CE 530 Molecular Simulation

Lecture 11
Molecular Dynamics Simulation

David A. Kofke
Department of Chemical Engineering
SUNY Buffalo
kofke@eng.buffalo.edu
Review and Preview

- **MD of hard disks**
  - intuitive
  - collision detection and impulsive dynamics

- **Monte Carlo**
  - convenient sampling of ensembles
  - no dynamics
  - biasing possible to improve performance

- **Molecular dynamics**
  - equations of motion
  - integration schemes
  - evaluation of dynamical properties
  - extensions to other ensembles
  - focus on atomic systems for now
Classical Equations of Motion

- Several formulations are in use
  - Newtonian
  - Lagrangian
  - Hamiltonian

- Advantages of non-Newtonian formulations
  - more general, no need for “fictitious” forces
  - better suited for multiparticle systems
  - better handling of constraints
  - can be formulated from more basic postulates

- Assume conservative forces
  \[ \mathbf{F} = -\nabla U \]
  Gradient of a scalar potential energy
Newtonian Formulation

- Cartesian spatial coordinates \( \mathbf{r}_i = (x_i,y_i,z_i) \) are primary variables
  - for \( N \) atoms, system of \( N \) 2nd-order differential equations
    \[
    m \frac{d^2 \mathbf{r}_i}{dt^2} = m \ddot{\mathbf{r}}_i = \mathbf{F}_i
    \]

- Sample application: 2D motion in central force field
  \[
  m \ddot{x} = \mathbf{F} \cdot \hat{e}_x = -f(r) \hat{r} \cdot \hat{e}_x = -xf\left(\sqrt{x^2 + y^2}\right)
  \]
  \[
  m \ddot{y} = \mathbf{F} \cdot \hat{e}_y = -f(r) \hat{r} \cdot \hat{e}_y = -yf\left(\sqrt{x^2 + y^2}\right)
  \]
  - Polar coordinates are more natural and convenient

- Constant angular momentum
  \[
  mr^2 \dot{\theta} = \ell
  \]
- Fictitious (centrifugal) force
  \[
  m \ddot{r} = -f(r) + \frac{\ell^2}{mr^3}
  \]
Generalized Coordinates

- Any convenient coordinates for description of particular system
  - use $q_i$ as symbol for general coordinate
  - examples
    - diatomic $\{q_1, \ldots, q_6\} = \{x_{com}, y_{com}, z_{com}, r_{12}, \theta, \phi\}$
    - 2-D motion in central field $\{q_1, q_2\} = \{r, \theta\}$

- Kinetic energy
  - general quadratic form
    $$K = M_0(q) + \sum M_j(q)\dot{q}_j + \frac{1}{2} \sum \sum M_{jk}(q)\dot{q}_j \dot{q}_k$$
    - examples
      - rotating diatomic
        $$K = \frac{1}{2} m \left( \dot{q}_1^2 + \dot{q}_2^2 + \dot{q}_3^2 \right) + \frac{1}{8} m \left[ \dot{r}^2 + r^2 \dot{\theta}^2 + (r \sin \theta)^2 \dot{\phi}^2 \right]$$
      - 2-D central motion
        $$K = \frac{1}{2} m \left( \dot{r}^2 + r^2 \dot{\theta}^2 \right)$$
Lagrangian Formulation

- Independent of coordinate system
- Define the Lagrangian
  \[ L(q, \dot{q}) \equiv K(q, \dot{q}) - U(q) \]
- Equations of motion
  \[ \frac{d}{dt} \left( \frac{\partial L}{\partial \dot{q}_j} \right) - \frac{\partial L}{\partial q_j} = 0 \quad j = 1 \ldots N \]
  \[ \text{• } N \text{ second-order differential equations} \]
- Central-force example
  \[ L = \frac{1}{2} m \left( \dot{r}^2 + r^2 \dot{\theta}^2 \right) - U(r) \]
  \[ \frac{d}{dt} \left( \frac{\partial L}{\partial \dot{r}} \right) = \frac{\partial L}{\partial r} \quad \Rightarrow \quad m \ddot{r} = mr\dot{\theta}^2 - f(r) \]
  \[ \frac{d}{dt} \left( \frac{\partial L}{\partial \dot{\theta}} \right) = \frac{\partial L}{\partial \theta} \quad \Rightarrow \quad \frac{d}{dt} \left( mr^2 \dot{\theta} \right) = 0 \]
  \[ \vec{F}_r = -\nabla_r U = -f(r) \]
Hamiltonian Formulation 1. Motivation

