Other Ising-like models

One of the possible generalizations of the Ising model is the so called n-vector model. Unlike the Potts model, it describes spins as vectors with n components. This model has applications in modelling magnetism or the Higgs mechanism. The Hamiltonian resembles the one of the Potts model in the sense that it favors spin alignment

$$\mathcal{H} = -J \sum_{\langle i,j \rangle} \vec{S}_i \cdot \vec{S}_j + \vec{H} \sum_i \vec{S}_i. \tag{11}$$

with $\vec{S}_i = (S_i^1, S_i^2, \ldots, S_i^n)$ and $\left|\vec{S}_i\right| = 1$.

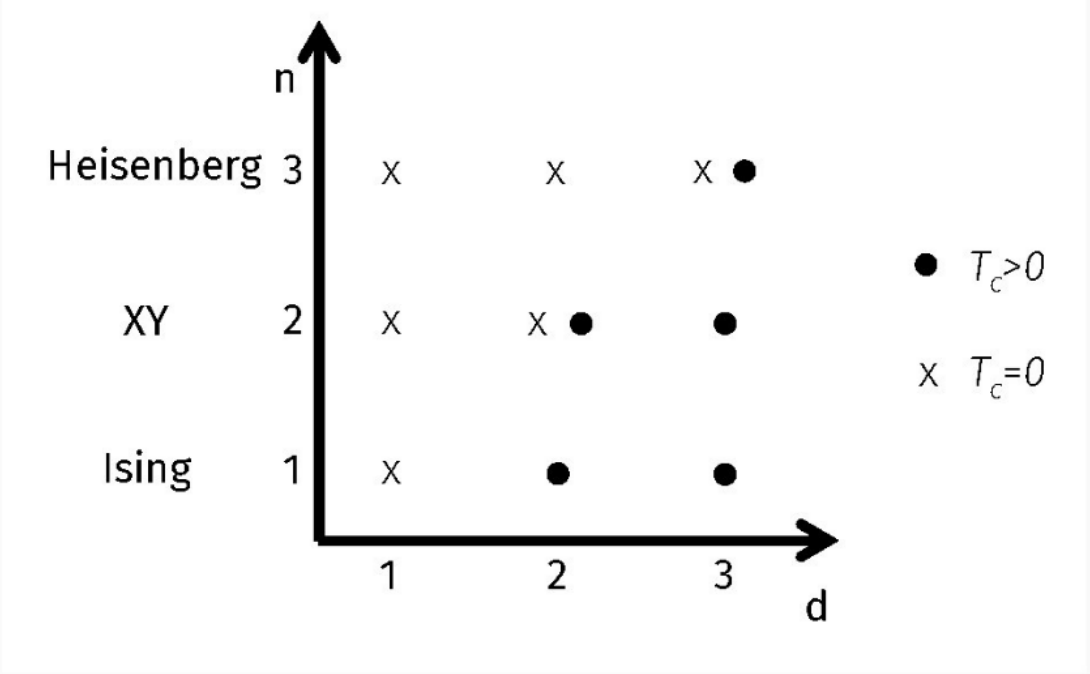


Figure 3: The dependence of the critical temperature on the number of vector components n.

For Monte Carlo simulations with vector-valued spins we have to adapt our simulation methods. The classical strategy is to flip spins by modifying the spin locally trough adding a small $\Delta\vec{S}$ such that $\vec{S}_i' = \vec{S}_i + \Delta\vec{S}$ and $\Delta\vec{S} \perp \vec{S}_i$. The classical Metropolis algorithm can then be used in the same fashion as in the Ising model.

For computing the thermal average

$$\langle Q \rangle = \frac{1}{Z_T} \sum_X Q(X) e^{-\frac{E(X)}{k_B T}}. \tag{12}$$

we need to sample different configurations at different temperatures. Another possibility would be to determine an average at a certain temperature T_0 and extrapolate to another temperature T. In the case of a canonical ensemble, an extrapolation can be achieved by reweighting the histogram of energies $p_{T_0}(E)$ with the Boltzmann factor $e^{\frac{E}{T} - \frac{E}{T_0}}$.

Such histogram methods have first been described by Salzburg et al. in 1959. We now reformulate the computation of the thermal average of a quantity Q and of the partition function as a sum over all possible energies instead of over all possible configurations and find

$$Q(T_0) = \frac{1}{Z_{T_0}} \sum_E Q(E) p_{T_0}(E) \quad \text{with} \quad Z_{T_0} = \sum_E p_{T_0}(E), \tag{13}$$

where $p_{T_0}(E) = g(E) e^{-\frac{E}{k_B T_0}}$ with $g(E)$ defining the *degeneracy of states*, i.e., the number of states with energy E.

