# Dangers of Multiple-Time-Step Methods*

Jeffrey J. Biesiadecki and Robert D. Skeel
Department of Computer Science and Beckman Institute
(University of Illinois at Urbana-Champaign)
1304 West Springfield Avenue
Urbana, Illinois 61801-2987, U.S.A.

May 6, 1993

**Keywords:** leapfrog method, Störmer method, Verlet method, symplectic method, canonical method, multiple time step methods, distance class methods, symplectic integrator, molecular dynamics simulation.

**AMS(MOS) Subject Classifications**: 65L07, 70H15

**CR Subject Classifications**: G.1.7

**Abstract**

In this work we demonstrate two potential dangers of multiple-time-step techniques for numerical integration of differential equations. This idea of using different time steps for different interactions was proposed in 1978 for molecular dynamics but undoubtedly has other useful applications. The use of multiple time stepping with the popular Verlet method was proposed in a 1991 paper. However, the method advocated in this paper does not retain the symplectic (or canonical) property of the Verlet method, which is an abstract property satisfied by the flow of any Hamiltonian system, of which molecular dynamics is an example. Recent work reported in the literature suggests that this property is important for the long-time integration of Hamiltonian dynamical systems. We perform experiments on linear and nonlinear problems comparing symplectic and nonsymplectic multiple-time-stepping extensions of the Verlet method. We observe that in the nonsymplectic case either instability or dissipation becomes evident after a long integration. However, our experiments also indicate that it is quite possible to obtain an artificial "resonance" for the symplectic method that is much worse than that for the nonsymplectic methods.

## 1 Introduction

In this work we demonstrate two potential dangers of multiple-time-step techniques for numerical integration of differential equations. The use of different time steps for different interactions was proposed [10] in 1978 for molecular dynamics (MD), but the idea undoubtedly has other

*This work was supported in part by National Science Foundation Grant DMS 90 15533 and National Institutes of Health Grant PHS 2 P41 RR05969-03

useful applications. In molecular dynamics only a small percentage of interactions require a very small stepsize in order to be resolved accurately; for most interactions much larger time steps are adequate. This idea is developed further in a recent paper [6], which uses multiple timesteps in a way that generalizes the popular Encke/Störmer/leapfrog/Verlet method. The theme of this paper is to evaluate different electrostatic interactions with different time increments depending on the distance. Pairs of particles are grouped into distance classes whose memberships change as the particles move. Method coefficients are chosen so that if all particle pairs were to belong to one distance class, the method would reduce to a single time step Verlet method. A speedup by a factor of 5.4 was observed with no significant loss of accuracy. (A popular alternative is to ignore completely long-range interactions beyond a certain "cut off" distance, but this can significantly change the behavior of the molecule [1, page 155].)

Recent work reported in the literature suggests that for the long-time integration of Hamiltonian dynamical systems, such as occurs in molecular dynamics, one should use methods that preserve the symplectic (or canonical) property of the *flow*. The multiple-time-stepping method advocated in [6] does not retain the symplectic property of the Verlet method and it requires additional storage of past positions. We have performed experiments comparing this method to a symplectic method and to another nonsymplectic method, neither of which require the additional storage. Experiments on linear and nonlinear problems (very small systems that model bondlength and Lennard-Jones interactions, respectively) show striking superiority of the symplectic methods on long enough time intervals. We observe that for nonsymplectic multiple time stepping either instability or dissipation becomes evident after a long integration.

On the other hand, it is suggested [6] that resonance may be a problem for the symplectic method, and we confirm this for linear and nonlinear problems. We show that it is quite possible to obtain an artificial "resonance" for the symplectic method that is much worse than that for the nonsymplectic methods. This situation arises because the large time-step discretization (which resolves the slowly varying interactions) introduces periodic impulses of a low enough frequency to excite natural high frequencies arising from the fast interactions, which are being integrated with a smaller time step. The resonance waxes and wanes on a short timescale. However, it might be the case that the irregularities of a large nonlinear system make such resonance very unlikely.

The simple experiments reported in this paper are for situations in which the stepsize used for each interaction remains fixed. This is insufficient for nonbonded interactions, and we are currently investigating a strategy for permitting the stepsize to increase or decrease as the distance between interacting particles changes without sacrificing symplecticness.

Section 2 states the equations used for molecular dynamics, and section 3 defines the property of being symplectic. Section 4 describes the multiple-time-step algorithms, and these are compared in sections 5 and 6 with three numerical experiments.

## 2 Molecular Dynamics Simulations

A molecular dynamics simulation of $\mathcal{N}$ particles based on principles of Newtonian mechanics requires the solution of the following system of second-order ordinary differential equations:

$$m_k \frac{d^2}{dt^2}\mathbf{r}_k(t) \;=\; \mathbf{F}_k(\mathbf{r}), \qquad\qquad k = 1, 2, \ldots, \mathcal{N}, \tag{1}$$

where $m_k$, $\mathbf{r}_k$, and $\mathbf{F}_k$ is the mass of, position of, and force acting upon particle $k$, respectively. The force $\mathbf{F}_k = -\nabla_k V(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_\mathcal{N})$ where the potential energy $V$ is a sum of contributions from nonbonded 2-body forces and bonded 2-, 3-, and 4-body forces. The nonbonded forces consist of Coulomb forces due to charges (or assigned fractional charges) of particles and the Van der Waals forces, and the number of such interactions is $\frac{1}{2}\mathcal{N}(\mathcal{N}-1)$, which is much greater than the number of bonded interactions.

The Verlet algorithm [12] is a standard way of solving system (1). It obtains the positions of the particles at discrete time steps, $ih$, where $i$ is the step number and $h$ is the size of an individual time step. Defining the vector $\mathbf{x} = (\mathbf{r}_1^{\mathrm{T}}, \mathbf{r}_2^{\mathrm{T}}, \ldots, \mathbf{r}_\mathcal{N}^{\mathrm{T}})^{\mathrm{T}}$ and the vector $\mathbf{f} = \left(\mathbf{F}_1^{\mathrm{T}}/m_1, \mathbf{F}_2^{\mathrm{T}}/m_2, \ldots, \mathbf{F}_\mathcal{N}^{\mathrm{T}}/m_\mathcal{N}\right)^{\mathrm{T}}$, the algorithm approximately satisfies $\ddot{\mathbf{x}}(ih) = \mathbf{f}(\mathbf{x}(ih))$ for each step $i$ of the simulation. The formula used by the Verlet algorithm to obtain approximations $\mathbf{x}_i \approx \mathbf{x}(ih)$ to new particle positions is

$$\mathbf{x}_{i+1} = 2\mathbf{x}_i \; - \; \mathbf{x}_{i-1} \; + \; h^2\mathbf{f}_i, \tag{2}$$

which is second-order accurate. If velocities are wanted, the Verlet algorithm specifies that they be calculated with

$$\dot{\mathbf{x}}_i = \frac{\mathbf{x}_{i+1} - \mathbf{x}_{i-1}}{2h}. \tag{3}$$

Typically one would start a simulation knowing $\mathbf{x}_0$ and $\dot{\mathbf{x}}_0$, so Eq. (3) can be used to eliminate the reference to $\mathbf{x}_{i-1}$ from (2) to get for $\mathbf{x}_1$ Eq. (7). of section 4.1.

# 3 Symplectic Integration Methods

Hamiltonian systems of differential equations have special properties that are retained by symplectic (also called canonical) integration methods. A Hamiltonian is a function $H(\mathbf{q}, \mathbf{p})$ where $\mathbf{q}, \mathbf{p} \in \mathbb{R}^d$. The system of differential equations associated with $H$ is defined to be [9]

$$\frac{dq_i}{dt} = \frac{\partial H}{\partial p_i}, \qquad \frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i}, \qquad 1 \le i \le d. \tag{4}$$

(Here subscript $i$ refers to the $i$th element of a vector.) Equivalently, (4) can be written as

$$\dot{\mathbf{z}} = J \cdot \nabla H \qquad \text{where} \qquad \mathbf{z} = \begin{pmatrix} \mathbf{q} \\ \mathbf{p} \end{pmatrix} \qquad \text{and} \qquad J = \begin{pmatrix} 0 & I_d \\ -I_d & 0 \end{pmatrix}.$$

In molecular dynamics simulations, the Hamiltonian of interest is the formula for total energy expressed as a function of positions and momenta. There are $d = 3\mathcal{N}$ degrees of freedom, $\mathbf{q}$ contains the position of each particle (referred to as $\mathbf{x}$ earlier), and $\mathbf{p}$ contains the momentum of each particle. Define $M$ to be the diagonal matrix of particle masses, with

$$M_{3k-2,3k-2} = M_{3k-1,3k-1} = M_{3k,3k} = m_k, \qquad k = 1, 2, \ldots, \mathcal{N}.$$

Then, the Hamiltonian is given by

$$H(\mathbf{q}, \mathbf{p}) = \frac{1}{2}\mathbf{p}^{\mathrm{T}} M^{-1} \mathbf{p} + V(\mathbf{q}) \tag{5}$$

where the first term is the kinetic energy and the second is the potential energy. The force is $-\nabla V(\mathbf{q})$.

The *t-flow* of a Hamiltonian system [9] is the mapping $\phi_t$ from $\mathbf{z}(0)$ to $\mathbf{z}(t)$ which is the solution of (4) at time $t$ with initial values $\mathbf{z}(0)$. A quick example, from [9], is the Hamiltonian with $d = 1$ and $H(q,p) = \frac{1}{2}q^2 + \frac{1}{2}p^2$, for which the flow is

$$\phi_t(q,p) = \begin{pmatrix} q\cos t + p\sin t \\ -q\sin t + p\cos t \end{pmatrix}.$$

Flow of a Hamiltonian system has an area-preserving property [2]. Let $\Sigma$ be an oriented 2-dimensional surface in $\mathbb{R}^{2d}$. In the case of a Hamiltonian with one degree of freedom, the signed area of $\Sigma$ is equal to the signed area of $\phi_t(\Sigma)$. With multiple degrees of freedom, the sum of the signed areas of the projection of $\Sigma$ onto the $(p_i, q_i)$-plane, $i = 1, 2, \ldots, d$, is preserved. Any mapping that possesses this property for arbitrary $\Sigma$ is called symplectic or canonical.

