Coding considerations for standalone molecular dynamics simulations of atomistic structures

R O Ocaya and J J Terblans

Department of Physics, University of the Free State, South Africa.

E-mail: `ocayaro@ufs.ac.za`

Abstract. The laws of Newtonian mechanics allow ab-initio molecular dynamics to model and simulate particle trajectories in material science by defining a differentiable potential function. This paper discusses some considerations for the coding of ab-initio programs for simulation on a standalone computer and illustrates the approach by C language codes in the context of embedded metallic atoms in the face-centred cubic structure. The algorithms use velocity-time integration to determine particle parameter evolution for up to several thousands of particles in a thermodynamical ensemble. Such functions are reusable and can be placed in a redistributable header library file. While there are both commercial and free packages available, their heuristic nature prevents dissection. In addition, developing own codes has the obvious advantage of teaching techniques applicable to new problems.

1. Introduction

The last decades have seen rapid growth of atomistic/molecular simulation (AMS) as a powerful tool in material science. The method can postulate material properties that are difficult or costly to study experimentally [1, 2, 3, 4, 5]. In AMS, macroscopic material properties are aggregations of atomic interactions and dynamics. A wide span of properties, such as structural changes due to temperature and applied forces, are better understood through parametric variations effected in the method. Examples include energetics of cluster formation [6, 7, 8], diffusion [9, 10], crack formation and propagation [11], the impact of vacancy or adatom defects, the load-bearing capacity, and so on.

There are two approaches to MD [12, 13, 14]: classical MD and ab-initio MD [15]. In ab-initio MD, the starting point is an incremental and iterative numerical basis arising from a concisely bounded mathematical model i.e. the forces can be deduced by differentiation of a potential energy function. The temporal and spatial evolution of each particle of the ensemble then permits the determination of the trajectories. While in principle simple, the analytical complexity increases with the number of system particles, the efficacy of the mathematical model and the required level of computational accuracy. Complexity bears directly on overall computational time and storage. Much effort is being expended to discover better potential energy functions and more efficient computational algorithms for them. Ongoing developments in internet and computer technology make it ever easier to simulate these systems. The current trend is to discover new efficient, high-speed interconnects of several multiple-core central or graphical processing units into super-computing clusters that collaborate to solve a complex problem [5]. This approach defines a clear futuristic research niche that is relevant to all areas of research. The single, standalone machine, however, is still an essential part in most institutional supercomputer set ups. Its participation is therefore vital in the computational scheme since, as the starting point, algorithms are presumed to run on each node in some form or other. More complex systems are then expected to integrate several such nodes through generalization and scaling.

Modern computer architectures are by their nature scalable i.e. programs can readily be extended to bigger systems with minimal modification. Thus, larger particle ensembles can exploit distributed processing to realize computational time and storage cost savings. There is also the ease of porting codes to run on more heterogeneous nodes that suit specific users. Typical ensembles are defined in terms of the total number of individual particles in the system in addition to macroscopic material properties such as pressure, volume and temperature. Perturbations then ultimately give an idea of the system dynamics i.e. the trajectories of individual particles, their interactions through collisions, repulsions, attractions, adhesion, steady state and equilibrium distances, angles and similar geometric quantities can be computed and over time. Deductions of connected macroscopic parameters such as kinetic and potential energy, diffusion and other transport coefficients, structure factors, spectral density functions, distribution functions, then follow. Numerical integration algorithms rely on the computation of the system differential equations at well-defined, linearly incremented instants of time t. There are a variety of integration schemes, each depending on the specific variables of interest and what is known or needs to be known at each time step δt.

This paper implements all integrations using the Störmer-Verlet (SV) technique for pairwise particle interactions. At each iteration, the computational end point per particle is said to be determined when the force F_i acting on the i-th particle is calculated. It is understood that all pre-models are based on approximations. Therefore, the outputs are accepted to be accurate to a specified degree for a set number of iterations. These trade-offs are usually made to reduce execution speed. Although the potential of modeling and simulation for research is clear, it remains largely in the domain of organizations that have large computational facilities with direct vested interests in materials modeling. This does not imply that a standalone computer typically found today is not up to many of the tasks. The algorithms can be ported to run on distributed clusters by implementing freely available message passing interfaces (MPI).

