

RICE UNIVERSITY

Multi-scale calculation based on dual domain
material point method combined with molecular
dynamics

by

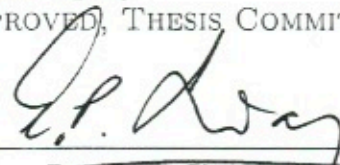
Tilak R. Dhakal

A THESIS SUBMITTED

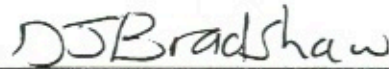
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Doctor of Philosophy

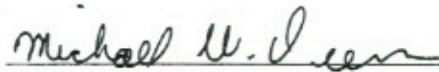
APPROVED, THESIS COMMITTEE:



Edison P. Liang, Chair
Andrew Hays Buchanan Professor of
Astrophysics



Stephen Bradshaw
Assistant Professor, Physics and
Astronomy



Michael W. Deem
John W. Cox Professor in Biochemical
and Genetic Engineering



Duan Z. Zhang
Staff Scientist, Los Alamos National
Laboratory

Marjorie D. Corcoran (deceased)
Professor of Physics and Astronomy

Houston, Texas

February, 2017

ABSTRACT

Multi-scale calculation based on dual domain material point method combined with
molecular dynamics

by

Tilak R. Dhakal

This dissertation combines the dual domain material point method (DDMP) with molecular dynamics (MD) in an attempt to create a multi-scale numerical method to simulate materials undergoing large deformations with high strain rates. In these types of problems, the material is often in a thermodynamically non-equilibrium state, and conventional constitutive relations are often not available. In this method, the closure quantities, such as stress, at each material point are calculated from a MD simulation of a group of atoms surrounding the material point. Rather than restricting the multi-scale simulation in a small spatial region, such as phase interfaces, or crack tips, this multi-scale method can be used to consider non-equilibrium thermodynamic effects in a macroscopic domain. This method takes advantage that the material points only communicate with mesh nodes, not among themselves; therefore MD simulations for material points can be performed independently in parallel.

First, using a one-dimensional shock problem as an example, the numerical properties of the original material point method (MPM), the generalized interpolation material point (GIMP) method, the convected particle domain interpolation (CPDI) method, and the DDMP method are investigated. Among these methods, only the DDMP method converges as the number of particles increases, but the large number

of particles needed for convergence makes the method very expensive especially in our multi-scale method where we calculate stress in each material point using MD simulation. To improve DDMP, the sub-point method is introduced in this dissertation, which provides high quality numerical solutions with a very small number of particles.

The multi-scale method based on DDMP with sub-points is successfully implemented for a one dimensional problem of shock wave propagation in a cerium crystal. The MD simulation to calculate stress in each material point is performed in GPU using CUDA to accelerate the computation. The numerical properties of the multi-scale method are investigated as well as the results from this multi-scale calculation are compared with direct MD simulation results to demonstrate the feasibility of the method. Also, the multi-scale method is applied for a two dimensional problem of jet formation around copper notch under a strong impact.

Acknowledgements

I would like to thank Physics and Astronomy Department of Rice University and Los Alamos National Laboratory for providing me the opportunity to pursue my graduate studies and research. Many thanks to my mentor Dr. Duan Z. Zhang, I appreciate your continuous support and encouragement. I would like to sincerely thank my thesis advisor Professor Edison P. Liang and advisory committee members Professor Pablo P. Yepes, Professor Marjorie Corcoran, Professor Michael W. Deems and Professor Stephen Bradshaw for your guidance. I also would like to thank my colleagues in Los Alamos National Laboratory T-3 group, Dr. Xia Ma, Dr. Christopher C. Long, Dr. Juan C. Padrino for all the helpful discussions.

I would like to specially thank my parents, my wife Shraddha Parajuli, my brother Badri R. Dhakal, my sister Tulsi D. Dhakal and all my family members for your selfless continuous support. I would also like to mention my friends Madan Lamichhane, Niraj Dhital, Kamal Dhungana, Bishnu Dahal, Binod Rai, Suman Khatiwada, Tej Lamichhane, Tirtha Joshi, Akhilesh Singh, Nirmal Ghimire, Krishna Acharya for your contribution towards the completion of my journey.

The research work conducted for this dissertation was supported by The Stockpile Safety and Surety Program, The Joint DoD/DoE Munitions Technology Development Program, the ASC program and the Next Generation Code program of Los Alamos National Laboratory.

Contents

Abstract	ii
List of Illustrations	vii
List of Tables	xi
1 Introduction	1
1.1 Multi-scale modeling: Review	1
1.2 Large material deformation under thermodynamic non-equilibrium . .	4
1.3 Continuum equation solution methods	7
2 Liouville Equation and Transport Equations	13
3 Material Point Methods	22
3.1 Original material point method	22
3.2 Material point methods: GIMP and CPDI	27
3.3 Dual domain material point method (DDMP)	30
4 Material point methods applied to one-dimensional shock waves and dual domain material point method with sub-points	35
4.1 Shock wave simulated using original MPM	35
4.2 Comparison of GIMP, CPDI and DDMP method	40
4.3 DDMP with Sub-points	45
4.4 Results using DDMP method with sub-points	50
4.5 Chapter Summary	52

5	Molecular Dynamics	55
5.1	Leap frog algorithm	56
5.2	Periodic boundary conditions	57
5.3	Embedded atom method	64
5.4	Stress calculation using EAM method potential	65
5.5	Molecular dynamics in GPU	67
6	Shock waves simulated using the dual domain material point method combined with molecular dynamics	71
6.1	Continuum level calculation	72
6.2	Molecular dynamics simulation of Cerium	74
6.3	Numerical Results and Discussion	77
6.4	Summary	88
7	Modeling of jet formation around copper notch using dual domain material point method combined with molec- ular dynamics	90
7.1	Molecular dynamics simulation of jet formation of copper	91
7.2	Multi-scale calculation for 2D problems	95
7.3	Numerical results and discussions	99
7.4	Chapter Summary	102
8	Conclusions	103
8.1	Summary	103
8.2	Future Work	107
	Bibliography	108

Illustrations

3.1	Typical initialization of material points in background mesh	26
3.2	(a) Illustration of shape function and (b) Gradient of the shape function used in traditional MPM (dotted line) and used in DDMP (solid line) at node i . Δx is the grid spacing.	27
4.1	Particles and cells in a one-dimensional weak shock calculation.	36
4.2	Particles pressure near one-dimensional shock front. Position 0 corresponds node j in Fig. 4.1.	38
4.3	Particle pressure calculated with MPM method using (a) 512 particles per cell, (b) 1024 particles per cell.	39
4.4	Particle pressure calculated with GIMP method using (a) 8 particles per cell, (b) 16 particles per cell, (c) 32 particles per cell, and (d) 64 particles per cell.	41
4.5	Particle pressure calculated with CPDI method using (a) 8 particles per cell, (b) 16 particles per cell, (c) 32 particles per cell, and (d) 64 particles per cell.	42
4.6	Particle pressure calculated with DDMP method using (a) 8 particles per cell, (b) 16 particles per cell, (c) 32 particles per cell, and (d) 64 particles per cell.	43

4.7	One-dimensional illustration of the sub-point algorithm. Four particles or material points are divided into 22 sub-points, represented by hollow or solid circles and squares. Each of these sub-points is an integration point in (6.2). The sub-point belonging to the same particle share the same value of the stress.	46
4.8	Results calculated using the sub-point method.	50
4.9	Comparison of strong shock results calculated using 2 particles per cell and 16 sub-points per particle with that obtained using 32 DDMP particles per cell (a). Strong shock results from CPDI method using 32 particles per cell (b).	51
4.10	Distribution of particles in the region of expanded material in the strong shock calculation using the DDMP method with sub-points at time $100\Delta x/c$. Initially two DDMP particles per cell are used. 16 sub-points per DDMP particle are used in this calculation.	52
5.1	Periodic boundary condition in 2 dimensions	58
5.2	The periodic boundary condition used to address shear deformation. The periodic domain is kept rectangular. For any particle x outside the rectangular domain is discarded and the image of x (i.e., y) is tracked.	62
5.3	Periodic boundary condition used for pure shear deformation. The particle and its image can both be inside the rectangular box.	63
5.4	Stress-strain Relation of Cerium at different temperatures. The strain used in this figure is the engineering strain $\varepsilon_{xx}^e = (L_{px}^n - L_{px}^0)/L_{px}^0$	66
5.5	Atoms in uniform grids. The atoms are later sorted according to the uniform grids.	69
6.1	Schematic diagram of the multi-scale method	72

6.2	Total energy per atom as a function of lattice constant for Cerium crystal	75
6.3	Shock wave propagation for weak shock at 0 K initial temperature calculated using 400 cells with 1 material point per cell and 8 sub-points per material point.	80
6.4	Propagation of strong shock at 0 K initial temperature calculated using 400 cells with 1 material point per cell and 8 sub-points per material point.	81
6.5	Propagation of strong shock at 300 K initial temperature calculated using 400 cells with 1 material point per cell and 8 sub-points per material point. In the combined DDMP-MD results, stresses at material points are obtained from the MD simulations using 256 and 500 atoms per material point.	82
6.6	Comparison of results from using the stress calculated from one MD time step stress and using the average stress over the one DDMP time step for cases of 0 K initial temperature.	84
6.7	Effects of artificial viscosity for cases of 0 K initial temperature.	85
6.8	Effects of number of material points or sub-points for cases of 0 K initial temperature.. . . .	86
6.9	Effect of mesh size in shock propagation at 0 K initial temperature calculated with 1 material point per cell and 8 sub-points per material point.	87
7.1	Initial configuration of the copper crystal sample with a piston and a notch	91
7.2	Total energy per atom as a function of lattice constant for copper crystal	92

7.3	The color plots of σ_{xx} component of stress field. Direct MD results are shown in (a) and (b) corresponding to $t = 10$ ps and $t = 20$ ps. Multi-scale DDMP-MD results are shown in (c) and (d) at $t = 10$ ps and $t = 20$ ps.	94
7.4	Jet formation in copper obtained by using direct MD simulation and the multi-scale calculation	100

Tables

7.1	Percentage errors of σ_{xx} calculated at $t = 20$ ps in comparison to direct MD simulation (A) For different particles per cell with 1250 cells, (B) For different number of cells with 4 particle per cell.	101
-----	--	-----

Chapter 1

Introduction

The equation of state or constitutive relation of a material is often obtained based on the assumption of thermodynamic equilibrium. As a consequence of this assumption, the equation of state can only be applied to the problems in which the time scale of the material deformation, defined by the inverse of the characteristic strain rate, is significantly larger than the time scale for the molecules in the material to relax to their thermodynamic equilibrium state after an external perturbation. For problems with a smaller ratio between the problem time scale and the relaxation time scale of the material, the equation of state cannot be used. These types of physical problems are the main focus of this dissertation. To address such problems involving materials undergoing extreme deformation, we combine dual domain material point method (DDMP) and molecular dynamics (MD) to build a novel multi-scale numerical method.

1.1 Multi-scale modeling: Review

Almost all the problems we encounter in science and engineering are multi-scale in nature. Materials are made up of atoms in microscopic level, and we characterize the materials in continuum level which are several order larger than the atomistic

level. Also the atomistic level processes occur in the order of femtoseconds whereas the time scale of physical problems occur in much slower pace. In many cases, the continuum characterization of material is sufficient to describe the material behavior. The effects of microscopic processes to the macroscopic processes are reflected in terms of constitutive relations or equations of state. There also exist many other problems, which are directly related to atomistic level processes, such as crack propagation in solids. Atomistic level calculation such as molecular dynamics can describe such phenomena very accurately, but it is impractical to perform such calculations in a domain of a material of engineering interest. Multi-scale methods which couple the atomistic level calculation with the continuum description of the material has become a popular choice to accurately and efficiently model such problems.

Multi-scale method was first originated in US DOE national labs to replace underground nuclear tests with simulation based experiments in mid 1980s [1]. Many of the early multi-scale methods are based on coarse-graining the energy [2]. One example is the quasicontinuum method [3] where a coarse grained hamiltonian is formed and the hamiltonian is minimized to find the equilibrium state. The quasicontinuum method is successfully applied to study the quasistatic structure of solids at zero temperature [4]. Coarse-grained molecular dynamics is another popular method involving coarse-grained hamiltonian [5]. Besides coarse-grained methods, the domain decomposition method [6] is another popular multi-scale method. In this method, the computational domain is decomposed into atomistic regions and continuum regions, and some

scale-bridging method is used at the interface. For modeling fluids, the atomistic and continuum domains can overlap. The overlapped domain can be used to validate the model by comparing the results from microscopic and macroscopic calculation. For modeling for solids, the two domains usually do not overlap.

The multi-scale modeling used in this dissertation is related to so called heterogeneous multi-scale method (HMM) [7]. The general strategy of HMM method is to start with a macroscopic solver, with missing macroscopic data, such as constitutive relation, calculated using microscopic calculation. Unlike in domain decomposition method, in HMM strategy, the macroscale solver is used over the whole domain. The HMM method can be applied to two different kind of problems, type A and type B problems. In type A problems, there are isolated defects, such as cracks or dislocations in solids, where macroscale solver requires coupling with microscale calculation near the defects to accurately model the material. Away from the defects, continuum model can be used. In type B problems, the constitutive relations to drive the macroscale solver are missing. Microscopic calculations are performed to obtain the closure quantities, such as stress. The multi-scale method studied in this dissertation can be categorized as type B HMM method, where we use molecular dynamics simulation to obtain stress bypassing the need for a constitutive relation for materials undergoing extreme deformation in thermodynamically non-equilibrium states.

1.2 Large material deformation under thermodynamic non-equilibrium

Since the time scale of the material deformation of concern in this dissertation is comparable or even smaller than the thermodynamic relaxation time, the material response at a given external disturbance depends on the micro structures of the material at that time, which depends on the history of the material. Therefore, the material behavior is generally history dependent for these types of problems. Furthermore, such high rates of material deformations often involve significant forces, for instance, in the cases of hypervelocity impacts. Significant material deformations are common in these cases.

To numerically simulate these types of problems at the macroscopic level, on the theoretical front, one needs a set of continuum scale equations with a method to calculate the closure quantities accurately representing the micro structure and history effects. On the numerical front, one needs a numerical method capable of accurately tracking material deformation history in cases of large material deformation.

The continuum scale equations can be derived from the Liouville equation using ensemble average for fairly general cases [8, 9]. The closure quantities, such as pressure, stress, and heat flux, are directly related to microscopic interactions among molecules and atoms. In this dissertation, we use these relations to calculate the necessary closure quantities directly from molecular dynamics (MD) simulations. Such closure quantities are used in continuum calculations to advance the macroscopic

quantities. For thermodynamically equilibrium systems in the limit of Continuum Mechanics, we show that such calculated stress is consistent with the traditional concept. The method presented in the current paper is in the same philosophy of the heterogeneous multiscale method (HMM) [10, 11, 12]. For selected regions in a macroscopic domain, lower length scale simulations are performed in synchrony with the macroscopic continuum level calculation. These lower length scale simulations are constrained by the macroscopic information passed to them. Closure quantities, such as stress or heat flux, calculated from these lower length scale systems are passed to the continuum level calculation to advance the macroscopic state. In this thesis work we emphasize the simulation method for history dependent systems undergoing large macroscopic deformations. Frequent reinitialization and time averaging are not allowed. While the non-linear terms are neglected in [10] and the work is limited to small material deformation, the Lagrangian coordinate system is capable of tracking material deformation history, which is an advantage of a Lagrangian method. The main disadvantage of a pure Lagrangian method, such as the Lagrangian finite element method, appears in problems involving large material deformation, where elements are often severely distorted and even entangled leading to loss of accuracy and eventual failure of the calculation. In this dissertation, we use the dual domain material point (DDMP) method to avoid this numerical difficulty. For the fluid dynamics problems studied in [11], the simulated molecular dynamics systems are fixed in the spatial locations intended for history independent materials, such as Newtonian

fluids, based on the pure Eulerian description. Unless special treatments, such as in [13, 14], are used, Eulerian methods often lead to significant numerical diffusion of history dependent variables. Rather than proposing another multiscale computation philosophy or a method to perform MD simulations, the main purpose of this work is to show that the general strategy of HMM can be efficiently implemented with the dual domain material point (DDMP) method for thermodynamically nonequilibrium or history dependent systems undergoing large material deformation by taking advantage of the simultaneous use of Lagrangian particles and Eulerian meshes in the method.

The Lagrangian particles, also called material points, are used to track the history of the material deformation. These particles can be regarded as pieces of the material experiencing the history. The Eulerian mesh is used to perform numerical analysis, such as taking spatial derivatives. Since the information about material history is carried by the Lagrangian particles, numerical diffusion for the history dependent quantities is avoided. Furthermore, because the mesh is Eulerian, it remains unchanged in cases of large material deformations, thereby avoiding the mesh distortion difficulty suffered by a pure Lagrangian method. In the DDMP method, the closure quantities, such as the stress, are needed only at material points following the motion of the material. The numerical method does not require direct molecular dynamics simulation of the entire macroscopic computational domain. Direct molecular dynamics simulations are only performed in small representative domains

around the material points. These simulations are driven by the continuum level DDMP calculation by imposing appropriate constraints to the molecular dynamics systems. The closure quantities calculated from these molecular systems are used to drive the macroscopic DDMP calculation. Such intimate communications between the macroscopic scale calculation and the microscopic scale simulations are unique for these types of multiscale calculations and need to be studied carefully to ensure physical consistency of the results.

In the DDMP method, material points only communicate with mesh nodes, and not directly among themselves. Therefore, there is no need to form neighbor lists of the particles. This feature of the numerical method is very advantageous for parallel computation, especially for heterogeneous computers, such as CPU-GPU combined platforms. In this dissertation, we take advantage of this feature and demonstrate the feasibility of such multiscale calculations.

1.3 Continuum equation solution methods

Since its first introduction [15, 16], the material point method has been used in many problems involving large material deformations [17, 18, 19, 20, 21, 22] in which a traditional finite element method encounters difficulties due to mesh or element distortion. The material point method is similar to finite element method. Both of them seek approximate weak solutions to the partial differential equations but there are two significant differences that result in different numerical properties of the methods.

The first difference is that the finite element method is a pure Lagrangian method in which elements move and deform with the material, while the material point method uses an arbitrary Eulerian mesh, which may stay fixed during a material motion and avoids the mesh or element distortion in cases of large material deformation. The Lagrangian capability of the material point method resides in its use of material points to carry history dependent quantities.

The second difference is the numerical integration method used to approximate the inner product of two functions. In the finite element method the integration for the internal force calculation is often calculated using the Gaussian quadrature method with Gauss integration points specified at given logical coordinates of the element for the purpose of accuracy. In the material point method, the material points are used not only to carry important field quantities, such as stress and damage of the material, but also used as integration points. Unlike in the finite element method, we do not have control on locations of these material points. In the material point method, the numerical integration in the internal force calculation is approximated by a low accuracy Riemann sum.

Comparing to the finite element method, the freedom or advantage of the material point method is gained at the cost of numerical accuracy and smoothness in the internal force calculation. One problem caused by this low accuracy integration is now the well known cell crossing noise [23] of the method. The generalized interpolation material point method (GIMP) reduces this noise. In this version of the

material point method, the concept of particle domain is introduced to perform integrations in the calculation of the nodal mass and the internal force. The smoothing effect is provided by averaging or integrating over the particle domain. To conserve momentum, the GIMP method requires that the particle domains cover the entire computational domain without overlap. This requirement is also called the partition of unity, since this is equivalent to requiring that the sum of the particle characteristic functions equals to unity everywhere in the computational domain. It is rather straightforward to satisfy this requirement in one-dimensional problems. For two- or three-dimensional problems with a significant material deformation, it is nearly impossible to numerically satisfy this requirement. This is similar to the difficulty encountered by the finite element methods due to element distortion. To alleviate this difficulty, there are two approximate versions of the GIMP method, uGIMP and cpGIMP [23]. In uGIMP, the partition of unity requirement is simply ignored, and the particle domains are assumed unchanged during the material deformation; therefore it can only be applied to problems with small material deformations. In cpGIMP, the particle domains are assumed to be rectangles all the time. Only deformations caused by the diagonal components of the deformation gradient are considered in the change of the particle domain. Therefore, the method cannot be used for problems with strong shear or rotation.

To improve this situation, another version of the material point method, called the convected particle domain interpolation (CPDI) [24] method, allows significant

rotation and shear deformation by using parallelograms (in two-dimension) and parallelepipeds (in three-dimension) to cover the computational domain. The parallelograms or parallelepipeds can move, rotate, and deform according to the velocity and the velocity gradient at the center. In this way, the CPDI method provides a local linear approximation to the deformation field in the calculation of the particle domains, but gaps and overlaps still happen among the deformed parallelograms and parallelepipeds.

Both the GIMP and CPDI methods rely on averages over particle domains to provide the smoothing effect in reducing the numerical noise caused by cell crossing of particles. In these methods, the use of particle domain introduces a new numerical length scale in addition to the mesh size and has numerical consequences. For instance, if the domain size is reduced, the smoothing effects are also reduced. As we show in Chapter 4 these methods encounter convergence issues as the number of particles is increased.

To eliminate the cell crossing noise, the third version of the improved material point method, called the dual domain material point (DDMP) method, uses an equivalent stress field to calculate the internal force. The equivalent stress is constructed using the stress at the material points and is only different from the original stress field by an amount of order of $(\Delta x)^2$ in the sense of weak solution, where Δx is the typical mesh size. The internal force calculated using such a stress field is a linear combination of the products of the particle volumes and the stresses, and the coeffi-

cients in this linear combination can be then regarded as the modified gradient of the shape function. The equivalent stress field is constructed in such a way that the corresponding modified gradient of the shape function is continuous on cell boundaries. Unaltered from the original MPM, material points in the DDMP method serve two roles: as Lagrangian markers carrying history information and as integration points. These two roles of material points in DDMP are separated in this work to further improve the performance of DDMP by using sub-points as described in Chapter 4.3. In this new method the material points or particles are used to carry physical quantities and history information, while the numerical integration is done by using the sub-points generated at each time step. This improved DDMP method preserves the conservation properties of the DDMP method without encountering issues of partition of unity as in the GIMP and CPDI methods. Mass and momentum are conserved exactly, and energy conservation is second order both in spatial and time discretizations to machine accuracy.

To investigate the properties of different versions of the material point method, we use them to calculate the propagation of one-dimensional weak isothermal shock waves in an ideal gas. Although this problem is very simple and even unphysical, it actually reveals many numerical properties of the MPM methods. One objective of this dissertation is to improve the material point method by exploiting advantages of different versions of the material point method, while avoiding their disadvantages. In this dissertation, the terms material points and particles are used interchangeably.

The improved DDMP method will be used as continuum solver in our multi-scale numerical modeling effort, since the main objective of this work is to develop a multi-scale method applicable to macroscopic problems.

To describe our multi-scale method, we start by examining the validity of continuum formulations. To study its applicability to thermodynamically non-equilibrium systems, we derive them from molecular dynamics.

Chapter 2

Liouville Equation and Transport Equations

Starting from the Liouville equation, one can derive the continuum level mass, momentum, and energy equations with the closure quantities directly related to interactions at the microscopic level [8, 9].

In classical Newtonian mechanics, a system can be completely and uniquely determined by a point $\mathcal{C} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N; \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_N\}$, in the $6N$ dimensional position-velocity phase space. Let $P(\mathcal{C}, t)$ be the probability density function corresponding to configuration \mathcal{C} so that $P(\mathcal{C}, t)d\mathcal{C}$ gives the probability of finding such a system with configuration \mathcal{C} at time t , where $d\mathcal{C} = d\mathbf{x}_1 \dots d\mathbf{x}_N d\mathbf{v}_1 \dots d\mathbf{v}_N$. The Liouville Equation describes the evolution of probability density function $P(\mathcal{C}, t)$ as,

$$\frac{\partial P}{\partial t} + \sum_{i=1}^N \nabla_{\mathbf{x}_i} \cdot (\mathbf{v}_i P) + \sum_{i=1}^N \nabla_{\mathbf{v}_i} \cdot (\dot{\mathbf{v}}_i P) = 0 \quad (2.1)$$

The Liouville equation is simply a statement of conservation of total number of states in phase space. In other words, it states that phase space points are neither created nor destroyed. A complete knowledge of the probability density function enables us to calculate any averaged physical quantities which are functions of position and/or velocity coordinates.