- Appropriate for application to statistical mechanics and quantum mechanics
- Newtonian and Lagrangian viewpoints take the $q_i$ as the fundamental variables
  - $N$-variable configuration space
  - $q_i$ appears only as a convenient shorthand for $dq/dt$
  - working formulas are 2nd-order differential equations
- Hamiltonian formulation seeks to work with 1st-order differential equations
  - $2N$ variables
  - treat the coordinate and its time derivative as independent variables
  - appropriate quantum-mechanically
Hamiltonian Formulation 2. Preparation

- Mathematically, Lagrangian treats $q$ and $\dot{q}$ as distinct
  - $L(q_j, \dot{q}_j, t)$
  - identify the generalized momentum as $p_j = \frac{\partial L}{\partial \dot{q}_j}$
  - e.g. if $L = K - U = \frac{1}{2}m\dot{q}^2 - U(q)$; $p = \frac{\partial L}{\partial \dot{q}} = m\dot{q}$
  - Lagrangian equations of motion $\frac{dp_j}{dt} = \frac{\partial L}{\partial q_j}$
- We would like a formulation in which $p$ is an independent variable
  - $p_i$ is the derivative of the Lagrangian with respect to $\dot{q}_i$, and we’re looking to replace $\dot{q}_i$ with $p_i$
  - we need ...?
Hamiltonian Formulation 3. Definition

- ...a Legendre transform!
- Define the Hamiltonian, $H$

\[
H(q,p) = - \left[ L(q,\dot{q}) - \sum p_j \dot{q}_j \right] = -K(q,q) + U(q) + \sum \frac{\partial K}{\partial \dot{q}_j} \dot{q}_j = -\sum a_j \dot{q}_j^2 + U(q) + \sum (2a_j \dot{q}_j) \dot{q}_j = +\sum a_j \dot{q}_j^2 + U(q) = K + U
\]

- $H$ equals the total energy (kinetic plus potential)
Hamiltonian Formulation 4. Dynamics

○ Hamilton’s equations of motion
  • From Lagrangian equations, written in terms of momentum

Differential change in L
\[ dL = \frac{\partial L}{\partial q} dq + \frac{\partial L}{\partial \dot{q}} d\dot{q} \]
\[ = \dot{p} dq + \dot{q} dp \]

Legendre transform
\[ H = -(L - p\dot{q}) \]
\[ dH = -(\dot{p} dq - \dot{q} dp) \]
\[ dH = -\dot{p} dq + \dot{q} dp \]

Lagrange’s equation of motion
\[ \frac{dp}{dt} = \dot{p} = \frac{\partial L}{\partial q} \]
Definition of momentum
\[ p = \frac{\partial L}{\partial \dot{q}} \]

Hamilton’s equations of motion
\[ \dot{q} = +\frac{\partial H}{\partial p} \]
\[ \dot{p} = -\frac{\partial H}{\partial q} \]

Conservation of energy
\[ \frac{dH}{dt} = -\dot{p} \frac{dq}{dt} + \dot{q} \frac{dp}{dt} = -\dot{p}q + \dot{q}\dot{p} = 0 \]
Hamiltonian Formulation 5. Example

- Particle motion in central force field

\[ H = K + U = \frac{p_r^2}{2m} + \frac{p_\theta^2}{2mr^2} + U(r) \]

