> **Today's lecture:** how to evolve a system of classical atoms in time using numerical approximations to Newton's equations of motion.

Background

Time evolution by Newton's equations

The time evolution of an atomic system is deterministic and described by Newton's equations:

$$\frac{d\mathbf{p}_i}{dt} = -\frac{dU(\mathbf{r}^N)}{d\mathbf{r}_i}$$

Alternatively,

$$m_i \frac{d^2 \mathbf{r}_i}{dt^2} = -\frac{dU(\mathbf{r}^N)}{d\mathbf{r}_i}$$

Notice that Newton's equations describe a set of $3N$ second-order, nonlinear, coupled partial differential equations for the coordinates of all of the atoms. In this lecture, we will determine numerical approximations to solutions to these equations.

Newton's equations **conserve total energy**:

$$
\begin{aligned}
\frac{dH}{dt} &= \sum \frac{dU}{d\mathbf{r}_i}\frac{d\mathbf{r}_i}{dt} + \sum \frac{d}{dt}\left(\frac{\mathbf{p}_i^2}{2m_i}\right) \\
&= \sum \left(\frac{dU}{d\mathbf{r}_i}\frac{d\mathbf{r}_i}{dt} + \frac{\mathbf{p}_i}{m_i}\frac{d\mathbf{p}_i}{dt}\right) \quad \text{combining the sums} \\
&= \sum \left(\frac{dU}{d\mathbf{r}_i}\frac{d\mathbf{r}_i}{dt} + \frac{d\mathbf{r}_i}{dt}\frac{d\mathbf{p}_i}{dt}\right) \quad \text{using the fact that } \mathbf{p}_i = m_i \frac{d\mathbf{r}_i}{dt} \\
&= \sum \frac{d\mathbf{r}_i}{dt}\left(\frac{dU}{d\mathbf{r}_i} + \frac{d\mathbf{p}_i}{dt}\right) \\
&= \sum \frac{d\mathbf{r}_i}{dt} \times 0 = 0 \quad \text{using Newton's law } \frac{d\mathbf{p}_i}{dt} = -\frac{dU}{d\mathbf{r}_i}
\end{aligned}
$$

Newton's equations also **conserve net linear momentum**:

$$
\begin{aligned}
\frac{d}{dt}\sum \mathbf{p}_i &= \sum \frac{d\mathbf{p}_i}{dt} \\
&= \sum -\frac{dU}{d\mathbf{r}_i}
\end{aligned}
$$

$$= \sum_i \mathbf{f}_i$$
$$= 0$$

This is a form of Newton's third law, which says that for every force a particle exerts on another, there is an equal and opposite force. It can also be shown that these equations **conserve net angular momentum.**

Importantly, Newton's equations of motion are **time reversible**. That is, if we start at one microstate m and proceed to another one n in time Δt, then reversing all of the atomic momenta at n and proceeding according to Newton's equations for Δt will take us along the exact trajectory back to m. We can see time reversibility by defining new coordinates:

$$\mathbf{p}' = -\mathbf{p}$$
$$t' = -t$$

Plugging back into Newton's equations,

$$\frac{-d\mathbf{p}_i'}{-dt'} = -\frac{dU(\mathbf{r}^N)}{d\mathbf{r}_i}$$

The two negatives cancel, leaving the same equation.

Lagrangian and Hamiltonian formulations of the equations of motion

While Newton's equations are valid for Cartesian coordinates, a more general formulation of classical mechanics enables us to develop equations of motion for *any* coordinate system.

Let $\mathbf{q}^N$ and $\mathbf{p}^N$ be **generalized atomic coordinates and conjugate momenta**. The coordinates could be Cartesian, $\mathbf{q}^N = (x_1, y_1, z_1, \dots)$ in which case the momenta conjugate to them would be $\mathbf{p}^N = (p_{x,1}, p_{y,1}, p_{z,1}, \dots)$.

Alternatively, the coordinates could be spherical, $\mathbf{q}^N = (r_1, \theta_1, \phi_1, \dots)$ and the conjugate momenta would be $\mathbf{p}^N = (p_{r,1}, p_{\theta,1}, p_{\phi,1}, \dots)$. What are the equations of motion for this coordinate set? We could derive it by transforming Newtons' equations. More informatively, we can use the Lagrangian formulation of classical mechanics:

$$\mathcal{L}(\mathbf{q}^N, \dot{\mathbf{q}}^N) = K(\dot{\mathbf{q}}^N) - U(\mathbf{q}^N)$$

Here, the Lagrangian is a function of the generalized coordinates and their time derivative. The kinetic energy is expressed as a function of the derivative and the potential energy as a function of the coordinates themselves.