One way to check if a mapping $\chi$ is symplectic is to see if

$$(\nabla\chi)^{\mathrm{T}} J (\nabla\chi) = J \tag{6}$$

where $\nabla\chi$ is the Jacobian matrix of the mapping. This equation and other characterizations are discussed in [9]. It is easy to verify that (6) holds for the example earlier.

It will be useful to know that the composition of two symplectic mappings $\phi = \phi^{[2]} \circ \phi^{[1]}$ is also symplectic and that the inverse of a symplectic mapping is symplectic.

Numerical approximations $\psi_h$ to $\phi_h$ are sought to calculate positions and momenta through iteration

$$\mathbf{z}_{n+1} = \psi_h(\mathbf{z}_n) = \underbrace{\psi_h(\psi_h(\cdots\psi_h}_{n+1}(\mathbf{z}_0)\cdots)) \approx \phi_{(n+1)h}(\mathbf{z}_0).$$

It is desirable for $\psi_h$ to be symplectic itself, at least disregarding round-off error; this mimics certain properties of the physical system. For example, because of the area-preservation of phase space, the system will not "damp out" under repeated iteration. In particular, there is considerable experimental evidence [3, 7, 8] suggesting symplectic methods perform better than non-symplectic ones.

# 4 Multiple-Time-Step Algorithms

Three multiple-time-step methods are compared: Verlet-I and Verlet-II, which are introduced by Grubmüller et al. [6], and Verlet-X, which is new. Simplified versions of these algorithms will be presented where it is assumed that there are only two distance classes, "near" and "far." Forces $\mathbf{f}^{\text{near}}$ in the "near" class are those that are changing rapidly, so they will be evaluated every time step. The rest of the forces $\mathbf{f}^{\text{far}}$ are in the "far" class and will be evaluated only every *macro*-step, because they change more slowly. The length of a macro-step is $N$ times the length of a normal time step, for some integer $N$. The more general case would be to have an arbitrary number of distance classes. In [6], distance classes are numbered 0 through $n$, with forces in class $j$ (referred to as $\mathbf{f}^{(j)}$) evaluated every $2^j$ time steps and $N = 2^n$.

Because particles move during the course of a simulation, forces between them may change distance classes. But in this paper membership will be considered fixed. Problems that arise even with this easier special case are illustrated in sections 5 and 6.

A constraint imposed on multiple-time-step algorithms by [6] is that they satisfy "Verlet equivalence." This means that if all forces are in a class with step size $Nh$, then the method should give the same results as the standard Verlet algorithm with step size $Nh$.

### 4.1 Verlet Algorithm

The standard Verlet algorithm, given earlier with (2) and (3), can be rewritten in such a way that velocities are made use of in the calculation of new positions. The description of distance-class methods will use the formulas derived below, but differ in how the $\mathbf{f}$ term (which specifies the forces to evaluate) is defined.

First, to get an equation for $\mathbf{x}_{i+1}$, use (3) to eliminate $\mathbf{x}_{i-1}$ from (2). Rearrangement yields

$$\mathbf{x}_{i+1} = \mathbf{x}_i + h\dot{\mathbf{x}}_i + \frac{h^2}{2}\mathbf{f}_i. \tag{7}$$

Then, to get an equation for $\dot{\mathbf{x}}_{i+1}$ use (3) and later plug in (2) to see

$$\begin{aligned}
\dot{\mathbf{x}}_{i+1} - \dot{\mathbf{x}}_i &= \frac{\mathbf{x}_{i+2} - \mathbf{x}_i}{2h} - \frac{\mathbf{x}_{i+1} - \mathbf{x}_{i-1}}{2h} \\
&= \frac{\mathbf{x}_{i+2} - 2\mathbf{x}_{i+1} + \mathbf{x}_i}{2h} + \frac{\mathbf{x}_{i+1} - 2\mathbf{x}_i + \mathbf{x}_{i-1}}{2h} \\
&= \frac{1}{2h}\left(h^2\mathbf{f}_{i+1} + h^2\mathbf{f}_i\right).
\end{aligned}$$

Rearranged, this is

$$\dot{\mathbf{x}}_{i+1} = \dot{\mathbf{x}}_i + \frac{h}{2}\mathbf{f}_i + \frac{h}{2}\mathbf{f}_{i+1}. \tag{8}$$

Eqs. (7) and (8) minimize round-off error and define a method proposed in [11].

Finally, notation is used that will be convenient for distance-class methods, and an intermediate velocity term is introduced to get

$$\dot{\mathbf{x}}_{Ni+k+1/2} = \dot{\mathbf{x}}_{Ni+k} + \frac{h}{2}\mathbf{f}_{Ni+k} \tag{9}$$

$$\mathbf{x}_{Ni+k+1} = \mathbf{x}_{Ni+k} + h\dot{\mathbf{x}}_{Ni+k+1/2} \tag{10}$$

$$\dot{\mathbf{x}}_{Ni+k+1} = \dot{\mathbf{x}}_{Ni+k+1/2} + \frac{h}{2}\mathbf{f}_{Ni+k+1} \tag{11}$$

where $i$ is the current macro-step, $k$ ranges from 0 to $N-1$, and $Ni+k$ is the current time step. For the standard method (without multiple time steps) $\mathbf{f}^{\text{far}}$ is evaluated every step, so the force term is just

$$\mathbf{f}_{Ni+j} = \mathbf{f}^{\text{near}}(\mathbf{x}_{Ni+j}) + \mathbf{f}^{\text{far}}(\mathbf{x}_{Ni+j}).$$

Eq. (9) is the transformation

$$\psi^{[1]}(\mathbf{q},\mathbf{p}) = \begin{pmatrix} \mathbf{q} \\ \mathbf{p} + \frac{h}{2}M\mathbf{f}(\mathbf{q}) \end{pmatrix} \qquad \text{which maps} \qquad \begin{pmatrix} \mathbf{x}_{Ni+k} \\ M\dot{\mathbf{x}}_{Ni+k} \end{pmatrix} \mapsto \begin{pmatrix} \mathbf{x}_{Ni+k} \\ M\dot{\mathbf{x}}_{Ni+k+1/2} \end{pmatrix}.$$

By using (6) and the fact that the Jacobian matrix of $M\mathbf{f}$ is symmetric (because it is the Hessian of the function $-V$), $\psi^{[1]}$ is symplectic. Mapping $\psi^{[2]}$ for Eq. (10) is similarly seen to be symplectic. $\psi^{[1]}$ is used again for (11). Therefore, the Verlet method is symplectic, since it is just the composition of these mappings,

$$\psi = \psi^{[1]} \circ \psi^{[2]} \circ \psi^{[1]}.$$

### 4.2 Verlet-I Algorithm

This method scales forces in each class by the ratio of its step size to the smallest step size. In [6],

$$\mathbf{f}_{Ni+j} \;=\; \sum_{\ell=0}^{n} 2^{\ell}\delta_{j,2^{\ell}\lfloor j/2^{\ell}\rfloor}\mathbf{f}^{(\ell)}(\mathbf{x}_{Ni+2^{\ell}\lfloor j/2^{\ell}\rfloor})$$

where $\delta_{i,j} = 1$ if $i = j$ and 0 otherwise.

In general the time steps of each class need not be $h$ times some power of 2. Using the assumption about having only two different time steps, one gets

$$\mathbf{f}_{Ni+j} \;=\; \begin{cases} \mathbf{f}^{\text{near}}(\mathbf{x}_{Ni+j}) + N\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}), & j \;=\; 0, \\ \mathbf{f}^{\text{near}}(\mathbf{x}_{Ni+j}), & j \;=\; 1, 2, \ldots, N-1. \end{cases}$$

The mapping corresponding to one macro-step of Verlet-I is the composition of simple symplectic mappings and thus is also symplectic.

Note that "far" forces are felt only at the beginning and end of each macro-step. A physical interpretation would be two equally large impulses at time $t = Nih-$ and $t = Nih+$. A possible problem arising from this scheme will be discussed and demonstrated in section 6.

### 4.3 Verlet-II Algorithm

This algorithm includes forces from the previous macro-step to increase accuracy, and to spread out the effects of "far" forces over the entire macro-step (unlike Verlet-I). Forces are to be of the form

$$\mathbf{f}_{Ni+j} \;=\; \mathbf{f}^{\text{near}}(\mathbf{x}_{Ni+j}) + a_j\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) + b_j\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N})$$

for some choice of coefficients $a_j$ and $b_j$. The idea is to approximate linear extrapolation of the "far" force, giving

$$a_j\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) + b_j\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N}) \approx \mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) + \frac{j}{N}\left(\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) - \mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N})\right). \tag{12}$$

However, the choice of $a_j$ and $b_j$ that satisfies extrapolation exactly gives a method that does not satisfy Verlet equivalence [6]. The Verlet-II algorithm is derived by finding coefficients that are as close as possible to those in linear extrapolation with the constraint of Verlet equivalence.