Lagrange’s equations

\[
\begin{align*}
\dot{q} &= + \frac{\partial H}{\partial p} \\
\dot{p} &= - \frac{\partial H}{\partial q}
\end{align*}
\]

\[
\begin{align*}
(1) \quad \frac{dr}{dt} &= \frac{p_r}{m} \\
(2) \quad \frac{d\theta}{dt} &= \frac{p_\theta}{mr^2} \\
(3) \quad \frac{dp_r}{dt} &= \frac{p_\theta^2}{mr^3} - f(r) \\
(4) \quad \frac{dp_\theta}{dt} &= 0
\end{align*}
\]

\[ \ddot{r} = -\nabla_r U = -f(r) \]

- Equations no simpler, but theoretical basis is better
Phase Space (again)

Return to the complete picture of phase space

- full specification of microstate of the system is given by the values of all positions and all momenta of all atoms
  \[ \Gamma = (p^N, r^N) \]

- view positions and momenta as completely independent coordinates
  \[ \Gamma \text{ connection between them comes only through equation of motion} \]

Motion through phase space

- helpful to think of dynamics as “simple” movement through the high-dimensional phase space
  \[ \Gamma \text{ facilitate connection to quantum mechanics} \]
  \[ \Gamma \text{ basis for theoretical treatments of dynamics} \]
  \[ \Gamma \text{ understanding of integrators} \]
Integration Algorithms

Equations of motion in cartesian coordinates

\[
\begin{align*}
\frac{d\mathbf{r}_j}{dt} &= \frac{\mathbf{p}_j}{m} \\
\frac{d\mathbf{p}_j}{dt} &= \mathbf{F}_j
\end{align*}
\]

\[
\mathbf{r} = (r_x, r_y) \quad \mathbf{p} = (p_x, p_y)
\]

\[
\mathbf{F}_j = \sum_{i=1}^{N} \mathbf{F}_{ij} \quad \text{pairwise additive forces}
\]

Desirable features of an integrator

- minimal need to compute forces (a very expensive calculation)
- good stability for large time steps
- good accuracy
- conserves energy and momentum
- time-reversible
- area-preserving (symplectic)

More on these later
Verlet Algorithm

1. Equations

- Very simple, very good, very popular algorithm
- Consider expansion of coordinate forward and backward in time

\[
\begin{align*}
\mathbf{r}(t + \delta t) &= \mathbf{r}(t) + \frac{1}{m} \mathbf{p}(t) \delta t + \frac{1}{2m} \mathbf{F}(t) \delta t^2 + \frac{1}{3!} \mathbf{\ddot{r}}(t) \delta t^3 + O(\delta t^4) \\
\mathbf{r}(t - \delta t) &= \mathbf{r}(t) - \frac{1}{m} \mathbf{p}(t) \delta t + \frac{1}{2m} \mathbf{F}(t) \delta t^2 - \frac{1}{3!} \mathbf{\ddot{r}}(t) \delta t^3 + O(\delta t^4)
\end{align*}
\]

- Add these together

\[
\mathbf{r}(t + \delta t) + \mathbf{r}(t - \delta t) = 2\mathbf{r}(t) + \frac{1}{m} \mathbf{F}(t) \delta t^2 + O(\delta t^4)
\]

- Rearrange

\[
\mathbf{r}(t + \delta t) = 2\mathbf{r}(t) - \mathbf{r}(t - \delta t) + \frac{1}{m} \mathbf{F}(t) \delta t^2 + O(\delta t^4)
\]

- *update without ever consulting velocities!*
Verlet Algorithm 2. Flow diagram

One MD Cycle

Configuration \( r(t) \)
Previous configuration \( r(t-dt) \)

Compute forces \( F(t) \) on all atoms using \( r(t) \)

Advance all positions according to \( r(t+\delta t) = 2r(t)-r(t-\delta t)+F(t)/m \, dt^2 \)

Add to block sum

End of block?