2. Motivation and significance

While computational materials science has grown substantially, the bulk of research output still remains largely experimental, inevitably leaving the avenue of computation largely unexplored. There are a number of likely reasons for this. One could be a misconception of computation as solely the domain of computer science. Another could be that the real, general shortage of programming skills required to code the underlying functions places a high starting inertia on most researchers. Interestingly, most established undergraduate physical and applied science curricula have some aspects of computer programming. It is likely that many researchers today have either some programming exposure or access to a computer science department, thus permitting collaboration. The aim of this article is therefore twofold. Firstly, it presents a toolkit of functions and examples that were developed and tested over a period of several months to solve AMS equations of face-centred cubic (fcc) crystalline structures using embedded atom methods (EAM) on a standalone computer. The algorithms are coded using Microsoft Visual C/C++ language and are easily ported to GNU compilers such as `gcc`. The software has been used by the authors to calculate for instance fcc metal bulk and surface energetics. The potential to use the codes for scientific visualization and discovery therefore exists. Secondly, it provides repeatable information to peers who have some basic knowledge of C-programming and an interest in materials modeling and simulation. The aim is to facilitate their progression through first simulations without the detractions of deep coding. The nature of materials science today is such that the work will be ongoing as newer functions are implemented and the package updated. The article also outlines the mathematical basis of the algorithms for parallel reading. Finally, an example of the usage of the package is given.

3. Software description

The illustration package implements a three-dimensional (3D) Sutton-Chen (SC) form of the Finnis-Sinclair (FS) EAM potential [16, 17, 18, 19] in the canonical ensemble. EAM methods are less accurate than first-order principles that consider all particle interactions individually but are computationally cheaper and can handle larger time scales and more particles [20]. For a monoatomic N-particle ensemble, the energy EAM Hamiltonian is expressed as [18, 20, 21]

$$E_{tot} = \frac{1}{2}\sum_{ij}^{N} V(r_{ij}) + \sum_{i}^{N} F(\bar{\rho}_i), \tag{1}$$

where $V{=}V(r_{ij})$ is the pairwise interaction potential energy between particles i and j separated as distance r_{ij}, and $F{=}F(\bar{\rho}_i)$ is the embedding energy of atom i as a function of the host electron density.

According to SC, the FS total potential energy for fcc metals is written in the translation and rotation invariant form [11, 16, 23, 24]

$$V = \varepsilon \sum_{i=1}^{N}\left(\sum_{j=i+1}^{N}\left(\frac{\sigma}{r_{ij}}\right)^{n} - c\sqrt{S_i}\right), \tag{2}$$

where

$$S_i = \sum_{j=1,j\neq i}^{N}\left(\frac{\sigma}{r_{ij}}\right)^{m}, \tag{3}$$

and σ is a length parameter e.g. lattice constant of the fcc unit cell, ε is an energy scaling factor for the system. The constants c, m and n, where $m < n$, are fitting parameters. Force is gradient of potential, i.e.

$$\overline{F} = -\overline{\nabla}V(r). \tag{4}$$

Hence the force on the i-th particle can be expressed as

$$\overline{F}_i = \varepsilon \sum_{j=1,\ j\neq i}^{N}\left[n\left(\frac{\sigma}{r_{ij}}\right)^{n} - \frac{cm}{2}\left(\frac{1}{\sqrt{S_i}} + \frac{1}{\sqrt{S_j}}\right)\left(\frac{\sigma}{r_{ij}}\right)^{m}\right]\frac{\bar{r}_{ij}}{r_{ij}^{2}}, \tag{5}$$

where $\bar{r}_{ij}$ is the vector between particles i and j and is given in terms of the positions of each particle i.e.

$$\bar{r}_{ij} = \bar{x}_j - \bar{x}_i. \tag{6}$$

Discretization of the continuous domain into a finite set of points achieves numerical approximations that are useful to solve a variety of difficult differential equations. In MD such equations are derived from the Schrödinger equation under specific boundary conditions [12, 13, 14]. The continuous closed-form solutions may be extremely difficult to solve analytically so that numerical approximations may be the only recourse. An necessary requirement in MD discretization is energy conservation i.e. the Hamiltonian (H) must not change over discrete steps [25]. However, discretization leads to unavoidable approximation errors. Effort must then be spent to keep the variations in H within acceptable bounds. A commonly used approach to discretize MD systems is the so-called velocity Störmer-Verlet method. It is a discretization of