The result of a least-squares derivation obtained in [6] for a full set of distance classes is

$$\mathbf{f}_{Ni+j} \;=\; \sum_{\ell=0}^{n}\left(a^{(\ell)}_{j-2^{\ell}\lfloor j/2^{\ell}\rfloor}\mathbf{f}^{(\ell)}(\mathbf{x}_{Ni+2^{\ell}\lfloor j/2^{\ell}\rfloor}) + b^{(\ell)}_{j-2^{\ell}\lfloor j/2^{\ell}\rfloor}\mathbf{f}^{(\ell)}(\mathbf{x}_{Ni+2^{\ell}\lfloor j/2^{\ell}\rfloor-2^{\ell}})\right)$$

with coefficients

$$\begin{aligned} a^{(\ell)}_j &= 3\cdot 2^{\ell+1}\frac{(2^{2\ell} - 2^{\ell} + 1) - j(2^{\ell} - 2)}{(2^{\ell} + 1)(2^{2\ell+1} + 1)}, \\ b^{(\ell)}_j &= \frac{2(-2^{2\ell+1} + 3\cdot 2^{\ell} - 1) + 6j(2^{\ell} - 1)}{2^{2\ell+1} + 1}. \end{aligned}$$

With our assumption that forces are in the innermost ($\mathbf{f}^{(0)} \equiv \mathbf{f}^{\text{near}}$) or outermost ($\mathbf{f}^{(n)} \equiv \mathbf{f}^{\text{far}}$) distance classes only, this leads to

$$\mathbf{f}_{Ni+j} \;=\; a_0^{(0)}\mathbf{f}^{\text{near}}(\mathbf{x}_{Ni+j}) + b_0^{(0)}\mathbf{f}^{\text{near}}(\mathbf{x}_{Ni+j-1}) + a_j^{(n)}\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) + b_j^{(n)}\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N}).$$

Note that $a_0^{(0)} = 1$, $b_0^{(0)} = 0$, and $N = 2^n$, so the superscripts are dropped and the simplified version of Verlet-II is

$$\begin{aligned}
\mathbf{f}_{Ni+j} &= \mathbf{f}^{\text{near}}(\mathbf{x}_{Ni+j}) + a_j\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) + b_j\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N}),\\
a_j &= 6N\frac{(N^2-N+1)-j(N-2)}{(N+1)(2N^2+1)},\\
b_j &= \frac{2(-2N^2+3N-1)+6j(N-1)}{2N^2+1}.
\end{aligned}$$

This method is not symplectic. One attractive feature it has, however, is that the coefficients $a_j$ and $b_j$ are bounded, unlike coefficients used in Verlet-I and Verlet-X, which makes Verlet-II less sensitive to variations in the forces.

### 4.4 Verlet-X Algorithm

This algorithm also approximates linear extrapolation. Rearrange the left side of (12) to get

$$\begin{aligned}
(a_j+b_j)\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) &- b_j\left(\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) - \mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N})\right)\\
&\approx \mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) + \frac{j}{N}\left(\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) - \mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N})\right). \qquad (13)
\end{aligned}$$

Presumably $\|\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) - \mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N})\| \ll \|\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni})\|$, so the error will be smaller if left terms of (13) match. This means $a_j + b_j = 1$. The formulas for satisfying Verlet equivalence derived in [6, page 129] are

$$\begin{aligned}
\sum_{k=0}^{N-1}(N-k)a_k &= N^2,\\
\sum_{k=1}^{N-1} ka_k + \sum_{k=0}^{N-1}(N-k)b_k &= 0, \qquad (14)\\
\sum_{k=1}^{N-1} kb_k &= 0.
\end{aligned}$$

With the given restriction on $a_j$ and $b_j$, the equations above reduce to

$$\sum_{k=0}^{N-1} ka_k \;=\; \frac{N(N-1)}{2} \qquad \text{and} \qquad \sum_{k=0}^{N-1} a_k \;=\; \frac{3N-1}{2}. \qquad (15)$$

If $b_1 = b_2 = \cdots = b_{N-1} = 0$ (which saves storage because $\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N})$ can be discarded before $\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni})$ is computed), then $a_1 = a_2 = \cdots = a_{N-1} = 1$. The condition on the left in (15) is

now satisfied, and the right condition gives $a_0 = \frac{N+1}{2}$ and $b_0 = \frac{1-N}{2}$. The method is then

$$\mathbf{f}_{Ni+j} \;=\; \begin{cases} \mathbf{f}^{\text{near}}(\mathbf{x}_{Ni+j}) + \frac{N+1}{2}\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) + \frac{1-N}{2}\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N}), & j \;=\; 0, \\ \mathbf{f}^{\text{near}}(\mathbf{x}_{Ni+j}) + \mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}), & j \;=\; 1, 2, \ldots, N-1. \end{cases}$$

This method is not symplectic either.

### 4.5 Starting Verlet-II and Verlet-X

The Verlet-I algorithm is self-starting. Verlet-II and Verlet-X, on the other hand, require $\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N})$, which is not available for the first macro step.

One possibility, used in [6], is to begin the simulation with all forces in the "near" class, and move "far" forces out to farther classes as the run progresses. The problem with this idea is that Verlet equivalence will be violated when switching methods at time $t = Nh$. This can be fixed by allowing the force used at the end of one time step (in Eq. (11)) and the beginning of the next (in Eq. (9)) to be different as long as the velocity used in (10) works out to be the same. This matters only between macro-steps when switching methods. Let $\mathbf{f}^{+}_{Ni}$ be the force used in (9) when $k = 0$ and $\mathbf{f}^{-}_{Ni+N}$ the force in (11) when $k = N-1$. Using

$$\mathbf{f}^{\pm}_{Ni} = \mathbf{f}^{\text{near}}(\mathbf{x}_{Ni}) + 2a_0^{\pm}\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) + 2b_0^{\pm}\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N}),$$

it is then required that

$$\begin{aligned} a_0^{-} + a_0^{+} &= a_0, \\ b_0^{-} + b_0^{+} &= b_0. \end{aligned}$$

To determine $a_0^{\pm}$ and $b_0^{\pm}$ that satisfy Verlet equivalence, Eq. (9) through (11) are used. Assuming $\mathbf{f}^{\text{near}} \equiv \mathbf{0}$, one macro-step of an extrapolation method yields

$$\begin{aligned} \dot{\mathbf{x}}_{Ni+N} \;=\; \dot{\mathbf{x}}_{Ni} \;+\; & h(b_0^{+} + \sum_{k=1}^{N-1} b_k)\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N}) \\ & + h(a_0^{+} + \sum_{k=1}^{N-1} a_k + b_0^{-})\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}) \;+\; ha_0^{-}\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni+N}), \end{aligned}$$

$$\begin{aligned} \mathbf{x}_{Ni+N} \;=\; \mathbf{x}_{Ni} \;+\; & Nh\dot{\mathbf{x}}_{Ni} \;+\; h^2(Nb_0^{+} + \sum_{k=1}^{N-1}(N-k)b_k)\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N}) \\ & + h^2(Na_0^{+} + \sum_{k=1}^{N-1}(N-k)a_k)\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni}). \end{aligned}$$

And comparing the above to one step of standard Verlet with step size $Nh$ (to ensure Verlet equivalence) gives

$$b_0^{+} + \sum_{k=1}^{N-1} b_k \;=\; 0,$$

$$
\begin{aligned}
a_0^+ + \sum_{k=1}^{N-1} a_k + b_0^- &= \frac{N}{2}, \\
a_0^- &= \frac{N}{2}, \\
N b_0^+ + \sum_{k=1}^{N-1} (N-k) b_k &= 0, \\
N a_0^+ + \sum_{k=1}^{N-1} (N-k) a_k &= \frac{N^2}{2}.
\end{aligned}
$$

Eqs. (14) are used to eliminate $\sum_{k=1}^{N-1} k a_k$, $\sum_{k=0}^{N-1} b_k$, and $\sum_{k=1}^{N-1} k b_k$, to get

$$
a_0^- = \frac{N}{2}, \qquad\qquad b_0^- = N - \sum_{k=0}^{N-1} a_k.
$$

Note that for the Verlet-X method $b_0^+ = 0$. This is important because otherwise $\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni-N})$ would be needed after $\mathbf{f}^{\text{far}}(\mathbf{x}_{Ni})$, ruining the storage-saving feature of Verlet-X.

# 5 Numerical Experiments: Long-Time Effects

The model problems described below are the simplest systems that illustrate the effects of the algorithms tested. The following two sets of experiments show how the non-symplectic methods Verlet-II and Verlet-X suffer from either instabilities or artificial dissipation, with both linear and non-linear forces.

The primary metric used here to quantify the accuracy of the results of a simulation is conservation of energy. This measure is convenient, and it has proven to be a useful gauge of accuracy for methods not specifically designed to conserve energy. Very often determining properties of the system as a whole is what is desired, and it is acceptable if particle trajectories are not at all accurate [1, pages 76-77].

It should be noted that a distance-class algorithm is intended to be run one full macro-step at a time, and positions should not be recorded and statistics sampled in the middle of a macro-step.

## 5.1 Experiments with Linear Forces

The first experiment is a one-dimensional simulation of three particles bound together by two springs of different stiffnesses, with no other forces considered. (This is a way of modeling bondlength forces.) The leftmost particle (particle 1) is bound to the middle particle (particle 2) by spring 1; the middle particle is also bound to the rightmost particle (particle 3) by spring 2.

The potential energy of the system is

$$
V(r_1, r_2, r_3) \;=\; \frac{1}{2} k_1 (|r_2 - r_1| - \ell_1)^2 \;+\; \frac{1}{2} k_2 (|r_3 - r_2| - \ell_2)^2.
$$

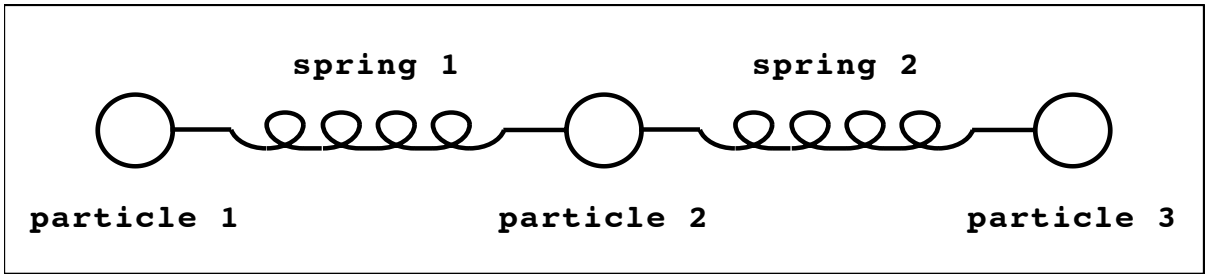


Figure 1: Spring experiment, system of particles.

The parameters and initial conditions for this experiment are

| | |
|---|---|
| particle masses | $m_1 = m_2 = m_3 = 1$ |
| initial positions | $r_1 = -7,\ r_2 = 0,\ r_3 = 7$ |
| initial velocities | all particles at rest |
| spring stiffnesses | $k_1 = 16,\ k_2 = 1$ |
| equilibrium spring lengths | $\ell_1 = 6,\ \ell_2 = 6$ |

The initial total energy of the system is $E_0 = 8.5$.