No

Yes

Block averages

Entire Simulation

Initialization

Reset block sums

New configuration

1 move per cycle

Add to block sum

cycles per block

Compute block average

blocks per simulation

Compute final results
Verlet Algorithm 2. Flow Diagram

Given current position and position at end of previous time step
### Verlet Algorithm 2. Flow Diagram

<table>
<thead>
<tr>
<th>t-δt</th>
<th>t</th>
<th>t+δt</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td></td>
<td></td>
</tr>
<tr>
<td>v</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Compute the force at the current position

*Schematic from Allen & Tildesley, *Computer Simulation of Liquids*
Verlet Algorithm 2. Flow Diagram

Compute new position from present and previous positions, and present force

Schematic from Allen & Tildesley, Computer Simulation of Liquids
Verlet Algorithm  2. Flow Diagram

Advance to next time step, repeat

Schematic from Allen & Tildesley, Computer Simulation of Liquids
Forces 1. Formalism

- Force is the gradient of the potential

\[ \mathbf{F}_{2 \rightarrow 1} = -\nabla u(r_{12}) \]

For force on 1 due to 2:

\[ \mathbf{F}_{2 \rightarrow 1} = -\frac{\partial u(r_{12})}{\partial x_1} \mathbf{e}_x - \frac{\partial u(r_{12})}{\partial y_1} \mathbf{e}_y \]

\[ = -\frac{du(r_{12})}{dr_{12}} \left[ \frac{\partial r_{12}}{\partial x_1} \mathbf{e}_x + \frac{\partial r_{12}}{\partial y_1} \mathbf{e}_y \right] \]

\[ = -\frac{f(r_{12})}{r_{12}} \left[ x_{12} \mathbf{e}_x + y_{12} \mathbf{e}_y \right] \]

\[ \mathbf{F}_{2 \rightarrow 1} = -\mathbf{F}_{1 \rightarrow 2} \]

\[ r_{12} = \left[ (x_2 - x_1)^2 + (y_2 - y_1)^2 \right]^{1/2} \]
Forces 2. LJ Model

- Force is the gradient of the potential

\[ F_{2 \rightarrow 1} = -\frac{f(r_{12})}{r_{12}} \left[ x_{12} e_x + y_{12} e_y \right] \]

e.g., Lennard-Jones model

\[ u(r) = 4\varepsilon \left[ \left( \frac{\sigma}{r} \right)^{12} - \left( \frac{\sigma}{r} \right)^{6} \right] \]

\[ f(r) = -\frac{du}{dr} \]

\[ = +\frac{48\varepsilon}{\sigma} \left[ \left( \frac{\sigma}{r} \right)^{13} - \frac{1}{2} \left( \frac{\sigma}{r} \right)^{7} \right] \]

\[ F_{2 \rightarrow 1} = -\frac{48\varepsilon}{\sigma^2} \left[ \frac{\sigma}{r_{12}} \right]^{14} - \frac{1}{2} \left( \frac{\sigma}{r_{12}} \right)^{8} \left[ x_{12} e_x + y_{12} e_y \right] \]

\[ r_{12} = \left[ (x_2 - x_1)^2 + (y_2 - y_1)^2 \right]^{1/2} \]
Verlet Algorithm. 4. Loose Ends

- **Initialization**
  - *how to get position at “previous time step” when starting out?*
  - *simple approximation*
    \[ r(t_0 - \delta t) = r(t_0) - v(t_0)\delta t \]

- **Obtaining the velocities**
  - *not evaluated during normal course of algorithm*
  - *needed to compute some properties, e.g.*
    - temperature
    - diffusion constant
  - *finite difference*
    \[ v(t) = \frac{1}{2\delta t}[r(t + \delta t) - r(t - \delta t)] + O(\delta t^2) \]
Verlet Algorithm 5. Performance Issues