For a generic quantity g_α pertaining to atom α , the corresponding average \bar{g} at

time t and position \mathbf{x} is defined as,

$$n(\mathbf{x}, t)\bar{g}(\mathbf{x}, t) = \int \sum_{\alpha=1}^N \delta(\mathbf{x}_\alpha(t) - \mathbf{x})g_\alpha(\mathcal{C}, t)P(\mathcal{C}, t)d\mathcal{C}, \quad (2.2)$$

where n is the number density of the atoms which corresponds to $g_\alpha = \bar{g} = 1$ in the above equation, i.e.,

$$n(\mathbf{x}, t) = \int \sum_{\alpha=1}^N \delta(\mathbf{x}_\alpha(t) - \mathbf{x})P(\mathcal{C}, t)d\mathcal{C}. \quad (2.3)$$

Taking moment of equation (2.1) yields continuum equations [8, 9], e.g. for any quantity $g_j(\mathcal{C}, t)$ pertaining to a particle j , multiplying equation (2.1) by $\delta(\mathbf{x} - \mathbf{x}_j)g_j(\mathcal{C}, t)$, summing over all the atoms in the system and then integrating over all possible configurations $\mathcal{C}'s$,

$$\int \sum_j \delta(\mathbf{x} - \mathbf{x}_j)g_j \left\{ \frac{\partial P}{\partial t} + \sum_{i=1}^N \nabla_{\mathbf{x}_i} \cdot (\mathbf{v}_i P) + \sum_{i=1}^N \nabla_{\mathbf{v}_i} \cdot (\dot{\mathbf{v}}_i P) \right\} d\mathcal{C} = 0. \quad (2.4)$$

The first term on the left hand side of equation (2.4),

$$\begin{aligned} \int \sum_j \delta(\mathbf{x} - \mathbf{x}_j)g_j \frac{\partial P}{\partial t} d\mathcal{C} &= \frac{\partial}{\partial t} \int \sum_j \delta(\mathbf{x} - \mathbf{x}_j)g_j P d\mathcal{C} - \int \sum_j P \delta(\mathbf{x} - \mathbf{x}_j) \frac{\partial g_j}{\partial t}, \\ &= \frac{\partial(n\bar{g})}{\partial t} - \int \sum_j P \delta(\mathbf{x} - \mathbf{x}_j) \frac{\partial g_j}{\partial t}, \end{aligned} \quad (2.5)$$

where equation (2.2) is used to define $n\bar{g}$. The second term of equation (2.4),

$$\begin{aligned} \int \sum_j \delta(\mathbf{x} - \mathbf{x}_j)g_j \sum_i \nabla_{\mathbf{x}_i} \cdot (\mathbf{v}_i P) d\mathcal{C} &= \int \sum_i \nabla_{\mathbf{x}_i} \cdot \left[\mathbf{v}_i P \sum_j \delta(\mathbf{x} - \mathbf{x}_j)g_j \right] d\mathcal{C} \\ &\quad - \int \sum_i \mathbf{v}_i P \nabla_{\mathbf{x}_i} \cdot \left[\sum_j \delta(\mathbf{x} - \mathbf{x}_j)g_j \right] d\mathcal{C}. \end{aligned} \quad (2.6)$$

After using Gauss divergence theorem, the first term on the right hand side of above equation becomes a surface integral, which vanishes because $P(\mathbf{x}_1, \dots, \mathbf{x}_{i-1}, \pm\infty, \mathbf{x}_{i+1}, \dots, \mathbf{x}_N; \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_N) = 0$. So we can write,

$$\begin{aligned} \int \sum_j \delta(\mathbf{x} - \mathbf{x}_j) g_j \sum_i \nabla \mathbf{x}_i \cdot (\mathbf{v}_i P) d\mathcal{C} &= - \sum_j \sum_i \int P \mathbf{v}_i (\nabla \mathbf{x}_i \cdot g_j) \delta(\mathbf{x} - \mathbf{x}_j) d\mathcal{C} \\ &+ \int \sum_i P \mathbf{v}_i g_i \nabla \mathbf{x} \cdot \delta(\mathbf{x} - \mathbf{x}_i) d\mathcal{C} \end{aligned} \quad (2.7)$$

Here we used $\nabla \mathbf{x}_i \cdot \delta(\mathbf{x} - \mathbf{x}_i) = -\nabla \mathbf{x} \cdot \delta(\mathbf{x} - \mathbf{x}_i)$. The third term of equation (2.4) is simplified as,

$$\begin{aligned} &\int \sum_i \nabla \mathbf{v}_i \cdot (\dot{\mathbf{v}}_i P) \sum_j \delta(\mathbf{x} - \mathbf{x}_j) g_j d\mathcal{C} \\ &= \int \sum_i \nabla \mathbf{v}_i \cdot \left[\dot{\mathbf{v}}_i P \sum_j \delta(\mathbf{x} - \mathbf{x}_j) g_j \right] d\mathcal{C} - \int \sum_i P \dot{\mathbf{v}}_i \nabla \mathbf{v}_i \cdot \left[\sum_j \delta(\mathbf{x} - \mathbf{x}_j) g_j \right] d\mathcal{C} \\ &= - \sum_j \sum_i \int P \dot{\mathbf{v}}_i (\nabla \mathbf{v}_i \cdot g_j) \delta(\mathbf{x} - \mathbf{x}_j) d\mathcal{C}. \end{aligned} \quad (2.8)$$

Here we used the similar argument that the surface integral vanishes after using Gauss divergence theorem. Substituting equations (2.5), (2.7) and (2.8) back into equation (2.4), after exchanging order of $\nabla \mathbf{x}$ with the integration, one finds,

$$\begin{aligned} &\frac{\partial(n\bar{g})}{\partial t} + \nabla \mathbf{x} \cdot \int \sum_i P \mathbf{v}_i g_i \delta(\mathbf{x} - \mathbf{x}_i) d\mathcal{C} \\ &= \int P \sum_j \left[\frac{\partial g_j}{\partial t} + \sum_i \mathbf{v}_i \nabla \mathbf{x}_i \cdot g_j + \sum_i \dot{\mathbf{v}}_i \nabla \mathbf{v}_i \cdot g_j \right] \delta(\mathbf{x} - \mathbf{x}_j) d\mathcal{C}. \end{aligned} \quad (2.9)$$

Using definition (2.2), equation (2.9) becomes a transport equation for a quantity g ,

Finally, the continuum level transport equation for g can be written as,

$$\frac{\partial(n\bar{g})}{\partial t} + \nabla \cdot (n\bar{g}\bar{\mathbf{v}}) = n\frac{d\bar{g}}{dt}. \quad (2.10)$$

Here, we used the definition of an averaged quantity as in equation (2.2) for $n\frac{d\bar{g}}{dt}$ noting that $g_j(\mathcal{C}, t) = g_j(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N; \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_N, t)$, the quantity inside the square bracket of equation (2.9) is the total time derivative $\frac{dg_j}{dt}$. Let $g_j = m_j$ (mass) in equation (2.10), we find the mass conservation equation,

$$\begin{aligned} \frac{\partial(n\bar{m})}{\partial t} + \nabla \cdot (n\bar{m}\bar{\mathbf{v}}) &= 0 \\ \frac{\partial\rho}{\partial t} + \nabla \cdot (\rho\tilde{\mathbf{v}}) &= 0 \end{aligned} \quad (2.11)$$

Here, the mass density ρ is defined as $\rho = n\bar{m}$. The Favre average velocity $\tilde{\mathbf{v}}$ is defined as $\tilde{\mathbf{v}} = n\bar{m}\bar{\mathbf{v}}/\rho$.

Similarly, letting $g_j = m_j\mathbf{v}_j$ (momentum), one finds the momentum equation,

$$\frac{\partial\rho\tilde{\mathbf{v}}}{\partial t} + \nabla \cdot (\rho\tilde{\mathbf{v}}\tilde{\mathbf{v}}) = \nabla \cdot \boldsymbol{\sigma}_k + n\bar{\mathbf{f}} + \rho\mathbf{b}, \quad (2.12)$$

where \mathbf{b} is the body force, $\boldsymbol{\sigma}_k$ is the stress due to velocity fluctuations,

$$\boldsymbol{\sigma}_k(\mathbf{x}, t) = - \int \sum_{\alpha=1}^N \delta(\mathbf{x}_\alpha - \mathbf{x}) m_\alpha (\mathbf{v}_\alpha - \tilde{\mathbf{v}})(\mathbf{v}_\alpha - \tilde{\mathbf{v}}) P(\mathcal{C}, t) d\mathcal{C}, \quad (2.13)$$

and

$$n\bar{\mathbf{f}} = \int \sum_{\alpha=1}^N \delta(\mathbf{x}_\alpha - \mathbf{x}) \mathbf{f}_\alpha(\mathcal{C}, t) P(\mathcal{C}, t) d\mathcal{C}, \quad (2.14)$$

with $\mathbf{f}_\alpha(\mathcal{C}, t)$ being the total force acting on atom α by other atoms in configuration \mathcal{C} . Let $\mathbf{f}_{\alpha\beta}$ be the force atom β acting on α . The total force can be written as

$$\mathbf{f}_\alpha = \sum_{\beta=1}^N \mathbf{f}_{\alpha\beta}. \quad (2.15)$$

With this decomposition of force, equation (2.14) can be written as,

$$n\bar{\mathbf{f}} = \int \sum_{\alpha=1}^N \sum_{\beta=1}^N \delta(\mathbf{x}_\alpha - \mathbf{x}) \mathbf{f}_{\alpha\beta} P(\mathcal{C}, t) d\mathcal{C}, \quad (2.16)$$

$$n\bar{\mathbf{f}} = \frac{1}{2} \int \sum_{\alpha=1}^N \sum_{\beta=1}^N [\delta(\mathbf{x}_\alpha - \mathbf{x}) - \delta(\mathbf{x}_\beta - \mathbf{x})] \mathbf{f}_{\alpha\beta} P(\mathcal{C}, t) d\mathcal{C}. \quad (2.17)$$

Here, we used $\mathbf{f}_{\alpha\beta} = -\mathbf{f}_{\beta\alpha}$. Using the theorem in Appendix A of [25], we can write,

$$n\bar{\mathbf{f}} = \nabla \cdot \boldsymbol{\sigma}_f, \quad (2.18)$$

where

$$\boldsymbol{\sigma}_f(\mathbf{x}, t) = \frac{1}{2} \int \sum_{\alpha=1}^N \sum_{\beta=1}^N \delta(\xi \mathbf{x}_\beta + (1 - \xi) \mathbf{x}_\alpha - \mathbf{x}) (\mathbf{x}_\beta - \mathbf{x}_\alpha) \mathbf{f}_{\alpha\beta}(\mathcal{C}, t) P(\mathcal{C}, t) d\mathcal{C}, \quad (2.19)$$

is the stress due to interaction forces among the atoms with $0 \leq \xi = \xi(\mathcal{C}, t) \leq 1$.

Here ξ determines Using (2.18), we can write momentum equation (2.12) in terms of the total Cauchy stress $\boldsymbol{\sigma} = \boldsymbol{\sigma}_k + \boldsymbol{\sigma}_f$,

$$\frac{\partial \rho \tilde{\mathbf{v}}}{\partial t} + \nabla \cdot (\rho \tilde{\mathbf{v}} \tilde{\mathbf{v}}) = \nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{b}. \quad (2.20)$$

Although the derivation of the stress in [8, 9] uses the abstract Taylor expansion of the δ -function, it is equivalent to translating configuration \mathcal{C} by a small distance of order of atomic length scale ℓ_a in the physical space [25, 26], assuming the probability distribution function $P(\mathcal{C}, t)$ is differentiable with respect to this translation. This assumption implies separation between the atomic length scale ℓ_a and the macroscopic length scale L_m ($\ell_a \ll L_m$). Under this assumption, within an error of order $(\ell_a/L_m)^2$ in momentum equation (2.20), we can set $\xi = 0$ in (2.19). In the derivation of

these continuum scale equations, the assumption of thermodynamic equilibrium is not necessary, therefore these equations are applicable to material deformations under high strain rates.

The stress $\boldsymbol{\sigma}_f$ defined in (2.19) is not the only one satisfying (2.18). Stress $\boldsymbol{\sigma}_f$ can differ by a divergence free tensor without a consequence to the dynamics of the system. Furthermore, for a given potential among atoms, the stress defined in (2.19) is still not unique, because there are different ways [27, 28] of decomposing the force acting on an atom into pair interaction forces while yielding the same total force in (2.15). Different decompositions lead to different $\mathbf{f}_{\alpha\beta}$, and different stresses calculated from (2.19). In continuum mechanics, the difference in the divergence free stress is often regarded as a residual stress, which depends on the definition of the “undeformed”, or the reference state.

In principle, this non-uniqueness does not have a consequence for us to perform the multiscale calculation described in this thesis work, because it is the divergence of the stress, not the stress itself, drives the evolution of the macroscopic momentum as in (2.20). However, it is still desirable, at least comforting, to know that this stress is consistent with the stress defined in continuum mechanics at the limit of thermodynamic equilibrium.

Let \mathbf{x} be the geometric center of a representative volume V_c with length scale $L_c (\gg \ell_a)$. If $L_c \ll L_m$, we have

$$\boldsymbol{\sigma}(\mathbf{x}, t) = \overline{\boldsymbol{\sigma}}^v(\mathbf{x}, t) + O\left(\frac{L_c^2}{L_m^2}\right) = \overline{\boldsymbol{\sigma}}_k^v(\mathbf{x}, t) + \overline{\boldsymbol{\sigma}}_f^v(\mathbf{x}, t) + O\left(\frac{L_c^2}{L_m^2}\right), \quad (2.21)$$

where over bar with superscript v , $(\bar{\cdot}^v)$, denotes average over the representative volume V_c . Using (2.19) after exchanging the order of the volume and ensemble averages, (the order of the probability integration with the volume integration), we have

$$\begin{aligned}\bar{\boldsymbol{\sigma}}^v(\mathbf{x}, t) &= \frac{1}{V_c} \int_{V_c} \boldsymbol{\sigma}(\mathbf{y}, t) d^3y \\ &= \int \frac{1}{V_c} \sum_{\mathbf{x}_\alpha \in V_c} \left[m_\alpha (\mathbf{v}_\alpha - \tilde{\mathbf{v}}) (\mathbf{v}_\alpha - \tilde{\mathbf{v}}) + \frac{1}{2} \sum_{\beta} (\mathbf{x}_\beta - \mathbf{x}_\alpha) \mathbf{f}_{\alpha\beta} \right] P(\mathcal{C}, t) d\mathcal{C} + O\left(\frac{\ell_a}{L_m}\right),\end{aligned}\tag{2.22}$$

where V_c is the representative volume, the first summation is over all the atoms in the representative volume V_c , and the second summation is over all the atoms interacting with atom α ($\mathbf{f}_{\alpha\beta} \neq \mathbf{0}$). If atom α is close to a boundary of V_c , atom β could be outside of the volume, but must be within the vicinity, (within a distance of order ℓ_a), of the boundary, if the range of the interaction forces is of that order. Such cross-boundary interacting atom pairs are located in the region near the boundary. The volume of the boundary region is of order $\ell_a L_c^2$. Neglecting such pair interactions in the boundary region causes an error of order ℓ_a/L_c in the stress, which is insignificant for $L_c \gg \ell_a$. Allowing this error, to calculate the volume averaged stress $\bar{\boldsymbol{\sigma}}^v$, we can only consider the interaction pairs inside V_c and approximate \sum_{β} in (2.22) with $\sum_{\mathbf{x}_\beta \in V_c}$. Noting $\mathbf{f}_{\alpha\beta} = -\mathbf{f}_{\beta\alpha}$, we have

$$\sum_{\mathbf{x}_\alpha \in V_c} \sum_{\mathbf{x}_\beta \in V_c} (\mathbf{x}_\beta - \mathbf{x}_\alpha) \mathbf{f}_{\alpha\beta} = - \sum_{\mathbf{x}_\beta \in V_c} \mathbf{x}_\beta \sum_{\mathbf{x}_\alpha \in V_c} \mathbf{f}_{\beta\alpha} - \sum_{\mathbf{x}_\alpha \in V_c} \mathbf{x}_\alpha \sum_{\mathbf{x}_\beta \in V_c} \mathbf{f}_{\alpha\beta} = -2 \sum_{\mathbf{x}_\alpha \in V_c} \mathbf{x}_\alpha \mathbf{f}_\alpha^{int},\tag{2.23}$$

where $\mathbf{f}_\alpha^{int} = \sum_{\mathbf{x}_\beta \in V_c} \mathbf{f}_{\alpha\beta}$ is the sum of the forces from the atoms inside the representative volume, or the total internal force. Using this relation in (2.22), under condition $\ell_a \ll L_c \ll L_m$ we find

$$\bar{\boldsymbol{\sigma}}^v = \int \frac{1}{V_c} \sum_{\mathbf{x}_\alpha \in V_c} [m_\alpha(\mathbf{v}_\alpha - \tilde{\mathbf{v}})(\mathbf{v}_\alpha - \tilde{\mathbf{v}}) + \mathbf{x}_\alpha \mathbf{f}_\alpha^{int}] P(\mathcal{C}, t) d\mathcal{C} + O\left(\frac{L_c^2}{L_m^2}\right) + O\left(\frac{\ell_a}{L_c}\right). \quad (2.24)$$

In almost all practical cases, when the representative volume is sufficiently large ($L_c \gg \ell_a$), the volume averaged value in (2.24) or (2.22) becomes insensitive to configuration \mathcal{C} . Noting that $\int P(\mathcal{C}, t) d\mathcal{C} = 1$, using (2.21), (2.22) and (2.24) under condition $\ell_a \ll L_c \ll L_m$ we have,

$$\begin{aligned} \boldsymbol{\sigma} &\approx \bar{\boldsymbol{\sigma}}^v \approx \frac{1}{V_c} \sum_{\mathbf{x}_\alpha \in V_c} \left[m_\alpha(\mathbf{v}_\alpha - \tilde{\mathbf{v}})(\mathbf{v}_\alpha - \tilde{\mathbf{v}}) + \frac{1}{2} \sum_{\beta} (\mathbf{x}_\beta - \mathbf{x}_\alpha) \mathbf{f}_{\alpha\beta} \right] \\ &= \frac{1}{V_c} \sum_{\mathbf{x}_\alpha \in V_c} [m_\alpha(\mathbf{v}_\alpha - \tilde{\mathbf{v}})(\mathbf{v}_\alpha - \tilde{\mathbf{v}}) + \mathbf{x}_\alpha \mathbf{f}_\alpha^{int}] + O\left(\frac{\ell_a}{L_c}\right). \end{aligned} \quad (2.25)$$

In deriving these relations, we have only used the separation of the atomic and the macroscopic length scales. This length scale separation allows us to calculate the stress as an average over a representative volume with a length scale between the atomic and the macroscopic length scales without explicitly specifying the probability distribution function $P(\mathcal{C}, t)$.

The second line of (2.25) is the virial expression of the Cauchy stress. In continuum mechanics under the assumption of thermodynamic equilibrium, the expression can be obtained by differentiating the Helmholtz free energy with respect to the deformation gradient tensor [27, 29]. Therefore, the stress $\boldsymbol{\sigma}$ used in this work is consistent with

the stress in classical continuum mechanics.

In our MD simulation, the representative volume is a periodic box described in Chapter 5.2. To avoid the error related to interaction pairs near the boundaries in our computational domain and to take advantage of the periodic boundary conditions, the potential part of stress $\bar{\sigma}^v$ is calculated using the first line of (2.25) with pair interaction forces.

Chapter 3

Material Point Methods

Material point method (MPM) is an advanced form of particle in cell method [15], developed by Frank Harlow. The current version of MPM is developed by Sulsky *et al.* [16]. Similar to finite element method (FEM), MPM is also based on weak formulation of partial differential equations. In MPM, the transport equations are solved in the predefined background grids. The material points are used as integration points as in FEM. Those grids can remain fixed or they can be reinitialized, and the material points move to follow the motion of the material. Therefore MPM uses both Lagrangian and Eulerian descriptions, making this method suitable for large deformation problems.

3.1 Original material point method

Let us write the momentum equation (2.20) in lagrangian form and neglecting the body force as,

$$\rho \frac{d\mathbf{v}}{dt} - \nabla \cdot \boldsymbol{\sigma} = 0. \quad (3.1)$$

For an arbitrary continuous function ϕ , we have,

$$\int (\rho \frac{d\mathbf{v}}{dt} - \nabla \cdot \boldsymbol{\sigma}) \phi dV = 0. \quad (3.2)$$

Above equation is the weak formulation of the momentum equation (3.1). Let us discretize \mathbf{v} and ϕ using shape functions as, $\mathbf{v} = \sum_{i=1}^N \mathbf{v}_i S_i(x)$ and $\phi = \sum_{i=1}^N \phi_i S_i(x)$, where N is the number of mesh nodes, \mathbf{v}_i is the value of \mathbf{v} at node i , ϕ_i is the value of ϕ at node i , S_i is the shape function associated with the node i . Equation (3.2) can now be written as,

$$\sum_i \phi_i \sum_j \frac{d\mathbf{v}_j}{dt} \int \rho S_j(\mathbf{x}) S_i(\mathbf{x}) dV - \sum_i \phi_i \int (\nabla \cdot \boldsymbol{\sigma}) S_i(\mathbf{x}) dV = 0. \quad (3.3)$$

$$\sum_i \phi_i \sum_j \frac{d\mathbf{v}_j}{dt} M_{ij} - \sum_i \phi_i \int \left\{ \nabla \cdot [\boldsymbol{\sigma} S_i(\mathbf{x})] - \boldsymbol{\sigma} \cdot \nabla S_i(\mathbf{x}) \right\} dV = 0. \quad (3.4)$$

After using Gauss theorem in the first term inside the integral and since ϕ_i is an arbitrary function, we can write,

$$\sum_{j=1}^N M_{ij} \frac{d\mathbf{v}_j}{dt} - \int \boldsymbol{\sigma} \cdot \mathbf{n} S_i(\mathbf{x}) dA + \mathbf{f}_i^{int} = 0, \quad (3.5)$$

where \mathbf{n} is the outward normal on the boundary of the computational domain, dA is an element of surface area, M_{ij} is an element of the mass matrix, \mathbf{f}_i^{int} is the internal force. The mass matrix element M_{ij} is defined as

$$M_{ij} = \int \rho S_i(\mathbf{x}) S_j(\mathbf{x}) dV, \quad (3.6)$$

and the internal force takes the form

$$\mathbf{f}_i^{int} = - \int \boldsymbol{\sigma} \cdot \nabla S_i(\mathbf{x}) dV. \quad (3.7)$$

By dividing the computational domain into material points with volume V_p , the mass matrix element M_{ij} and internal force are approximated as,

$$M_{ij} \approx \sum_{p=1}^{n_p} m_p S_i(\mathbf{x}_p) S_j(\mathbf{x}_p), \quad (3.8)$$

$$\mathbf{f}_i^{int} = - \sum_{p=1}^{n_p} V_p \boldsymbol{\sigma}_p \cdot \nabla S_i(\mathbf{x}_p). \quad (3.9)$$

Here m_p is the mass of the material point p , n_p is the total number of material points, and $\boldsymbol{\sigma}_p$ is the stress at the material point. Equation (3.5) is a system of coupled linear equations. Noting that M_{ij} is non-zero only for the node j 's that are within the support of the shape function S_i , within an error $O[\Delta x]^2$, $\frac{d\mathbf{v}_j}{dt}$ can be approximated using $\frac{d\mathbf{v}_i}{dt}$, if node i corresponds to its surrounding cells [30].

Using this approximation, equation (3.5) can be decoupled as,

$$M_i \frac{d\mathbf{v}_i}{dt} - \int \boldsymbol{\sigma} \cdot \mathbf{n} S_i(\mathbf{x}) dA + \mathbf{f}_i^{int} = 0, \quad (3.10)$$

where

$$M_i = \sum_{j=1}^N M_{ij} \approx \sum_{p=1}^{n_p} m_p S_i(\mathbf{x}_p). \quad (3.11)$$

The approximation used in equation (3.10) is called lumped mass matrix approximation. Here, we have replaced the mass matrix M_{ij} by a diagonal matrix with the diagonal element being the sum of the elements in the same row. This approximation is known to cause artificial energy dissipation of order $[\Delta t]^2$ [30].

After we calculate the acceleration on node i using equation (3.10), the lagrangian velocity of node i is updated as,

$$\mathbf{v}_i^L = \mathbf{v}_i^n + \frac{d\mathbf{v}_i}{dt} \Delta t, \quad (3.12)$$

where the superscript L denotes the lagrangian step and the superscript n denotes the value at time step n . The material point velocity is now updated as,

$$\mathbf{v}_p^{n+1} = \mathbf{v}_p^n + \sum_{i=1}^N (\mathbf{v}_i^L - \mathbf{v}_i^n) S_i(\mathbf{x}_p^n). \quad (3.13)$$

Here we have interpolated the nodal velocity difference ($\mathbf{v}_i^L - \mathbf{v}_i^n$) to the material point to prevent the numerical diffusion, since this prevents from changing the particle velocity if there is no nodal acceleration in equation (3.12) and subsequently no driving force in equation (3.10). Similarly, the positions of material points are updated using the average nodal velocities interpolated to the material point, i.e.,

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \frac{1}{2}\Delta t \sum_{i=1}^N (\mathbf{v}_i^L + \mathbf{v}_i^n) S_i(\mathbf{x}_p^n). \quad (3.14)$$

The nodal velocity corresponding to time step $n + 1$ is updated as,

$$\sum_{j=1}^N M_{ij} \mathbf{v}_j \approx \sum_{p=1}^{n_p} m_p \mathbf{v}_p S_i(\mathbf{x}_p). \quad (3.15)$$

The above equation is also decoupled approximating \mathbf{v}_j with \mathbf{v}_i using the similar argument about local support to decouple equation (3.5) and the lumped mass matrix approximation,

$$M_i \mathbf{v}_i \approx \sum_{p=1}^{n_p} m_p \mathbf{v}_p S_i(\mathbf{x}_p). \quad (3.16)$$

Putting them together, following are the steps for a typical MPM calculation.

- 1) Initialize the grid structure as well as the material point positions as shown in figure 3.1.
- 2) Initialize particle quantities such as mass, stress, etc.
- 3) Compute nodal masses $M_i = \sum_{p=1}^{n_p} m_p S_i(\mathbf{x}_p)$.
- 4) Compute nodal velocities $\mathbf{v}_i = \frac{\sum_{p=1}^{n_p} m_p \mathbf{v}_p S_i(\mathbf{x}_p)}{M_i}$.
- 5) Compute internal forces $\mathbf{f}_i^{int} = - \sum_{p=1}^{n_p} V_p \boldsymbol{\sigma}_p \cdot \nabla S_i(\mathbf{x}_p)$.
- 6) Compute Lagrangian nodal velocities $\mathbf{v}_i^L = \mathbf{v}_i^n + \frac{d\mathbf{v}_i}{dt} \Delta t$.

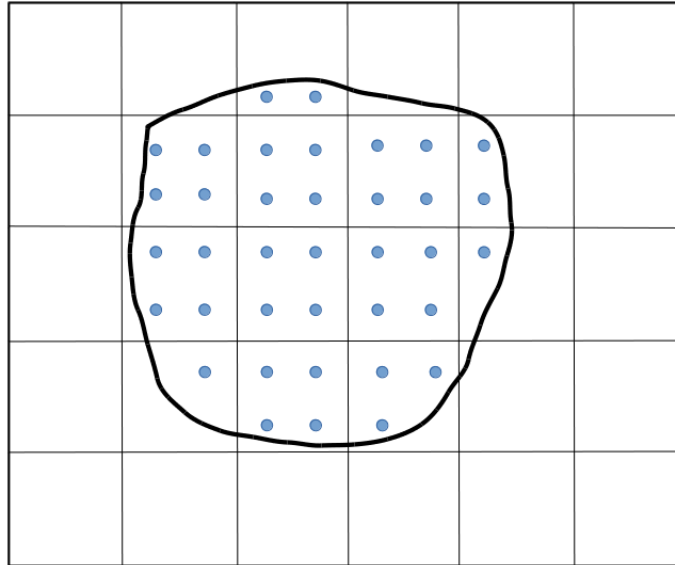


Figure 3.1 : Typical initialization of material points in background mesh

- 7) Compute velocity gradient tensor $\nabla \mathbf{v}(\mathbf{x}, t) = \sum_{i=1}^N \mathbf{v}_i(t) \nabla S_i(\mathbf{x})$.
- 8) Compute stresses in each particle.
- 9) Update particle velocities $\mathbf{v}_p^{n+1} = \mathbf{v}_p^n + \sum_{i=1}^N (\mathbf{v}_i^L - \mathbf{v}_i^n) S_i(\mathbf{x}_p^n)$.
- 10) Update particle positions $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \frac{1}{2} \Delta t \sum_{i=1}^N (\mathbf{v}_i^L + \mathbf{v}_i^n) S_i(\mathbf{x}_p^n)$.
- 11) Proceed to step (3).