Newtonian equations of motion [11, 26, 27]. In the variant of the method that is used in this software, using the discretized notation $\bar{p}_i^n := \bar{p}_i(t_n)$ which describes the vector $\bar{p}$ sampled at the discrete n-th time interval, where $\bar{p}=\{\bar{x},\ \bar{v},\ \bar{F}\}$, it follows that the approximation of $\bar{p}_i^{n+1}$, i.e. on the next or $(n+1)$-st time interval or $(t+\delta t)$ gives the new positions

$$\overline{x}_i^{n+1} = \overline{x}^n + \delta t \overline{v}_i^{n+1} + \frac{\overline{F}_i^n}{2m_i}\delta t^2. \tag{7}$$

Similarly, the next velocities are

$$\overline{v}_i^{n+1} = \overline{v}_i^n + \left(\frac{\overline{F}_i^n + \overline{F}_i^{n+1}}{2m_i}\right)\delta t. \tag{8}$$

The discretization error in the method is $O(\delta t^2)$. The method requires starting values of positions and velocities for each particle. Therefore in the software the user-created ASCII text file e.g. `data.txt` contains initial particle data in the 3D format:

```
mass x_pos y_pos z_pos v_x v_y v_z
```

It is necessary to scale all variables to reference values and dimensionless equations. The user then interprets the output in that context [11].

The canonical ensemble is a statistical arrangements of particles and their states in a mechanical system that is in thermal equilibrium with a heat bath at fixed temperature [20]. For a system of N particles, volume V and temperature T, the system is said to be a NVT canonical ensemble. Specifying T facilitates the estimation of the system energies using the energy equipartition theorem (EPT) of thermodynamics. Conversely, a knowledge of the energies allows an estimate of the system temperature [11, 28]. Thus, the evolution of the macroscopic variables T, V and P that portray system phases may be followed by considering particle interactions. According to EPT, the kinetic energy E_k of a particle with three degrees of freedom is

$$E_k = \frac{3}{2}Nk_BT, \tag{9}$$

where k_B is Boltzmann constant.

3.1. Software Architecture

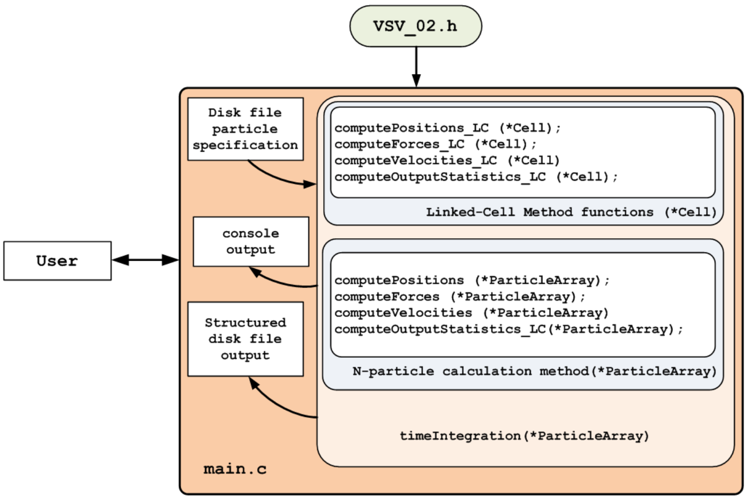
The package combines a compiled executable and C-style library file. The aforementioned MD functions are in the `VSV_02.h` header whose functions are called from the main program `main.c`. The particle structure is defined, initial particle boundary conditions are loaded from a user-edited ASCII disk data file containing unambiguous masses, 3D coordinates and initial velocity components of each particle. The 3D-space particle structure is

```
typedef struct{
double m;
double x[DIM];
double v[DIM];
double F[DIM];
double F_old[DIM];
} Particle;  // 72 bytes

// ParticleList structure
typedef struct ParticleList{
Particle p;
struct ParticleList *next;
} ParticleList;
```

The central coding theme employed is function reusability and a by-reference passable RAM-based particle array for speedier computations. Reusable codes are readily adaptable to new MD problems.

3.2. Software Functionalities
Fig. 1 shows the main functionalities of the software in pictorial form. Further information about the most current build of the program code and user manuals is available for download under the GNU General Public License v3.0 [22]. The code has evolved to include the calculation of ergodics such as vacancy energy, cohesion energy, deterministic thermostats and Monte-Carlo thermostats.