- **Time reversible**
  - *forward time step*
    \[ \mathbf{r}(t + \delta t) = 2\mathbf{r}(t) - \mathbf{r}(t - \delta t) + \frac{1}{m} \mathbf{F}(t) \delta t^2 \]
  - *replace \( \delta t \) with \(-\delta t\)*
    \[ \mathbf{r}(t + (-\delta t)) = 2\mathbf{r}(t) - \mathbf{r}(t - (-\delta t)) + \frac{1}{m} \mathbf{F}(t)(-\delta t)^2 \]
    \[ \mathbf{r}(t - \delta t) = 2\mathbf{r}(t) - \mathbf{r}(t + \delta t) + \frac{1}{m} \mathbf{F}(t) \delta t^2 \]
  - *same algorithm, with same positions and forces, moves system backward in time*

- **Numerical imprecision of adding large/small numbers**

\[
\mathbf{r}(t + \delta t) - \mathbf{r}(t) = \mathbf{r}(t) - \mathbf{r}(t - \delta t) + \frac{1}{m} \mathbf{F}(t) \delta t^2
\]

\( \mathcal{O}(\delta t^1) \quad \mathcal{O}(\delta t^1) \quad \mathcal{O}(\delta t^1) \quad \mathcal{O}(\delta t^0) \quad \mathcal{O}(\delta t^0) \quad \mathcal{O}(\delta t^0) \quad \mathcal{O}(\delta t^2) \)
Initial Velocities
(from Lecture 3)

- Random direction
  - randomize each component independently
  - randomize direction by choosing point on spherical surface

- Magnitude consistent with desired temperature. Choices:
  - Maxwell-Boltzmann: \( \text{prob}(v_x) \propto \exp\left(-\frac{1}{2}mv_x^2 / kT \right) \)
  - Uniform over (-1/2, +1/2), then scale so that \( \frac{1}{N} \sum v_{i,x}^2 = kT / m \)
  - Constant at \( v_x = \pm \sqrt{kT / m} \)
  - Same for y, z components

- Be sure to shift so center-of-mass momentum is zero

\[
P_x \equiv \frac{1}{N} \sum p_{i,x}
\]

\( p_{i,x} \rightarrow p_{i,x} - P_x \)
Leapfrog Algorithm

- Eliminates addition of small numbers $O(\delta t^2)$ to differences in large ones $O(\delta t^0)$
- Algorithm
  \[
  \mathbf{r}(t + \delta t) = \mathbf{r}(t) + \mathbf{v}(t + \frac{1}{2} \delta t) \delta t
  \]
  \[
  \mathbf{v}(t + \frac{1}{2} \delta t) = \mathbf{v}(t - \frac{1}{2} \delta t) + \frac{1}{m} \mathbf{F}(t) \delta t
  \]
Leapfrog Algorithm

- Eliminates addition of small numbers $O(\delta t^2)$ to differences in large ones $O(\delta t^0)$

- Algorithm

\[
\begin{align*}
\mathbf{r}(t + \delta t) &= \mathbf{r}(t) + \mathbf{v}(t + \frac{1}{2} \delta t) \delta t \\
\mathbf{v}(t + \frac{1}{2} \delta t) &= \mathbf{v}(t - \frac{1}{2} \delta t) + \frac{1}{m} \mathbf{F}(t) \delta t
\end{align*}
\]

- Mathematically equivalent to Verlet algorithm

\[
\begin{align*}
\mathbf{r}(t + \delta t) &= \mathbf{r}(t) + \left[ \mathbf{v}(t - \frac{1}{2} \delta t) + \frac{1}{m} \mathbf{F}(t) \delta t \right] \delta t
\end{align*}
\]
Leapfrog Algorithm

- Eliminates addition of small numbers $O(\delta t^2)$ to differences in large ones $O(\delta t^0)$