Using the Lagrangian, the equations of motion for any coordinate set are given from the condition:

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}_i} = \frac{\partial \mathcal{L}}{\partial \mathbf{q}_i}$$

For example, in Cartesian coordinates $\dot{\mathbf{q}}^N = \mathbf{v}^N$ and $\mathbf{q}^N = \mathbf{r}^N$,

$$\mathcal{L}(\mathbf{r}^N, \mathbf{v}^N) = \sum \frac{m|\mathbf{v}_i|^2}{2} - U(\mathbf{r}^N)$$

and the equations of motion are:

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \mathbf{v}_i} = \frac{\partial \mathcal{L}}{\partial \mathbf{r}_i}$$

$$\frac{d}{dt} m\mathbf{v}_i = -\frac{\partial U(\mathbf{r}^N)}{d\mathbf{r}_i}$$

The Lagrangian formulation has important advantages over the Cartesian Newtonian formulation. Importantly, it generalizes principles that have deep connections with quantum mechanics.

An alternative to the Lagrangian formulation is the Hamiltonian formulation of the equations of motion. The major distinction is that the latter uses the conjugate momenta $\mathbf{p}^N$ instead of the coordinate time derivatives $\dot{\mathbf{q}}^N$:

$$H(\mathbf{q}^N, \mathbf{p}^N) = K(\mathbf{p}^N) + U(\mathbf{q}^N)$$

Here, as we have seen before, the Hamiltonian simply returns the total energy of the system. Mathematically, the Hamiltonian is a Legendre transform of the Lagrangian.

With these generalized coordinates, the equations of motion can be expressed as:

$$\frac{\partial \mathbf{p}_i}{\partial t} = -\frac{\partial H}{\partial \mathbf{q}_i} \qquad \frac{\partial \mathbf{q}_i}{\partial t} = \frac{\partial H}{\partial \mathbf{p}_i}$$

In Cartesian coordinates, $\mathbf{q}^N = \mathbf{r}^N$ and:

$$\frac{\partial \mathbf{p}_i}{\partial t} = -\frac{\partial U}{\partial \mathbf{r}_i} \qquad \frac{\partial \mathbf{r}_i}{\partial t} = \frac{\partial K}{\partial \mathbf{p}_i}$$

$$\frac{\partial \mathbf{p}_i}{\partial t} = -\mathbf{f}_i \qquad \frac{\partial \mathbf{r}_i}{\partial t} = \frac{\mathbf{p}_i}{m_i}$$

The LHS is just Newton's law and the RHS gives the relationship between the momentum and the velocity.

Why should we care about these alternate formulations of classical mechanics if we can always perform simulations in Cartesian space and thus use Newton's equations?

- It turns out that we do *not* always perform simulations in Cartesian space—for example, systems with **rigid bonds** present require special treatment.

- The introduction of various **thermostats and barostats** are rigorously treated using these formulations, as we shall see in a later lecture.

- **Statistical mechanics** relies on the Hamiltonian formulation as it is the generalized coordinates in which one can most easily construct the classical statistical mechanical integrals.

Phase space

The concept of **phase space** is often discussed with regards to simulation. Phase space denotes a $6N$ dimensional space in which there is an axis for every position and every conjugate momentum. In Cartesian coordinates, this means there are x, y, z and p_x, p_y, p_z axes for each atom. A single microstate corresponds to a point in phase space.

The time-evolution of a system can be described as a trajectory through phase space. For classical systems, the energy is conserved and the **phase space trajectory** adheres to a surface of constant energy.

Systems that are **ergodic** and able to reach equilibrium are able to explore all parts of phase space that have the same energy as the constant total. That is, the trajectory of an ergodic system will visit all points in phase space on the constant energy hypersurface with equal probability. Systems that are non-ergodic have parts of phase space that are said to be **inaccessible**. This means that the system is not able to explore the complete constant energy surface in the time scales of interest.