To calculate internal force, equation (3.9) is used directly in the original MPM [16]. A significant problem arises due to the discontinuity in the shape function gradient, $\nabla S_i(\mathbf{x}_p)$, which is illustrated for a one-dimensional case in Fig. 3.2. The discontinuity of the gradient of shape function at node i causes the internal force, to suddenly switch signs as a particle crosses a cell boundary, creating numerical noise that could lead to numerical instability. There are many existing methods available

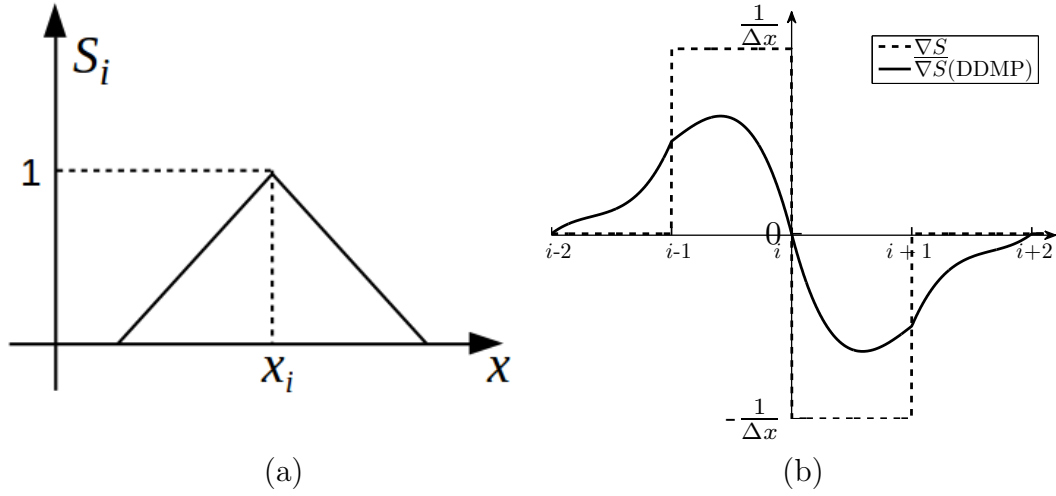


Figure 3.2 : (a) Illustration of shape function and (b) Gradient of the shape function used in traditional MPM (dotted line) and used in DDMP (solid line) at node i . Δx is the grid spacing.

to solve this issue. We will discuss the existing methods in the next section.

3.2 Material point methods: GIMP and CPDI

The generalized interpolation material point method (GIMP) introduces effective averages of the shape functions and gradient of shape functions over particle domain [23]. In GIMP, the modified shape function and gradient of shape function can be written as,

$$S_{ip} = \frac{1}{V_p} \int_{\Omega_x} \chi_p(\mathbf{x} - \mathbf{x}_p) S_i(\mathbf{x}) d\mathbf{x}, \quad (3.17)$$

$$\nabla S_{ip} = \frac{1}{V_p} \int_{\Omega_x} \chi_p(\mathbf{x} - \mathbf{x}_p) \nabla S_i(\mathbf{x}) d\mathbf{x}, \quad (3.18)$$

where S_i and ∇S_i are the shape function and gradient of shape function of the original material point method, V_p is the volume of material point p , $\chi_p(\mathbf{x})$ is the particle characteristic function, and Ω_χ is the support of this function. A typical choice for $\chi_p(\mathbf{x})$ in GIMP is,

$$\chi_p(\mathbf{x}) = \begin{cases} 1, & \text{if } \mathbf{x} \in \Omega_p, \\ 0, & \text{otherwise.} \end{cases} \quad (3.19)$$

For one-dimensional problems, the particle characteristic function can simply be written as,

$$\chi_p(x) = \begin{cases} 1, & \text{if } |x| \leq l_p/2, \\ 0, & \text{otherwise.} \end{cases} \quad (3.20)$$

We can see from equation (3.17) and (3.18) that if we choose a dirac delta function as particle characteristic function, the shape function and gradient of shape function of GIMP retain original MPM formulation. As discussed earlier in Chapter 1.3, there are two strategies in GIMP method to track the particle domains during calculation. The first approach is uGIMP (unchanged GIMP) where the particle domain χ_p is kept fixed, and the second one is cpGIMP (contiguous particle GIMP) where the particle domain χ_p is updated but remains rectangular.

Thus, in both uGIMP and cpGIMP method, the particle domain is tracked as rectangular. The GIMP method has been successfully implemented to reduce the cell crossing noise of the original MPM method for problems involving small material deformations [23, 31, 24].

In case of shear deformation, the shape of the particle domain becomes parallelogram. In order to track the particle domain as parallelogram, the GIMP method requires a complicated subdivision of particle domain across the cell boundaries [24]. For this reason, the CPDI method is developed.

In CPDI method, the particle domains are tracked as parallelograms [24, 32]. In this method, the equations (3.17) and (3.18) are written as,

$$S_{ip} = \frac{1}{V_p} \sum_{j=1}^4 \left(\int_{\Omega_p} Q_j^p(\mathbf{x}) d\mathbf{x} \right) S_i(\mathbf{x}_j^p), \quad (3.21)$$

$$\nabla S_{ip} = \frac{1}{V_p} \sum_{j=1}^4 \left(\int_{\Omega_p} \nabla Q_j^p(\mathbf{x}) d\mathbf{x} \right) S_i(\mathbf{x}_j^p), \quad (3.22)$$

where Q_j^p is the standard finite element 4-node shape function corresponding to the j th corner for the particle domain related to material point p and \mathbf{x}_j^p is the position of the j th corner of the particle domain. In one-dimensional problems, these shape function and gradient of shape function can be calculated as,

$$S_{ip} = \frac{1}{2} [S_i(\mathbf{x}_1^p) + S_i(\mathbf{x}_2^p)], \quad (3.23)$$

$$\nabla S_{ip} = \frac{1}{l_p} [(S_i(\mathbf{x}_2^p) - S_i(\mathbf{x}_1^p))]. \quad (3.24)$$

The CPDI method has been successfully used for problems involving large shear deformation [24, 32]. In cases of extreme material deformation, the particle domains can be distorted severely. This is similar to element distortion problem associated with pure Lagrangian methods as in FEM. Also in both CPDI and GIMP methods, the particle domains must cover the entire computational domain without overlap to

conserve momentum. For extreme deformation, this condition becomes difficult to satisfy.

In DDMP method, instead of modifying the shape functions, the gradient of shape function is modified so that it is continuous across the cell boundaries. This method does not use the particle domain, thus does not have the domain overlap problem. The detailed description of the DDMP method is given in the following section.

3.3 Dual domain material point method (DDMP)

Instead of introducing the concept of finite particle domain as in the GIMP and CPDI methods, the DDMP method reduces the cell crossing noise by adding an auxiliary stress [33]

$$\boldsymbol{\sigma}_A = A(\mathbf{x}, t) + \sum_j \frac{S_j(\mathbf{x})}{V_j} \sum_{p=1}^{N_p} V_p(t) \boldsymbol{\sigma}_p(t) S_j(\mathbf{x}_p) - \boldsymbol{\sigma}(\mathbf{x}, t) \quad (3.25)$$

to the stress $\boldsymbol{\sigma}$ in (3.9) to improve the numerical properties of the integral, where

$$\mathbf{A}(\mathbf{x}) = \sum_{p=1}^{N_p} \alpha(\mathbf{x}_p) V_p \boldsymbol{\sigma}_p \delta(\mathbf{x} - \mathbf{x}_p) - \sum_{j=1}^N \frac{S_j(\mathbf{x})}{V_j} \sum_{p=1}^{N_p} \alpha(\mathbf{x}_p) V_p \boldsymbol{\sigma}_p S_j(\mathbf{x}_p), \quad (3.26)$$

and $\alpha(\mathbf{x})$ is a continuous function whose value is zero on cell boundaries. A typical choice for $\alpha(\mathbf{x})$ is [33]

$$\alpha(\mathbf{x}) = 0.5 \left\{ \prod_{k=1}^{n_c} [n_c S_k(\mathbf{x})] \right\}^{\frac{3}{2(n_c-1)d}}, \quad (3.27)$$

where n_c is the number of nodes in the cell, and $d(= 1, 2, 3)$ is the dimension of the problem.

To ensure that the addition of the auxiliary stress $\boldsymbol{\sigma}_A$ in (3.9) does not affect the accuracy of the numerical solution to the original equations, which is second order in Δx , one needs to ensure the auxiliary stress is of the same order. It has been proved in [33] that $\mathbf{A}(\mathbf{x}) = O(\Delta x)^2$, independent of choice of the material points, their volumes, and function $\alpha(\mathbf{x})$. The proof is based on the relation

$$h(\mathbf{x}) = \sum_{j=1}^N h_j S_j(\mathbf{x}) + O(\Delta x)^2 = \sum_{j=1}^N \left[\frac{1}{V_j} \int_{\Omega} S_j(\mathbf{y}) h(\mathbf{y}) dV_y \right] S_j(\mathbf{x}) + O(\Delta x)^b, \quad (3.28)$$

where $h(\mathbf{x})$ is a smooth function, h_j is the value of $h(\mathbf{x})$ at node j , and $b = 2$ if \mathbf{x} is located in an interior cell, and $b = 1$ if \mathbf{x} is in a boundary cell. The first identity comes from the property of the shape functions, and the second relation holds because $h_j = \int_{\Omega} S_j(\mathbf{y}) h(\mathbf{y}) dv_y / V_j + O(\Delta x)^b$ and $V_j = \int_{\Omega} S_j(\mathbf{y}) dV_y$, or in other words, the value of h at a mesh node can be approximated by the average value in the square brackets in (3.28).

The summation over material points in (3.25) can be regarded as a Riemann sum. In Riemann sum, a definite integral is approximated by dividing the interval into several rectangles and summing over the area of all the rectangles. As the maximum volume of the particles v_p approaches zero, the sum becomes $\int_{\Omega} S_j(\mathbf{y}) \boldsymbol{\sigma}(\mathbf{y}, t) dv_y$. Using relation (3.28), one finds

$$\begin{aligned} \lim_{\max(V_p) \rightarrow 0} \sum_j \frac{S_j(\mathbf{x})}{V_j} \sum_{p=1}^{N_p} V_p(t) \boldsymbol{\sigma}_p(t) S_j(\mathbf{x}_p) &= \sum_{j=1}^N \left[\frac{1}{V_j} \int_{\Omega} S_j(\mathbf{y}) \boldsymbol{\sigma}(\mathbf{y}) dV_y \right] S_j(\mathbf{x}) \\ &= \boldsymbol{\sigma}(\mathbf{x}, t) + O(\Delta x)^b. \end{aligned} \quad (3.29)$$

Thus, we have proved that, in the sense of weak solutions, the auxiliary stress defined

in (3.25) is second order in Δx , $\boldsymbol{\sigma}_A(\mathbf{x}) = O(\Delta x)^2$, when a sufficient number of material points is used, because the total volume of the boundary cells is proportional to Δx .

With this property of $\boldsymbol{\sigma}_A$, we can add $\boldsymbol{\sigma}_A$ to the original stress without changing the order of accuracy in the internal force calculation, and the force integral can be calculated as,

$$\mathbf{f}_i^{int} \approx - \int_{\Omega} (\boldsymbol{\sigma} + \boldsymbol{\sigma}_A) \cdot \nabla S_i dV = - \sum_{p=1}^{N_p} V_p \boldsymbol{\sigma}_p \cdot \overline{\nabla S_i}(\mathbf{x}_p), \quad (3.30)$$

where

$$\overline{\nabla S_i}(\mathbf{x}) = \alpha(\mathbf{x}) \nabla S_i(\mathbf{x}) + [1 - \alpha(\mathbf{x})] \sum_{j=1}^N \frac{S_j(\mathbf{x})}{V_j} \int S_j(\mathbf{y}) \nabla S_i(\mathbf{y}) dV_y, \quad (3.31)$$

with N being the number of the nodes in the computational domain. In this way, one can also regard that the DDMP method replaces the discontinuous derivative of the shape function ∇S_i with $\overline{\nabla S_i}$ defined in (3.31). $\overline{\nabla S_i}$ is continuous because the shape function $S_j(\mathbf{x})$ is continuous, and $\alpha(\mathbf{x}) = 0$ on cell boundaries, where the discontinuity in ∇S_i occurs.

Both the original MPM and the DDMP method use Riemann sums, but they are used for different purposes. In the original MPM, the Riemann sum is used essentially for numerical differentiation of the stress by summing over the products of the stresses and the gradients of the shape functions. The increase of the number of material points causes more frequent numerical noise generation as more material points can move across cell boundaries. Although the magnitude of the noise generated by each particle is reduced because the resulting discontinuity in the internal force is

proportional to the particle volume, the overall effect on the numerical solution quality is very limited as discussed in Chapter 4.

In the DDMP method, the Riemann sum is used to approximate the integral of the stress around a mesh node. The increase of the material points provides an improved accuracy in the calculation of the average nodal stress as required in (3.29), leading to a better solution quality. Although the first term in (3.31) is similar to the Riemann sum used in the MPM for numerical differentiation of the stress, its presence in DDMP is for the purpose of numerical stability, not for numerical differentiation. It comes from stress \mathbf{A} defined in (3.26). The two terms in (3.26) almost cancel each other and $\mathbf{A} = O(\Delta x)^2$ independent of material points [33]. The presence of \mathbf{A} eliminates a null space of stress in the force calculation and provides the effect of using staggered grids. For instance, in cases of a uniform mesh in a one-dimensional problem or a uniform rectangular mesh in a two-dimensional problem, if the stresses in the neighbor cells take values of equal magnitudes but opposite signs, the stress “checker boarding” situation, the second term in (3.25) vanishes. Without \mathbf{A} , the internal nodal force calculated using $\boldsymbol{\sigma} + \boldsymbol{\sigma}_A$ in (3.9) or from (3.30) is exactly zero, therefore provides no resistance to this spurious stress mode. With the presence of \mathbf{A} , while the second term of (3.26) also vanishes, the first term of (3.26), which results in the first term in (3.31), causes a nonzero internal force leading to a material motion reducing the magnitude of the spurious stress.

With this understanding of the DDMP method and the particle sum in (3.30), we

find that the increase of material points provides a better approximation to the nodal stress as shown in (3.29) and a better solution quality as shown in Fig. 4.6. However, in many applications, such as the combined MD and DDMP calculation studied in this thesis work, the calculation of stress at a material point is expensive.

In Chapter 4, we will introduce a new method, DDMP with Sub-points, that has the accuracy equivalent to the use of many particles in the DDMP method but with the amount of computation greatly reduced. This new method incorporates the idea of particle domains of GIMP and CPDI, but without the need to track their shapes exactly.

Chapter 4

Material point methods applied to one-dimensional shock waves and dual domain material point method with sub-points

In this Chapter, we use a simple one-dimensional shock problem to investigate the numerical properties of the original material point method (MPM), the generalized interpolation material point (GIMP) method, the convected particle domain interpolation (CPDI) method, and the dual domain material point (DDMP) method. The experimental set up is similar to Sod shock tube problem. Suppose a one-dimensional tube initially separated by a partition located at node j in Fig. 4.1. To the left of the partition the gas pressure is slightly higher than that to the right. The gas on the both sides of the partition is at rest initially. At time $t = 0$ the partition is suddenly removed. A shock wave starts propagating to the right side of the tube, where as an expansion wave starts propagating to the left.

4.1 Shock wave simulated using original MPM

In original MPM, the internal force is calculated simply as a Riemann sum of the material points [16] as described in Chapter 3.1

$$\mathbf{f}_i^{int} \approx - \sum_{p=1}^{N_p} \boldsymbol{\sigma}(\mathbf{x}_p, t) V_p \cdot \nabla S_i(\mathbf{x}_p), \quad (4.1)$$

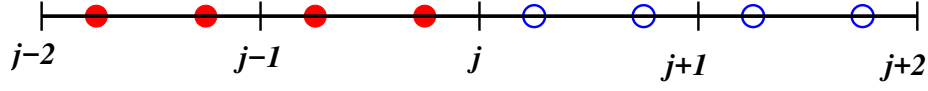


Figure 4.1 : Particles and cells in a one-dimensional weak shock calculation.

where \mathbf{x}_p is the material point location, N_p is the number of particles in the computational domain, and V_p is the volume of the material point.

While this approximation has been successfully used in many MPM calculations, mostly for solid materials, we now show this approximation fails if we use it to calculate a small perturbation of an ideal gas under isothermal condition, in which material points do not move across cells. For this shock tube problem, physically we expect to see a weak shock and expansion wave to propagate either direction in the computational domain. However if we use approximation (4.1), we have

$$\mathbf{f}_i^{int} \approx \sum_{p=1}^{N_p} V_p P_p \nabla S_i, \quad (4.2)$$

where P_p is the pressure P at material point p . In an isothermal case, $V_p P_p$ equals to two constants C_L or C_R respectively for the particles initially located at the left or right of the partition. Since ∇S_i is piecewise constant, for particle distributions shown in Fig. 4.1 only node j , the location of initial partition, experiences a net non-zero force. The net force at all other nodes is exactly zero. As a result, only node j acquires an acceleration to the right, while all other nodal accelerations remain zero. Then only particles in cells between nodes $j - 1$ to $j + 1$ obtain velocity increases,

while other particle velocities remain zero, according to node-particle velocity relation

$$\mathbf{v}_p^{n+1} = \mathbf{v}_p^n + \sum_{i=1}^N S_i(\mathbf{x}_p^n) \frac{d\mathbf{v}_i}{dt} \Delta t, \quad (4.3)$$

where the superscripts denote the time step. After the particles move according to the velocity interpolated from nodes, their velocities are mapped to nodes using

$$\mathbf{v}_i^{n+1} = \frac{\sum_{p=1}^{N_p} m_p \mathbf{v}_p^{n+1} S_i(\mathbf{x}_p^{n+1})}{\sum_{p=1}^{N_p} m_p S_i(\mathbf{x}_p^{n+1})}. \quad (4.4)$$

Only nodes $j - 1$, j , and $j + 1$ achieve nonzero velocities, while other nodal velocities are zero. When these nodal velocities are used to calculate the stress or pressure at the particles, only the pressure between these nodes $j - 2$ and $j + 2$, the four cells on both sides of j as shown in Fig. 4.2 with $j = 0$, are changed, while other particle pressures remain unaltered. As time advances, the velocities at node j and the two neighboring nodes continue to increase, while other nodal velocities remain zero. The waves fail to propagate. During this time the particle pressures in the two cells left of node j continue to decrease and even become less than the initial value on the right. Similarly, the particle pressures in the two cells right of node j continue to increase and exceed the initial value on the left. The resulting high pressures at the particles on the right cannot push nodal velocity back because in this isothermal case, the particle volume is reduced, and $V_p P_p = C_R$. Similarly the resulting low pressures at the left particles cannot pull nodal velocity back because $V_p P_p = C_L$. Since $C_L > C_R$, the nodal force from (4.2) is always positive for node $j = 0$, and the process accelerates. This process is shown in Fig. 4.2 for particle pressure values at different times.

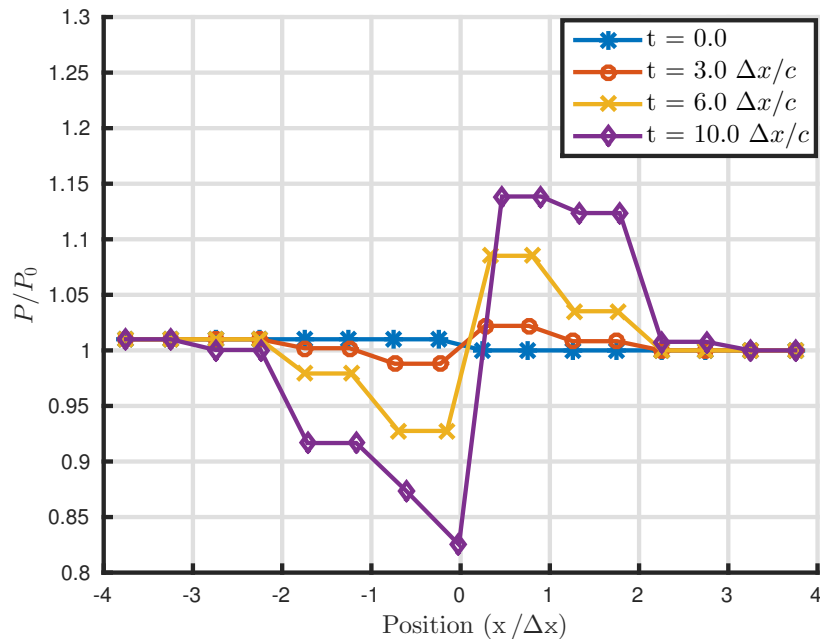


Figure 4.2 : Particles pressure near one-dimensional shock front. Position 0 corresponds node j in Fig. 4.1.

This process continues until the first particle left of the partition moves into the right cell, as shown in Fig. 4.2 at $t = 10\Delta x/c$, where c is the speed of sound. At that time the forces on nodes $j - 1$ and $j + 1$ become nonzero, and then the particle pressures in the next neighboring cells start being affected. However this propagation of the disturbance is purely numerical, is not continuous, occurs only after the particle has been displaced by a quarter of the cell length for the particle arrangement shown in Fig. 4.1, and is too late for a proper propagation of the waves.

Despite the incorrect wave propagation results explained above, the process of material point motion resembles material motion in the physical situation: as time advances, some of the material in the left cell moves across the nodes into the cell

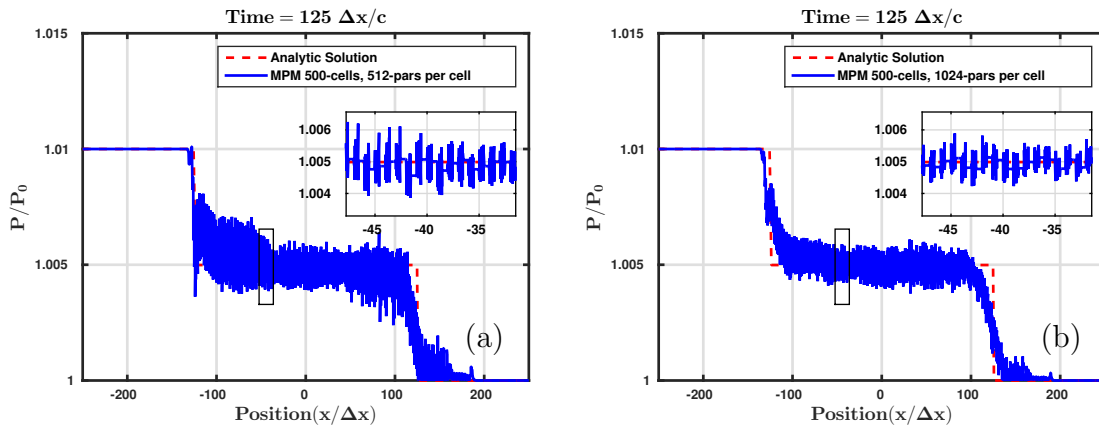


Figure 4.3 : Particle pressure calculated with MPM method using (a) 512 particles per cell, (b) 1024 particles per cell.

on the right to cause a pressure change. In the MPM method, because the mass is lumped at material points, this cell crossing motion does not occur until the material point moves across a cell. One might think that if sufficient number of material points are used, the cell crossing can happen earlier so that the pressure disturbance can propagate to the other nodes earlier and the pressure inversion can be avoided. This physical intuition is correct. However, the use of many material points results in noise and pressure spikes as shown in Fig. 4.3 calculated using 512 and 1024 particles per cell in the initial particle placement with time step $0.1\Delta x/c$. In the figure the results roughly approximate the correct behavior. The numerical quality is apparently unsatisfactory, although the results are slightly better with the larger amount of particles.

The spike is caused by cell crossing of the particles. Suppose one particle moves from a high pressure cell to a low pressure cell carrying a high pressure value with it.

While this new member in the low pressure cell contributes to the resistance to the compression of the low pressure cell, the majority of particles in the cell are still at low pressure. The cell is still being compressed causing a pressure increase for particles in the low pressure cell, including the new member carrying a high pressure value. This trend is correct for most of the particles in the cell, but for the particle just arrived from the high pressure cell, this compression further increases its pressure, instead of relieving it. Meanwhile, the upstream particles remaining in the high pressure cell experience a pressure release from a positive velocity divergence in the cell. The particle that just moved into the low pressure cell experiences a further pressure increase, while its upstream neighbors experience a pressure release, and its downstream neighbors are still at their low pressure values. Thus, the particle that just entered the low pressure cell has a higher pressure value than its neighbors, and it generates a pressure spike.

These pressure spikes are caused by particles moving across cell boundaries, and currently there are three versions of the material method, GIMP, CPDI and DDMP available to reduce the cell crossing noise. In the next section we use them to see how they perform in this case of one-dimensional shock propagation.