This takes into account the fact that multiple configurations can have the same energy. The goal is to compute the quantity Q at another temperature T

$$Q(T) = \frac{1}{Z_T} \sum_E Q(E) p_T(E). \tag{14}$$

The degeneracy of states contains all the information needed. Using the definition of $g(E)$ yields

$$p_T(E) = g(E) e^{-\frac{E}{k_B T}} = p_{T_0}(E) \exp\left[-\frac{E}{k_B T} + \frac{E}{k_B T_0}\right] \tag{15}$$

and with $f_{T_0,T}(E) = \exp\left[-\frac{E}{k_B T} + \frac{E}{k_B T_0}\right]$ we finally obtain

$$Q(T) = \frac{\sum_E Q(E) p_{T_0}(E) f_{T_0,T}(E)}{\sum_E p_{T_0}(E) f_{T_0,T}(E)}. \tag{16}$$

Broad histogram methods

Let N_{up} and N_{down} be the numbers of processes which lead to an increasing and decreasing energy, respectively. Furthermore, we have to keep in mind that the degeneracy of states increases exponentially with energy E since the number of possible configurations increases with energy. To explore all energy regions equally, we find a condition equivalent to the one of detailed balance, i.e.,

$$g\left(E+\Delta E\right)N_{\mathrm{down}}\left(E+\Delta E\right)=g\left(E\right)N_{\mathrm{up}}\left(E\right). \qquad (17)$$

The motion in phase space towards higher energies can then be penalized with a Metropolis-like dynamics:

- Choose a new configuration,
- if the new energy is lower, accept the move,
- if the new energy is higher then accept with probability $\frac{N_{\mathrm{down}}(E+\Delta E)}{N_{\mathrm{up}}(E)}$.

We obtain the function $g(E)$ by taking the logarithm of Eq. (17) and divide by ΔE

$$\log\left[g\left(E+\Delta E\right)\right]-\log\left[g\left(E\right)\right]=-\log\left[N_{up}\left(E\right)\right]-\log\left[N_{down}\left(E+\Delta E\right)\right]. \qquad (18)$$

In the limit of small energy differences, we can approximate the latter equation by

$$\frac{\partial\log\left[g\left(E\right)\right]}{\partial E}=\frac{1}{\Delta E}\log\left[\frac{N_{\mathrm{up}}\left(E\right)}{N_{\mathrm{down}}\left(E+\Delta E\right)}\right] \qquad (19)$$

which we can numerically integrate to obtain $g\left(E\right)$.

Distributions of N_{up} and N_{down} can be obtained by keeping track of these numbers for each configuration at a certain energy. In addition, we also need to store the values of the quantity $Q\left(E\right)$ we wish to compute as a thermal average according to

$$Q\left(T\right)=\frac{\sum_{E}Q\left(E\right)g\left(E\right)e^{-\frac{E}{k_{B}T}}}{\sum_{E}g\left(E\right)e^{-\frac{E}{k_{B}T}}}. \qquad (20)$$

Based on a known degeneracy of states $g\left(E\right)$, we can now compute quantities at any temperature.

Renormalization and free energy

To build some intuition for renormalization approaches, we consider a scale transformation of the characteristic length L of our system with that leads to a rescaled characteristic length $\tilde{L} = L/l$

Moreover, we consider the partition function of an Ising system. A scale transformation with $\tilde{L} = L/l$ leaves the partition function

$$Z = \sum_{\{\sigma\}} e^{-\beta H} \tag{1}$$

and the corresponding free energy invariant.

Free energy density of the system also stays invariant under scale transformations. Since the free energy F is an *extensive* quantity[1], it scales with the system size and

$$F(\epsilon, H) \quad = l^{-d} \tilde{F}\left(\tilde{\epsilon}, \tilde{H}\right) \text{ with } \quad \epsilon = T - T_c, \tag{2}$$

where $\tilde{F}$ is the renormalized free energy.

We can rescale previous equation by setting

$$\tilde{\epsilon} = l^{y_T} \epsilon \quad \text{and} \quad \tilde{H} = l^{y_H} H \tag{3}$$

and obtain in terms of the renormalized free energy

$$\tilde{F}\left(\tilde{\epsilon}, \tilde{H}\right) = \tilde{F}\left(l^{y_T}\epsilon, l^{y_H} H\right). \tag{4}$$

Since renormalization also affects the correlation length

$$\xi \sim |T - T_c|^{-\nu} = |\epsilon|^{-\nu} \tag{5}$$

we can relate the critical exponent ν to y_T.

The renormalized correlation length $\tilde{\xi} = \xi/l$ scales as

$$\tilde{\xi} \sim \tilde{\epsilon}^{-\nu}. \tag{6}$$

And due to

$$l^{y_T}\epsilon = \tilde{\epsilon} \sim \epsilon l^{\frac{1}{\nu}}, \tag{7}$$

we find $y_T = 1/\nu$.

The critical point is a fixed point of the transformation since $\epsilon = 0$ at T_c and ϵ does not change indepedent of the value of the scaling factor.