Technically, in simulation we can never explore exhaustively all accessible regions of phase space. However, we can explore a subset of the accessible regions that are statistically identical or that are **representative**. In this sense, our simulations can be ergodic and reach equilibrium.

A **differential volume element** in phase space is a very small region of $6N$-dimensional volume that spans differential elements $d\mathbf{r}^N d\mathbf{p}^N = dx_1 dy_1 \ldots dz_N dp_{x,1} dp_{y,1} \ldots dp_{z,N}$.

Configuration space is the subset of phase space corresponding to the $3N$ variables $\mathbf{r}^N$.

Microcanonical ensemble and density function

Isolated systems at equilibrium correspond to the microcanonical ensemble. This is the natural setting for microscopic evolution per Newton's equations of motion. These systems do not interact with the outside world and rigorously maintain constant values of total energy E, volume V, and number of atoms N.

Consider starting an isolated system in a particular microstate. As the system evolves and approaches equilibrium, the total energy E remains constant, at the same value that it had for the initial microstate. The **principle of equal a priori probabilities** then states that the system will ultimately visit all of the microstates with that value of E with the same frequency. That is, the system will spend an equal amount of time in each of its microstates with the same energy.

The function $\Omega(E, V, N)$ counts the number of microstates for N atoms in volume V that have energy E, and is called the **density of states** or **microcanonical partition function**. It is specific to the particular system or substance of interest. For classical systems, it is given by

$$\Omega(E, V, N) = \frac{1}{h^{3N} N!} \int \int \dots \int \delta[H(\mathbf{p}^N, \mathbf{r}^N) - E] \, d\mathbf{p}_1 d\mathbf{p}_2 \dots d\mathbf{p}_N d\mathbf{r}_1 d\mathbf{r}_2 \dots d\mathbf{r}_N$$

$$= \frac{1}{h^{3N} N!} \int \delta[H(\mathbf{p}^N, \mathbf{r}^N) - E] \, d\mathbf{p}^N d\mathbf{r}^N$$

The notation $d\mathbf{p}$ indicates $dp_x dp_y dp_z$ such that the integral over $d\mathbf{p}$ for one particle is actually a three-dimensional integral. A similar case exists for $d\mathbf{r}$. Therefore, the total number of integrals is $6N$. We say that this is a $6N$-dimensional integral.

We need to comment a little bit on the limits of integration:

- For each momentum variable, the limits are $-\infty < p < \infty$. In other words, the momenta can take on any real value. These degrees of freedom and the associated integrals are **unbounded**.

- For each position variable, the values it can take on depend on the volume of the container into which it was placed. For a cubic box of side length L, one has $0 < r < L$. Therefore these position variables are **bounded** and the integrals are definite. The limits of the position integrals also introduce a dependence of the partition function on V.

Here, the delta function is a Dirac delta function. In simple terms, the Dirac delta function is infinitely peaked around the point where its argument is zero, but zero otherwise. Here, it "selects out" or "filters" those configurations that have the same energy as the specified energy. It can be shown that evaluating the multidimensional integral with the delta function is the same as finding the $6N$ dimensional area in the space of all of the degrees of freedom where the energy equals specified energy.

According to the law of equal a priori probabilities, an isolated system at equilibrium spends an equal amount of time in each of its $\Omega(E, V, N)$ microstates. The probabilities of each microstate can be written compactly as:

$$\wp(\mathbf{p}^N, \mathbf{r}^N) = \frac{\delta[H(\mathbf{p}^N, \mathbf{r}^N) - E]}{\Omega(E, V, N)}$$

where E is the specified total energy (a property of the ensemble).

The **microcanonical partition function** relates to the macroscopic **entropy**:

$$S = k_B \ln \Omega(E, V, N)$$

The entropy can be related to other (macroscopic) thermodynamic properties by the **fundamental equation**,

$$dS = \frac{1}{T} dE + \frac{P}{T} dV - \frac{\mu}{T} dN$$

This equation can also be rearranged to the **energy version**,

$$dE = TdS - PdV + \mu dN$$

These two equations simply summarize the partial derivatives of the functions $S(E, V, N)$ and $E(S, V, N)$, respectively.

In the limit of large numbers of particles, the fluctuating macroscopic properties of the microcanonical ensemble (T, P, μ) appear constant. Thus, for large enough systems, we can apply macroscopic relations to understand this ensemble.