- Algorithm

\[
\begin{align*}
\mathbf{r}(t + \delta t) &= \mathbf{r}(t) + \mathbf{v}(t + \frac{1}{2} \delta t) \delta t \\
\mathbf{v}(t + \frac{1}{2} \delta t) &= \mathbf{v}(t - \frac{1}{2} \delta t) + \frac{1}{m} \mathbf{F}(t) \delta t \\
\mathbf{r}(t + \delta t) &= \mathbf{r}(t) + \left[ \mathbf{v}(t - \frac{1}{2} \delta t) + \frac{1}{m} \mathbf{F}(t) \delta t \right] \delta t
\end{align*}
\]

- Mathematically equivalent to Verlet algorithm

\[
\mathbf{r}(t) = \mathbf{r}(t - \delta t) + \mathbf{v}(t - \frac{1}{2} \delta t) \delta t
\]

\(\mathbf{r}(t)\) as evaluated from previous time step
Leapfrog Algorithm

- Eliminates addition of small numbers $O(\delta t^2)$ to differences in large ones $O(\delta t^0)$

- Algorithm

\[
\mathbf{r}(t + \delta t) = \mathbf{r}(t) + \mathbf{v}(t) + \frac{1}{2} \delta t \mathbf{v}(t) + \frac{1}{2} \mathbf{F}(t) \delta t
\]

\[
\mathbf{v}(t + \frac{1}{2} \delta t) = \mathbf{v}(t) + \frac{1}{2} \delta t \mathbf{v}(t) + \frac{1}{m} \mathbf{F}(t) \delta t
\]

- Mathematically equivalent to Verlet algorithm

\[
\mathbf{r}(t + \delta t) = \mathbf{r}(t) + \left( \mathbf{v}(t) - \frac{1}{2} \delta t \mathbf{v}(t) + \frac{1}{m} \mathbf{F}(t) \delta t \right) \delta t
\]

\[
\mathbf{r}(t) = \mathbf{r}(t - \delta t) + \mathbf{v}(t) - \frac{1}{2} \delta t \mathbf{v}(t) - \frac{1}{2} \delta t \mathbf{r}(t - \delta t) + \frac{1}{2} \delta^2 \mathbf{F}(t) \delta t + \frac{1}{2} \delta^2 \mathbf{F}(t - \delta t) \delta t
\]

\[
\mathbf{r}(t + \delta t) = \mathbf{r}(t) + \left( \mathbf{r}(t) - \mathbf{r}(t - \delta t) \right) + \frac{1}{m} \mathbf{F}(t) \delta t^2
\]
Leapfrog Algorithm

- Eliminates addition of small numbers $O(\delta t^2)$ to differences in large ones $O(\delta t^0)$
- Algorithm

$$
\mathbf{r}(t + \delta t) = \mathbf{r}(t) + \mathbf{v}(t + \frac{1}{2} \delta t)\delta t
$$

$$
\mathbf{v}(t + \frac{1}{2} \delta t) = \mathbf{v}(t - \frac{1}{2} \delta t) + \frac{1}{m} \mathbf{F}(t)\delta t
$$

- Mathematically equivalent to Verlet algorithm

$$
\mathbf{r}(t + \delta t) = \mathbf{r}(t) + \left[\mathbf{v}(t - \frac{1}{2} \delta t) + \frac{1}{m} \mathbf{F}(t)\delta t\right]\delta t
$$

$\mathbf{r}(t)$ as evaluated from previous time step

$$
\mathbf{r}(t) = \mathbf{r}(t - \delta t) + \mathbf{v}(t - \frac{1}{2} \delta t)\delta t
$$

$$
\mathbf{r}(t + \delta t) = \mathbf{r}(t) + \left[(\mathbf{r}(t) - \mathbf{r}(t - \delta t)) + \frac{1}{m} \mathbf{F}(t)\delta t^2\right]
$$

$$
\mathbf{r}(t + \delta t) = 2\mathbf{r}(t) - \mathbf{r}(t - \delta t) + \frac{1}{m} \mathbf{F}(t)\delta t^2 \quad \text{original algorithm}
$$
Leapfrog Algorithm 2. Flow Diagram

Given current position, and velocity at last half-step.