VSV_02.h
User
Disk file particle specification
console output
Structured disk file output
Linked-Cell Method functions (*Cell)
N-particle calculation method(*ParticleArray)
timeIntegration(*ParticleArray)
main.c

```
computePositions_LC (*Cell);
computeForces_LC (*Cell);
computeVelocities_LC (*Cell)
computeOutputStatistics_LC (*Cell);
```

```
computePositions (*ParticleArray);
computeForces (*ParticleArray);
computeVelocities (*ParticleArray)
computeOutputStatistics_LC(*ParticleArray);
```

Figure 1. Pictorial depiction of the functionalities of the program.

3.3. Main code snippets
All included functions (`func`) are declared in the form

```
return_type func(arrangement *particle, types other_variables)
```

where `arrangement` can be either particle, particlelist or cell array, the latter two being dedicated to the LCM method. The `other_variables` can be summation indices, etc. The 3D inter-particle distance r_{ij} in Eq.(6) for all MD processes is calculated using

```
Particle *p;
double r_ij;
for (d=0; d<3; d++)
  r_ij += sqr(p[j].x[d]-p[i].x[d]);
r_ij = pow(r,0.5);    // 3D distance
```

Stating the condition: $r_{ij}>r_{cut}$, where r_{cut} is a cut-off radius within this snippet will improve calculation speed by ignoring interactions beyond r_{cut}.

4. Illustrative Example

Doye & Wales [23] have investigated global minima in transition metal SC clusters described by various (m-n) pairs i.e. (12-6), (9-6) and (10-8) for clusters up to $N = 80$ atoms. Their published work provide sufficient data to illustrate the calculation of energy minima using the present software. For instance, specifying the data points in Table 1 for $N = 3$ for copper with normalized energy and lattice parameters $\varepsilon = 1$, $\sigma = 1$, $m = 6$, $n = 9$, $m_o = 63$ and $c = 39.432$

Table 1. Sample data for fcc copper obtained from [23].

m	x	y	z	v_x	v_y	v_z
63	-0.5264206369	0.1019469049	0.0252070114	0	0	0
63	-0.8377547075	-0.1828574677	-0.4214018482	0	0	0
63	-0.2345929501	-0.0832085752	-0.4827850722	0	0	0

Doye and Wales [23] developed a method to improve structural identifications of metallic atomic clusters based on experimental data. Their Monte Carlo based method deforms an arbitrarily constructed hypersurface of the atomic clusters for up to 80 atoms. By searching for a global potential energy minimum they are able to suggest the most stable structure of the collection of atoms. A difficulty of their method, hence the need for the globalized search, is that the global minimum of adjacent surfaces are not necessarily related and therefore a newer local surface is needed for the atoms on the two surfaces. Therefore, the difficulty with the method is a complexity of $\approx\mathcal{O}(N^2)$. In contrast, the method of the present paper is deterministic, has a complexity of $\approx\mathcal{O}(N \log N)$ and relies on the complete solution of the potential energy of the atomic cluster as is. Thus, we make no assumptions about the stability or structure of the cluster, and consequently we can apply it to a much larger number of atoms. Table 2 shows the output of global minima calculation for a stable copper atom cluster. The output energy is scaled by ε. While Table 2 only compares 3-atom clusters with those in Doye and Wales [23], we verify their outputs for all their 80 atoms with no noticeable increase in calculation time.

Table 2. Program output for Table 1 input data.

N	12-6	9-6	10-8
3	-1704.6905	-480.8560	-633.7771

Our present interest is to show how initial experiments in MD simulations for fcc metals, for instance, may be facilitated in a standalone setting. It has been written from personal experiences that confirm how first computer-based experiments can be difficult to initiate. By coding a dissectable software and presenting alongside it the mathematical basis for the algorithms the progression from simple to more advanced MD systems is easier. FCC metal properties were correctly calculated such as the bulk and surface formation energetics down to single atoms, forces on individual particles, diffusion coefficients, cluster formation mechanisms, that successfully reproduced published data in similar simulation programs.

5. Conclusions

This article emphasizes that the software program tool, like any tool, has to be coded and then applied correctly for any measure of effectiveness. The ease of interpreting the outputs of the program can lead to postulations and predictions about new configurations of the material. With increasing interest in the use of software to investigate materials, more researchers are increasingly having to code material behavior. In this work, the authors make no assumptions about the C-compilers, nor presume to recommend any particular programming environment as being more suitable for a given purpose. Rather, the emphasize is on underlying methods and pathways to successful coding. Having a software that can be updated easily, ported over many platforms and scalable are advantageous for wider applicability. The programs codes presented here have been tested successfully on Microsoft C/C++ compiler (Windows 7) and `gcc` in Ubuntu Linux with little modification for various fcc metal parameters. The outputs compare favorably with published literature sources and confirm the applicability of the software.

