption in overprovisioned systems, in: 2024 IEEE International Conference on Cluster Computing Workshops (CLUSTER Workshops), IEEE, 2024, pp. 1–8.
[51] G. Spur, E. Uhlmann, F. Elbing, Dry-ice blasting for cleaning: process, optimization and application, Wear 233–235 (1999) 402–411, https://doi.org/10.1016/S0043-1648(99)00204-5.
[52] Y.-H. Liu, G. Calvert, C. Hare, M. Ghadiri, S. Matsusaka, Size measurement of dry ice particles produced from liquid carbon dioxide, J. Aerosol Sci. 48 (2012) 1–9, https://doi.org/10.1016/j.jaerosci.2012.01.007.
[53] A.H. Nguyen, M. Mizoguchi, H. Itoh, Unsteady flow field around NACA0012 airfoil undergoing constant pitch rates at low Reynolds numbers, in: AIAA AVIATION 2020 FORUM, American Institute of Aeronautics and Astronautics, 2020.
[54] J. Liu, D. Li, Z. Zuo, C. Liu, H. Wang, Aerodynamic performance of a characteristic airfoil at low- Reynolds number and transonic flow under Mars sand-containing environment, Phys. Fluids 35 (2023), https://doi.org/10.1063/5.0158003.
[55] S.C.R. Rafkin, D. Banfield, On the problem of a variable Mars atmospheric composition in the determination of temperature and density from the adiabatic speed of sound, Planet. Space Sci. 193 (2020) 105064, https://doi.org/10.1016/j.pss.2020.105064.