Integrators

In the following sections we derive discrete-time numerical approximations to Newton's equations of motion. For simplicity, we will consider a single coordinate r but keep in mind that the following relations can be generalized to vector Cartesian coordinates $\mathbf{r}^N$.

The basic idea in each of the following is to solve for the trajectory of atoms as a function of time:

$$\mathbf{r}^N(t)$$

This is called a **molecular dynamics** simulation. We actually find the positions at discrete time intervals,

$$\mathbf{r}^N(0), \mathbf{r}^N(\delta t), \mathbf{r}^N(2\delta t), \mathbf{r}^N(3\delta t), \dots$$

Here, δt is called the **time step.** The smaller the values of δt, the more accurate our numerical solution of Newton's equations of motions are. Our approach will to be to take steps forward in time, at each point solving for the position r and the velocity v at the next time step.

Verlet and velocity Verlet algorithms

Consider a Taylor expansion of the position vector in time:

$$r(t + \delta t) = r(t) + \frac{dr(t)}{dt}\delta t + \frac{d^2 r(t)}{dt^2}\frac{\delta t^2}{2} + \frac{d^3 r(t)}{dt^3}\frac{\delta t^3}{6} + \vartheta(\delta t^4)$$
$$= r(t) + v(t)\delta t + \frac{f(t)}{m}\frac{\delta t^2}{2} + \frac{d^3 r(t)}{dt^3}\frac{\delta t^3}{6} + \vartheta(\delta t^4)$$

In the second line, we used Newton's equation of motion to replace the acceleration with the force. Similarly,

$$r(t - \delta t) = r(t) - v(t)\delta t + \frac{f(t)}{m}\frac{\delta t^2}{2} - \frac{d^3 r(t)}{dt^3}\frac{\delta t^3}{6} + \vartheta(\delta t^4)$$

Adding these two equations together and moving the $r(t - \delta t)$ term to the RHS:

$$r(t + \delta t) = 2r(t) - r(t - \delta t) + \frac{f(t)}{m}\delta t^2 + \vartheta(\delta t^4)$$

This equation forms the basis of the **Verlet algorithm** for molecular dynamics. Here, we propagate a system forward in time by a **time step** δt. To do so, we use the positions at the previous two time steps as well as the forces at the current time step. To get the forces, we use the force field and the current position set at time t:

$$f(t) = -\frac{dU(r(t))}{dr}$$

The accuracy of this equation is of order $\vartheta(\delta t^4)$. In other words, smaller time steps increase the accuracy of this discrete time approximation.

The Verlet algorithm does not use the velocity to determine a solution to the atomic positions at the next time step. We can approximate the velocities using:

$$v(t) = \frac{r(t + \delta t) - r(t - \delta t)}{2\delta t} + \vartheta(\delta t^3)$$

This equation can be derived by forming $r(t + \delta t) - r(t - \delta t)$ and using the above Taylor expansions.

One disadvantage of the Verlet algorithm is that it requires us to store in memory two sets of positions, $r(t)$ and $r(t - \delta t)$. An alternative is the so-called **velocity Verlet algorithm**, which is a reformulation of the Verlet algorithm (by manipulating the expansions) that uses the velocity directly:

$$r(t + \delta t) = r(t) + v(t)\delta t + \frac{f(t)}{2m}\delta t^2$$

$$v(t + \delta t) = v(t) + \frac{f(t + \delta t) + f(t)}{2m}\delta t$$

This algorithm is one of the most frequently used in molecular simulations because of its ease of implementation. A general strategy is the following:

1. Given $r(t)$ and $v(t)$ at one time point t, first compute the forces on each atom using the force field.

2. Do an update of the positions $r \leftarrow r + v\delta t + \frac{f}{2m}\delta t^2$.

3. Do a partial update of the velocity array based on the current forces:
$$v \leftarrow v + \frac{f}{2m}\delta t$$

4. Compute the new forces $f(t + \delta t)$ using the new positions $r(t + \delta t)$.

5. Finish the update of the velocity array:
$$v \leftarrow v + \frac{f}{2m}\delta t$$

6. Go back to 1.

Other integrators

Several other integrators exist. The **leap frog** scheme is equivalent to the Verlet algorithm, but solves for the velocities at half time step intervals:

$$v(t + \delta t/2) = v(t - \delta t/2) + \frac{f(t)}{m}\delta t$$

$$r(t + \delta t) = r(t) + v(t + \delta t/2)\delta t$$

One disadvantage of the leap frog approach is that the velocities are not known at the same time as the positions, making it difficult to evaluate the total energy (kinetic + potential) at any one point in time. We can get an estimate from:

$$v(t) = \frac{v(t + \delta t/2) + v(t - \delta t/2)}{2}$$

Higher-order integrations schemes are possible, in which the Taylor expansions are taken out to the δt^3 term or more. The **predictor-corrector** approach, on the other hand, tries to use the new forces after a time step to back-correct for the extrapolation in time. Generally speaking, these are not frequently used. Higher-order methods require additional computational overhead; the same accuracy might be achieved with a lower-order algorithm by shrinking the time step size. The velocity Verlet and leap frog algorithms are often the methods of choice.

Practical considerations

Energy conservation
Rigorously, Newton's equations conserve total energy and therefore our numerical solutions should as well. We can diagnose the accuracy of our solutions by the extent to which the energy remains constant, calculated as the instantaneous sum of the kinetic and potential energies.

Typically it is desirable to conserve energy to one part in 10^4-10^5. That is, we would like to see at least 4-5 significant figures that do not fluctuate over the course of the simulation.

There are often two kinds of energy conservation:

- **short-term** – This refers to the fluctuations in total energy from step to step in our simulation, considered over just a few steps in sequence. The Verlet-like algorithms are actually not as good at short term energy conservation as many higher-order or corrector schemes.

- **long-term** – Over large numbers of time steps, one finds that the total energy can **drift** away from its initial value. The Verlet-like algorithms are actually very good at avoiding energy drift, while many other schemes are not.

Time step considerations
For numerical stability and accuracy in conserving the energy, one typically needs to pick a time step that is at least an order of magnitude smaller than the fastest time scale in the system.

system (fastest motion)	time step (1 fs = 10^{-15} s)
molecules (bond vibrations)	0.5 – 1.0 fs
molecules, rigid bonds (angle bending)	2.0 fs
atoms (translation)	5 – 10 fs
Lennard-Jones system	0.001 (dimensionless units)

Too large a time step can cause a molecular dynamics simulation to become unstable, with the total energy rapidly increasing with time. This behavior is often colloquially termed "exploding" and it is caused by devastating atomic collisions that occur when a large time step propagates the positions of two atoms to be nearly overlapping; the repulsive interactions then create a strong force that propels these atoms apart.
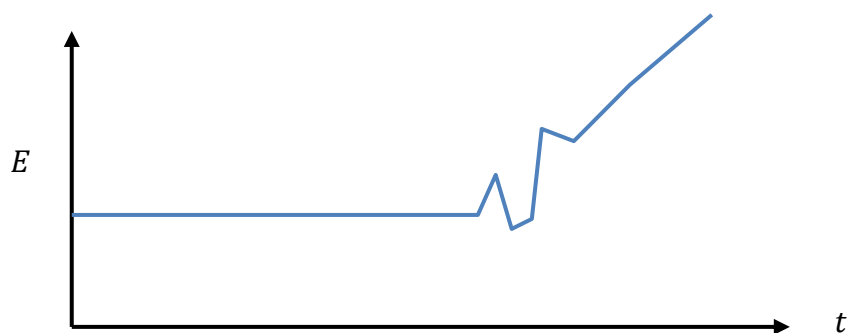
Practically speaking, the time step limits the length of our molecular dynamics trajectory. At each time step, the slowest computational operation is the pairwise loop to compute the potential energy and forces, which scales as N^2 in computational expense. A total molecular dynamics trajectory length of n such time steps is $t_{total} = n\delta t$. The computational time is $t_{CPU} = n\delta t_{CPU}$. Some sample results for a Lennard-Jones system ($\delta t = 0.001$) on a 2.4 GHz processor:

N	t_{CPU} for 1000 time steps
256	4 s
512	12 s
1024	44 s
2048	165 s

Initial positions

At the start of the simulation we must specify initial positions and velocities for each atom. For the initial positions, there are several common approaches:

- For bulk phases, we can start the simulation by placing all of the atoms on a **crystalline lattice**, the simplest being the cubic lattice. If our simulation conditions correspond to the liquid phase, we must first simulate long enough until the system melts.