4.2 Comparison of GIMP, CPDI and DDMP method

The results of GIMP are shown in Fig. 4.4. There is a significant improvement compared to the original MPM results. Instead of needing a few hundred particles per cell

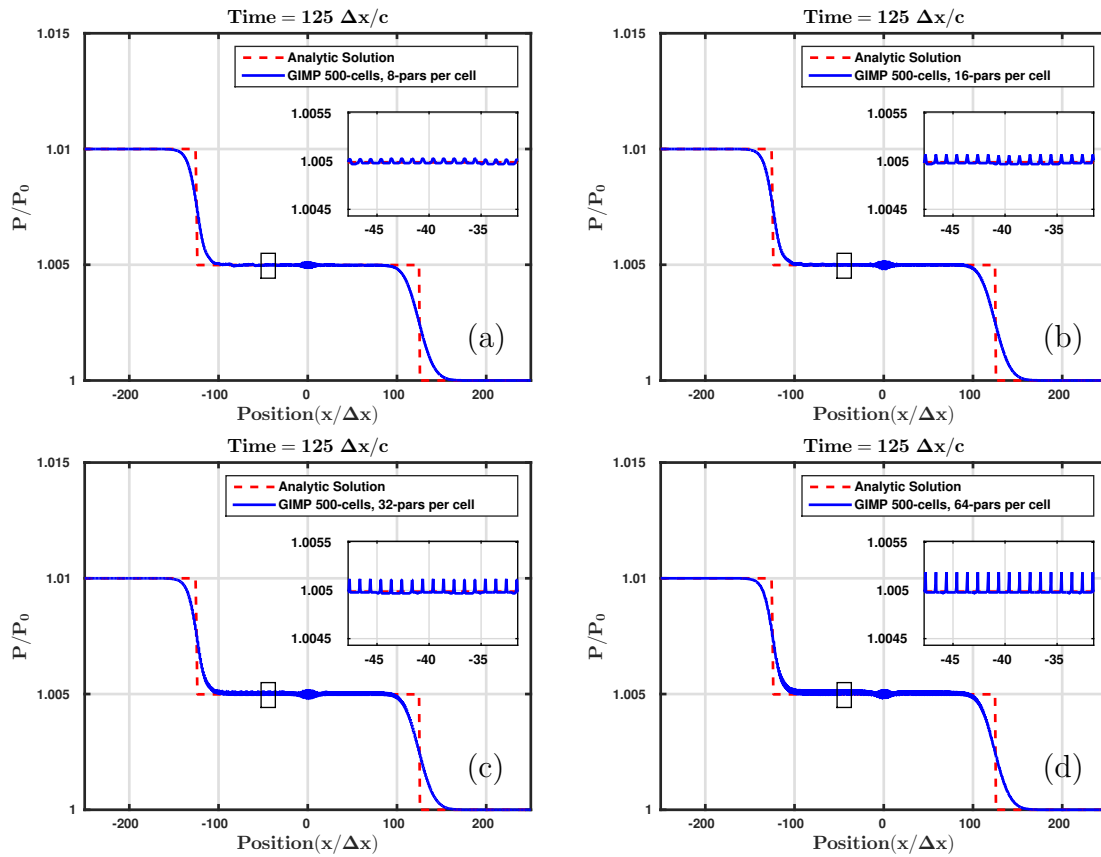


Figure 4.4 : Particle pressure calculated with GIMP method using (a) 8 particles per cell, (b) 16 particles per cell, (c) 32 particles per cell, and (d) 64 particles per cell.

to show just the rough shape of the results as in Fig.4.3, the GIMP results obtained using 8 particles per cell in the initial particle placement result in acceptable results compared to the analytic solution. However, the pressure spikes are evident as more particles per cell are used. Fig. 4.4 shows results calculated using 8, 16, 32, and 64 particles per cell in the initial particle placement. This behavior of the solution is qualitatively similar for the CPDI method as shown in Fig 4.5. The improvement of these two methods over the original MPM comes from the introduction of finite

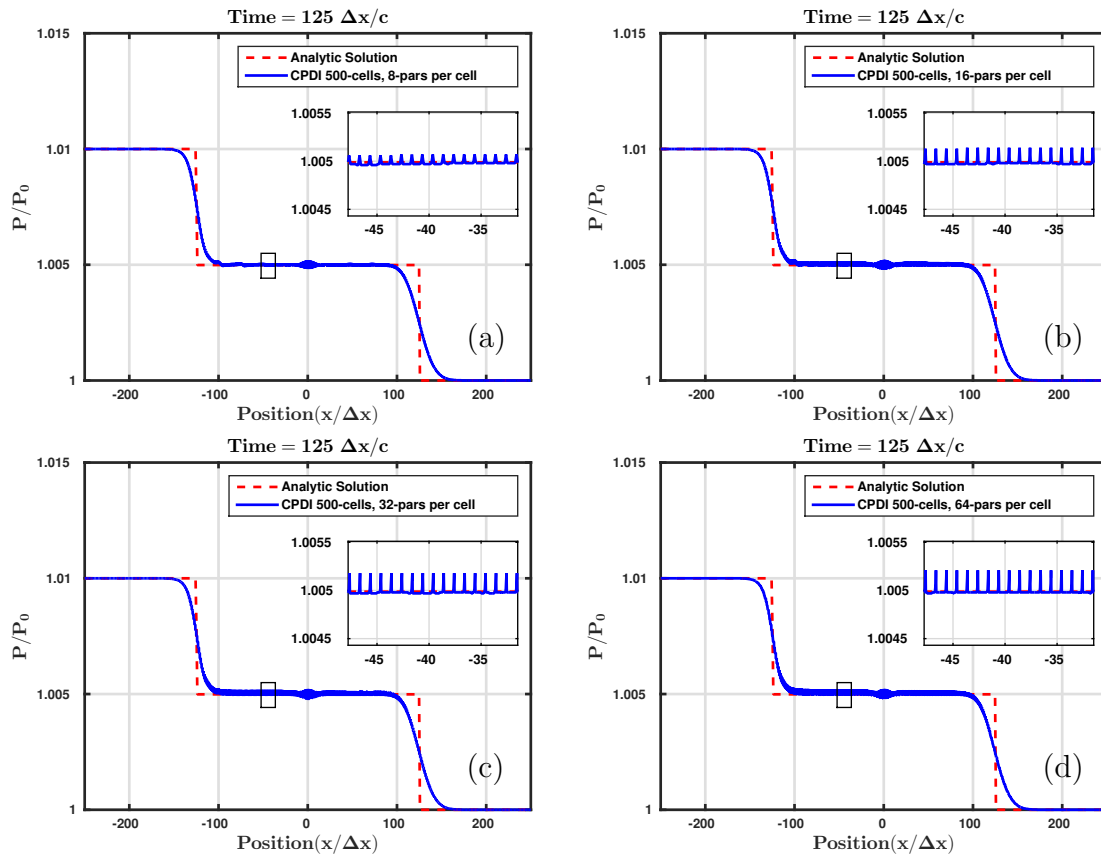


Figure 4.5 : Particle pressure calculated with CPDI method using (a) 8 particles per cell, (b) 16 particles per cell, (c) 32 particles per cell, and (d) 64 particles per cell.

particle size. In the internal force calculation, instead of using the gradient of the shape function evaluated directly at a particle location, which abruptly changes by $2/\Delta x$ in a time step in this one-dimensional case as the particle moves across a cell boundary in the original MPM, the GIMP and CPDI methods replace this discontinuous value of the shape function gradient at a material point by its average over the finite particle domain. With this replacement, for a particle near a cell boundary, the change in the value of the modified gradient of shape function in a time step is

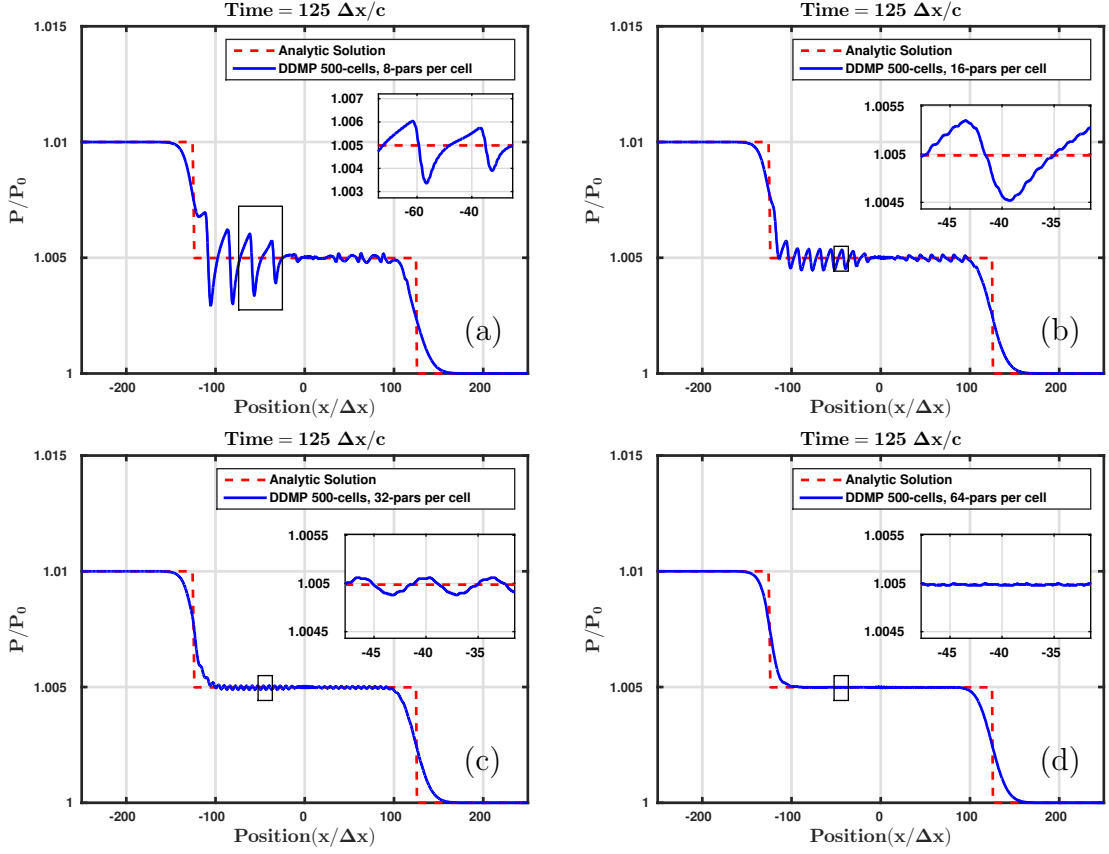


Figure 4.6 : Particle pressure calculated with DDMP method using (a) 8 particles per cell, (b) 16 particles per cell, (c) 32 particles per cell, and (d) 64 particles per cell.

limited by $2|\mathbf{v}_p|\Delta t/(\Delta x \ell_p)$, if $|\mathbf{v}_p|\Delta t \leq \ell_p$, where ℓ_p is the particle length in this one-dimensional case. These methods require that, at least initially, the particle domains cover the entire computational domain without gap or overlap. As the number of material points per cell increases, the size ℓ_p of the particle is reduced, and change in the modified shape function gradient in a time step is increased. Although the use of a large number of particles increases the numerical integration accuracy in the internal force calculation, the effect of this accuracy increase is very limited because

the use of small number of particles already produces reasonable results as shown in Figs. 4.4 and 4.5. The pressure spikes are not caused by the lack of the integration accuracy, but rather by the discontinuity or rapid change of the value of the shape function gradient in a time step. When the particle size ℓ_p becomes less than $|\mathbf{v}_p|\Delta t$, or the finite-domain particle moves across a cell boundary completely in a time step, the change becomes $2/\Delta x$, the same value as in the original MPM, and the benefit of using a finite particle domain is lost completely. In other words, the introduction of finite particle domain also introduces a concept of the particle Courant number, $|\mathbf{v}_p|\Delta t/\ell_p$, based on the particle size. The benefit of GIMP and CPDI methods reduces with increasing particle Courant number. Therefore, as the number of particles per cell increases, the pressure spikes increase. The appearance of the spikes at large number (> 8) of particles per cell might not be a significant issue for this problem with a small deformation, because one can always use a small number of particles per cell, but this is a special case of isothermal shock. For problems with large deformation, however, we do not have control on the number of particles per cell, and particles can aggregate in certain regions of the computational domain.

This weak shock problem is also calculated using the DDMP method, as shown in Fig. 4.6. With 8 particles per cell in the initial particle placement, the results show spurious oscillation in Fig. 4.6(a) and are worse than the GIMP and CPDI methods. But different from the GIMP and CPDI methods, DDMP results improve as the number of particles increases.

From these results calculated using different versions of the material point method, we conclude that

1. The original MPM cannot be used for this weak shock propagation problem.
2. The finite-sized particle domains used in GIMP and CPDI methods provide smoothing effects, and therefore, when a small number of particles are used, the GIMP and CPDI methods give reasonable results for this weak shock problem. But these methods can only benefit calculations with a sufficiently small particle Courant number, and fail as number of particles increases.
3. The DDMP method is inaccurate when a small number of particles are used, but improves as the number of particles increases. However, the large number of particles required to achieve accurate results renders this method very expensive, especially for similar problems in two- or three- dimensional cases.

Since the DDMP method is the only one among the four versions of the material point method that converges to the correct solution with increasing particle number, in the rest of this Chapter we focus on improvements to the DDMP method.

4.3 DDMP with Sub-points

In the Riemann sum (3.29), the length scale of the stress variation is determined by the physical problem, while the length scale of shape function is the cell size Δx as shown in figure 4.7, which is much smaller compared to the physical length scale in a

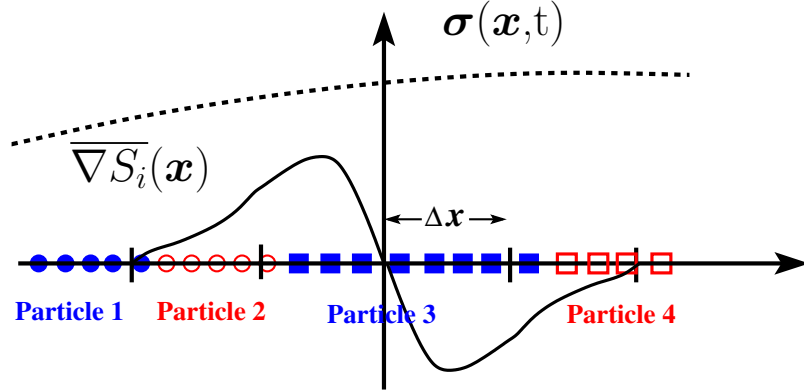


Figure 4.7 : One-dimensional illustration of the sub-point algorithm. Four particles or material points are divided into 22 sub-points, represented by hollow or solid circles and squares. Each of these sub-points is an integration point in (6.2). The sub-point belonging to the same particle share the same value of the stress.

reasonable calculation. The need for many material points in (3.29) arises not from the stress variation but from the variation of the shape functions over the cell size Δx . The new method originates from this observation and uses a small number of particles, say two particles per spatial dimension per cell, to adequately represent the stress variation over the physical length scale. We split each particle (p) into a group of (n_p) sub-particles or sub-points having the same stress as the original particle. These sub-points are distributed around the original material point. With these sub-points we re-write (3.30) as,

$$\mathbf{f}_i^{int} = - \sum_{p=1}^{N_p} \boldsymbol{\sigma}_p \cdot \sum_{s=1}^{n_p} V_{ps} \overline{\nabla S}_i(\mathbf{x}_{ps}) = - \sum_{p=1}^{N_p} \boldsymbol{\sigma}_p V_p \cdot \overline{\overline{\nabla S}_i}(\mathbf{x}_p), \quad (4.5)$$

where V_{ps} is the volume of sub-point s belonging to original particle p satisfying $\sum_{s=1}^{n_p} V_{ps} = V_p$, \mathbf{x}_{ps} is the position of the sub-point, and

$$\overline{\overline{\nabla S}_i}(\mathbf{x}_p) = \frac{1}{V_p} \sum_{s=1}^{n_p} V_{ps} \overline{\nabla S}_i(\mathbf{x}_{ps}), \quad (4.6)$$

is the sub-point volume weighted average of $\overline{\nabla S_i}$. The method of specifying volumes and positions of the sub-points will be discussed later. Equation (4.5) can be derived by following the derivations from (3.25) to (3.30). One can first treat all the sub-points as ordinary DDMP particles, calculate the internal force using (3.30), and then factor out stresses from the group of sub-points belonging to the same main particles p . With many sub-points, the approximation of (3.29) is better leading to enhanced quality of the numerical solution.

As number of sub-points approaches infinity, the new gradient of the shape function defined in (4.6) approaches $\int_{\Omega_p(t)} \overline{\nabla S_i} dV / V_p$, where $\Omega_p(t)$ is the particle domain. Although $\overline{\nabla S_i}$ is a function independent of the particles as defined in (3.31), direct evaluation of the integral is not an easy task, because the particle domain $\Omega_p(t)$ is a function of time. Approximately tracking the particle domain is a main task of the CPDI method. Unfortunately, the approximation introduced in the CPDI method does not satisfy the conservation properties required for high quality numerical solutions, but the scheme of approximately tracking the particle domains is useful in our new method introduced here. Let us first study the conservation properties of this new method.

Relation (4.6) can be viewed as another modification to the gradient of the shape function. With this modified gradient of the shape function, we can still easily prove that

$$\sum_i^N \mathbf{f}_i^{int} = \mathbf{0}, \quad (4.7)$$

because $\sum_i^N \overline{\nabla S}_i(\mathbf{x}_{ps}) = \mathbf{0}$ according to (30) in [33], where N is number of mesh nodes. This relation shows that the internal forces sum to zero over the computational domain and ensures momentum conservation. Because the method of computing nodal mass is not changed in this new method, mass conservation is unaffected.

To consider energy conservation, we note that kinetic energy difference in a time step n to $n + 1$ can be calculated using (46) in [33]

$$K^{n+1} - K^n = \Delta t \sum_{i=1}^N \mathbf{v}_i^{n+1/2} \cdot \sum_{\ell=1}^N C_{\ell i} \mathbf{f}_\ell^{int}, \quad (4.8)$$

after neglecting the boundary force and body force terms, where $C_{\ell i}$ is the force transfer coefficient [33, 34] and $\mathbf{v}_i^{n+1/2} = (\mathbf{v}_i^L + \mathbf{v}_i^n)/2$ is the half time Lagrangian velocity, with \mathbf{v}_i^L being the Lagrangian velocity obtained from (3.5).

Substituting (4.5) into (4.8), we find

$$K^{n+1} - K^n = -\Delta t \sum_{p=1}^{N_p} V_p \boldsymbol{\sigma}_p : \overline{\nabla \mathbf{v}}^{n+1/2}(\mathbf{x}_p^n) \quad (4.9)$$

where

$$\overline{\nabla \mathbf{v}}^{n+1/2}(\mathbf{x}_p^n) = \sum_{\ell=1}^N \sum_{i=1}^N C_{\ell i} \mathbf{v}_i^{n+1/2} \overline{\overline{\nabla S}_\ell}(\mathbf{x}_p^n) = \frac{1}{V_p} \sum_{s=1}^{n_p} V_{ps} \left(\sum_{\ell=1}^N \overline{\nabla S}_\ell(\mathbf{x}_{ps}) \sum_{i=1}^N C_{\ell i} \mathbf{v}_i^{n+1/2} \right), \quad (4.10)$$

and the second identity comes from the use of (4.6). In the first identity of (4.10), the velocity gradient at the main particle is calculated using the modified DDMP gradient of the shape function defined in (4.6), which is the average of the DDMP gradient of the shape function weighted by the sub-point volumes. This form is used in our numerical calculation of the velocity gradient. The second identity shows that

this velocity gradient can also be regarded as the sub-point volume weighted average of the velocity gradients at the sub-points, since the quantity inside the brackets can be identified as the velocity gradient at the sub-point located at \mathbf{x}_{ps} .

The internal energy change due to the material deformation in the time step can be calculated as eq. (51) in [33]

$$U^{n+1} - U^n = \sum_{p=1}^{N_p} V_p \boldsymbol{\sigma}_p : \dot{\boldsymbol{\epsilon}}_p \Delta t + O[(\Delta t)^2], \quad (4.11)$$

Comparing (4.9) and (4.11), if we use velocity gradient defined in (4.10) to calculate the strain rate, i.e. $\dot{\boldsymbol{\epsilon}}_p = [\overline{\nabla \mathbf{v}}^{n+1/2} + (\overline{\nabla \mathbf{v}}^{n+1/2})^T]/2$, we can ensure energy conservation error is second order in both the time step size and the cell size.

In this method, the information about the domain of the main particles is used to place the sub-points, not to perform an exact calculation. The error on the determination of particle domain does not affect the conservation properties of the new method, although it affects solution accuracy. In cases of extreme deformation, where a local linear approximation to the displacement field fails, and severe gaps or overlaps by the parallelograms and parallelepipeds appear, to maintain accuracy one has the option to use CPDI2, the improved version of the CPDI method [32], which ensures no gap and overlap among the particle domains, or to directly track the position and volume evolution of the sub-points in the calculation. For cases of complex constitutive relations, if the second option is chosen, the computational cost of a sub-point is still very small compared to that of a main particle, because the stresses on the sub-points are not calculated directly using the constitutive relation, but copied from

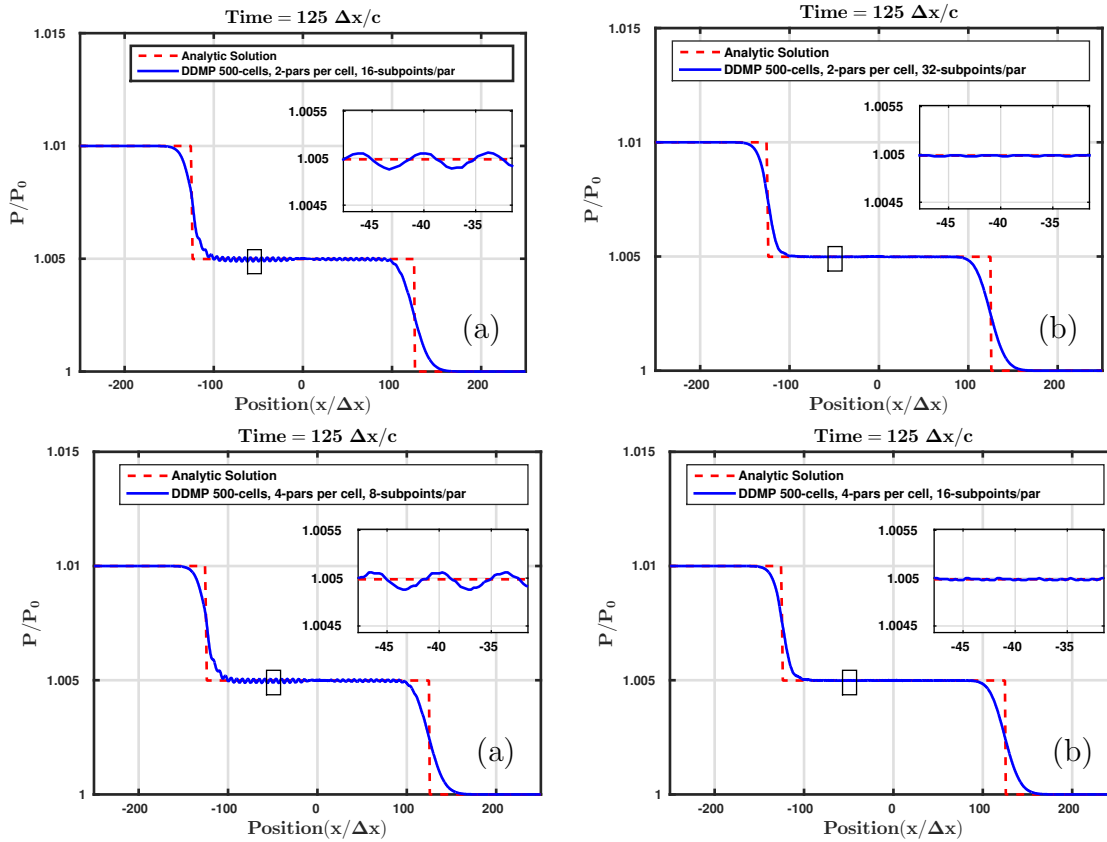


Figure 4.8 : Results calculated using the sub-point method.

their main particles. These options are left for future exploration. In this thesis we limit ourselves to the study of the numerical properties of this new sub-point method when applied to one-dimensional shock waves.

4.4 Results using DDMP method with sub-points

We now use the sub-points in DDMP method for the one dimensional shock problem as discussed in Chapter 4.3. The results plotted in Fig. 4.8 are obtained by placing 2 or 4 particles per cell initially in the computational domain. Their volumes (actually,

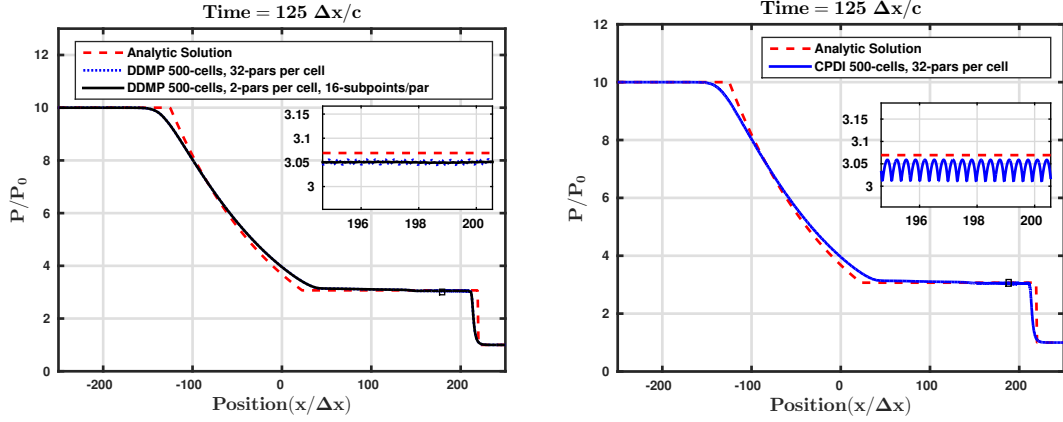


Figure 4.9 : Comparison of strong shock results calculated using 2 particles per cell and 16 sub-points per particle with that obtained using 32 DDMP particles per cell (a). Strong shock results from CPDI method using 32 particles per cell (b).

the lengths in this one-dimensional case) are updated using the velocity divergence at the particle location \mathbf{x}_p calculated using the nodal velocity and the DDMP gradient of the shape function, $dV_p/dt = V_p \sum_{i=1}^N \mathbf{v}_i \cdot \nabla \overline{S}_i(\mathbf{x}_p)$. Eight to 32 sub-points are evenly placed in each of the deformed particle domains at every time step. By comparison to Fig. 4.6, we find that the quality of the solution obtained from using this sub-point method is equivalent to that obtained using the number of main particles equal to the number of sub-points.

The results in Fig. 4.8 are obtained for a weak shock. In Fig. 4.9, we apply the new sub-point method to a much stronger shock. We compare the results calculated by initially placing 32 particles per cell using the DDMP method and the results obtained by using only 2 DDMP particles initially in a cell with 16 sub-points per DDMP particle. Only very small differences are observed. For the calculation with 2 DDMP particles per cell, we plot the particle distribution in the region with expanded

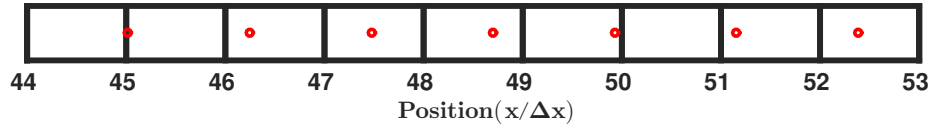


Figure 4.10 : Distribution of particles in the region of expanded material in the strong shock calculation using the DDMP method with sub-points at time $100\Delta x/c$. Initially two DDMP particles per cell are used. 16 sub-points per DDMP particle are used in this calculation.

material in Fig. 4.10 at time $100\Delta x/c$, earlier than the time $125\Delta x/c$ for Fig. 4.8. In this region there is less than one particle per cell. In this case, the tension instability has not happened because of the larger support of the modified gradient of the shape functions compared to that of the shape function, a feature inherent in DDMP. To test the limit, we also used one particle per cell in the initial particle placement. The calculation fails around time $25\Delta x/c$ due to the tension instability.