Acknowledgments

The authors hereby acknowledge the National Research Foundation (NRF) for funding support through KIC 2016, Grant No. 104078.

References

[1] Mendelev M I, Han S, Srolovitz D J, Ackland G J, Sun D Y and Asta M 2003 *Philos. Mag.* **83** 3977–3994 doi: `10.1080/14786430310001613264`.

[2] Sutton A P and Chen J 1990 *Philos. Mag. Lett.* **61** 139–156

[3] Das A and Ghosh M M 2015 *Comput. Mater. Sci.* **101** 88–95 doi: `10.1016/j.commatsci.2015.01.008`

[4] van der Walt C, Terblans J J and Swart H C 2014 *Comput. Mater. Sci.* **83** 70–77 doi: `10.1016/j.commatsci.2013.10.039`

[5] Abraham M J, Murtolad T, Schulz R, Pálla S, Smith J C, Hessa B and Lindahl E 2015 *SoftwareX* doi: `10.1016/j.softx.2015.06.001`

[6] Smirnov B M 1995 *Zh. Eksp. Teor. Fiz.* **107** 2080–2091

[7] Terblans J J 2002 *Surf. Interface Anal.* **33** 767–770 doi: `10.1002/sia.1451`

[8] Mattsson T R and Mattsson A E 2002 *Phys. Rev. B* **66** 214110

[9] Sebastian I S, Aldazabal J, Capdevila C and Garcia-Mateo C 2008 *Phys. Status Solidi A* **205** 6 1337–1342 doi: `10.1002/pssa.200778124`

[10] Jian-Min Z, Fei M and Ke-Wei X 2004 *Appl. Surf. Sci.* **13** 7 1009-1963/2004/13(07)/1082-09

[11] Griebel M, Knapek S and Zumbusch G 2007 *in Barth TJ et al. (Eds.), Numerical Simulation in Molecular Dynamics, in: Texts in Computational Science and Engineering 5* (Berlin: Berlin)

[12] Car R and Parrinello M 1985 *Phys. Rev. Lett* **55** 22 2471–2474

[13] Car R and Parrinello M 1989 *The Unified Approach for Molecular Dynamics and Density Functional Theory, in Simple Molecular Systems at Very High Density* **186** *of NATO ASI Series B, Loubeyre PP and Boccara N (eds.)* (New York: Plenum Press) 455–476

[14] Remler D K and Madden P A 1990 *Mol. Phys.* **70** 6 921–966

[15] Tuckerman M E 2002 *J. Phys.: Condens. Matter* **14** R1297–R1355 Online at `stacks.iop.org/JPhysCM/14/R1297`

[16] Finnis M W and Sinclair J E 1984 *Philos. Mag. A* **50** 45

[17] Daw M S, Foiles S M and Baskes M I 1993 *Mater. Sci. Rep.* **9** 251

[18] Daw M S and Baskes M I *Phys. Rev. B* **29** 6443

[19] Todd B D and Lynden-Bell R M 1993 *Surface Science* **281** 191–206

[20] Chamati H, Papanicolaou N I, Mishin Y and Papaconstantopoulos D A 2006 *Surface Science* 1–11 doi: `10.1016/j.susc.2006.02.10`

[21] Mishin Y 2005 *in: Yip S (Ed.), Handbook of Materials Modeling* (Springer: The Netherlands)

[22] Ocaya R O and Terblans J J 2016 *SoftwareX* doi: `10.1016/j.softx.2016.05.005`

[23] Doye J P K and Wales D J 1998 *New J. Chem.* 733–744

[24] Swope W, Andersen H, Berens P and Wilson K 1982 *J. Chem. Phys.* **76** 637–649

[25] Landau L and Lifschitz E 1976 *Mechanics, Course of Theoretical Physics, Vol. 1* (Pergamon Press: Oxford)

[26] Hockney R 1970 *Methods Comp. Phys.* **9** 136–211

[27] Verlet L 1967 *Phys. Rev.* **159** 98–103

[28] Chandler D 1987 *Introduction to modern statistical mechanics* (Oxford University Press: New York)