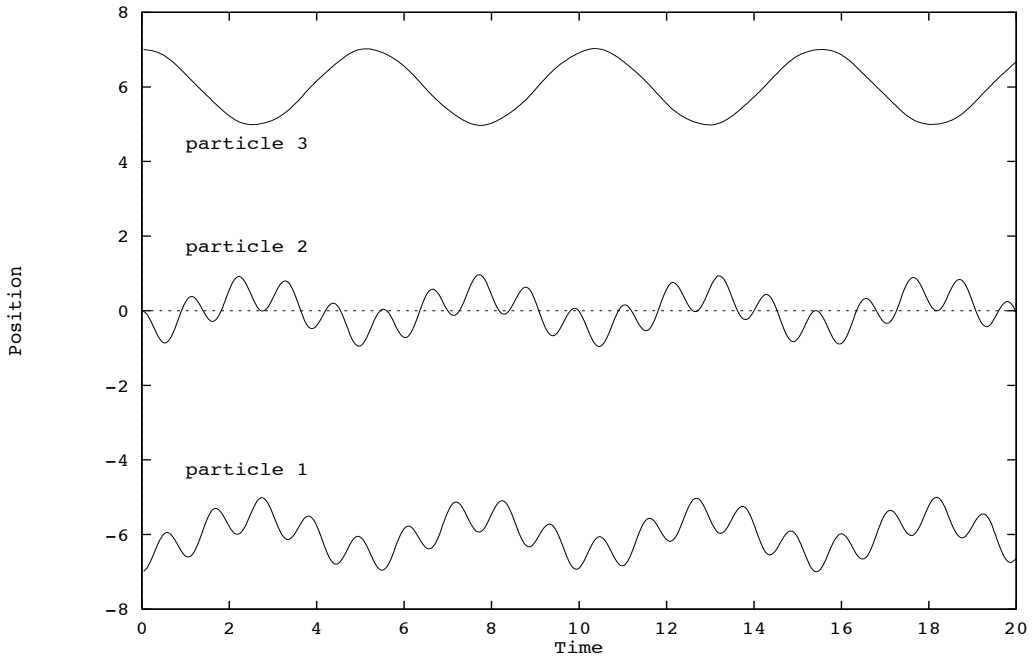


Figure 2: Spring experiment, analytic solution of particle trajectories.

Since all of the forces in this experiment are linear, the analytic solution can be determined to compare against the approximations. The fundamental frequencies are $\omega_1 = \sqrt{17 + \sqrt{241}} \approx$

5.702997 and $\omega_2 = \sqrt{17 - \sqrt{241}} \approx 1.214836$. The solution is

$$\mathbf{r}(t) = \begin{pmatrix} -6 + (-\frac{1}{2} + \frac{1}{2}\frac{\sqrt{241}}{241})\cos(\omega_1 t) + (-\frac{1}{2} - \frac{1}{2}\frac{\sqrt{241}}{241})\cos(\omega_2 t) \\ \frac{15}{2}\frac{\sqrt{241}}{241}\cos(\omega_1 t) - \frac{15}{2}\frac{\sqrt{241}}{241}\cos(\omega_2 t) \\ 6 + (\frac{1}{2} - 8\frac{\sqrt{241}}{241})\cos(\omega_1 t) + (\frac{1}{2} + 8\frac{\sqrt{241}}{241})\cos(\omega_2 t) \end{pmatrix}.$$

Figure 2 shows the analytic solution for a short amount of time. Formulas for the lengths $u_k$ of each spring are

$$\begin{aligned} u_1(t) &= r_2(t) - r_1(t) \approx 6 + 0.95\cos(5.7t) + 0.05\cos(1.2t), \\ u_2(t) &= r_3(t) - r_2(t) \approx 6 - 0.50\cos(5.7t) + 1.5\cos(1.2t). \end{aligned}$$

Because spring 1 is stiffer than spring 2, the force it generates varies more rapidly. So the force of spring 1 is evaluated each time step, while the force of spring 2 is evaluated once every $N$ steps. The error in approximating spring 1 with a second order method will be proportional to $16h^2$, and that from spring 2 proportional to $1(Nh)^2$. This suggests $N = 4$ will be most efficient for this problem, because it balances the errors.

Comparisons between the algorithms are made with a total simulation time of $t_{\text{total}} = 2000$. Thirteen values of $h$ are tested, geometrically increasing from 0.001 to 0.0112. $N$ is tested at 2, 4, and 8 for each of the $h$ values above.

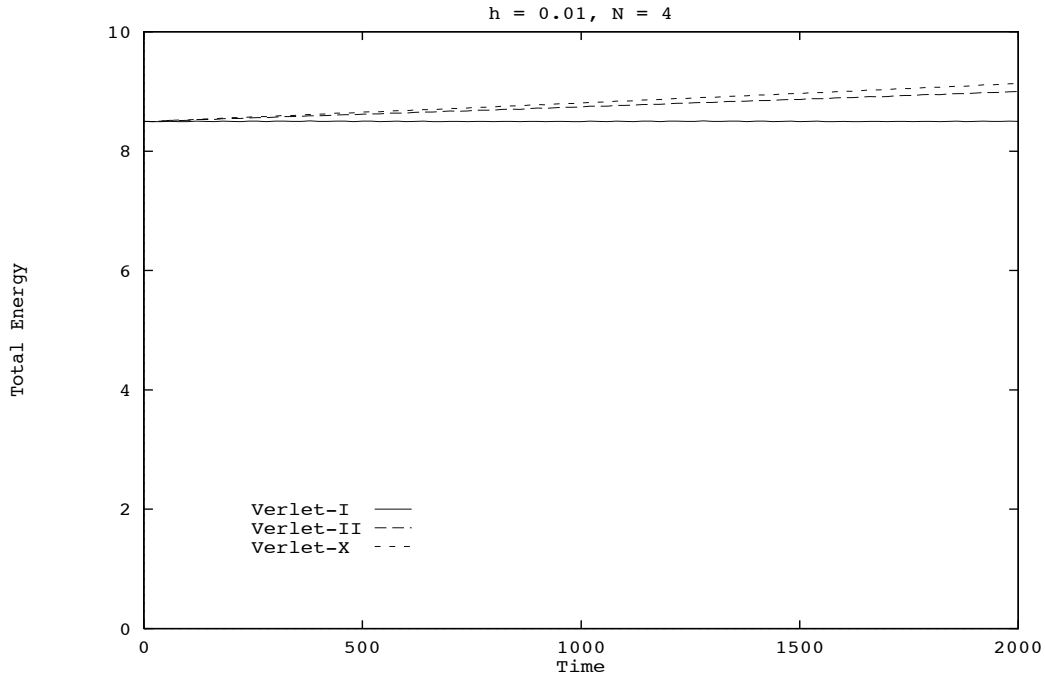


Figure 3: Spring experiment, stability of the multiple-time-step algorithms.

Figure 3 shows the results of a single run for each method. The non-symplectic methods Verlet-II and Verlet-X produce substantial energy drift after long-time integration, even when

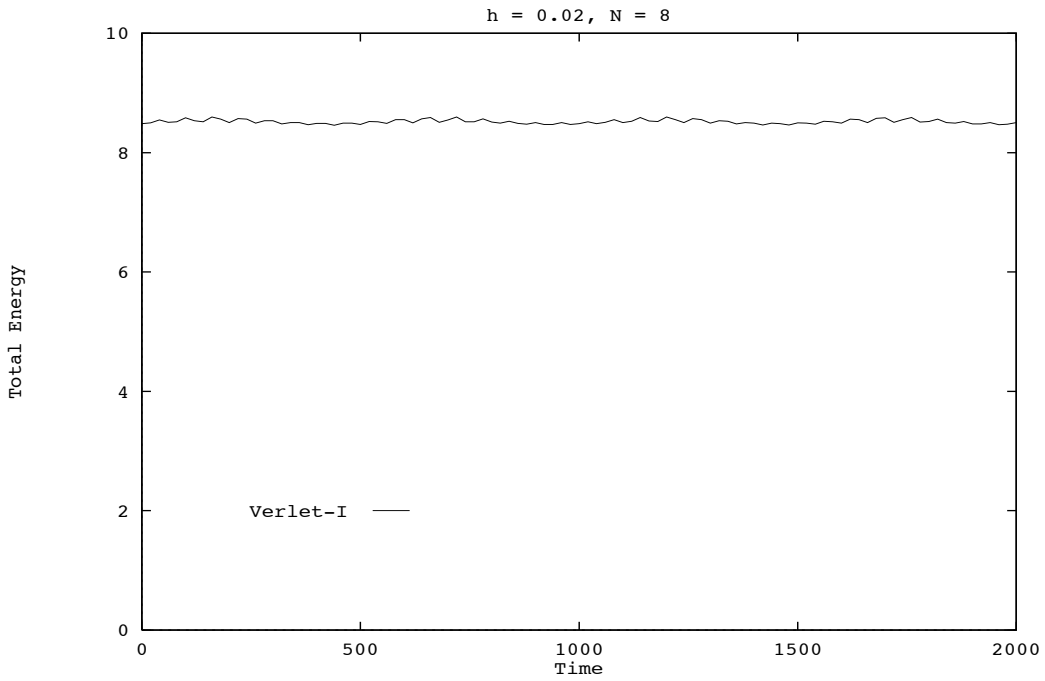


Figure 4: Spring experiment, stability of Verlet-I with doubled step sizes.

very small time steps are used. The Verlet-I algorithm does not experience this problem, and conserves energy well with both $N$ and $h$ doubled (figure 4). How rapidly errors accumulate is related to the stability of an algorithm.