- Atoms in molecules should be placed in **idealized or approximate geometries** that satisfy the known bond lengths and angles.

- Atoms can be **placed randomly** in the simulation volume. However, this can often be dangerous for large or dense systems, where there is a high probability that we will place two or more atoms too close together and create a strong repulsion between them as a result of **core overlap**.

- The initial configuration can be taken **from experiment** (typically xray or NMR).

In any case, it can be a good idea to first **energy-minimize** the initial configuration to **relax** the system to a nearby local minimum. This can also remove any core overlaps present in the system, and sometimes can fix up bond lengths and angles.

Initial velocities

For the initial velocities, we typically draw these randomly from a Maxwell-Boltzmann distribution at the desired temperature:

$$\wp(v_{x,i}) = \left(\frac{m_i}{2\pi k_B T}\right)^{\frac{1}{2}} \exp\left(-\frac{m_i v_{x,i}^2}{2k_B T}\right)$$

Notice that the form of this distribution is Gaussian with zero mean:

$$\wp(v_{x,i}) = (2\pi\sigma_v^2)^{-\frac{1}{2}} \exp\left(-\frac{v_{x,i}^2}{2\sigma_v^2}\right)$$

$$\sigma_v^2 = \frac{k_B T}{m_i}$$

Random velocities must be drawn for each of the $3N$ components, and are both positive and negative.

However, randomly picking each velocity component this way can lead to a nonzero net momentum of the system, which would create a systematic translational drift of our system. Thus, we typically shift the velocities so as to remove this effect. The procedure is:

1. Pick random velocities on each atom, either from a Maxwell-Boltzmann distribution or from a random uniform distribution.

2. Find the net momentum: $\mathbf{P} = \sum m_i \mathbf{v}_i$.

3. Subtract from each of the atomic velocities so as to remove the net momentum:
$$\mathbf{v}_i \leftarrow \mathbf{v}_i - \mathbf{P}/Nm_i$$

For systems without boundaries or fields, we typically also remove the net angular momentum, which prevents our system from rotating.

Analysis of integrators

Lyapunov instability

How well does our numerical solution to Newton's equations of motion $\mathbf{r}^N(t)$ approximate the true solution, $\mathbf{r}_{\text{true}}^N(t)$? One way of examining this issue is to quantify how sensitive our trajectories are to small errors in the positions or momenta at each point in time.

Consider two trajectories that both start at $\mathbf{r}_1^N(t=0) = \mathbf{r}_2^N(t=0)$. The first trajectory has momenta $\mathbf{p}_1^N(t=0)$ and the second has each of the momenta shifted by a small amount $\mathbf{p}_2^N(t=0) = \mathbf{p}_1^N(t=0) + \delta p$. It can be shown that the distance between the two trajectories in phase space grows exponentially in time according to a **Lyapunov divergence**:

$$|\mathbf{r}_1^N(t) - \mathbf{r}_2^N(t)| \sim (\delta p) \exp(\lambda t)$$

Here, λ is called the **Lyapunov exponent.** This relationship shows that very small deviations in the starting conditions of a trajectory result in huge differences in the trajectories at long times. These small initial condition differences cause an **exponential divergence** of the trajectories. The cause of this behavior is the **chaotic** behavior of many-body mechanical systems. The theory of chaos in dynamical systems constitutes a major research effort.

There are two consequences to this behavior:

- It is difficult to reproduce exact trajectories in molecular simulation. Care must be taken that *all* of the initial conditions are exactly the same, down to every bit of precision. Even differences in velocities of 1 in 10^{10} can result in major differences in the trajectories after only 100 time steps.

- It is impossible for us to reproduce the true Newtonian solution exactly, even as we take very small time steps, $\delta t \to 0$. Small differences in the predicted positions and momenta with each time step, relative to the true solution, will ultimately result in big deviations at longer times.

Time-reversible and area-preserving integrators

The Lyapunov analysis would seem to suggest that we cannot use numerical integration to solve effectively Newton's equations of motion. However, this is not a problem because the properties in which we are interested never really depend on specific, single, short trajectories, but rather

- statistical properties of long simulations, i.e., average behavior over time

- statistical properties many independent simulations, averaged over different initial velocities and, sometimes also, different initial configurations

Ultimately the reason for our *statistical* rather than *deterministic* interest is because it is the statistics that relate to experiment (multiple measurements, or measurements over periods of time) and to thermodynamics (statistical mechanics).

