Computational Materials Science 156 (2019) 135–141

Contents lists available at ScienceDirect

Computational Materials Science

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

Influence of integration formulations on the performance of the fast inertial relaxation engine (FIRE) method

Fei Shuang[a,b], Pan Xiao[a,*], Ronghao Shi[a,b], Fujiu Ke[c], Yilong Bai[a]

[a] LNM, Institute of Mechanics, Chinese Academy of Sciences, Beijing 100190, China
[b] School of Engineering Sciences, University of Chinese Academy of Sciences, Beijing 100049, China
[c] School of Physics and Nuclear Energy Engineering, Beihang University, Beijing 100191, China

ARTICLE INFO

Keywords:
Energy minimization
Fast inertial relaxation engine method
Integration formulations
Molecular simulations

ABSTRACT

As a simple and robust minimization algorithm with high efficiency, the fast inertial relaxation engine (FIRE) method has been widely applied in various fields. Different from the viewpoint of previous reports, the present study found out that integration formulations in the FIRE method have huge influence on its convergence performance and capability of critical analysis. Specifically, the Forward Euler (FE) integration is found ill-suited for all the trial applications due to the outdated velocity update; the Velocity Verlet (VV) integration shows robust convergence and superior efficiency, but loses the ability of critical analysis; the Semi-implicit Euler (SE) integration endows the FIRE method with the critical analysis ability as well as good efficiency, but its convergence is conditional. It is also found that the FIRE method using a modified energy monitor shows more robust convergence than using the original power monitor. Further investigation indicates that the SE integration combined with the energy monitor should be the first choice for the FIRE method in general molecular statics simulations. These findings extend the capability of FIRE and provide practical suggestions for selecting minimization algorithms in molecular simulations.

1. Introduction

Energy minimization (EM) of atomistic systems is one of the most common tasks in computational material sciences [1], solid-state physics [2], chemistry and biology [3]. EM is also the core algorithm in many simulation methods, such as molecular statics (MS) [4], coarse-grained methods [5] and recently developed multi-scale methods [6]. Improving the efficiency of EM, therefore, is of great importance to reduce the computational cost in simulations. A variety of well-established mathematical optimization methods, like the steepest decent (SD), conjugate gradient (CG) and quasi-Newton method have been widely used [7,8]. When combined with multigrid method, EM shows the superior efficiency in simulations of elastic deformation [9] and dislocation relaxation [10,11]. Some other EM methods are variants of molecular dynamics (MD) and have been applied in MS simulations, such as QuickMin (QM) [12] and Fast Inertial Relaxation Engine (FIRE) [13]. Among the MD-like methods, it is reported that FIRE is significantly faster than standard implementations of CG method and often competitive with the more sophisticated quasi-Newton schemes typically in ab initio calculations. The robustness and versatility in critical point analysis also make FIRE more intriguing than other EM methods.

Due to its high efficiency, simplicity and low computational consumption, FIRE has been implemented as an effective minimizer into various simulation packages, such as LAMMPS [14], HOOMD-blue [15,16], BigDFT [17] and EON [18], which play an important role in the frontier research of material sciences. Rogan et al. adopted FIRE as an efficient local minimizer to search for the global and local minimal energy states of freestanding nanoclusters [19]. Fayon et al. utilized FIRE to optimize geometry in HOOD-blue to study the formation mechanism of ultra-porous framework materials [20]. Hwang et al. used FIRE to study bubble super-diffusion and soft glassy materials [21]. As emphasized by Bitzek [13], integration formulation has little influence on the FIRE method. However, our preliminary results indicate that FIRE with the default integration formulation is much less efficient than CG in LAMMPS package, which is inconsistent with the conclusion in aforementioned work. Recently, a new structure relaxation algorithm based on micro-canonical ensemble (NVERE) proposed by Yang et al. is reported faster than FIRE in relaxing soft structures like graphene sheet [22], which implies that FIRE is still worth further exploring in different applications. Here in this work, we are focusing on the influence of integration formulation on the performance of FIRE so as to extend its capability and provide practical suggestions on general applications.

* Corresponding author.
E-mail address: xiaopan@lnm.imech.ac.cn (P. Xiao).

https://doi.org/10.1016/j.commatsci.2018.09.049
Received 7 August 2018; Received in revised form 21 September 2018; Accepted 22 September 2018

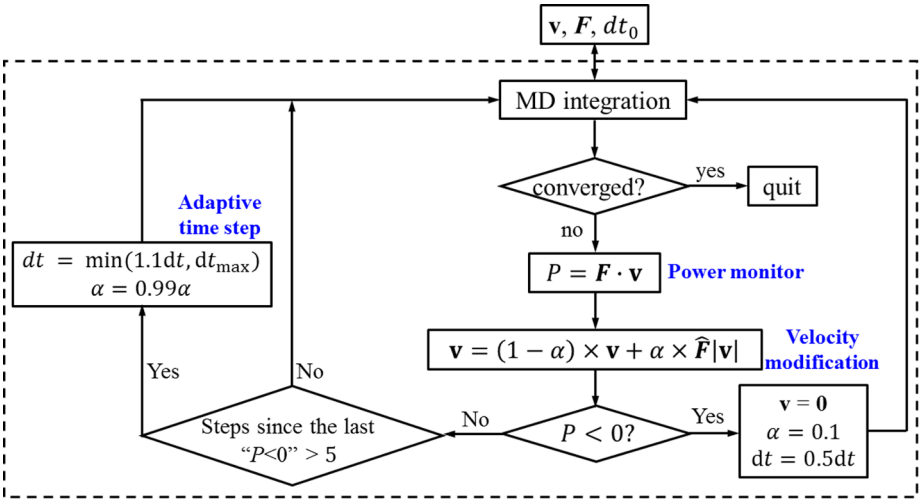

Fig. 1. Computational flow chart of the FIRE method in energy minimization of atomic systems.