Schematic from Allen & Tildesley, *Computer Simulation of Liquids*
Leapfrog Algorithm 2. Flow Diagram

Compute current force

Schematic from Allen & Tildesley, Computer Simulation of Liquids
Leapfrog Algorithm 2. Flow Diagram

Schematic from Allen & Tildesley, *Computer Simulation of Liquids*
Leapfrog Algorithm  2. Flow Diagram

Schematic from Allen & Tildesley, Computer Simulation of Liquids
Leapfrog Algorithm  2. Flow Diagram

Schematic from Allen & Tildesley, Computer Simulation of Liquids
Leapfrog Algorithm. 3. Loose Ends

- **Initialization**
  - *how to get velocity at “previous time step” when starting out?*
  - *simple approximation*
    \[
    \mathbf{v}(t_0 - \frac{1}{2} \delta t) = \mathbf{v}(t_0) - \frac{1}{m} \mathbf{F}(t_0) \frac{1}{2} \delta t
    \]

- **Obtaining the velocities**
  - *interpolate*
    \[
    \mathbf{v}(t) = \frac{1}{2} \left[ \mathbf{v}(t + \frac{1}{2} \delta t) + \mathbf{v}(t - \frac{1}{2} \delta t) \right]
    \]
Velocity Verlet Algorithm

- Roundoff advantage of leapfrog, but better treatment of velocities

- Algorithm

\[
\begin{align*}
\mathbf{r}(t + \delta t) &= \mathbf{r}(t) + \mathbf{v}(t)\delta t + \frac{1}{2m} \mathbf{F}(t)\delta t^2 \\
\mathbf{v}(t + \delta t) &= \mathbf{v}(t) + \frac{1}{2m} [\mathbf{F}(t) + \mathbf{F}(t + \delta t)] \delta t
\end{align*}
\]

- Implemented in stages
  - given current force
  - compute \( \mathbf{r} \) at new time
  - add current-force term to velocity (gives \( \mathbf{v} \) at half-time step)
  - compute new force
  - add new-force term to velocity

- Also mathematically equivalent to Verlet algorithm (in giving values of \( \mathbf{r} \))
### Velocity Verlet Algorithm

#### 2. Flow Diagram

<table>
<thead>
<tr>
<th></th>
<th>t-(\delta t)</th>
<th>t</th>
<th>t+(\delta t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\vec{r})</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\vec{v})</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\vec{F})</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Given current position, velocity, and force

*Schematic from Allen & Tildesley, *Computer Simulation of Liquids*
Velocity Verlet Algorithm
2. Flow Diagram

Schematic from Allen & Tildesley, Computer Simulation of Liquids
Velocity Verlet Algorithm
2. Flow Diagram

Compute velocity at half step

Schematic from Allen & Tildesley, Computer Simulation of Liquids
Velocity Verlet Algorithm

2. Flow Diagram

Schematic from Allen & Tildesley, *Computer Simulation of Liquids*
Velocity Verlet Algorithm
2. Flow Diagram

Compute velocity at full step

*Schematic from Allen & Tildesley, Computer Simulation of Liquids*
Velocity Verlet Algorithm  
2. Flow Diagram

Advance to next time step, repeat

Schematic from Allen & Tildesley, *Computer Simulation of Liquids*
Other Algorithms

- **Predictor-Corrector**
  - *not time reversible*
  - *easier to apply in some instances*
    - → constraints
    - → rigid rotations

- **Beeman**
  - *better treatment of velocities*

- **Velocity-corrected Verlet**
Summary

- Several formulations of mechanics
  - Hamiltonian preferred
    - independence of choice of coordinates
    - emphasis on phase space

- Integration algorithms
  - Calculation of forces
  - Simple Verlet algorithms
    - Verlet
    - Leapfrog
    - Velocity Verlet

- Next up: Calculation of dynamical properties