Figures 5 and 6 show statistics of all runs together, with two different metrics to quantify accuracy. Each mark represents a single run with a particular $N$ and $h$ value, and statistics were sampled $s = 100$ times for each run. The number of force evaluations done during the entire simulation was compared against the average error in an attempt to quantify efficiency. The number of force evaluations ($\eta$) is equal to the number of macro-steps times the sum of the number of near force evaluations and far force evaluations per macro-step. For this experiment,

$$\eta = \frac{t_{\text{total}}}{Nh}(N+1). \tag{16}$$

For figure 5, average relative error in position ($\nu$) is determined using

$$\nu = \frac{1}{s}\sum_{i=1}^{s}\frac{\|\mathbf{r}_i - \tilde{\mathbf{r}}_i\|_2}{\|\tilde{\mathbf{r}}_i\|_2} \tag{17}$$

where $s$ is the number of samples taken, and $\mathbf{r}_i$ and $\tilde{\mathbf{r}}_i$ are the computed and analytic solutions, respectively, at the time of the $i$th sample. Not shown are the results with $N = 2$, because for all methods $N = 4$ is the most efficient (needing the fewest number of force evaluations to attain a desired error level) and $N = 8$ the least. Verlet-I was the most efficient of the three algorithms for all values of $N$ tested. From the slopes of the lines, each method is second-order accurate (note that $h$ is proportional to $1/\eta$).

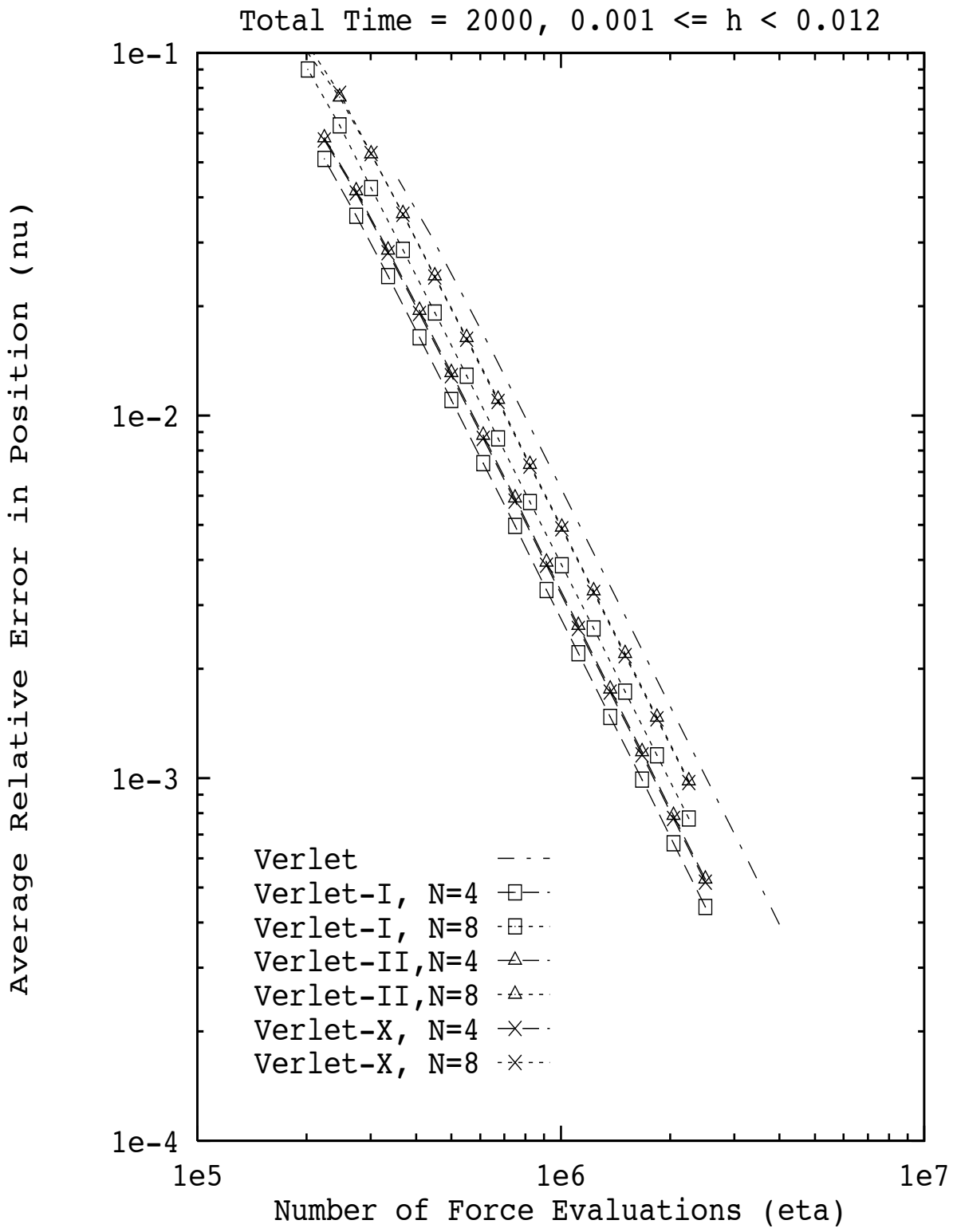


Figure 5: Spring experiment, accuracy of trajectory.

For figure 6, average relative error in total energy ($\xi$) is computed with the formula

$$\xi = \frac{1}{s} \sum_{i=1}^{s} \frac{|E_i - E_0|}{|E_0|} \tag{18}$$

where $s$ is again the number of samples taken, and $E_i$ is the total energy at the time of sample $i$.

Total energy appears to be more sensitive than actual particle positions (perhaps because it involves terms for velocity, which are not taken into account for figure 5). The graph for total energy shows that each of the symplectic methods is far superior to the non-symplectic methods. Note also that Verlet-I is more efficient than the standard algorithm, even though for this experiment there is not too much room for improvement (the stiffness ratio is only sixteen for the system). Again, Verlet-I with $N = 4$ is the most efficient. Interestingly, for the other algorithms $N = 2$ is most efficient with this metric. The symplectic methods are second-order accurate in energy. The extrapolation methods exhibit a higher order of accuracy for energy (third order) with the range of $h$ tested. The error of the two extrapolation methods is likely to be of the form $h^2 e_1(t) + h^3 e_2(t) + O(h^4)$ with $e_2 \gg e_1$ so that for fixed $t_{\text{total}}$ and small enough $h$, the second-order effects will dominate. Because they are more accurate, for any fixed $t_{\text{total}}$ there may be an $h$ small enough so that an extrapolation method will beat the symplectic methods. This can be seen at the right side of figure 6. However, because they are not stable, for any fixed $h$ given sufficiently large $t_{\text{total}}$ the extrapolation methods lose to the symplectic methods.

For large enough $h$ and $N$, Verlet-I also behaves poorly. The results of this experiment, and of other runs with different parameters and initial conditions, indicates that Verlet-II and Verlet-X are unstable for all $h$ but that there exists $h_{\text{threshold}}(N) > 0$ such that Verlet-I is stable for $h < h_{\text{threshold}}(N)$.

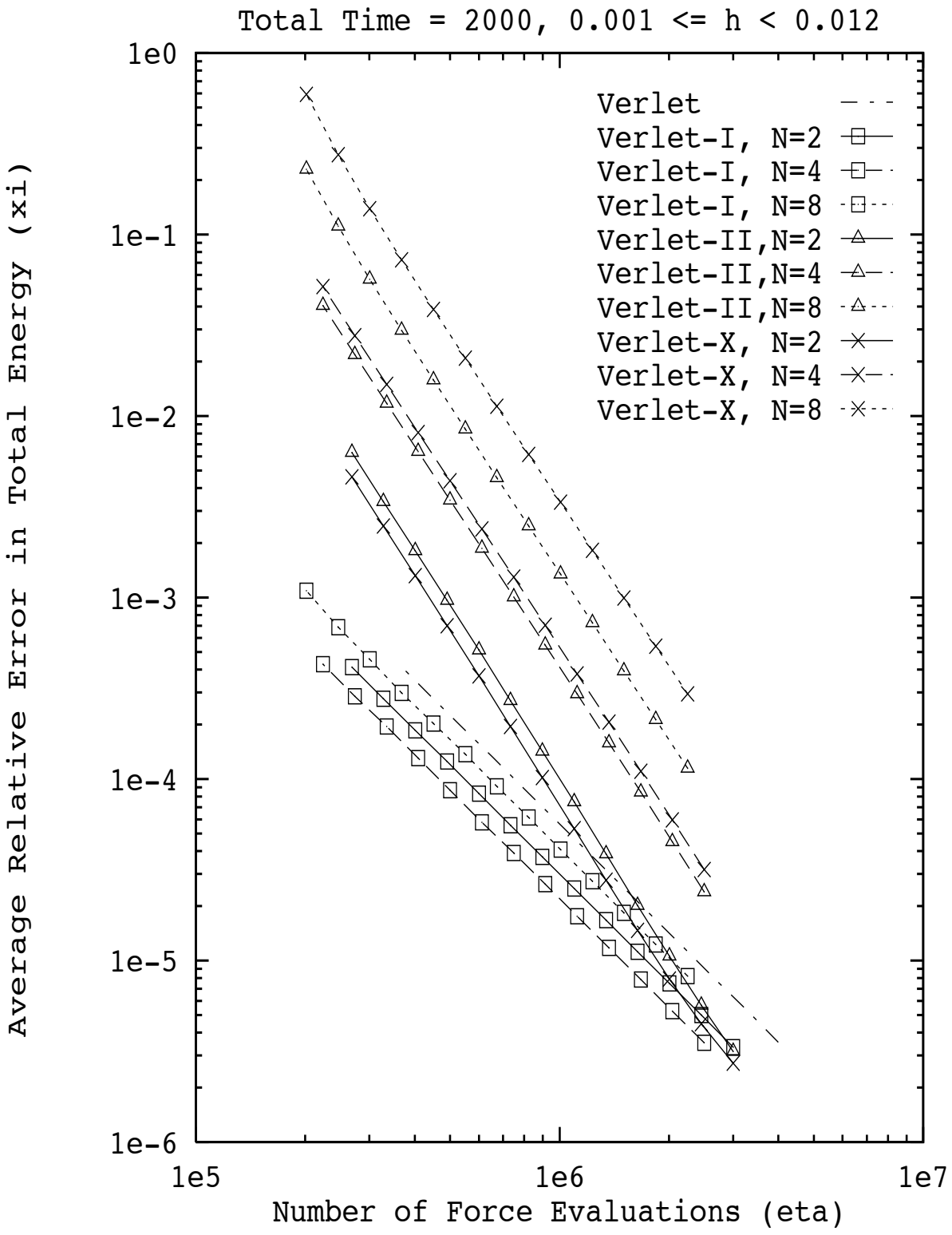


Figure 6: Spring experiment, accuracy of total energy.