Table 1
Details of integration formulations used in the work.

Formulations	Position update	Velocity update	Storage
Forward Euler (FE)	$x_{n+1} = x_n + v_n\,\Delta t$	$v_{n+1} = v_n + a_n\,\Delta t$	3*N*
Semi-implicit Euler (SE)	$x_{n+1} = x_n + v_n\,\Delta t$	$v_{n+1} = v_n + a_{n+1}\,\Delta t$	3*N*
Velocity Verlet (VV)	$x_{n+1} = x_n + v_n\,\Delta t + \frac{1}{2}a_n\Delta t^2$	$v_{n+1} = v_n + \frac{1}{2}(a_{n+1} + a_n)\,\Delta t$	4*N*

2. FIRE method and integration algorithms

Fig. 1 shows the flow chart of FIRE when minimizing the energy of a given system. Like QM, FIRE removes kinetic energy from the given system conditionally to find a local minimal energy state. However, its superiority over QM lies in two key aspects: velocity modification with more inertia effect and adaptive time step by monitoring power state P of the system. In an atomic system, let $\mathbf{x}$ be position vector, $E(\mathbf{x})$ be total potential energy, then $F = -\nabla E(\mathbf{x})$ is atomic force vector; $\mathbf{v} = \dot{\mathbf{x}}$ is velocity vector. The power P is defined as $P = F{\cdot}v$ and used to monitor whether the system goes toward the lower energy state or not. If P becomes negative, the system will be frozen by setting $\mathbf{v} = 0$ to avoid further uphill movement; and time step will be reduced by half in the next iteration. While if P is positive, the system will go further with increasing time step by 10% so as to promote the convergence rate. The contribution of force in velocity modification, α, will also be adjusted automatically during iterations.

The MD integration in Fig. 1 can be performed with different formulations. Three commonly used expressions are Forward Euler (FE), Semi-implicit Euler (SE) and Velocity Verlet (VV) as listed in Table 1. One of the most popular MD simulation packages, LAMMPS [14], takes FE as the only integrator for FIRE in the latest version. The difference of integration formulations rests on how to update atomistic positions and velocities. FE and SE use forces of the last and current step respectively to update velocities. VV uses velocities and forces of the last step to update positions, and then uses averaged forces of the last and current step to update velocities. Given a system with N atoms, both FE and SE are first-order integrator, and 3N vectors are required to store position, velocity and acceleration data of the current step. VV is a second-order integrator and needs extra N vectors to store acceleration data of the last step.

In order to explore the influence of integration formulations on the performance of FIRE, three typical minimization systems are investigated: (i) optimization of the two-dimensional (2D) spiral potential energy function, which is the trial problem in the original FIRE paper [13], (ii) relaxation of a 2D crystal atomic system which is a typical problem in molecular simulations and (iii), propagation of a 2D edge dislocation with atoms jumping over minor energy barriers. Results are discussed in comparison with that of the conventional CG and the limited memory Broyden-Fletcher-Goldfarb-Shanno (LBFGS) methods.

3. Results and discussions

3.1. Spiral-shaped potential energy function

With a corrugated and intricate landscape, the spiral-shaped potential energy function (Eq. (1) offers a heuristic benchmark to compare different minimization methods. The inset in Fig. 2(a) shows the landscape of the function with the minimum point locating at the center of the "disc", and an initial guess near the edge is selected as the starting point. Fig. 2(a) shows the evolution of the azimuthal angle θ in Eq. (1) versus the number of function calls of different methods. Obviously, LBFGS is always ahead of CG. FIRE with different integration formulations present huge difference. FIRE using SE takes only 296 function calls to reach the minimum point, which is even faster than LBFGS of 200 function calls. FIRE using VV is slightly slower than using SE, and surpasses LBFGS after 230 function calls. FIRE using FE moves to θ of only about $\pi/12$ after 296 functions calls, which is much slower than CG and LBFGS. For the rate of change, SE and VV increase nonlinearly, whereas CG and LBFGS increase linearly. Although FIRE using SE and VV are slower at the beginning, they show better efficiency than CG and LBFGS overall.

The above difference can also be observed from their trajectories. Fig. 2(b) gives the whole trajectory of SE and the partial trajectories of FE, CG and LBFGS within the dashed box. SE moves almost along the bottom spiral curve smoothly and perfectly, and each function call makes a positive contribution to the minimization process. VV is roughly close to SE, and the final difference comes from the 296th iteration. Different from the former two integrators, FE moves with random distances and wild directions which often deviates from the bottom spiral curve and goes uphill. Each function call in FE makes less or even negative contribution to the minimization process, so that the efficiency is much worse than that of SE and VV. As for classical minimization methods, LBFGS goes farther than CG in each iteration,