In Fig. 4.9, the results of the CPDI method are also plotted on the right as a comparison. The sub-figure shows that the pressure noise exists in the CPDI result, although small relative to the large pressure difference in this strong shock problem.

4.5 Chapter Summary

The present Chapter shows that the original material point method cannot be used for weak isothermal shocks, because of the low accuracy of the Riemann sum used in the internal force calculation. Any attempt to increase the Riemann sum accuracy by increasing the number of particles is met with the cell crossing noise, because the use of more particles leads to more frequent cell crossing of the particles. Better

numerical accuracy in the internal force calculation can be obtained by using GIMP and CPDI methods. However, these two methods fail to converge and generate noise as the number of particles increases. This non-convergence is caused by the use of a finite particle domain. As the number of particles increases, the particle domain is decreased, and the methods reduce to the original material point method, which suffers from cell-crossing noise. Although significant noise only occurs when a large number of particles (> 16) per cell are used, for materials undergoing a significant compression, the number of particles per cell can go well beyond 16, leading to a compression instability.

The dual domain material point (DDMP) method is also used to study this simple example. Although the method converges as the number of particles increases, an impractically large amount of particles are needed to produce smooth results. The cause of the lack of smoothness is identified to be the insufficient accuracy in the calculation of the internal force integral by using the Riemann sum with a small amount of particles. A sub-point scheme is then introduced to increase the numerical integration accuracy. In this new scheme, the role of integration points is separated from the particles and given to the sub-points. In this way the accuracy of the numerical integration increases, and accurate results can be achieved with very small number of particles using the DDMP method. In this improved DDMP method, the sub-points can be generated and placed around the original DDMP particles at every time step. There is no need to track their history, therefore the computational cost

incurred by these sub-points is negligible compared to that needed for the original DDMP particles. To place these sub-points, this new scheme needs an approximate particle domain. In one-dimensional cases, accurately tracking a particle domain is an easy task, but is very difficult for a multidimensional problem. In this sub-point scheme, the particle domain is not directly used in the integration for the internal force calculation; therefore errors, such as gaps and overlaps, of particle domains can be better tolerated than in the GIMP and CPDI methods. Furthermore, we have proved that placements of the sub-points do not alter conservation properties of the DDMP method, which conserves mass and momentum exactly and conserves energy to the second order of spatial and temporal discretization.

Although the numerical examples provided in this Chapter are one-dimensional, all derivations are in a multidimensional form. Extension of this sub-point method to multidimensional problems is rather straightforward. One can start from a DDMP code, and add the parallelograms (in two-dimension) and parallelepipeds (in three-dimension) tracking method of CPDI. Since both the DDMP and CPDI methods have been implemented for multidimensional problems, the only additional work to implement this sub-point method is to place sub-points evenly in the parallelograms or parallelepipeds and to calculate Riemann sums over these sub-points.

Chapter 5

Molecular Dynamics

Molecular dynamics (MD) is a method in which the motion of each particle is computed by integrating the equation of motion i.e., Newton's second law. An interatomic potential specific to a system is defined and the forces are calculated as the negative gradients of the potential. The first MD simulation was accomplished by Alder and Wainwright in 1957 [35]. They used hard sphere model, where the atoms move at constant velocity in between perfectly elastic collisions, to investigate the phase diagrams of such system. After several years, Rahman [36] used Lennard-Jones potential to study a number of properties of liquid argon using MD simulation. The use of continuous potential would open up several applications of MD simulation. Since then several pair potentials and multi-body potentials are developed to simulate more complex systems such as, diatomic molecular system, metals, polymers, etc. The purpose of molecular dynamics calculation in this thesis work is to calculate the closure quantities to drive the continuum level calculation.

The equation of motion for a system of N atoms can be written as

$$m_\alpha \ddot{\mathbf{r}}_\alpha = \mathbf{f}_\alpha, \quad (5.1)$$

where m_α is the mass of atom α , $\ddot{\mathbf{r}}_\alpha$ is the acceleration and \mathbf{f}_α is the force acting on

the atom. We calculate force \mathbf{f}_α using the potential V as,

$$\mathbf{f}_\alpha = -\nabla_\alpha V \quad (5.2)$$

A typical MD simulation consists of the following principal steps:

- 1) Initialize all the positions and velocities of atoms.
- 2) Calculate force on each atoms using equation (5.2).
- 3) Update the position and velocities of atoms by solving equation (5.1).

5.1 Leap frog algorithm

There are many integration schemes to solve equation (5.1). In this thesis, leap frog algorithm is used for integration purpose. Leap frog algorithm is a second order method with discretization error $O(\Delta t^2)$, where Δt is the MD time step. In this method, the half step velocities are updated according to the force on i th atom \mathbf{f}_α as,

$$\mathbf{v}_\alpha^{n+1/2} = \mathbf{v}_\alpha^{n-1/2} + \frac{\mathbf{f}_\alpha^n}{m_\alpha} \Delta t. \quad (5.3)$$

In the very first time step, the half step velocity $\mathbf{v}_\alpha^{1/2}$ is calculated as,

$$\mathbf{v}_\alpha^{1/2} = \mathbf{v}_\alpha^0 + \frac{\mathbf{f}_\alpha^0}{2m_\alpha} \Delta t. \quad (5.4)$$

In each time step, the atomic positions are updated as,

$$\mathbf{x}_\alpha^{n+1} = \mathbf{x}_\alpha^n + \mathbf{v}_\alpha^{n+1/2} \Delta t, \quad (5.5)$$

where $\mathbf{v}^{n+1/2}$ is the half step velocity. If the velocity corresponding to n^{th} step is required to calculate statistical quantities, such as temperature of the system, \mathbf{v}^n can

be calculated as,

$$\mathbf{v}_\alpha^n = (\mathbf{v}_\alpha^{n-1/2} + \mathbf{v}_\alpha^{n+1/2})/2. \quad (5.6)$$

A few advantages of using leap frog algorithm include, it is symplectic, time reversal and it conserves angular momentum exactly [37].

5.2 Periodic boundary conditions

In this thesis work, molecular dynamics simulation seeks to compute closures of the system using statistical average which usually requires a large number of atoms. It is impractical to simulate entire computational domain of interest (order of 10^{23} atoms). We can only simulate relatively small number ($N \approx 10^3 - 10^7$) of atoms. This introduces the problem of surface effects, where a significant number of atoms reside on the surface. For example, for a cubic simulation box containing 1000 atoms, about half of them will be on the surface. The periodic boundary condition is widely used to overcome the surface effect problems. In periodic boundary condition, it is imagined that the MD simulation box is replicated to infinity by rigid translation through out the space to infinite domain. In our multi-scale numerical method, we calculate closure quantities using MD simulation in a small representative volume of a material point. We use periodic boundary condition to eliminate any surface effects. Figure (5.1) shows an illustration of periodic boundary condition in two dimensions. The atoms corresponding to each color are replicated through out the space. In periodic boundary condition, each particle inside a box is interacting not only with

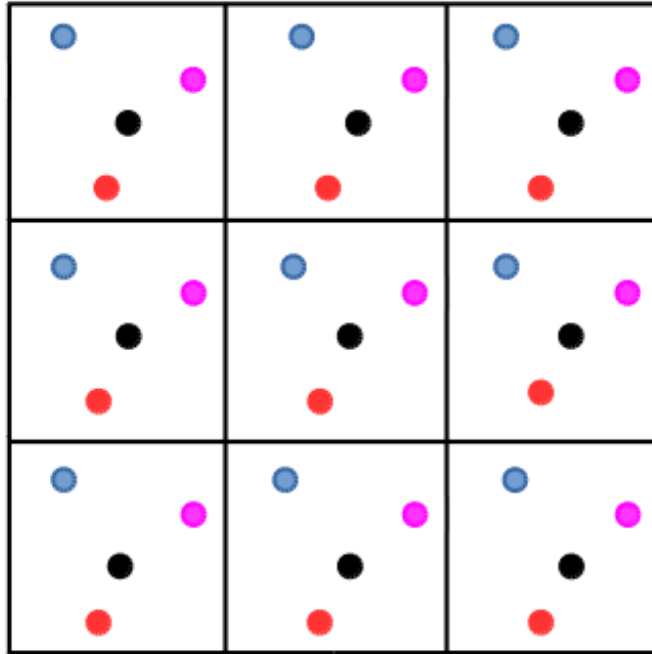


Figure 5.1 : Periodic boundary condition in 2 dimensions

the other particles inside the same box, but also with their images in the neighbor boxes. Also, after each integration step, the coordinates of the atoms are examined. If any atom lies outside the periodic box, the coordinate of the atom is adjusted to bring it back to the simulation box. For example, in one dimension, suppose the simulation domain extends from $-L/2$ to $L/2$. If the coordinate $r \geq L/2$, bring the atom back to $r - L$. Similarly, if $r \leq -L/2$, bring the atom back to $r + L$.

In this thesis work, the communication between the macroscopic system and the molecular dynamics system is through the strain rate and stress. In every MD time step, using the strain rate $\dot{\epsilon}(\mathbf{x}_p, t)$, calculated from DDMP calculation for the macroscopic time t and material point p , we update the periodic domain length in the

x -direction as

$$L_{px}^{n+1} = L_{px}^n [1 + \dot{\epsilon}_{xx}(\mathbf{x}_p, t)(\Delta t)_{md}], \quad (5.7)$$

where L_{px} is the periodic domain length in x -direction and $\dot{\epsilon}_{xx}$ is the xx - component of the strain rate tensor. Similarly, the periodic domain sizes in y and z - directions are updated using $\dot{\epsilon}_{yy}$ and $\dot{\epsilon}_{zz}$. To ensure all atoms are within the periodic domain, atoms left from one side of the periodic domain are put back through the other side. More precisely, the atom locations are further updated using \mathbf{x}_α^* from (5.5). For the x - coordinate we use

$$x_\alpha^{n+1} = \left(x_\alpha^* + \frac{L_{px}^{n+1}}{2} \right) \bmod(L_{px}^{n+1}) - \frac{L_{px}^{n+1}}{2}, \quad (5.8)$$

and similarly for y and z - coordinates. If there is a velocity gradient in x - direction, if $x_\alpha^{n+1} \neq x_\alpha^*$, then velocity $(x_\alpha^{n+1} - x_\alpha^*)\dot{\epsilon}_{xx}(\mathbf{x}_p, t)$ is added to the full-step velocity \mathbf{v}_α^n and the half-step velocity $\mathbf{v}_\alpha^{n+1/2}$ of the atom. Similar velocity adjustment is performed if there is velocity gradient in y or z - direction.

If there is a shear strain, the periodic MD box can change the shape and no longer remains rectangular. It is inconvenient to apply periodic boundary condition for such non-rectangular shaped box. In our calculation, we keep the shape of the MD box, called fundamental box, rectangular all the time. In periodic boundary condition, a particle in the fundamental box represents all of its image particles. To ensure the fundamental box contains one and only one of the particle or its image, we first

assume that the velocity gradient tensor takes the following form

$$\begin{bmatrix} g_{xx} & g_{xy} & g_{xz} \\ 0 & g_{yy} & g_{yz} \\ 0 & 0 & g_{zz} \end{bmatrix}, \quad (5.9)$$

where g 's are the components of velocity gradient tensor.

In periodic boundary condition, the coordinates of images of a particle can be written as,

$$\begin{bmatrix} x_i(t) \\ y_i(t) \\ z_i(t) \end{bmatrix} = \begin{bmatrix} x_0(t) + n_x L_x(t) \\ y_0(t) + n_y L_y(t) \\ z_0(t) + n_z L_z(t) \end{bmatrix}, \quad (5.10)$$

where $x_i(t)$ is x - coordinate of images at time t , $x_0(t)$ is x - coordinate of particle inside the fundamental box at time t , n_x is an integer, $L_x(t)$ is the x -length of the MD box at time t and similarly for y and z - components.

Using the velocity gradient tensor from equation (5.9), the co-ordinates of images at later time $t + \Delta t$ become,

$$\begin{bmatrix} x_i(t + \Delta t) \\ y_i(t + \Delta t) \\ z_i(t + \Delta t) \end{bmatrix} = \begin{bmatrix} g_{xx} & g_{xy} & g_{xz} \\ 0 & g_{yy} & g_{yz} \\ 0 & 0 & g_{zz} \end{bmatrix} \begin{bmatrix} x_i(t) \\ y_i(t) \\ z_i(t) \end{bmatrix}. \quad (5.11)$$

Here,

$$z_i(t + \Delta t) = g_{zz}z_i(t) \quad (5.12)$$

$$= g_{zz}(z_0 + n_z L_z(t)) \quad (5.13)$$

$$= g_{zz}z_0 + n_z L_z(t + \Delta t). \quad (5.14)$$

Since n_z is an integer and $L_z(t + \Delta t)$ is the updated z -length according to equation (5.7), there exists a unique n_z such that $z_i(t + \Delta t) \in [-L_z/2, L_z/2)$.

Similarly,

$$y_i(t + \Delta t) = g_{yy}y_i(t) + g_{yz}z_i(t) \quad (5.15)$$

$$= g_{yy}(y_0 + n_y L_y(t)) + g_{yz}(z_0 + n_z L_z(t)) \quad (5.16)$$

$$= n_y L_y(t + \Delta t) + (g_{yy}y_0 + g_{yz}z_0 + n_z g_{yz} L_z(t)). \quad (5.17)$$

For a fixed n_z , the terms inside the parenthesis are fixed. Using the similar argument for z -component as above, there exists a unique n_y such that $y_i(t + \Delta t) \in [-L_y/2, L_y/2)$. Similarly, we can show that there exists a unique n_x when n_z and n_y are fixed such that $x_i(t + \Delta t) \in [-L_x/2, L_x/2)$. This proves that if the velocity gradient tensor is in the form of equation (5.9), it is ensured that there is always one and only one image of a particle (or particle itself) inside the MD box.

As shown in figure 5.2, a rectangular domain can be deformed into a parallelogram if the non-diagonal terms of the velocity gradient tensor are non-zero. Instead of using parallelogram, the domain is kept rectangular. For any atoms who are left outside the rectangular domain due to this transformation such as atom x in figure 5.2 is

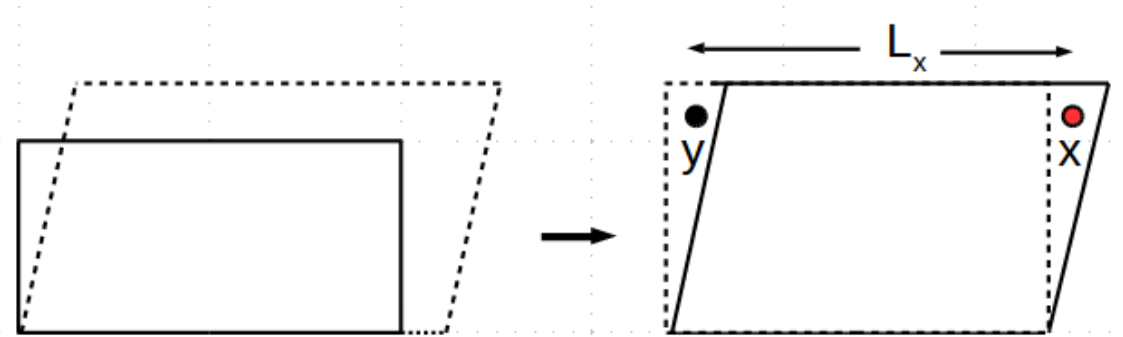


Figure 5.2 : The periodic boundary condition used to address shear deformation. The periodic domain is kept rectangular. For any particle x outside the rectangular domain is discarded and the image of x (i.e., y) is tracked.

discarded and its image (y) inside the rectangular box is tracked. This ensures the number of atoms inside the fundamental box remains constant.

However if there is a non-zero element in the lower triangle of equation (5.9), this property can not be guaranteed. As a counter example, let us consider an example of pure shear in 2D where the deformation gradient tensor is,

$$\begin{bmatrix} 0 & \gamma \\ \gamma & 0 \end{bmatrix}, \quad (5.18)$$

where γ is the shear strain. As shown in figure (5.3), a rectangular box is deformed to a parallelogram (dotted lines). After deformation, particle x shifts to x' . Also the image of x (i.e., y) shifts to y' after deformation. Since the diagonal elements of velocity gradient tensor are zero, according to equation (5.7), the fundamental box does not change. Both x' and y' are inside the fundamental box. In this way, total number of particles inside the MD box will not remain constant.

Equation (5.9) is a special case of gradient of velocity. Generally, the velocity

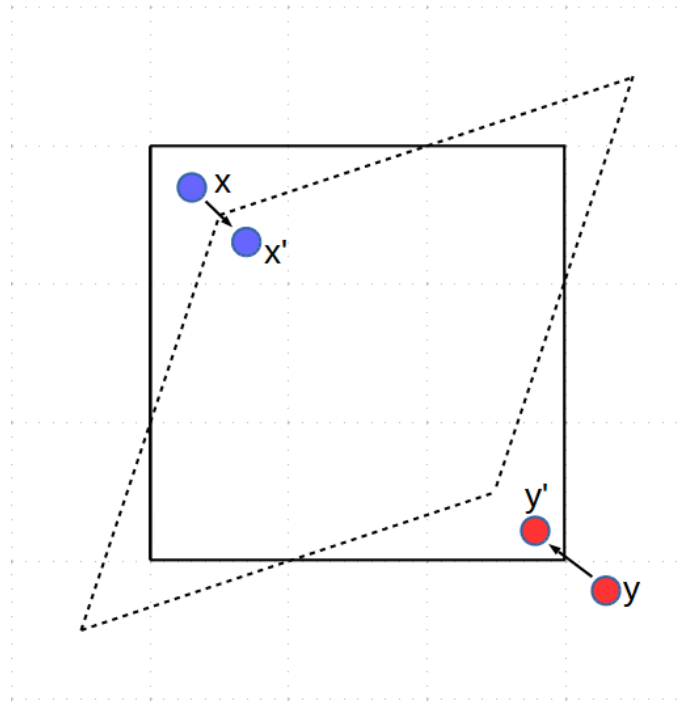


Figure 5.3 : Periodic boundary condition used for pure shear deformation. The particle and its image can both be inside the rectangular box.

gradient tensor has the following form,

$$\nabla \mathbf{v} = \begin{bmatrix} \frac{dv_x}{dx} & \frac{dv_x}{dy} & \frac{dv_x}{dz} \\ \frac{dv_y}{dx} & \frac{dv_y}{dy} & \frac{dv_y}{dz} \\ \frac{dv_z}{dx} & \frac{dv_z}{dy} & \frac{dv_z}{dz} \end{bmatrix}. \quad (5.19)$$

We wish to convert the velocity gradient tensor to the form of upper triangular matrix as in equation (5.9). To achieve this, we add a rotational tensor to the velocity gradient tensor,

$$\nabla \mathbf{v}' = \begin{bmatrix} \frac{dv_x}{dx} & \frac{dv_x}{dy} & \frac{dv_x}{dz} \\ \frac{dv_y}{dx} & \frac{dv_y}{dy} & \frac{dv_y}{dz} \\ \frac{dv_z}{dx} & \frac{dv_z}{dy} & \frac{dv_z}{dz} \end{bmatrix} + \begin{bmatrix} 0 & \frac{dv_y}{dx} & \frac{dv_z}{dx} \\ -\frac{dv_y}{dx} & 0 & \frac{dv_z}{dy} \\ -\frac{dv_z}{dx} & -\frac{dv_z}{dy} & 0 \end{bmatrix} = \begin{bmatrix} \dot{\epsilon}_{xx} & 2\dot{\epsilon}_{xy} & 2\dot{\epsilon}_{xz} \\ 0 & \dot{\epsilon}_{yy} & 2\dot{\epsilon}_{yz} \\ 0 & 0 & \dot{\epsilon}_{zz} \end{bmatrix}, \quad (5.20)$$

where we used the definition of strain rate, for e.g., the xy - component is defined as $\dot{\epsilon}_{xy} = \frac{1}{2} \left[\frac{dv_x}{dy} + \frac{dv_y}{dx} \right]$. Adding a rotational tensor is equivalent to using a rotational frame of reference. We assume that this addition of rotational tensor does not affect stress calculation for fairly general cases except for problems involving extreme shear deformation where the non-inertial effects become important. The examples considered in this thesis work do not involve high shear strain rate even though the normal component of strain rate can be extreme. Further examination of validity of this method for high shear strain rate problems are left for future study.

5.3 Embedded atom method

The use of pair potential where the interaction between a pair of atoms is independent of the surrounding atoms, is a popular choice for potential in MD simulation. One example is the well-known Lenard-Jones potential. But in metals, the valence electrons can be shared among many atoms. The pair potential itself will not be enough to capture the physics of metallic bonding. The embedded atom method (EAM) potential [38, 39] takes into account the effect of electron density surrounding an atom, thereby, incorporates the many-body effect among atoms.

The total energy of a system in EAM method can be written as

$$E = \frac{1}{2} \sum_{\substack{\alpha, \beta \\ \alpha \neq \beta}} \phi(r_{\alpha\beta}) + \sum_{\alpha} F(\Gamma_{\alpha}), \quad (5.21)$$

where the first summation is over all atom pairs, $\phi_{\alpha\beta}$ is the pair potential, $r_{\alpha\beta}$ is the distance between atoms α and β , F is the embedding energy as a function of host

electron density Γ_α . The embedding energy term represents the many-body effect. The host electron density can be written as the sum of the electron densities $\gamma(r_{\alpha\beta})$,

$$\Gamma_\alpha = \sum_{\beta(\neq\alpha)} \gamma(r_{\alpha\beta}). \quad (5.22)$$

In this thesis, instead of using analytical form of potential, a tabulated potential is used. In the study of cerium metal, a tabulated potential generated by Sheng *et al.* [40, 41] has been used. For EAM potential of copper, we use the potential data developed by Mishin *et al.* [42] in tabulated form. The potential data consists of three tables, F_α as a function of Γ_α , $\gamma(r_{\alpha\beta})$ and $\phi(r_{\alpha\beta})$ as functions of $r_{\alpha\beta}$. If necessary, the data are interpolated using Lagrangian interpolation. By differentiating this energy, we can calculate total force acting on atom α as [43]

$$\mathbf{f}_\alpha = -\nabla_\alpha E, \quad (5.23)$$

where ∇_α denotes the gradient acting on the position of the α -th atom.

5.4 Stress calculation using EAM method potential

To calculate stress in the periodic computational domain while avoiding errors related to the interaction pairs near the boundary of our MD domain, we use pair interaction forces and the first line of (2.25). This is similar to [10], but without the time averaging. As shown in (2.25), such calculated stress is consistent with virial stress expression of the Cauchy stress, which is independent of force decomposition in the limit of continuum mechanics in a thermodynamic equilibrium. In EAM method

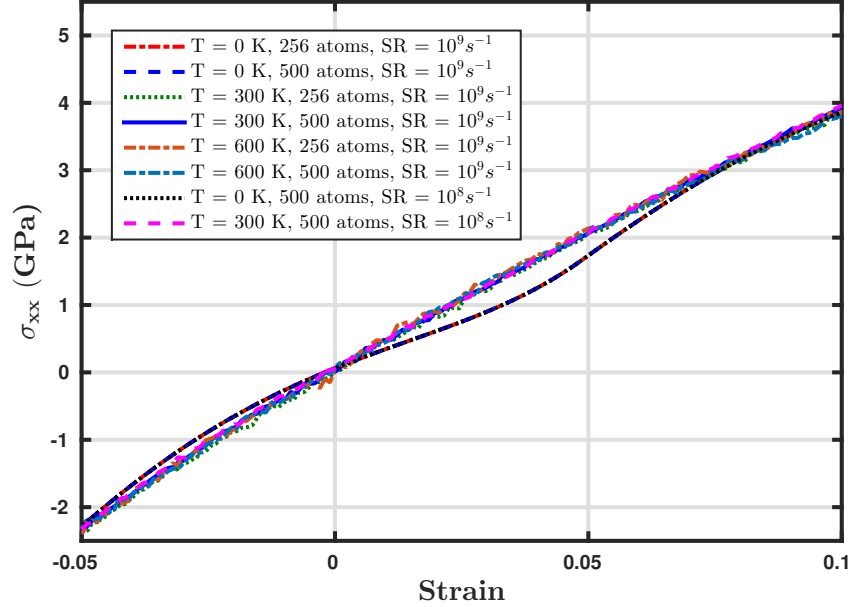


Figure 5.4 : Stress-strain Relation of Cerium at different temperatures. The strain used in this figure is the engineering strain $\varepsilon_{xx}^e = (L_{px}^n - L_{px}^0)/L_{px}^0$.

potential, the pair interaction force between a pair of atoms is calculated as [43, 44]

$$\mathbf{f}_{\alpha\beta} = - \left[\frac{\partial\phi(r_{\alpha\beta})}{\partial r_{\alpha\beta}} + \frac{\partial F}{\partial\Gamma_i} \frac{\partial\gamma}{\partial r_{\alpha\beta}} + \frac{\partial F}{\partial\Gamma_j} \frac{\partial\gamma}{\partial r_{\alpha\beta}} \right] \frac{\mathbf{r}_{\alpha\beta}}{r_{\alpha\beta}}. \quad (5.24)$$

The pair interaction force in equation (5.24) satisfies the Newton's 3rd law i.e., $\mathbf{f}_{\alpha\beta} = -\mathbf{f}_{\beta\alpha}$. So $\mathbf{f}_{\alpha\beta}$ defined in equation (5.24) can be used for stress calculation [27]. In our simulation the molecular dynamics calculations are performed on a GPU. With such pair interaction force, the stress calculated is passed to CPU, which performs the continuum DDMP calculation as described in the previous chapter.