### 5.2 Experiments with Non-linear Forces

This experiment is a simulation of a two-dimensional frozen argon crystal. As shown in figure 7, six argon atoms are arranged symmetrically around a center atom with a Lennard-Jones potential between each atom pair.

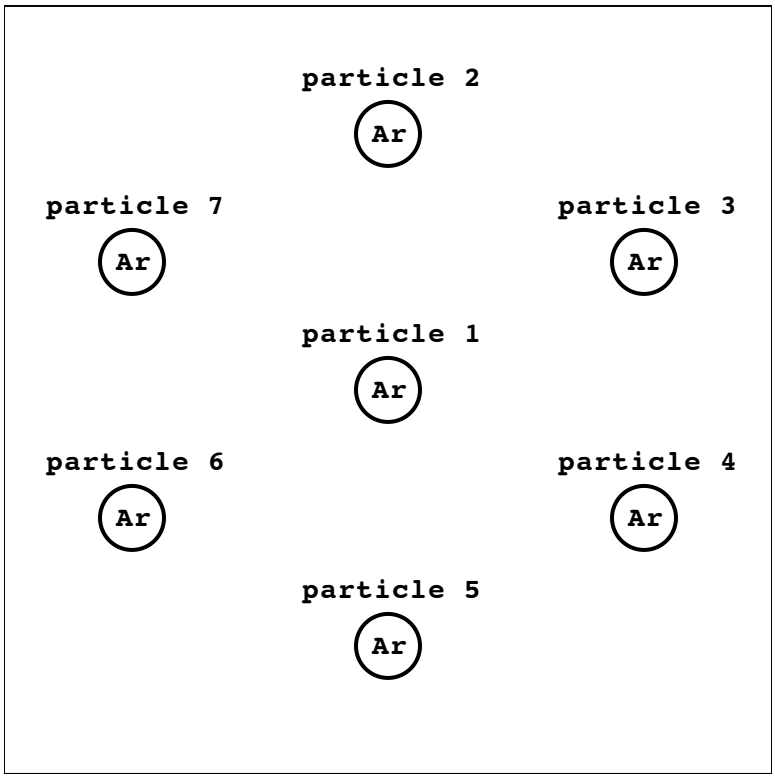


Figure 7: Argon experiment, system of particles.

The potential energy of the system is

$$\begin{aligned} W(r) &= 4\varepsilon\left(\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right), \\ V(\mathbf{x}) &= \sum_{i=1}^{6}\sum_{j=i+1}^{7} W(\|\mathbf{r}_j - \mathbf{r}_i\|_2). \end{aligned}$$

Recall $\mathbf{x} = (\mathbf{r}_1^{\mathrm{T}}, \mathbf{r}_2^{\mathrm{T}}, \ldots, \mathbf{r}_{\mathcal{N}}^{\mathrm{T}})^{\mathrm{T}}$.

The Lennard-Jones potential $W(r)$ is shown in figure 8. The position of lowest potential (and no force) is at a distance of $r = \sqrt[6]{2}\sigma$, and the potential energy at this point is $-\varepsilon$. The force repulses the particles strongly when they are closer than this and attracts them when they are farther. See also [1, pages 8 and 21] for more details and a table of $\varepsilon$ and $\sigma$ for various elements.

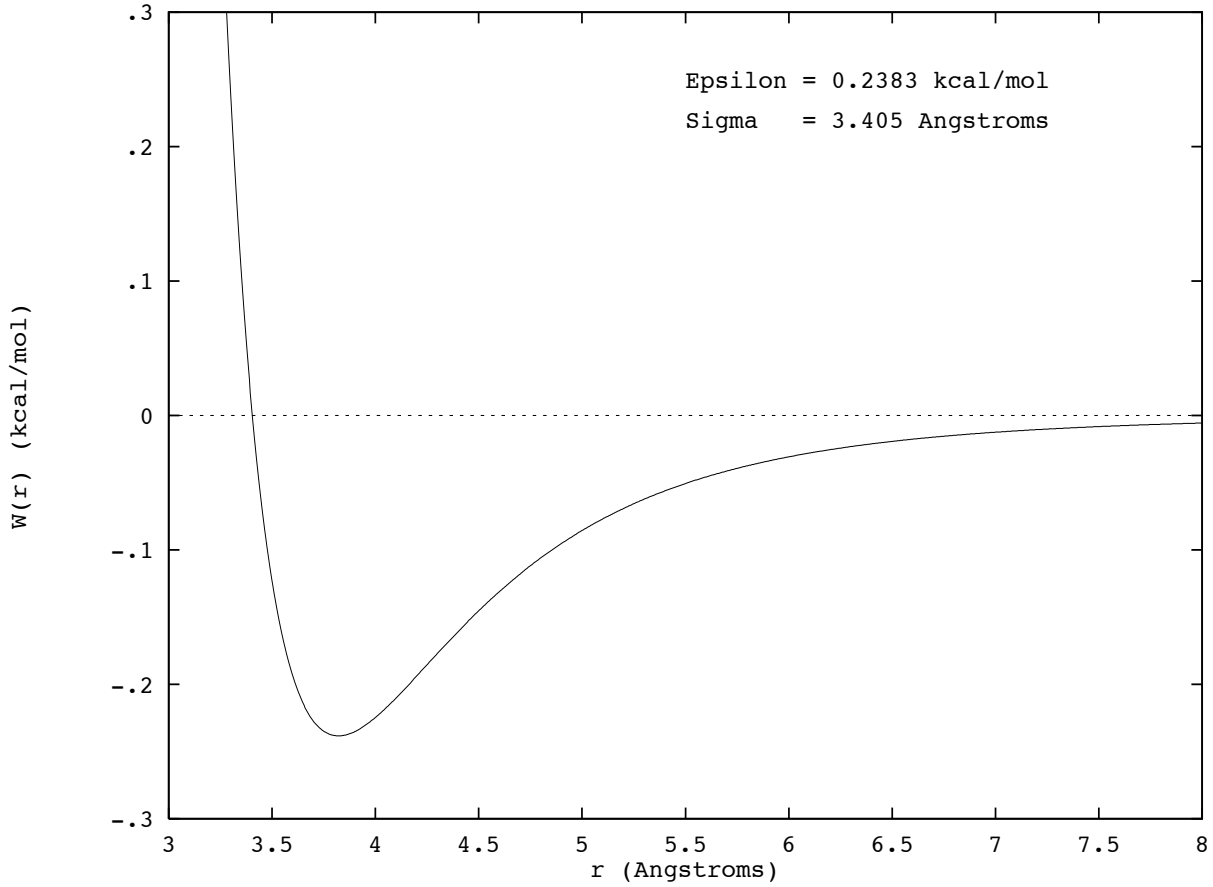


Figure 8: Lennard-Jones pair potential for argon.

The initial positions of the particles are slightly perturbed about those for lowest potential energy (the corners and center of a perfect hexagon with sides of length $\sqrt[6]{2}\sigma$). They are given low initial velocities such that the total momentum of the system is 0.

| | |
|---|---|
| particle masses | 39.95 amu $=$ $66.34 \times 10^{-27}$ kg |
| initial positions (Å) | $\mathbf{r}_1 = (0.0, 0.0), \mathbf{r}_2 = (0.2, 3.9), \mathbf{r}_3 = (3.4, 1.7),$ $\mathbf{r}_4 = (3.6, -2.1), \mathbf{r}_5 = (-0.2, -4.0), \mathbf{r}_6 = (-3.5, -1.6),$ $\mathbf{r}_7 = (-3.1, 2.1)$ |
| initial velocities (m/s) | $\mathbf{v}_1 = (-30, -20), \mathbf{v}_2 = (50, -90), \mathbf{v}_3 = (-70, -60),$ $\mathbf{v}_4 = (90, 40), \mathbf{v}_5 = (80, 90), \mathbf{v}_6 = (-40, 100),$ $\mathbf{v}_7 = (-80, -60)$ |
| $\varepsilon$ | $120°\mathrm{K}k_\mathrm{B} = 120 \cdot 1.380658 \times 10^{-23}$ joules $= 0.2383$ kcal/mol |
| $\sigma$ | 3.405Å |

The initial total energy of the system is $-2.5$ kcal/mol. Temperature can be calculated with the formula

$$T = \frac{2}{k_\mathrm{B}} \frac{1}{d} \sum_{i=1}^{\mathcal{N}} \frac{1}{2} m_i \mathbf{v}_i \cdot \mathbf{v}_i$$

For this two-dimensional experiment $d = 2\mathcal{N}$ and $\mathcal{N} = 7$. Using the conditions above, the initial temperature is 22.72°K, which is very low.

This time the analytic solution to this set of differential equations is not known. Error in energy will be used to measure accuracy here.

The six forces along the outside of the hexagon (between particles 2 and 3, 3 and 4, etc.) are put in the near class. So are the six forces involving the center particle, particle 1. The remaining nine forces are put in the far class. This includes six forces between alternating particles on the outside (particles 2 and 4, 3 and 5, etc.) and the three forces directly across the hexagon (particles 2 and 5, 3 and 6, and 4 and 7). Because the particles lie nearly on the corners and center of a hexagon, distances between far particles are either $\sqrt{3}$ or 2 times as far as distances between near particles.

The simulation is run for a time of $t_{\text{total}} = 1$ nanosecond. Eighteen values of $h$ are tested, geometrically increasing from 2 femtoseconds to 22.25 femtoseconds. $N$ is tested as before at 2, 4, and 8. Some of these macro-step sizes are fairly large—they work because the temperature is so low.