What we want are numerical solutions to Newton's equations that maintain the correct statistical sampling of phase space. In other words, we want our numerical solution to populate different parts of phase space with the same probabilities that would be observed for the true solution.

Two properties of numerical integration schemes will maintain correct statistical sampling:

- **time reversibility** – Newton's equations are time reversible. We would like integrators that preserve this property in our numerical trajectories. In other words, reversing the velocities should result in a trajectory that re-traces itself. The Verlet and Verlet-derived integrators all satisfy this property.

- **area preserving** – This is a subtle feature of the equations of motion. Imagine launching a large (essentially infinite) number of trajectories a particular energy hypersurface in phase space, $E = E_0$. The initial conditions all have the same energy, and the collection of points can be used to define a $6N - 1$ dimensional area of the E_0 hypersurface. As we simultaneously and independently evolve all of these trajectories in time, we expect the area they define after some time t to remain the same, since the area of the E_0 hypersurface does not change. If the area does change (typically grows larger), we have moved to another hypersurface $E \neq E_0$ and energy is not conserved at very long time, i.e., there is an energy drift. The Verlet and Verlet-derived integrators can be shown to be **area preserving** (by way of computing a Jacobian of the coordinate and velocity updates for time steps).

Some integrators are not time reversible or area preserving. These should be avoided.

Since the original formulations of molecular dynamics algorithms, a rigorous theory has been developed for deriving time-reversible and area-preserving integrators to arbitrary order. This formalism is called the **Loiuville formulation** of integration schemes [Tuckerman, Berne, and Martyna, J. Chem. Phys. 97, 1990 (1992)]. This formulation has been central to the development of advanced molecular dynamics implementations (such as the constant-temperature methods we will cover later).

Discontinuous potentials

Oftentimes we are interested in simple, coarse-grained systems that are modeled with pairwise interatomic potentials that have discontinuities in their derivatives.

hard sphere potential

square well potential sphere
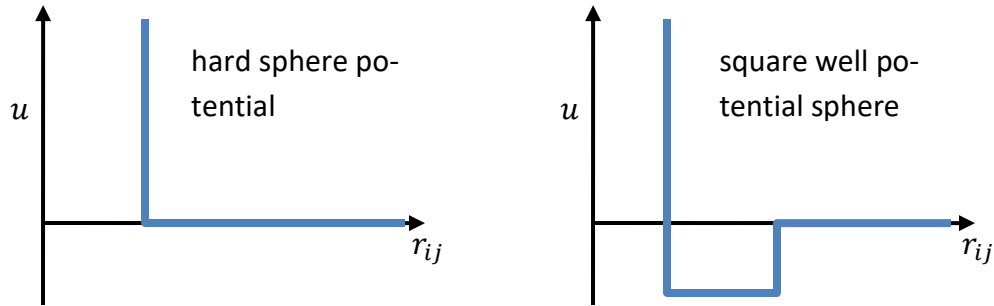
For these systems, we cannot use Newton's equations of motions as written above because the forces are either zero or infinite. Instead, we need to perform a **discontinuous molecular dynamics** simulation.

The basic approach is the following:

1. At time t, examine all pairwise interactions and predict for each, using the velocities, the point in time Δt_{ij} at which two atoms will first cross over a distance that corresponds to a force discontinuity in the potential. For the hard sphere system, this simply amounts to finding the times at which each pair of particles will first collide.

2. Find the minimum $\Delta t = \min_{ij} \Delta t_{ij}$.

3. Propagate all particles according to their velocities a time Δt.

4. For the two particles corresponding to the minimum $\Delta t_{ij} (= \Delta t)$, update their velocities based on conservation of momentum, conservation of energy, and the collision geometry. For hard spheres, this amounts to reversing the components of velocities of each of the two particles along the vector joining them at the collision.

Discontinuous MD can be useful modeling non-dense systems with few intramolecular bond and angle constraints. The time scales that this approach can reach for these are often much longer than continuous MD integration because the time steps Δt can be larger. However, for very dense systems, collisions occur very often and the time steps in discontinuous MD simulations can become extremely short, making this approach inefficient. While discontinuous MD does not require computation of forces, it still requires a pairwise interaction loop that scales as N^2 in computational expense.