To confirm that such calculated stress is in agreement with the classical concept of stress in elasticity theory for continuum mechanics, in Fig. 5.4 we show stress-strain relation for cerium at initial temperatures of 0 K, 300 K, and 600 K. The results are

obtained using 256 atoms and 500 atoms in the periodic domain with initial domain sizes of 20.5Å and 25.7Å. These results are obtained by imposing strain rates 10^8 1/s and 10^9 1/s to the system using time step $(\Delta t)_{md} = 1$ fs in the MD simulation. Time step $(\Delta t)_{md} = 1$ fs is about 1% of the phonon period for cerium [45]. For cases of nonzero initial temperature, according to the initialization process, at zero strain (undeformed crystal with lattice constant 5.132Å), there is a small stress due to thermal fluctuations. The stress plotted in Fig. 5.4 has been offset by this stress to ensure that the curve passes the origin in the figure. Other than small fluctuations, for a given initial temperature, the stress-strain relation is insensitive to the number of atoms and the strain rates used in our MD simulations. The Young's moduli can be calculated from the slopes of the curves at zero strain in this plot and are found to be 16.9 GPa at temperature 0 K and 25.6 GPa at temperature 300 K, compared to the value of 25.3 GPa at 300 K obtained by Sheng *et al.* [40, 41].

5.5 Molecular dynamics in GPU

The increasing computing power of Graphics Processing Units (GPU) has opened up many possibilities of using GPUs, not only for graphics purpose but also for high performance computing. For GPU based software development, there are many graphics-oriented languages such as CUDA, OpenGL, Cg, etc. The CUDA language is based on standard C and makes it easy for users who are already familiar with C or C++. In our numerical experiments, the molecular dynamics calculations are

performed in GPU using CUDA. Since the molecular dynamics calculations are performed to calculate the ensemble averaged quantities, such as stress in our model, the MD simulation in GPU is performed with single precision to achieve higher speed up.

To perform MD simulation in parallel, there are basically three different methods [46]. The first is atom decomposition method, where each processor is assigned to N/P atoms when simulating N atoms on P processors. In this method, each processor computes forces and updates the positions and velocities of the assigned atoms. The second is domain decomposition method. The computational domain is divided into different blocks and those blocks are assigned to different processors. Each processor computes forces on the atoms in its block and also tracks if any atom entered or left the block. The third is force decomposition method, where rather than assigning a processor to compute all forces in N/P atoms as in the atom decomposition method, each processor is assigned to calculate a block of the force matrix. The force matrix is $N \times N$ matrix with all the pair interaction force information.

The most computationally expensive part of any MD simulation is the force calculation. For a system of N atoms, it typically requires $N(N - 1)/2$ operations. For short range forces, there are different algorithms to achieve near $O(N)$ operations for force calculations. In neighbor list method, we create a list of neighbors for each atom with a radius of slightly more than cutoff distance ($r_c + \Delta r$) corresponding to that potential. The calculation of force experienced by an atom α involves searching through all the atoms in its neighbor list instead of searching all the atoms in the com-

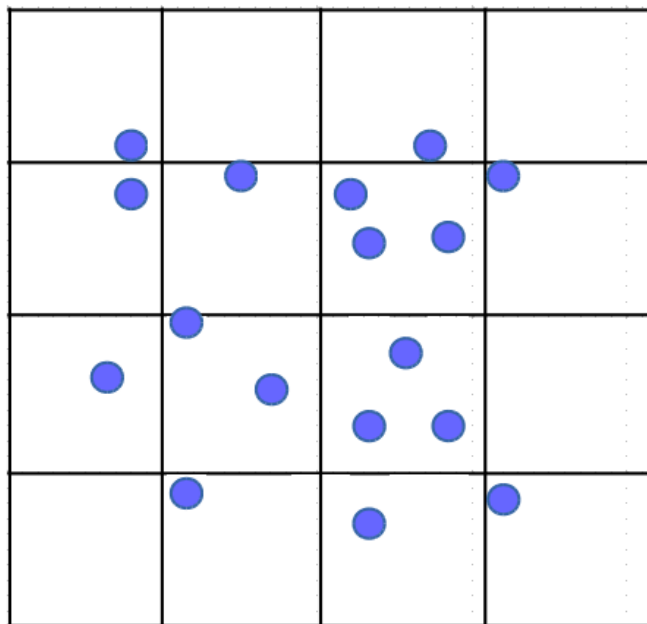


Figure 5.5 : Atoms in uniform grids. The atoms are later sorted according to the uniform grids.

putational domain. The neighbor list must be updated before any atom can travel a distance Δr . Although this method is very efficient, the memory requirement is high since each atom will have several atoms in the list. The large memory requirement makes this method not suitable for GPU. Another widely used algorithm is the linked list method, in which the spatial domain is divided into many cells of length equal to or slightly larger than r_c . For each cell, a linked list of atoms is created in every time step. The creation of linked list requires $O(N)$ operations. To calculate force, for each atom, the atoms in the same cell and the neighbor cells are examined. For example, there will be 27 cells to be examined in 3D. This method is slightly more expensive than the neighbor list method since the search volume is greater. Even

though this method does not require additional memory, looping through the linked list is not ideal for parallel computation.

The algorithm implemented in this thesis to perform MD simulation is similar to what described in [47] for any particle simulation. Similar to domain decomposition method, the computational domain is subdivided into uniform cells. The size of the cells is equal to the cutoff radius (r_c) corresponding to the inter-atomic potential. Then for each atom, its cell ID is calculated based on its position. Figure 5.5 depicts the way the atoms are assigned to the cells corresponding to their positions. The atoms are sorted based on their cell ID. For sorting we use *sort_by_key* function of CUDA thrust library. This function creates a list of atom IDs in cell order. To make the sorted list useful, we find the start and end address of any given cell in the sorted list. For example, in order to find the start address of a cell, the cell ID of current atom with the previous atom is compared. If the cell IDs are different, this indicates the start of a new cell. Now reorder the position and velocity arrays of all atoms into sorted order. This step is optional, but this improves the memory access coherency since the atoms close in space tend to be closer in memory address. Finally, the force calculation is performed by searching neighbor atoms in the neighbor cells.

Chapter 6

Shock waves simulated using the dual domain material point method combined with molecular dynamics

In this Chapter we use a one-dimensional strong shock propagation in cerium single crystal to demonstrate the feasibility of the proposed multi-scale method. This multi-scale method is based on the dual domain material point method. The continuum level calculation is performed using DDMP method. Instead of using constitutive relation, we perform molecular dynamics (MD) simulations to calculate stress in the material points. In this multi-scale computation, each material point is a MD system. The MD systems communicate with the continuum level calculation through strain rate and stress. The strain rate calculated from the DDMP calculation are used as a boundary condition to constrain the MD system, and the stress obtained from the MD simulation is used to drive the DDMP calculation. Since the material points do not need to communicate among each other, the MD simulations are performed in parallel in GPU using CUDA to accelerate the computation. Figure 6.1 outlines the multi-scale calculation method used in this thesis.

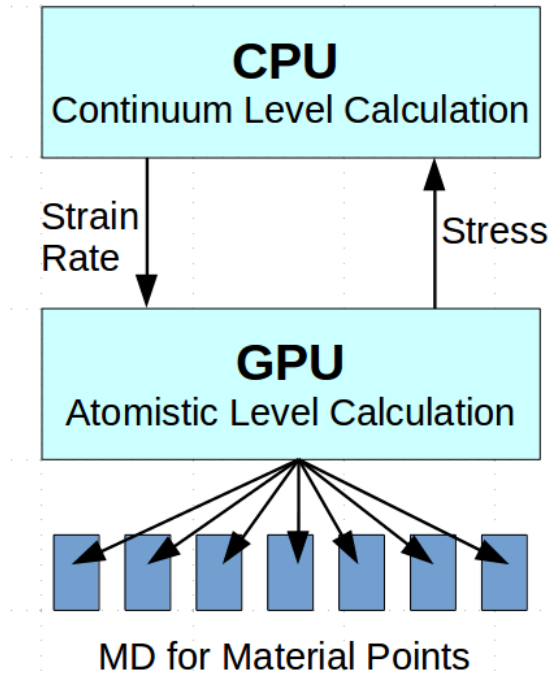


Figure 6.1 : Schematic diagram of the multi-scale method

6.1 Continuum level calculation

For continuum level calculation, we use DDMP method with subpoints as described in Chapter 4.3. The DDMP method can be used to compute large material deformation, provided the stresses at the material points can be obtained. We study a method of performing macroscopic simulations for systems far away from their thermodynamic equilibriums by computing these stresses directly from molecular dynamics simulations at each material point. We take the advantage that the material points are Lagrangian points. These material points are tracked throughout the computation so that the history of the material can be considered in the stress calculation. In these

types of calculations, the stress evaluation at material points is expensive. To reduce the number of material points needed, while having sufficient numerical accuracy in the DDMP internal force calculation, the recently developed sub-point algorithm as described in Chapter 4.3 [48] is used. The algorithm exploits the difference between the stress variation length scale, which is the macroscopic length scale L_m , and the length scale of the gradient of the shape function, which is the typical mesh size Δx . As illustrated in Fig. 4.7, in a meaningful calculation, the typical mesh size $\Delta x \ll L_m$. The stress varies much slower than the modified gradient $\overline{\nabla S_i}$ of the shape function. To ensure numerical accuracy in the internal force calculation, we split a material point, or particle, into many integration points, called sub-points. The values of $\overline{\nabla S_i}$ are evaluated individually on these sub-points to account for rapid variation of the modified gradient of the shape function, while the sub-points belonging to the same particle share the same stress. These sub-points can be generated at each time step around a particle. They are introduced purely to account for variations of $\overline{\nabla S_i}$ and do not carry any physical information; therefore there is no need to track the history of these sub-points. As in the ordinary material point methods, the history information is carried by the particles. The MD simulation to calculate stresses are performed for the particles not for sub-points. Since the generation and calculation associated with these sub-points are very cheap compared to the stress calculation on the particles, a large number of sub-points can be used without incurring too much computational cost. As shown in Chapter 4.3 [48], the numerical

accuracy of this sub-point algorithm is very close to the DDMP calculation as if using the same number of particles instead of sub-points. This sub-point DDMP method is equivalent to further replacing the gradient of the shape function in (3.31) by

$$\overline{\overline{\nabla S_i}}(\mathbf{x}_p) = \frac{1}{n_{ps}} \sum_{s=1}^{n_{ps}} \nabla S_i(\mathbf{x}_{ps}), \quad (6.1)$$

where n_{ps} is the number of sub-points for material point p , and \mathbf{x}_{ps} is the position of a sub-point, which is generated around the material point at each time step. In this way, the internal force in (3.31) is calculated by

$$\mathbf{f}_i^{int} = - \int_{\Omega} \boldsymbol{\sigma} \cdot \nabla S_i(\mathbf{x}) dv \approx - \sum_{p=1}^{N_p} \frac{V_p \boldsymbol{\sigma}_p}{n_{ps}} \cdot \sum_{s=1}^{n_{ps}} \overline{\overline{\nabla S_i}}(\mathbf{x}_{ps}). \quad (6.2)$$

Sub-point algorithm is described in details in Chapter 4.3 [48]. To ensure energy conservation (to the second order of the spatial and temporal discretization), the velocity gradient at a particle, which is used to calculate stress, also needs to be computed using new gradient of the shape function as

$$\overline{\overline{\nabla \tilde{\mathbf{v}}}}(\mathbf{x}_p, t) = \sum_{i=1}^{N_g} \tilde{\mathbf{v}}_i(t) \overline{\overline{\nabla S_i}}(\mathbf{x}_p) = \sum_{i=1}^{N_g} \frac{\tilde{\mathbf{v}}_i(t)}{n_{ps}} \sum_{s=1}^{n_{ps}} \nabla S_i(\mathbf{x}_{ps}). \quad (6.3)$$

6.2 Molecular dynamics simulation of Cerium

Although the continuum problem is in one dimension, the MD calculation is performed in a three-dimensional periodic domain to minimize the effects of boundary [11, 10, 12]. In the MD calculation, the Embedded Atom Method (EAM) [38, 39] potential is used. The EAM method potential as well as the integration scheme and

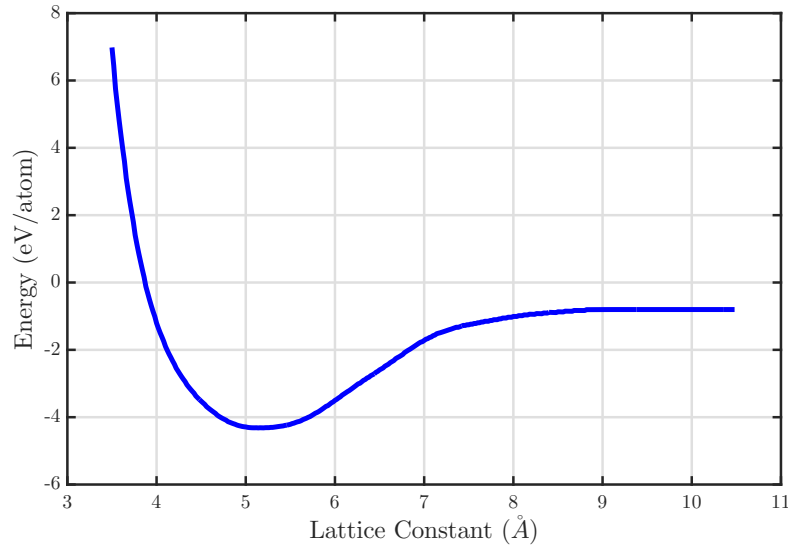


Figure 6.2 : Total energy per atom as a function of lattice constant for Cerium crystal

the boundary condition for MD is described in Chapter 5. In this study of cerium metal, a tabulated potential generated by Sheng *et al.* [40, 41] has been used.

At the beginning of our calculation, one molecular dynamics (MD) system is generated for each material point. To initialize the MD systems, we first place Cerium atoms in the cubic periodic domains $[-L_{px}^0/2, L_{px}^0/2] \times [-L_{py}^0/2, L_{py}^0/2] \times [-L_{pz}^0/2, L_{pz}^0/2]$ with $L_{px}^0 = L_{py}^0 = L_{pz}^0$, according to face center cubic (FCC) array. The lattice constant of the arrays is set to the value (5.132 Å) where the potential energy E is the minimal. The periodic length is a multiple of this value. To specify initial engineering strain $\varepsilon_{xx}^{e0}(\mathbf{x}_p)$ of the material point in the MD system, for our one-dimensional macroscopic problem, the x -coordinate of the atoms and x -length (L_{px}^0) of the periodic domain are multiplied by $1 + \varepsilon_{xx}^{e0}(\mathbf{x}_p)$.

To set nonzero initial temperature in the system, the velocities of the atoms are set

randomly according to the Maxwell distribution with their magnitudes corresponding to twice the intended initial temperature, and the velocity directions are set according to uniform distribution in the solid angles. The final procedure in the MD system initialization is to relax the system to its thermodynamic equilibrium. In this procedure the equation of motion for atoms are solved as describe below, but with the strain rate set to zero. According to the virial theorem, which is numerically verified, in our cases the total initial kinetic energy is split equally between the kinetic energy and the potential energy among the atoms at the thermodynamic equilibrium; therefore, the relaxed system has the temperature of the intended value. The MD systems are assumed adiabatic in our simulations after this temperature initialization. This initialization of the MD system is done once and only once for every material point in the computation. The MD systems are never re-initialized, and their histories are preserved in the present work. The end states of the MD systems of the previous DDMP time step are the initial states for the MD calculations in the next DDMP time step. The only difference in the MD simulations in two consecutive DDMP time steps are the velocity gradients, or the strain rates, used to update the sizes of the periodic computational domains. We use the leap frog algorithm to advance the MD systems as described in Chapter 5.1.

The method presented in this work is intended to resolve the effects of thermodynamic nonequilibrium in macroscopic problems, where the time scale of the macroscopic strain rate is comparable to the thermodynamic relaxation time. We have no

choice but to use time steps in the continuum DDMP calculation much smaller than the time scale of the strain rate. Clearly, this method should not be used directly for problems in which the strain rate is small compared to the inverse of the thermodynamic relaxation time. For such systems, for each continuum DDMP time step, we only need to perform sufficient MD time steps for each of the MD systems to reach its local thermodynamic equilibrium [49], then perform a time average and extrapolate the stress to be used in the continuum DDMP time step. In this limit this method reduces to category B of the HMM as described by [10, 11, 12]. The only difference is then in the continuum solvers. The finite element and the finite volume methods are used in [10, 11], while we use the DDMP method, which is more advantageous if the history dependence and large material deformation is of concern.

The stress calculation method in the periodic computational domain corresponding to each material point is described in Chapter 5.4.

6.3 Numerical Results and Discussion

In this section we explore the numerical properties of the multiscale calculation described above. Since the main purpose of this work is to study a numerical method to perform multiscale calculations, instead of improving the MD method or the EAM potential used in our calculations, we compare our results from our multiscale simulations to those obtained from the direct MD simulations, rather than to experimental values. We study a one-dimensional cerium bar of length about 500 nm. Although

the multiscale method described in the present work can be used to compute macroscopic problems, we limit our comparison with the MD results to this small length because of length limit of the MD simulations. Initially, the left half of the bar is compressed to a specified strain and held. In the combined MD-DDMP simulation, after properly initializing densities on mesh nodes, the MD systems for the material points are initialized with the initial strain and temperature as described in the last section 6.2. Initialization of the direct MD simulation is similar. Atoms are placed in the computational domain according to FCC array. In the direct MD calculation we use periodic boundary condition in y and z directions and reflective boundary condition on both ends of the cerium bar. There are 12 lattices in the y and z directions and 1200 lattices in the x direction. To specify an initial engineering strain ε_{xx}^{e0} in left half of the bar x -distance between atoms on the left is adjusted by a factor of $1 + \varepsilon_{xx}^{e0}$.

At the time $t = 0$, the hold is released causing a compression shock to propagate to the right and a rarefaction wave to propagate to the left. In the direct MD simulation, the equations of motion for the atoms are solved as described in the Chapter 5.1. To calculate the stress from this direct MD simulation, the entire MD domain is divided into 100 sections along the x -direction. We then treat each section as a representative volume, and use the first line of equation (2.25) to calculate the stress in this section.

For DDMP and combined DDMP-MD calculations, the computational domain consists of 400 cells for the results shown in Figs. 6.3 to 6.8. To avoid difficulty

associated with the initial discontinuity, as in many particle based methods [50, 51], we use a hyperbolic tangent function to smooth it across 26 cells, with 13 cells on each side of the initial discontinuity. As shown in Figs. 6.3 in the case of weak shock, the shock front is broadened to about 50 cells for both the continuum DDMP and the multiscale MD-DDMP calculations due to artificial viscosity. In this work the pressure from the artificial viscosity only involves the linear term $0.5\rho c_s|\Delta v|$ [52], where ρ is the density of the material point, c_s is the sound speed. The artificial viscosity is applied only if the strain rate is negative. Using the same artificial viscosity for strong shocks, shock front is only broadened by 4 cells to total of 17 cells for the multiscale MD-DDMP calculation and is broadened by 29 cells to total about 42 cells for the continuum DDMP calculations as shown in Figs. 6.4 to 6.9.

Figure 6.3 shows the comparison among the results from the full MD simulation using about 700,000 atoms, the continuum DDMP calculation using 400 equal cells with initially 1 material point per cell and 8 sub-points per material point [48], and the multiscale DDMP-MD simulation with the same number of cells and material points as in the DDMP calculation. For this case of moderate initial compressive strain of 0.5%, corresponding to about 0.164 GPa compressive stress on the left, although the combined DDMP and MD simulation result is slightly closer to the MD calculation, the overall results are in good agreement with each other.

Using the same numerical arrangement, we calculated a case with ten times higher initial compressive strain of 5.0 %, corresponding to compressive stress 2.3 GPa, on

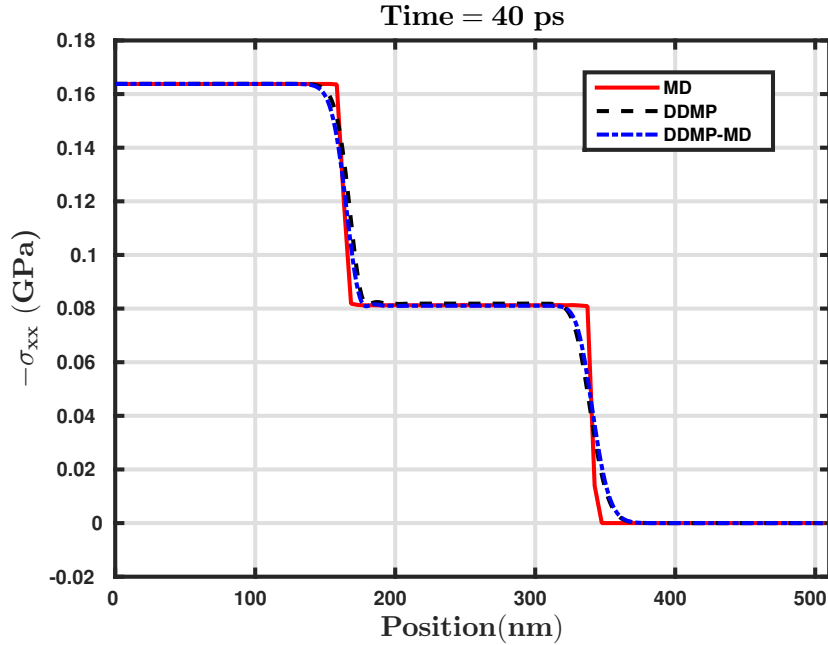


Figure 6.3 : Shock wave propagation for weak shock at 0 K initial temperature calculated using 400 cells with 1 material point per cell and 8 sub-points per material point.

the left of the bar. The results are shown in Fig. 6.4. In this figure, the continuum DDMP results deviate from the MD results showing smaller wave speeds for both the compression and rarefaction waves. The deviation is not caused by the DDMP numerical method, but rather by the constitutive assumption about the linear elasticity of the material. The multiscale DDMP-MD results are much closer to the MD results, especially in the center region of the figure. In this region, the material is significantly perturbed by the compression or rarefaction wave, and is not in the thermodynamic equilibrium. Such thermodynamic nonequilibrium is not considered in the linear elastic model used in the DDMP calculation, resulting the difference between the DDMP and MD results. The effect of this thermodynamic nonequilibrium

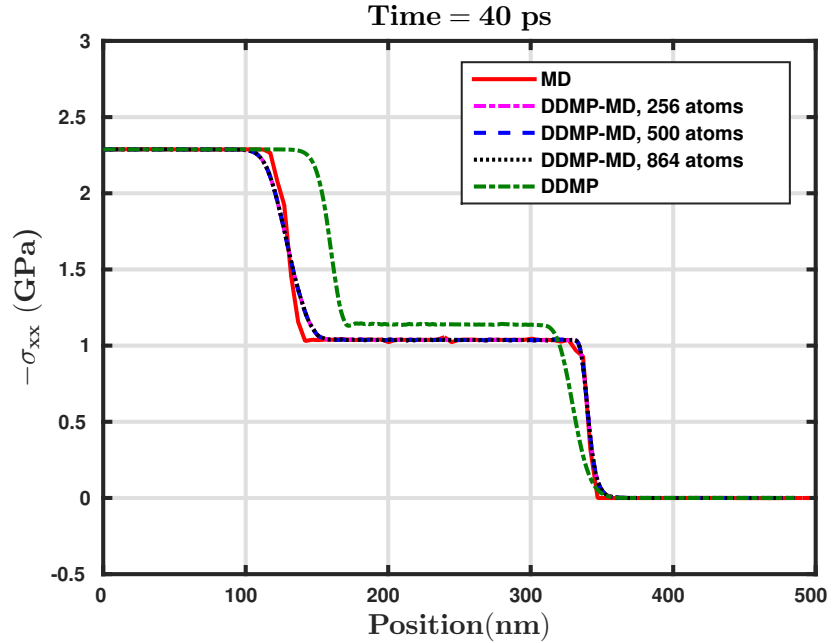


Figure 6.4 : Propagation of strong shock at 0 K initial temperature calculated using 400 cells with 1 material point per cell and 8 sub-points per material point.

is captured in the MD simulation performed at each material point leading to good agreement between the results from the MD and the DDMP-MD combined multiscale calculations. In this figure, we also show the combined DDMP-MD results obtained from different number of atoms in a MD system. There are only slight differences between the results by using different number of atoms.

Figure 6.5 shows similar calculation for a case with 300 K initial temperature. In this figure, again the multiscale DDMP-MD result is very close to that from the MD simulation. In this figure the comparison between the continuum DDMP result and the MD results is much better than in Fig. 6.4, because compared to the system starting from zero absolute temperature, the system with a finite temperature relaxes

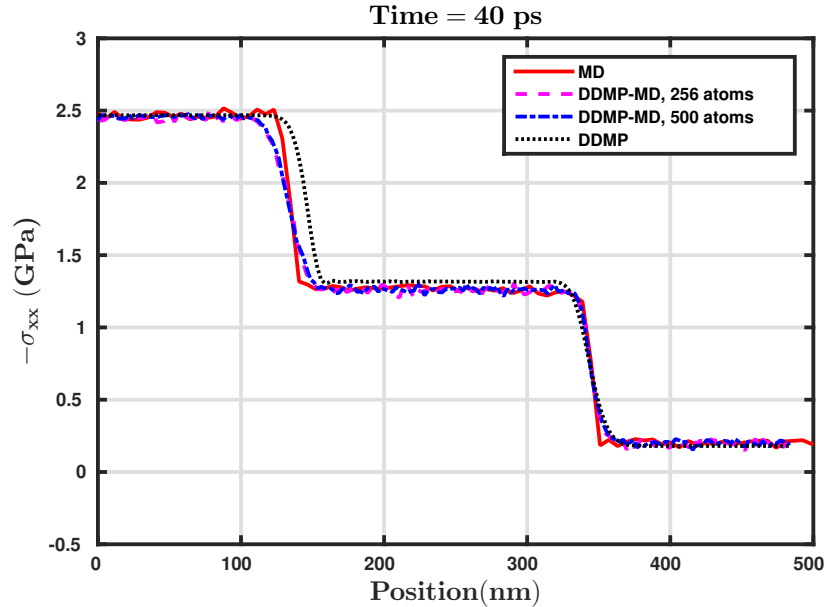


Figure 6.5 : Propagation of strong shock at 300 K initial temperature calculated using 400 cells with 1 material point per cell and 8 sub-points per material point. In the combined DDMP-MD results, stresses at material points are obtained from the MD simulations using 256 and 500 atoms per material point.

faster to its thermodynamic equilibrium, where the continuum theory provides a good approximation. This is also the reason for good agreement among all the results in Fig. 6.3 since the system was never far away from thermodynamic equilibrium for the weak shock. Figure 6.5 also shows noise in the stresses obtained from both the direct MD and DDMP-MD combined calculations. This noise can be reduced by using larger number of atoms in both calculations. The result obtained using 500 atoms per material point is less noisy than that obtained using 256 atoms per material point. These results show that even with small number (256 or 500) of atoms in our MD simulation for each material point, the results still capture the significant effects of thermodynamic nonequilibrium with reasonable accuracy. In this calculation, for the

unperturbed material on the right, the initial stress is negative because of velocity fluctuations of the atoms as represented by first term in (2.25) for this case of finite temperature.