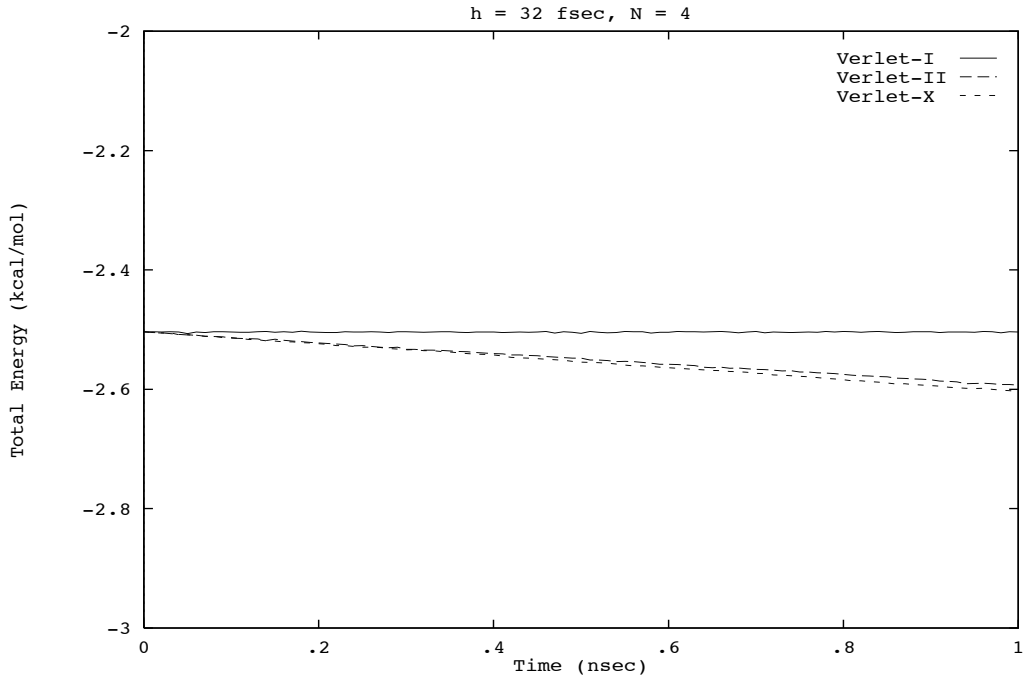


Figure 9: Argon experiment, stability of the multiple-time-step algorithms.

In these experiments, the non-symplectic methods tend to drift to states of lower energy (figure 9). Figure 10 shows that Verlet-I still conserves energy well with $h$ doubled.

Figure 11 shows similar behavior to that of figure 6 of the first experiment. Again, statistics were sampled 100 times during each run. Error in total energy $\xi$ is computed using (18). The formula for the number of force evaluations $\eta$ is analogous to (16), and is given by

$$\eta = \left\lceil \frac{t_{\text{total}}}{Nh} \right\rceil (12N + 9).$$

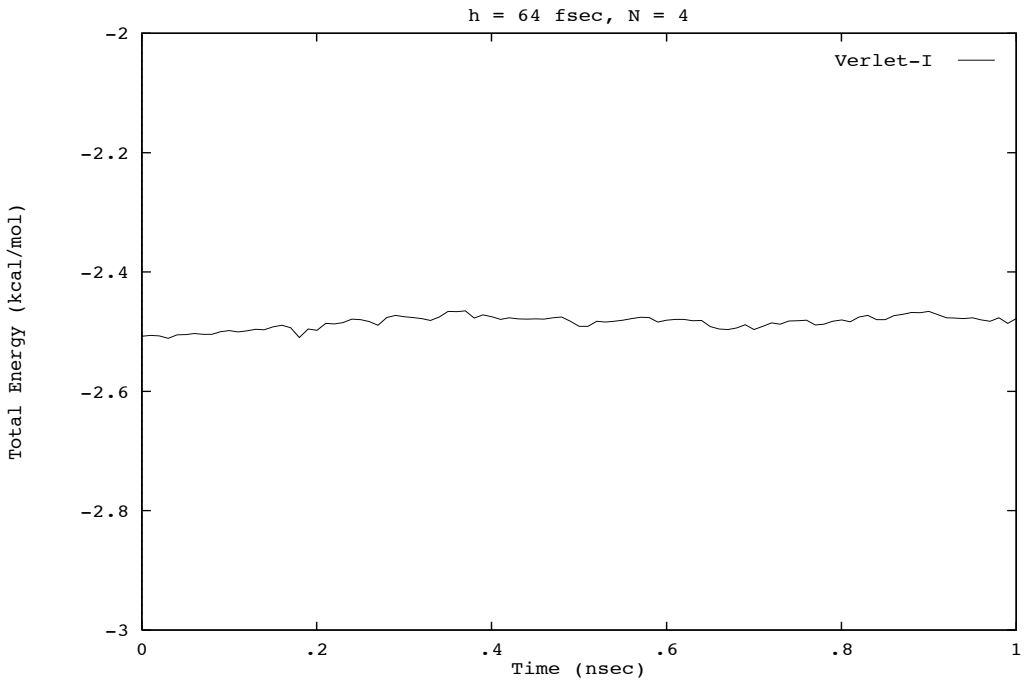


Figure 10: Argon experiment, stability of Verlet-I with doubled step size $h$.

In this experiment, Verlet-I with $N = 4$ was the most efficient. The symplectic methods were again roughly second-order accurate in energy and the non-symplectic methods roughly third order for the range of $h$ tested. The same conclusions can be drawn from this experiment as from the previous one—for long-time integration, the non-symplectic methods perform poorly compared to symplectic methods (even the method that does not use distance classes).

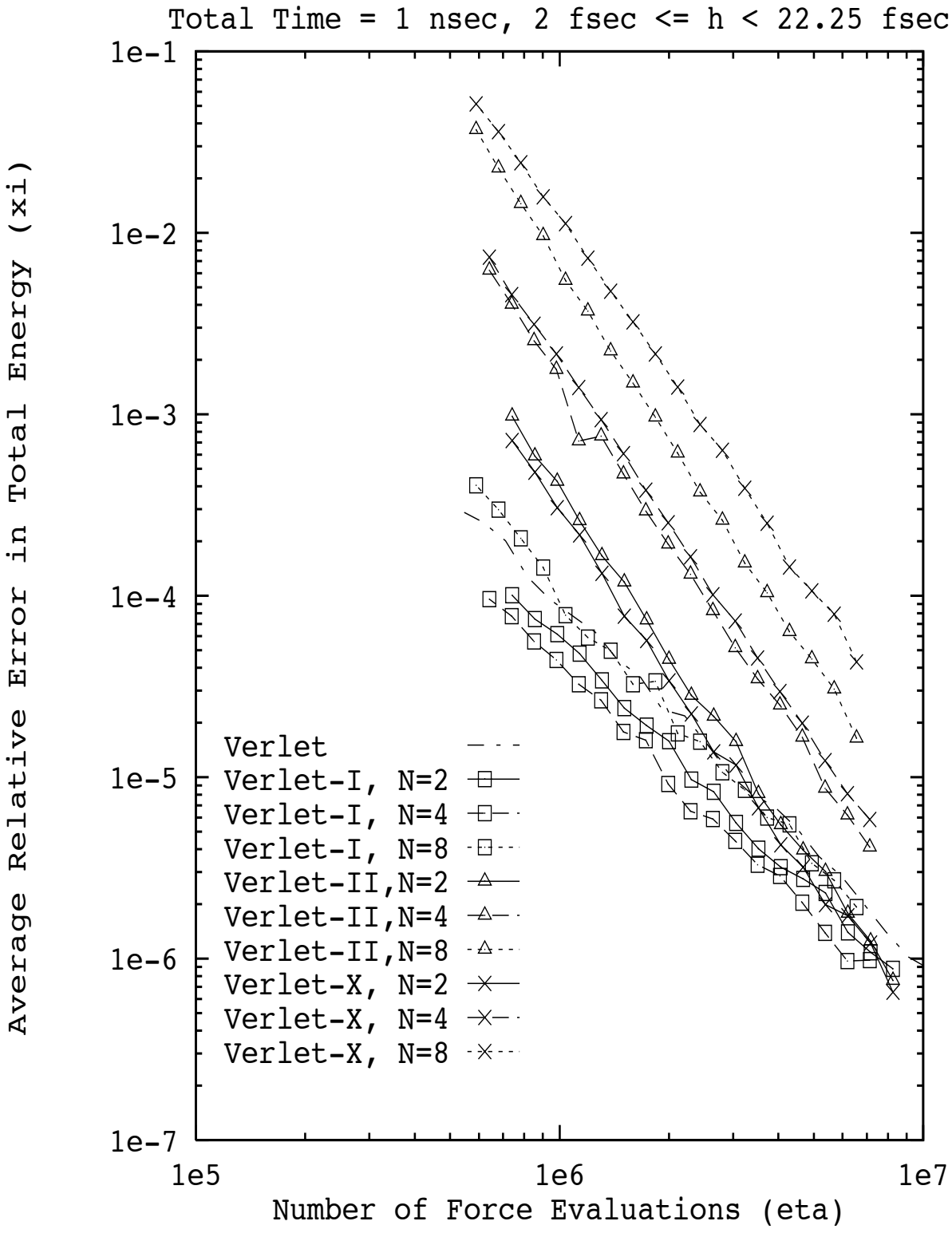


Figure 11: Argon experiment, accuracy of total energy.

# 6 Numerical Experiments: Artificial Resonance

In section 4, we discussed how the different algorithms incorporate the "far" force, with the effects felt as intermittent pulses in Verlet-I and more evenly distributed across the macro-step in the extrapolation methods. This difference could be important if the impulses of Verlet-I occur near a natural frequency of the system, because this could cause resonance in the system. This effect is documented here by experimentation. Another situation where this might occur is hypothesized in [6] and later confirmed in [5].