In Figs. 6.3, 6.4 and 6.5, the entire domain MD results are obtained from calculations performed in parallel on a NVIDIA Tesla K20 GPU. Each took about 6 hours. The MD calculations for the multiscale DDMP-MD method are also performed on the GPU, whereas the continuum scale calculations are performed in a CPU. The DDMP-MD calculations took about 15 minutes each, and the continuum DDMP calculation took only a few CPU seconds. We use the time steps of 500 fs in the DDMP calculations, while the time step for the MD simulation is 1 fs. The 500 fs time step in the continuum part of the simulation corresponds Courant number about 0.8, depending on the wave speed, which varies during the calculation. For the combined DDMP-MD calculation, the stress used in the continuum DDMP calculation is calculated using (2.25) at the last MD time step immediately before the continuum time step. As another option one can also use the average stress calculated at each MD time step. The difference shown in Fig. 6.6 is not significant, but the computation cost differs remarkably. Since the stress calculation requires global sum over the atom pairs, which cannot be performed effectively in parallel on a GPU, the calculation using the average stress is about 1.8 times slower than the method that simply evaluates the stress at the last time step of the MD simulation before the DDMP calculation. For this reason, the latter method is used to obtain the results reported in the rest of the

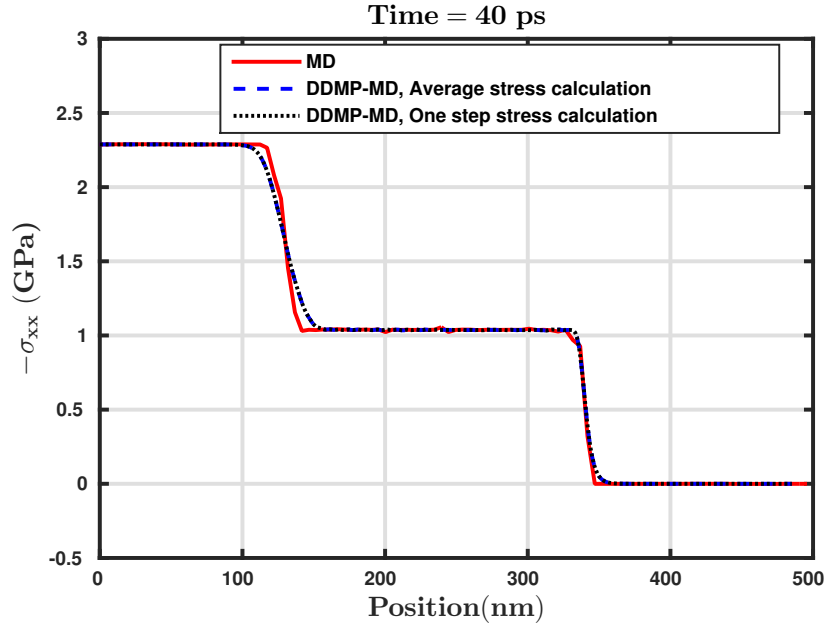


Figure 6.6 : Comparison of results from using the stress calculated from one MD time step stress and using the average stress over the one DDMP time step for cases of 0 K initial temperature.

present work.

To study numerical properties of the combined DDMP-MD simulation, without the contamination of the noise from thermal fluctuations, in the following, we use systems with initial temperature of absolute zero to study the effect of artificial viscosity, the effect of the number of material points, and mesh sizes used in the calculation. The effect of artificial viscosity is shown in Fig. 6.7. Artificial viscosity is needed to suppress the Gibbs phenomenon near the compression wave front.

In DDMP calculation, we place a number of material points in cells evenly at the beginning of a calculation. The results shown in Fig. 6.8 are obtained with different number of material points in such placement. Because the material points

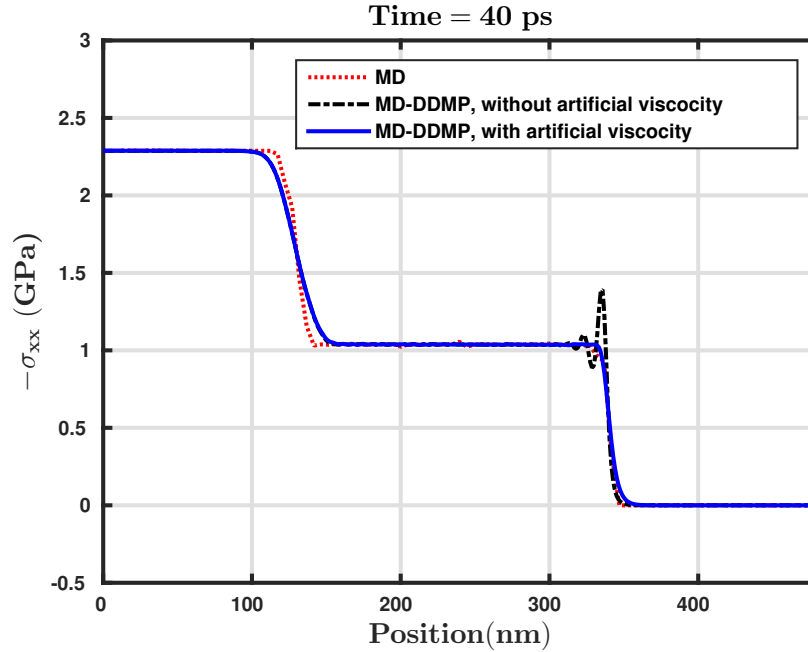


Figure 6.7 : Effects of artificial viscosity for cases of 0 K initial temperature.

are Lagrangian, they move with the material leading to increase or decrease of the number of material points in the cell during the calculation. The number of material points per cell in the figure only refers to the initial placement of the points. In the figure, we compare the results obtained by initially placing 1 material point per cell to that obtained by initially placing 8 material points per cell for the case of strong shock. The result from using 1 material point per cell is noisier as can be seen in the sub figure. The noise is a result of low accuracy of the numerical integration scheme used in the material point method. By using the sub-point algorithm [48], the noise can be significantly reduced. In Fig. 6.8, we also plot the result obtained using 1 material point per cell and 8 sub-points per material point. This result is barely

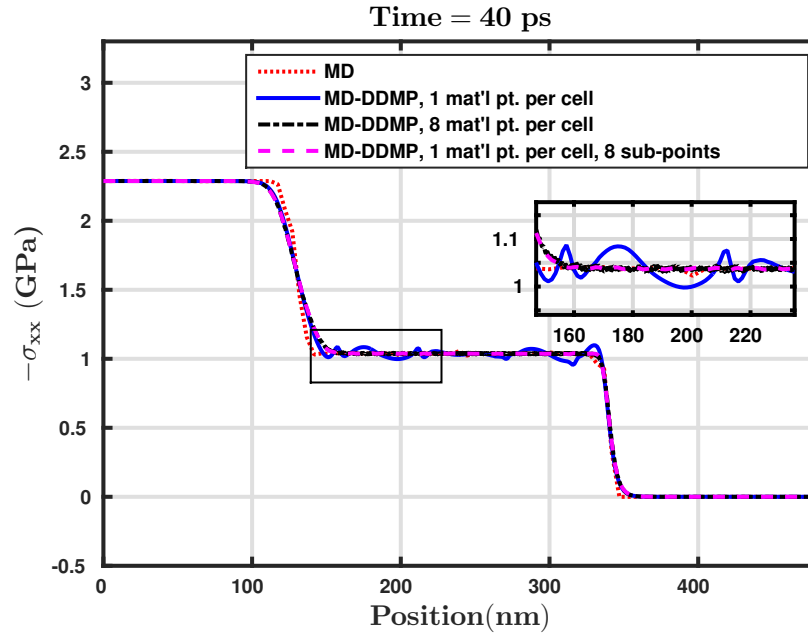


Figure 6.8 : Effects of number of material points or sub-points for cases of 0 K initial temperature..

different from the result obtained using 8 material points per cell. The CPU times are about 15 minutes for the two calculations using 1 material point per cell with or without using the sub-points. These results show that the use of sub-points leads to a significant speedup compare to the time needed for calculation using 8 material points per cell, which took about 110 minutes, without sacrificing the numerical accuracy. The use of sub-points is especially advantageous in this multiscale calculation, where stress evaluation at the material points is expensive.

In Fig. 6.9, we show effects of cell size on this multiscale calculation. This shows the results converge to the correct solution as mesh is refined. The results of the shock propagation calculated in this work are in a very small (500 nm) domain. This

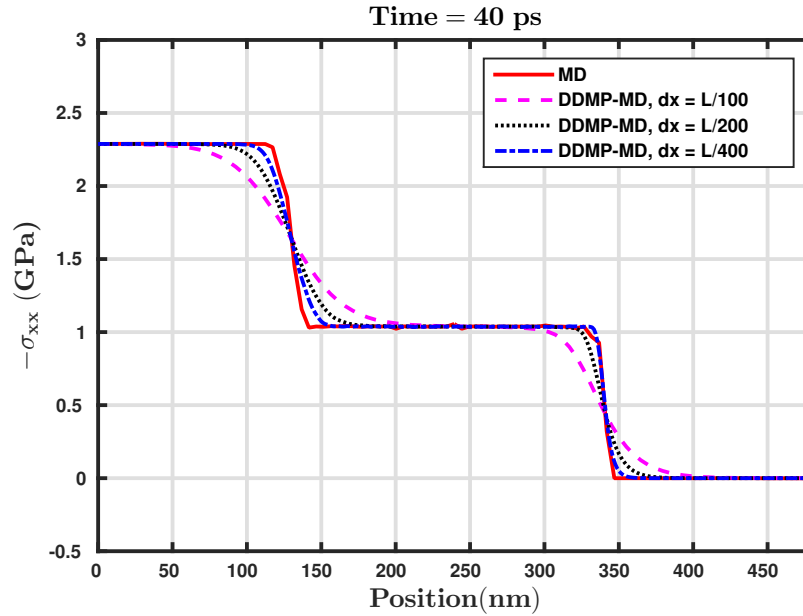


Figure 6.9 : Effect of mesh size in shock propagation at 0 K initial temperature calculated with 1 material point per cell and 8 sub-points per material point.

small domain is chosen because we want to compare our multiscale calculation with the MD calculation. While the MD calculation is restricted to a small domain, the combined DDMP-MD simulation is not because there is no difference in the stress calculation method when this combined DDMP-MD is used for much larger domains. A significant advantage is that this method can be used in a large domain. It can be used to simulate thermodynamic nonequilibrium phenomena involving large spatial domains instead of limited to some special areas, such as phase interfaces or crack tips [53, 54, 55]. To further develop this method and to consider effect of crystal defects, such as twinning and dislocation, large MD systems for the material points are needed.

The continuum level time step size (500 fs) used in the present paper is quite small for a typical continuum scale calculation. However, we do not think this is a significant issue preventing the application of this multiscale method for the following reasons. If, in the systems of interest, the effect of thermodynamic nonequilibrium is important, physical interactions at this small time scale have to be resolved to capture the nonequilibrium effects. Fortunately, for most macroscopic problems, the system relaxes to a thermodynamically equilibrium state quickly compared to the macroscopic time scale. After that time if the constitutive relation or the equation of state for the material is available, this multiscale method is no longer needed. In cases where the the constitutive relation is not available even in the thermodynamically equilibrium state, one can explore the time extrapolation methods [10, 12] by only performing this multiscale simulation in selected time intervals.

6.4 Summary

By taking advantage of the lack of necessity of communication among the material points, an efficient multiscale simulation method is developed based on the dual domain material point (DDMP) method. In this method, the stresses on each material point are calculated independently without the need to communicate with other material points. The material points only communicate with mesh nodes. For problems, where the traditional constitutive relations are not available, we perform molecular dynamics (MD) simulation to obtain the stress and use it in the continuum scale

DDMP calculation. Because no communication among material points is required, in this method, the MD simulations are performed independently and in parallel on a GPU.

The choice of the DDMP method is based on its capability of tracking material deformation history for cases involving extreme material deformations, which are often accompanied by high strain rates, where the assumption of thermodynamic equilibrium becomes invalid, and history dependence becomes important. In the DDMP method, especially with the recent enhancement using sub-points, the noise caused by material points moving across cell boundaries is suppressed, and highly accurate results can be obtained.

Compared to other multiscale methods, this method has an advantage in its capability of simulation a macroscopic domain because each material point is treated as a representative piece of the material that moves and deforms with the material, and carries the history with it.

Chapter 7

Modeling of jet formation around copper notch using dual domain material point method combined with molecular dynamics

In the previous Chapter, we used a one-dimensional shock wave propagation in cerium single crystal to demonstrate the viability of the multi-scale method based on dual domain material point method. In this Chapter, we use the phenomena of jet formation around a notch of a copper crystal due to a strong impact as an example to better validate our new multi-scale computation method intended for modeling extreme material deformation and thermodynamic non-equilibrium. Also, in this example, the DDMP method is used for continuum level calculation and the stress in the material points are calculated using MD simulation. The MD systems communicate with the continuum level calculation through the strain rate and stress tensors. The strain rate calculated from the DDMP calculation are used as a boundary condition to constrain the MD system, and the stress obtained from the MD simulation is used to drive the DDMP calculation. We take advantage of the property of the DDMP method that the material points do not need to communicate among each other in order to perform MD simulation in parallel. The MD simulation in each material point is performed in GPU using CUDA to accelerate the computation. The results

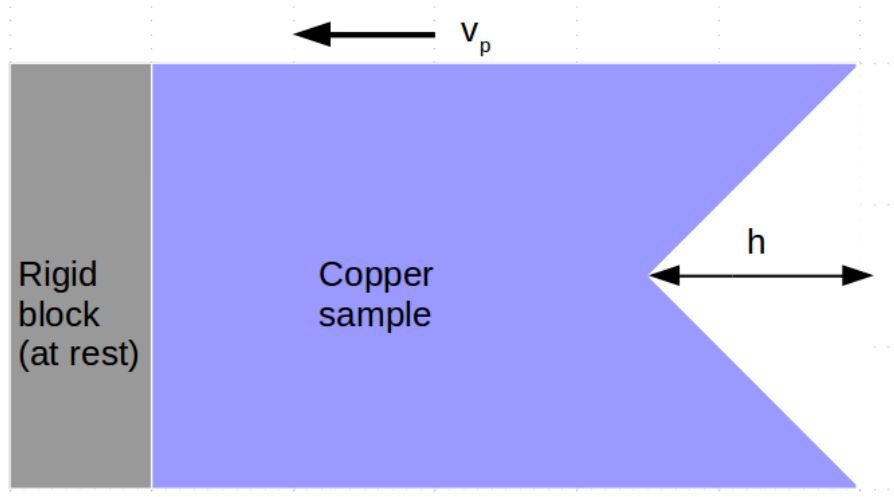


Figure 7.1 : Initial configuration of the copper crystal sample with a piston and a notch

obtained from the multi-scale calculation are compared with direct MD simulation results. First we discuss the direct MD simulation methodology.

7.1 Molecular dynamics simulation of jet formation of copper

We consider a sample of a block of copper single crystal with a groove of depth h on one edge as shown in figure 7.1. A shock wave is generated along the length of the copper using momentum mirror method where the copper crystal is struck into a rigid block at rest with an impact velocity v_p along the shock direction. The shock wave starts propagating along the length of the crystal away from the piston. After the shock wave reaches the groove, the interaction of planar shock wave with the groove causes the material around the groove to form a jet shooting outward. After some time the jet can eventually break up to form ejecta.

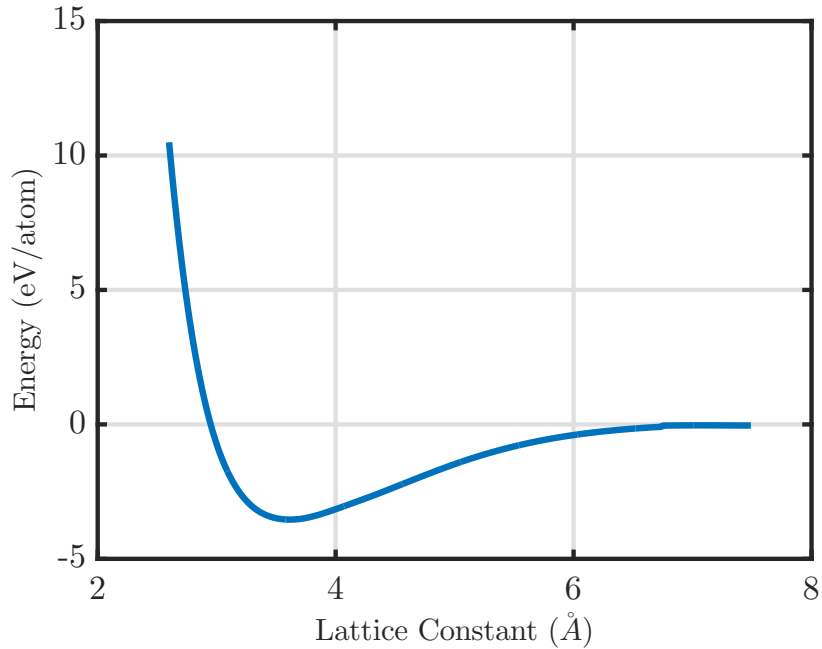


Figure 7.2 : Total energy per atom as a function of lattice constant for copper crystal

The Embedded Atom Method (EAM) potential [38, 39], as described in chapter 5.3, is used for MD calculation. For EAM potential of copper, we use the potential data developed by Mishin *et al.* [42]. The description of how to perform molecular dynamics simulation is in chapter 5.

To initialize the MD simulation, copper atoms are placed in the 3D computational domain to form a perfect face centered cubic (FCC) crystal structure. Similar to the cerium crystal in the previous Chapter, the lattice constant is set to the value where the potential energy is minimum. Figure 7.2 depicts the relationship between lattice constant and total energy per atom for copper single crystal at temperature of 0 K. The minimum energy corresponds to lattice constant (a_0) = 3.615 Å with cohesive

energy -3.54 eV/atom. In this experiment, the x -direction length of the sample is 144 nm corresponding to 400 lattice cells, which also includes the rigid block of size 14.4 nm. For simplicity, the rigid block is also chosen to be made up of copper single crystal. The y -direction length of the sample is 72 nm corresponding to 200 lattice cells. Although the problem of interest is 2-dimensional, the MD simulation is performed in a 3-dimensional box with 10 lattice cells along z -direction. A triangular notch is created with depth $h = 36$ nm as shown in figure 7.1. To create the notch, the atoms in that region are simply removed. To set up a finite temperature, the velocities of the atoms are randomly initialized according to Maxwell distribution for their magnitudes corresponding to twice of the desired temperature. After few time steps, the temperature of the system will relax to the desired temperature. In this experiment, the initial temperature of the sample is set to 10 K before shock loading. In this typical calculation, there are about 2.8 million atoms in the simulation box. Periodic boundary conditions are used in y and z -directions, where in x - direction corresponding to the shock direction, no such boundary condition is used. The time step used for MD is 1 fs. The MD calculation is performed in NVIDIA Tesla K20 GPU using CUDA as described in Chapter 5.5.

At time $t = 0$, the atoms in the sample are given an initial velocity $u_p = -2.5$ km/s in x - direction with the atoms in the rigid block at rest. A planar shock wave starts propagating through the copper crystal. To calculate stress field and any other statistical quantities, the sample block is divided into 400×50 two-dimensional bins.

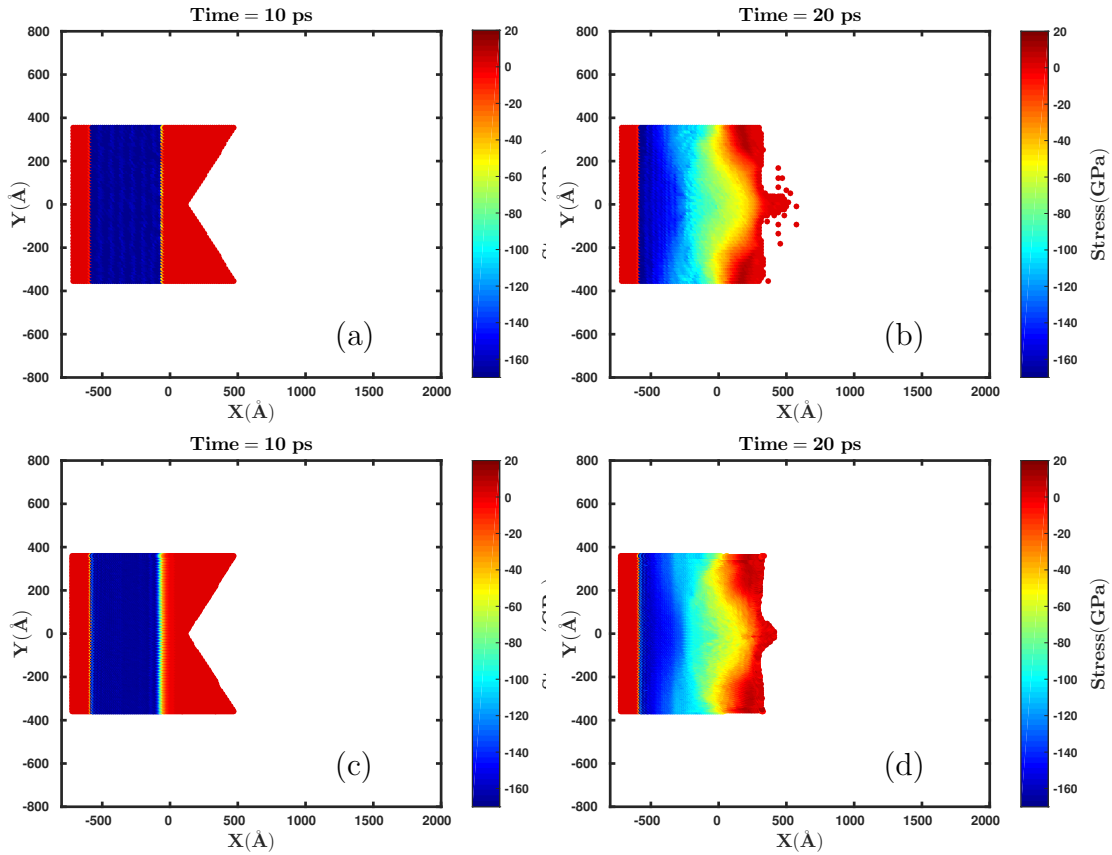


Figure 7.3 : The color plots of σ_{xx} component of stress field. Direct MD results are shown in (a) and (b) corresponding to $t = 10$ ps and $t = 20$ ps. Multi-scale DDMP-MD results are shown in (c) and (d) at $t = 10$ ps and $t = 20$ ps.

The stress in each bin is calculated using the pair interaction forces and the first line of (2.25). The upper two figures of figure 7.3 show the xx -component of stress (σ_{xx}) field at time $t = 10$ ps and $t = 20$ ps obtained by using the direct MD simulation. The maximum stress developed is 162 GPa. This value of stress agrees to previous similar work by Cherne *et al.* [56]. The shock velocity is calculated to be 5.2 km/s.

7.2 Multi-scale calculation for 2D problems

In Chapter 6 we successfully implemented the multi-scale method for one dimensional problems. The results are compared with full MD simulation. In this Chapter, we implement this multi-scale method for higher dimensional problems. For continuum level calculation, we use the DDMP method as described in Chapter 3.3. Molecular dynamics simulation is used in each material point to calculate stress without using a closure model. These material points being lagrangian, can be used to track the history of the material deformation.

In DDMP method, the internal force integral can be approximated as,

$$\mathbf{f}_i^{int} \approx - \sum_{p=1}^{N_p} V_p \boldsymbol{\sigma}_p \cdot \overline{\nabla S}_i(\mathbf{x}_p), \quad (7.1)$$

with N_p being the total number of material points, $\boldsymbol{\sigma}_p$ is the stress at material point p , and V_p is the volume of the material point. The gradient of shape function takes the form

$$\overline{\nabla S}_i(\mathbf{x}) = \alpha(\mathbf{x}) \nabla S_i(\mathbf{x}) + [1 - \alpha(\mathbf{x})] \sum_{j=1}^N \frac{S_j(\mathbf{x})}{V_j} \int S_j(\mathbf{y}) \nabla S_i(\mathbf{y}) dV_y, \quad (7.2)$$

with N being number of nodes in the computational domain and $\alpha(\mathbf{x})$ is a continuous function which vanishes on cell boundaries. In our calculations, we choose $\alpha(\mathbf{x})$ as [33]

$$\alpha(\mathbf{x}) = 0.5 \left\{ \prod_{k=1}^{n_c} [n_c S_k(\mathbf{x})] \right\}^{\frac{3}{2(n_c-1)d}}, \quad (7.3)$$

where n_c is the number of nodes in the cell, and d is the dimension of the problem.

As discussed in Chapter 3.3, the DDMP method modifies the gradient of shape function (figure 3.2) to make it continuous across a cell boundary to eliminate the cell crossing noise. The modified shape function gradient extends across 4 cells (in 1D) where as the shape function extends only across 2 cells. This mismatch between the support of $\overline{\nabla S_i}$ and the support of S_i can cause instability when explicit calculation are performed. To address this issue, we need to distribute the nodal force \mathbf{f}_l^{int} corresponding to node l away from zero mass nodes as well as very small mass nodes [33]. Let us define the distribution coefficient C_{li}^0 as,

$$C_{li}^0 = \begin{cases} \delta_{li}, & \text{if } m_l > 0, \\ \frac{\sqrt{m_i}}{\sum_{j'l} \sqrt{m_j}}, & \text{if } m_l = 0 \text{ and nodes } i \text{ and } l \text{ share a cell,} \\ 0, & \text{otherwise,} \end{cases} \quad (7.4)$$

where the summation $\sum_{j'l}$ is over all the nodes (j' 's) sharing a cell with node l and the node l itself. Let us define another distribution coefficient C_{li}^1 as,

$$C_{li}^1 = \begin{cases} \delta_{li}, & \text{if } m_l > m_\epsilon, \\ \frac{\sqrt{m_i}}{\sum_{j'l} \sqrt{m_j}}, & \text{if } m_l \leq m_\epsilon \text{ and nodes } i \text{ and } l \text{ share a cell,} \\ 0, & \text{otherwise,} \end{cases} \quad (7.5)$$

where m_ϵ is the lower bound of the nodal mass below which the distribution of mass algorithm is activated. For the calculation in this thesis, we take m_ϵ to be 4% of the mass of a material point. Finally, the nodal force corresponding to node i is calculated

as,

$$\mathbf{f}_i^{int} = \sum_{j=1}^N C_{li}^1 \sum_{l=1}^N C_{li}^0 \mathbf{f}_l^{int} \quad (7.6)$$

Similarly, the nodal velocity corresponding to node i is also calculated as,

$$\mathbf{v}_i = \sum_{j=1}^N C_{li}^1 \sum_{l=1}^N C_{li}^0 \mathbf{v}_l \quad (7.7)$$

An artificial viscosity is used to suppress the Gibbs phenomenon near the compression shock wave front. An additional stress in the form of artificial viscosity is added to the total stress of the system. The amount of artificial viscosity used in this experiment is [57]

$$q_{ij} = \sum_{k=1}^d \rho l Q_{ij_k} (\tau_0 c_L a - c_q l \lambda_k), \quad (7.8)$$

$$\mathbf{Q}_k = \lambda_k \mathbf{g}_k \mathbf{g}_k^T, \quad (7.9)$$

with \mathbf{g}_k being the eigenvectors of the strain rate tensor, d is the dimension of the problem, λ_k are the negative eigenvalues of the strain rate tensor, ρ is the density of the material point, l is the characteristic length of the material point and τ_0 is defined as,

$$\tau_0 = \frac{|\nabla \cdot \mathbf{v}|}{\|\nabla \mathbf{v}\|_2} \quad (7.10)$$

In our calculations, we choose the coefficients $c_L = 1.5$ and $c_q = 2.0$. We use l as an average of x and y -lengths of the material point.

Each material point is associated with a MD system for stress calculation purpose. Regardless of the dimensionality of continuum level problem, the MD calculation is performed in a three-dimensional periodic domain to minimize the boundary effects

[11, 10, 12]. First, the MD system is initialized by placing copper atoms in cubic periodic domains $[-L_{px}/2, L_{px}/2] \times [-L_{py}/2, L_{py}/2] \times [-L_{pz}/2, L_{pz}/2]$ with $L_{px} = L_{py} = L_{pz}$, according to face center cubic (FCC) array. The periodic length of an MD box is multiple of the lattice constant ($a = 3.615\text{\AA}$). The molecular dynamics simulation is performed using the same EAM potential as described in Chapter 7.1. The MD calculations are performed in GPU using CUDA.

The communication between continuum level calculation and atomistic calculation is through the stress and strain rate tensor. The periodic domain lengths of MD system in x and y - directions are updated according to equation (5.7). The periodic domain size in z -direction is kept constant. To make sure all atoms are inside the periodic domain, periodic boundary condition is used as described in Chapter 5.2. To deal with shear strain, a modified gradient of velocity is used as described in Chapter 5.2.

Different from the one dimensional shock wave example in previous Chapter, the strain rate developed in this example of jet formation can go very high (as high as order of $10^{11}s^{-1}$) across the shock. For such problems, we need to take into account of how much energy of a MD box is changing compared to continuum level change in energy. To ensure consistency in change in total energy, the continuum level change in energy in a continuum level time step is calculated as,

$$dE = \boldsymbol{\sigma} : \dot{\boldsymbol{\epsilon}} dV, \quad (7.11)$$

where $\boldsymbol{\sigma}$ is the stress tensor, $\dot{\boldsymbol{\epsilon}}$ is the strain rate tensor, and dV is the change in volume

in a time step. The difference between change in energy according to equation (7.11) and the change in energy in the MD system is added to the MD system in the form of kinetic energy (or temperature).

7.3 Numerical results and discussions

In this section we explore the numerical properties of our multi-scale method for the case of jet formation of copper around a notch under a strong impact. To prove the feasibility of this method, we compare the results from the multi-scale method with the results from direct MD simulation as described in the previous section 7.1. The computational domain is the same as shown in figure 7.1. A block of copper single crystal with length 144 nm and width 72 nm is considered. The length of the copper also consists of a piston of length 14.4 nm. A triangular shaped notch is created on the other end of the block with depth of 36 nm. At $t = 0$, the block of copper is slammed into the piston (at rest) with an impact velocity 2.5 km/s. A shock wave starts propagating through the copper crystal away from the piston.

The lower two figures in Figure 7.3 show the results obtained by using DDMP-MD multi-scale calculation at time $t = 10$ ps and $t = 20$ ps. In this calculation, we use 100×50 cells with 4 particles per cell to cover the sample domain. The triangular notch is formed by eliminating some material points, and there are 17,524 material points in total. Each MD box associated with each material point consists of 500 atoms. The dimensions of the MD box are approximately 18 Å corresponding to 5

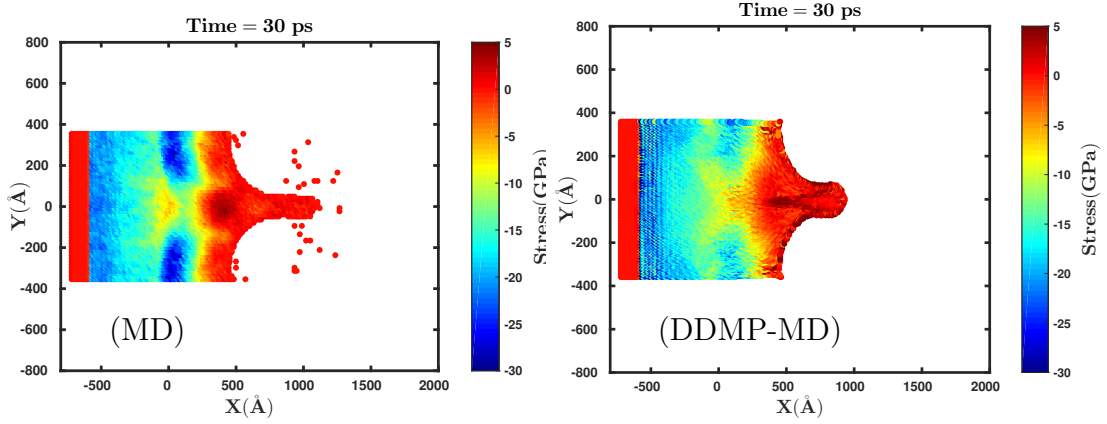


Figure 7.4 : Jet formation in copper obtained by using direct MD simulation and the multi-scale calculation

lattice cells with lattice constant 3.615 \AA . To mimic the periodic boundary condition of the direct MD calculation along y -direction, in this multi-scale calculation, we also use boundary condition along y -direction in the DDMP calculation such that the y -component of nodal velocities corresponding to boundary cells in y -direction are always kept zero. The stress (σ_{xx}) field in the multi-scale DDMP-MD results (Figure 7.3 (c)) for $t = 10 \text{ ps}$ is very similar to the results obtained by using direct MD simulation (Figure 7.3 (a)). At $t = 20 \text{ ps}$, we can see the similar jet formation in multi-scale calculation, too. The jet is not as sharp as in MD calculation because of the lower resolution of multi-scale calculation.

Figure 7.4 shows the jet shooting outward at time $t = 30 \text{ ps}$. The result obtained from combined DDMP-MD calculation shows a clear jet formation. The multi-scale method developed in this thesis work is able to capture the non-equilibrium phenomena across the shock and produces a reasonable result compared to the direct MD

(A)		(B)	
Particles/cell	Error (%)	Number of cells	Error (%)
1	24.7	50×25	10.1
4	10.1	100×50	6.0
16	9.8	160×80	5.6

Table 7.1 : Percentage errors of σ_{xx} calculated at $t = 20$ ps in comparison to direct MD simulation (A) For different particles per cell with 1250 cells, (B) For different number of cells with 4 particle per cell.

simulation result.

To study the effect of number of particles per cell and the grid resolution in the multi-scale calculation, we calculate the percentage error in the calculation of σ_{xx} in comparison with direct MD simulation results. To calculate the error in σ_{xx} calculation, for each material point, we find the spatially corresponding bin as described in Chapter 7.1 in the direct MD simulation and compare the results. The percentage error is calculated as

$$\epsilon = \sqrt{\frac{\sum(\sigma_{MD} - \sigma_{MS})^2}{\sum\sigma_{MD}^2}}, \quad (7.12)$$

where σ_{MD} is the stress calculated using direct MD simulation and σ_{MS} is the stress calculated using multi-scale method. Table 7.1 shows the results for varying number of material points per cell and varying grid resolutions. As expected, the error is smaller if we use more material points per cell keeping the number of cells fixed. Also, the results converge with respect to direct MD simulation if we refine the mesh

size. The figures 7.3 and 7.4 are obtained by using 4 material points per cell with 100×50 cells, and the calculated error is 6.0%.

7.4 Chapter Summary

Using the similar strategy as in one dimensional shock propagation in the previous Chapter, our multi-scale method is applied for the case involving much extreme material deformation. The capability of DDMP method to handle extreme material deformation has made it possible to apply this multi-scale method for such problem. This multi-scale method has been shown to be able to capture the thermodynamically non-equilibrium phenomena. After the shock passes, the material behind the shock is melted. Since we do not use any constitutive model to calculate stress, we do not even need to worry about whether the material is elastic, plastic or melt as long as the potential used for MD simulation can capture these change of phases.

Chapter 8

Conclusions

8.1 Summary

This dissertation attempts to build a multi-scale numerical method based on DDMP method which can be applied for problems involving extreme material deformation with high strain rates. For such problems, the time scale of material deformation is shorter than the material takes to reach a thermodynamic equilibrium state. Instead of using conventional constitutive relations or equations of state (which are often unavailable for such problems) to calculate closure quantities, such as stress tensor, we perform MD simulation to calculate those quantities. The continuum level calculation is performed using DDMP method to consider extreme deformation of the material. We choose DDMP method because of its capability of tracking material deformation history for cases involving extreme material deformations. History dependency becomes important for such systems in thermodynamically non-equilibrium state. The proposed multi-scale method is similar to type B heterogeneous multi-scale method (HMM) described in [7].

In Chapter 2, we derive macroscopic momentum equations starting from Liouville equation. Such derived momentum equation (2.12) can also be used for systems in

thermodynamically non-equilibrium state as long as we can calculate the stress tensor ($\boldsymbol{\sigma}$). The stress tensor ($\boldsymbol{\sigma}$) is directly related to the microscopic interactions among molecules and atoms as in equation (2.19). We also show that the stress defined in equation (2.19) is equivalent to the virial expression of cauchy stress in the limit of $l_a \ll l_m$, where l_a is the atomic length scale and l_m is the macroscopic length scale.

In Chapter 3, we introduce MPM method. Similar to FEM, MPM also seeks weak solution to PDEs. The MPM uses material points as integration points. These material points can move with the material and thus carry the history of the material deformation. The material points provide Lagrangian description of the material and fixed Eulerian grids are used to solve the momentum equations. Thus MPM eliminates diffusion problems associated with pure Eulerian methods and mesh distortion issues associated with pure Lagrangian methods. This makes MPM suitable for large deformation problems. The original MPM method suffers from so called cell crossing noise because of the discontinuity in gradient of shape function across cell boundaries. Currently, there are three different versions of MPM methods available to eliminate the cell crossing noise, GIMP, CPDI and DDMP. Chapter 3 gives a brief description of GIMP and CPDI methods as well as the detailed description of DDMP method. The DDMP method reduces the numerical noise by introducing gradient of shape function that is continuous across cell boundaries without violating the conservation properties of original MPM method.

In Chapter 4, we investigate the properties of original MPM, GIMP, CPDI and

DDMP methods for a one-dimensional shock problem in an ideal gas. We show that the original MPM method can not be used for weak isothermal shocks. GIMP and CPDI methods can provide reasonable results with minimal particles per cell but they fail to converge as number of particles per cell increases. The DDMP method converges as the number of particles increases but it requires large number of particles to achieve accurate results. This makes DDMP method very expensive especially for the proposed multi-scale method in this dissertation where the stress in each material point is calculated using MD simulation. The cause for producing the unsatisfactory DDMP results is identified. We introduce the sub-point method in which sub-points are generated around the material points and are used only as integration points. We show that we can achieve accurate results by using many sub-points without having to use many material points for the case of weak as well as strong shock problems. The computational cost can be greatly reduced using the sub-point scheme.

In our multi-scale method, we use MD simulation to calculate stress in each material point. Chapter 5 describes the basics of MD simulation. Leap frog algorithm is used as an integration scheme to solve the equations of motion of atoms (equation 5.1). EAM method potential is used to calculate force on each atoms. We use periodic boundary condition to eliminate any surface effects. For problems involving shear deformation, we modify the periodic boundary condition keeping the shape of the MD box rectangular. To ensure conservation of number of atoms inside a MD box, we transform the gradient of velocity tensor to upper rectangular form. The transforma-

tion is equivalent to using a rotational frame of reference. It is commonly assumed that this does not affect stress calculation. Further verification of this assumption is necessary and left for future work.

In Chapter 6, we use a one-dimensional strong shock propagation in cerium single crystal to demonstrate the feasibility of the proposed multi-scale method. The continuum level calculation is performed using DDMP method with sub-points, and MD simulation is performed in each material point to calculate stress. Since the material points do not need to communicate among each other, the MD simulations are performed independently and parallel on a GPU. The results obtained from the multi-scale DDMP-MD calculation are compared with direct MD calculation as well as pure DDMP method. For strong shock problems, the combined DDMP-MD results are in good agreement with direct MD simulation results where the pure DDMP method deviates from MD results. This proves that the combined DDMP-MD calculation can capture the thermodynamically non-equilibrium phenomena not captured by the elastic model used in pure DDMP method.

In Chapter 7, we apply the multi-scale method for a two-dimensional problem of jet formation around a notch of a copper single crystal. A block of copper single crystal is slammed into a rigid block with an initial velocity of 2.5 km/s. A jet formation occurs after shock reaches the other end of the block around a notch. In figure 7.3, we show that the combined DDMP-MD simulation results are very similar to the direct MD simulation results.

8.2 Future Work

Although, the experiments considered in this dissertation are limited to perfect crystal systems, the main purpose of this dissertation is to show that our multi-scale method can capture the thermodynamically non-equilibrium effects. To simulate more complex systems, such as, crystal defects and dislocations, a large scale MD simulation for each material point is needed. The larger MD system per material point requires a larger capability of computing machine. Since the large scale MD simulation corresponding to each material point can be performed independently and in parallel, this multi-scale method can be implemented in modern high performance computing systems in almost embarrassingly parallel fashion with good efficiency. Even though we believe its extension to complex problems in multi-dimension and other loading conditions is rather straightforward because the DDMP method has been implemented to three-dimensional calculations with complex deformation, the efficiency of the combined multi-scale calculation still needs to be explored, and the consistency conditions for the communication schemes between MD scale and continuum scale calculations have yet to be studied.

Bibliography

- [1] M. F. Horstemeyer, *Multiscale Modeling: A Review*, pp. 87–135. Dordrecht: Springer Netherlands, 2010.
- [2] X. Li and W. E, “Multiscale modeling of the dynamics of solids at finite temperature,” *Journal of the Mechanics and Physics of Solids*, vol. 53, no. 7, pp. 1650 – 1685, 2005.
- [3] E. B. Tadmor, M. Ortiz, and R. Phillips, “Quasicontinuum analysis of defects in solids,” *Philosophical Magazine A*, vol. 73, no. 6, pp. 1529–1563, 1996.
- [4] R. E. Miller and E. Tadmor, “The quasicontinuum method: Overview, applications and current directions,” *Journal of Computer-Aided Materials Design*, vol. 9, no. 3, pp. 203–239, 2002.
- [5] R. E. Rudd and J. Q. Broughton, “Coarse-grained molecular dynamics and the atomic limit of finite elements,” *Phys. Rev. B*, vol. 58, pp. R5893–R5896, Sep 1998.
- [6] F. F. Abraham, J. Q. Broughton, N. Bernstein, and E. Kaxiras, “Spanning the continuum to quantum length scales in a dynamic simulation of brittle fracture,” *EPL (Europhysics Letters)*, vol. 44, no. 6, p. 783, 1998.

- [7] W. E and B. Engquist, “The heterogeneous multi-scale methods,” *Comm. Math. Sci.*, vol. 1(1), pp. 87 – 132, 2003.
- [8] J. H. Irving and I. G. Kirkwood, “The statistical theory of transport process, iv. the equations of hydrodynamics,” *J. Chem. Phys.*, vol. 18, pp. 817–829, 1950.
- [9] D. Z. Zhang, X. Ma, and R. Rauenzahn, “Interspecies stress in momentum equations for dense binary particulate systems,” *Phys. Rev. Lett.*, vol. 97, no. 4, p. 048301, 2006.
- [10] X. Li and W. E, “Multiscale modeling of the dynamics of solids at finite temperature,” *Journal of the Mechanics and Physics of Solids*, vol. 229, pp. 1650 – 1685, 2005.
- [11] W. Ren and W. E, “Heterogeneous multiscale method for the modeling of complex fluids and micro-fluidics,” *Journal of Computational Physics*, vol. 240, pp. 1–26, 2005.
- [12] A. Abdulle, W. E, B.Engquist, and E. Vanden-Eijnden, “The heterogeneous multiscale method,” *Acta Numerica*, pp. 1–87, 2012.
- [13] M. Jemison, M. Sussman, and M. Shashkov, “A coupled level set and volume-of-fluid method for computing 3d and axisymmetric incompressible two-phase flows,” *Journal of Computational Physics*, vol. 162, pp. 301 – 337, 2000.
- [14] M. Jemison, M. Sussman, and M. Shashkov, “Filament capturing with the multi-

- material moment-of-fluid method,” *Journal of Computational Physics*, vol. 285, pp. 149 – 172, 2015.
- [15] F. H. Harlow, “The particle-in-cell computing method for fluid dynamics,” *Methods Comput. Phys*, vol. 3, p. 319, 1964.
- [16] D. Sulsky, Z. Chen, and H. L. Schreyer, “A particle method for history-dependent materials,” *Computational Methods in Applied Mechanics and Engineering*, vol. 118, pp. 179–196, 1994.
- [17] Z. Chen, Y. Han, S. Jiang, Y. Gan, and T. D. Sewell, “A multiscale material point method for impact simulation,” *Theoretical and Applied Mechanics Letters*, vol. 2, pp. 051003: 1–4, 2012.
- [18] Y. Lian, X. Zhang, F. Zhang, and X. Cui, “Tied interface grid material point method for problems with localized extreme deformation,” *International Journal of Impact Engineering*, vol. 70, pp. 50–61, 2014.
- [19] R. Tian, “Simulation at extreme-scale: Co-design thinking and practices,” *Arch Computat Methods Eng*, vol. 21, pp. 39–58, 2014.
- [20] D. Sulsky, H. Schreyer, K. Peterson, R. Kwok, and M. Coon, “Using the material-point method to model sea ice dynamics,” *J. Geophysical Res.*, vol. 112, p. C02S90, 2007.
- [21] F. Hamad, D. Stolle, and P. Vermeer, “Modelling of membranes in the material

- point method with applications,” *Int. J. Numer. Anal. Meth. Geomech.*, vol. 39, pp. 833–853, 2015.
- [22] P. C. Wallstedt and J. E. Guilkey, “A weighted least squares particle-in-cell method for solid mechanics,” *Int. J. Numer. Meth. Engng*, vol. 85, pp. 1687–1704, 2011.
- [23] S. G. Bardenhagen and E. Kober, “The generalized interpolation material point method,” *Computer modeling in engineering and science*, vol. 5(6), pp. 477–495, 2004.
- [24] A. Sadeghirad, R. Brannon, and J. Burghardt, “A convected particle domain interpolation technique to extend applicability of the material point method for problems involving massive deformations,” *Int. J. Numer. Meth. Engng.*, vol. 286, pp. 1435–1456, 2011.
- [25] D. Z. Zhang, C. Liu, and F. H. Harlow, “Effects of nonuniform segment deformation on the constitutive relation of polymeric solids,” *Physical Review E*, vol. 66, pp. 051806 (1–15), 2002.
- [26] D. Z. Zhang and R. M. Rauenzahn, “A viscoelastic model for dense granular flows,” *J. Rheol.*, vol. 41, no. 6, pp. 1275–1298, 1997.
- [27] N. C. Admal and E. B. Tadmor, “A unified interpretation of stress in molecular systems,” *Journal of Elasticity*, vol. 100, pp. 63 – 143, 2010.

- [28] N. C. Admal and E. B. Tadmor, “The non-uniqueness of the atomistic stress tensor and its relationship to the generalized beltrami representation,” *Journal of the mechanics and physics of solids*, vol. 93, pp. 72 – 92, 2016.
- [29] L. E. Malvern, *Introduction to the mechanics of a continuous medium*. Prentice-Hall, Upper Saddle River, 1969.
- [30] D. Burgess, D. Sulsky, and J. Brackbill, “Mass matrix formulation of the flip particle-in-cell method,” *J. Comput. Phys*, vol. 103, pp. 1–15, 1992.
- [31] P. C. Wallstedt and J. E. Guilkey, “An evaluation of explicit time integration schemes for use with the generalized interpolation material point method,” *J. Comput. Phys.*, vol. 227, pp. 9628–9642, Nov. 2008.
- [32] A. Sadeghirad, R. M. Brannon, and J. E. Guilkey, “Second-order convected particle domain interpolation (CPDI2) with enrichment for weak discontinuities at material interfaces,” *Int. J. Numer. Meth. Engng*, vol. 95, pp. 928–952, 2013.
- [33] D. Z. Zhang, X. Ma, and P. Giguere, “Material point method enhanced by modified gradient of shape function,” *J. Comp. Phys.*, p. doi:10.1016/j.jcp.2011.04.032, 2011.
- [34] X. Ma, P. T. Giguere, B. Jayaraman, and D. Z. Zhang, “Distribution coefficient algorithm for small mass nodes in material point method,” *J. Comput. Phys.*, vol. 229, pp. 7819–7833, 2010.

- [35] B. Alder and T. Wainwright, “Phase transition for a hard sphere system,” *J. Chem. Phys.*, vol. 27, p. 1208, 1957.
- [36] A. Rahman, “Correlations in the motion of atoms in liquid argon,” *Physical Review*, vol. 136, pp. A405–A411, 1964.
- [37] P. Young, “The leapfrog method and other symplectic algorithms for integrating newtons laws of motion,” 2014.
- [38] M. S. Daw and M. I. Baskes, “Embedded-atom method: Derivation and application to impurities, surfaces, and other defects in metals,” *Phys. Rev. B*, vol. 29, pp. 6443–6453, 1984.
- [39] S. M. Foiles, M. I. Baskes, and M. S. Daw, “Embedded-atom-method functions for the fcc metals cu, ag, au, ni, pd, pt, and their alloys,” *Phys. Rev. B*, vol. 33, pp. 7983–7991, 1986.
- [40] H. W. Sheng, M. J. Kramer, A. Cadien, T. Fujita, and M. W. Chen, “Highly optimized embedded-atom-method potentials for fourteen fcc metals,” *Phys. Rev. B*, vol. 83, p. 134118, 2011.
- [41] A. Cadien, Q. Y. Hu, Y. Meng, Y. Q. Cheng, M. W. Chen, J. F. Shu, H. K. Mao, and H. W. Sheng, “First-order liquid-liquid phase transition in cerium,” *Phys. Rev. Lett*, vol. 110, p. 125503, 2013.
- [42] Y. Mishin, M. Mehl, D. Papaconstantopoulos, A. Voter, and J. Kress, “Structural

- stability and lattice defects in copper: Ab initio, tight-binding, and embedded-atom calculations,” *Phys. Rev. B*, vol. 63, p. 224106, 2001.
- [43] Y. Mishin, *Handbook of Materials Modeling*, ed. S. Yip. Springer, 2005.
- [44] X. Wu and X. Li, “On consistent definitions of momentum and energy fluxes for molecular dynamics models with multi-body interatomic potentials,” *Modelling and Simulation in Materials Science and Engineering*, vol. 23, p. 015003, 2015.
- [45] J. Fu and J. Zhao, “Gupta potential for rare earth elements of the fcc phase: lanthanum and cerium,” *Modelling Simul. Mater. Sci. Eng.*, vol. 21, p. 065003, 2013.
- [46] J. Broughton, P. Bristowe, and J. Newsam, “Parallel molecular dynamics with the embedded atom method,” *MRS Preceedings*, vol. 291, pp. 37 – 42, 1993.
- [47] S. Green, *Particle Simulation using CUDA*, 2010.
- [48] T. R. Dhakal and D. Z. Zhang, “Material point methods applied to one-dimensional shock waves and dual domain material point method with sub-points,” *J. Comp. Phys.*, vol. 325, pp. 301– 313, 2016.
- [49] R. Luzzi and A. Vasconcellos, “On the nonequilibrium statistical operator method,” *Fortsclir. Phya.*, vol. 38, pp. 887– 922, 1990.
- [50] J. J. Monaghan, “SPH and riemann solvers,” *Journal of Computational Physics*, vol. 136, pp. 298 – 307, 1997.

- [51] L. D. G. Sigalotti, H. López, A. Donoso, E. Sira, and J. Klapp, “A shock-capturing sph scheme based on adaptive kernel estimation,” *Journal of Computational Physics*, vol. 212, pp. 124 – 149, 2005.
- [52] E. J. Caramana, M. J. Shashkov, and P. P. Whalen, “Formulations of artificial viscosity for multi-dimensional shock wave computations,” *Journal of Computational Physics*, vol. 144, pp. 70 – 97, 1998.
- [53] Y. Lee and C. Basaran, “A multiscale modeling technique for bridging molecular dynamics with finite element method,” *Journal of Computational Physics*, vol. 253, pp. 68 – 85, 2013.
- [54] E. Saether, V. Yamakov, and E. H. G. 1, “An embedded statistical method for coupling molecular dynamics and finite element analyses,” *Int. J. Numer. Meth. Engng*, vol. 78, pp. 1292 – 1319, 2009.
- [55] K. Gu, C. B. Watkins, and J. Koplik, “Atomistic hybrid dsmc/nemd method for nonequilibrium multiscale simulations,” *Journal of Computational Physics*, vol. 229, pp. 1381 – 1400, 2010.
- [56] F. J. Cherne, J. E. Hammerberg, M. J. Andrews, V. Karkhanis, and P. Ramaprabhu, “On shock driven jetting of liquid from non-sinusoidal surfaces into a vacuum,” *Journal of Applied Physics*, vol. 118, no. 18, 2015.
- [57] C. C. Long, D. Z. Zhang, C. A. Bronkhorst, and G. T. Gray, “Representing

ductile damage with the dual domain material point method,” *Comput. Methods Appl. Mech. Engrg.*, vol. 300, pp. 611 – 627, 2016.