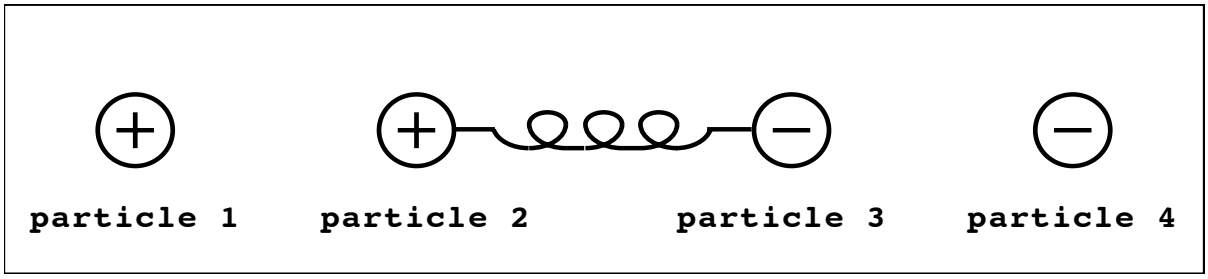


Figure 12: Resonance experiment, system of particles.

Figure 12 shows the system of four particles for this experiment. There is a Coulomb interaction between all particle pairs, and a bond (spring) between particles 2 and 3.

The potential energy of the system is

$$V(r_1, r_2, r_3, r_4) = \frac{1}{2}k(|r_3 - r_2| - \ell)^2 + \sum_{i=1}^{3}\sum_{j=i+1}^{4}\frac{q_i q_j}{|r_i - r_j|}.$$

The parameters and initial conditions are

| | |
|---|---|
| particle masses | $m_1 = 6,\ m_2 = 1,\ m_3 = 1,\ m_4 = 6$ |
| particle charges | $q_1 = +3,\ q_2 = +1,\ q_3 = -1,\ q_4 = -3$ |
| initial positions | $r_1 = -20,\ r_2 = -2.5,\ r_3 = 2.5,\ r_4 = 20$ |
| initial velocities | all particles at rest |
| spring stiffness | $k = 19.7$ |
| equilibrium spring length | $\ell = 7$ |

The initial total energy of the system is 39.05119.

The experiment is set up so that the Coulomb repulsion between particles 1 and 2 is slightly smaller than the sum of attractions between particles 1 and 3 and between 1 and 4. This way particle 1 does not move very much during the simulation, oscillating slightly around its starting position without colliding into the center particles or being flung away from the center. The same is true for particle 4. Since these particles move only a little, and the center particles do not get far past $\pm 4.5$, the Coulomb forces involving particles 1 and 4 do not change much and are put in the "far" class. The spring and Coulomb interaction between particles 2 and 3 are put in the "near" class.

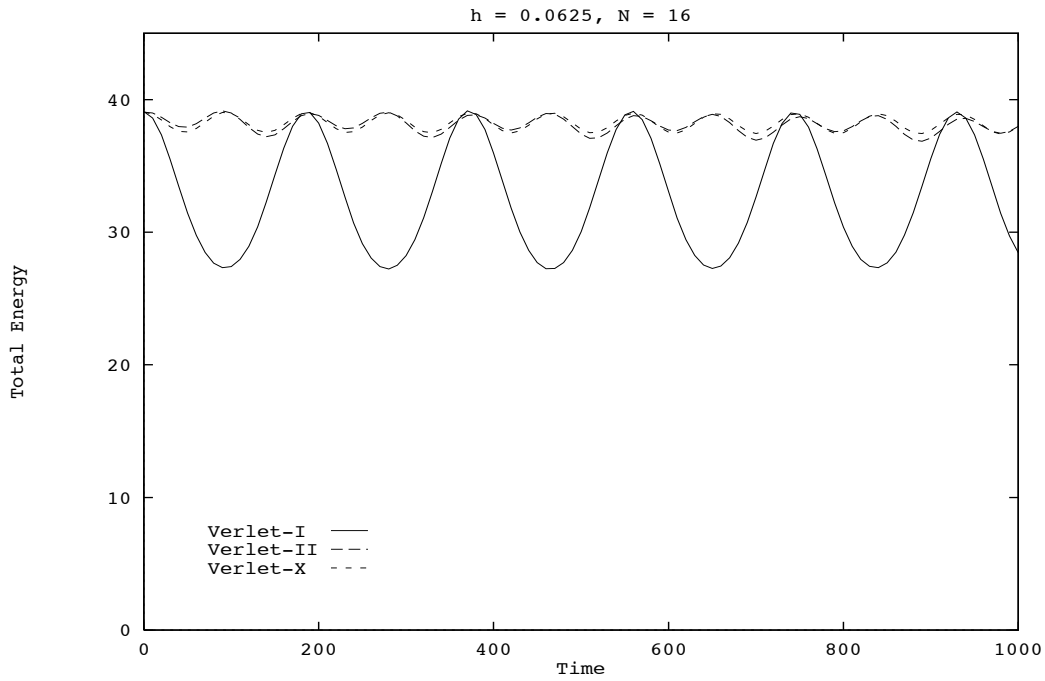


Figure 13: Resonance experiment, artificial resonance of Verlet-I.

If the system consisted of only particles 2 and 3, they would oscillate with some period $\tau \approx 1.0$ (measured numerically). Now considering the system including particles 1 and 4, suppose the new forces are evaluated with a macro-step size of $\tau$. In this scenario, Verlet-I produces resonance because the far forces add energy to the spring in pulses at its natural frequency. The actual physical system has no resonance because the far forces are not impulses but continuous. The extrapolation methods approximate this better than Verlet-I, but for long enough time their inherent instabilities may dominate.

Figure 13 shows the results of the experiment with a macro-step size of $1.0 \approx \tau$, and it can be seen that the Verlet-I algorithm is producing larger oscillations of total energy than the other two algorithms. Figure 14 shows that by *increasing* the step size the Verlet-I algorithm actually conserves energy much better, which would normally be unexpected. Also note that the extrapolation methods are showing signs of instability seen in section 5, though the step size of 0.08 is perhaps fairly large compared to $\tau \approx 1.0$.

Clearly this experiment is contrived. In a complex molecule it seems unlikely that an artificial resonance could sustain itself for any length of time.

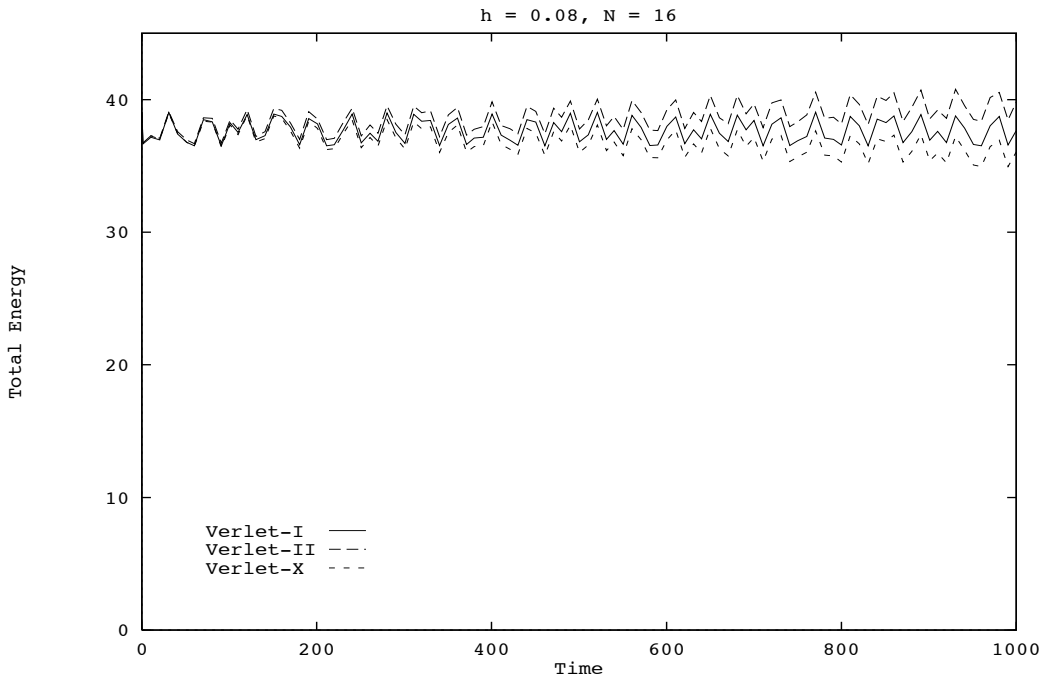


Figure 14: Resonance experiment, increased step size.

# References

[1] M.P. Allen and D.J. Tildesley, *Computer Simulation of Liquids* (Clarendon Press, Oxford, 1987).

[2] V.I. Arnold, *Mathematical Methods of Classical Mechanics, 2nd Edition* (Springer, New York 1989).

[3] P.J. Channell and J.C. Scovel, Symplectic integration of Hamiltonian systems, *Nonlinearity* **3**, 231–259, (1990).

[4] R.D. Skeel and C.W. Gear, Does variable step size ruin a symplectic integrator?, *Physica D* **60**, 311–313, (1992).

[5] H. Grubmüller, private correspondence, (September 1991).

[6] H. Grubmüller, H. Heller, A. Windemuth, and K. Schulten, Generalized Verlet algorithm for efficient molecular dynamics simulations with long-range interactions, *Molecular Simulation* **6**, 121–142 (1991).

[7] D. Okunbor, Canonical methods for Hamiltonian systems: numerical experiments, *Physica D*, **60**, 314–322, (1992).

[8] D. Okunbor and R.D. Skeel, Explicit canonical methods for Hamiltonian systems, *Math. Comp.* **59**, 439–455 (1992).

[9] J.M. Sanz-Serna, Symplectic integrators for Hamiltonian problems: an overview, *Acta Numerica* **1**, 243–286 (1992).

[10] W.B. Streett, D.J. Tildesley, and G. Saville, Multiple time step methods in molecular dynamics, *Mol. Phys.* **35**, 639–648 (1978).

[11] W.C. Swope, H.C. Andersen, P.H. Berens, and K.R. Wilson, A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: application to small water clusters, *J. Chem. Phys.* **76**, 637–649 (1982).

[12] L. Verlet, Computer 'experiments' on classical fluids II. Thermodynamic properties of Lennard-Jones molecules, *Phys. Rev.* **159**, 98–103 (1967).