Decimation of the one-dimensional Ising model

Generalization

In general, multiple coupling constants are necessary, e.g., in the two-dimensional Ising model. Thus, we have to construct a renormalized Hamiltonian based on multiple renormalized coupling constants, i.e.,

$$\tilde{H} = \sum_{\alpha=1}^{M} \tilde{K_\alpha} \tilde{O}_\alpha \text{ with } \tilde{O}_\alpha = \sum_i \prod_{k \in c_\alpha} \tilde{\sigma}_{i+k} \tag{18}$$

where c_α is the configuration subset over which we renormalize and

$$\tilde{K}_\alpha \left(K_1, \ldots, K_M\right) \quad \text{with} \quad \alpha \in \{1, \ldots, M\}.$$

At T_c there exists a fixed point $K_\alpha^* = K_\alpha \left(K_1^*, \ldots, K_M^*\right)$. A first ansatz to solve this problem is the linearization of the transformation. Thus, we compute the Jacobian $T_{\alpha,\beta} = \frac{\partial \tilde{K}_\alpha}{\partial K_\beta}$ and obtain

$$\tilde{K}_\alpha - K_\alpha^* = \sum_\beta T_{\alpha,\beta}|_{K^*} \left(K_\beta - K_\beta^*\right) \tag{19}$$

To analyze the behavior of the system close to criticality, we consider eigenvalues $\lambda_1, \ldots, \lambda_M$ and eigenvectors $\phi_1, \ldots, \phi_M$ of the linearized transformation defined by Eq. (19). The eigenvectors fulfill $\tilde{\phi_\alpha} = \lambda_\alpha \phi_\alpha$ and the fixed point is unstable if $\lambda_\alpha > 1$.

The largest eigevalue dominates the iteration and we can identify the scaling field $\tilde{\epsilon} = l^{y_T} \epsilon$ with the eigenvector of the transformation, and the scaling factor with eigenvalue $\lambda_T = l^{y_T}$. Then, we compute the exponent ν according to

$$\nu = \frac{1}{y_T} = \frac{\log\left(l\right)}{\log\left(\lambda_T\right)}. \tag{20}$$

Since we are dealing with generalized Hamiltonians with many interaction terms, we compute the thermal average using the operators O_α, i.e.,

$$\langle O_\alpha \rangle = \frac{\sum_{\{\sigma\}} O_\alpha e^{\sum_\beta K_\beta O_\beta}}{\sum_{\{\sigma\}} e^{\sum_\beta K_\beta O_\beta}} = \frac{\partial F}{\partial K_\alpha} \tag{21}$$

where F is the free energy.

Using the fluctuation-dissipation theorem, we can also numerically calculate the response functions:

$$\chi_{\alpha,\beta} = \frac{\partial \langle O_\alpha \rangle}{\partial K_\beta} = \langle O_\alpha O_\beta \rangle - \langle O_\alpha \rangle \langle O_\beta \rangle ,$$

$$\tilde{\chi}_{\alpha,\beta} = \frac{\partial \left\langle \tilde{O}_\alpha \right\rangle}{\partial K_\beta} = \left\langle \tilde{O}_\alpha O_\beta \right\rangle - \left\langle \tilde{O}_\alpha \right\rangle \langle O_\beta \rangle .$$

Using the chain rule, one can calculate with equation (21) that

$$\tilde{\chi}^{(n)}_{\alpha,\beta} = \frac{\partial \left\langle \tilde{O}^{(n)}_\alpha \right\rangle}{\partial K_\beta} = \sum_\gamma \frac{\partial \tilde{K}_\gamma}{\partial K_\beta} \frac{\partial \left\langle \tilde{O}^{(n)}_\alpha \right\rangle}{\partial K_\gamma} = \sum_\gamma T_{\gamma,\beta} \chi^{(n)}_{\alpha,\gamma}.$$

It is thus possible to derive a value of $T_{\gamma,\beta}$ from the correlation functions by solving a set of M coupled linear equations. At point $K = K^*$, we can apply this method in an iterative manner to compute critical exponents as suggested by Eq. 20.

There are many error sources in this technique, that originate from the fact that we are using a combination of several tricks to obtain our results:

- Statistical errors,
- Truncation of the Hamiltonian to the M^{th} order,
- Finite number of scaling iterations,
- Finite size effects,
- No precise knowledge of K^*.

Hopfield Network

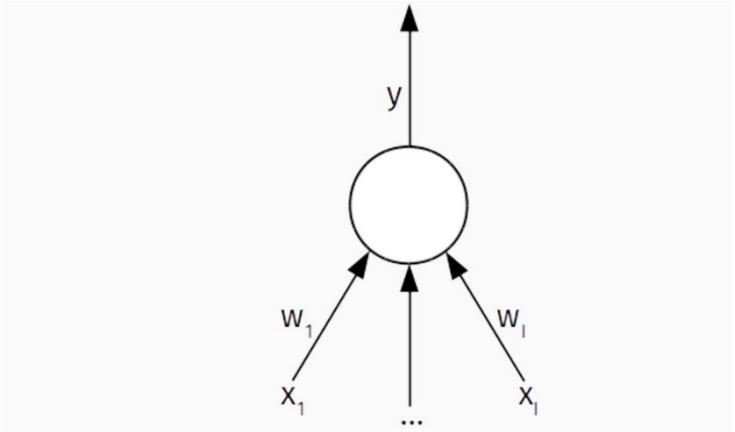


Figure 2: An illustration of a single neuron with output y, inputs $\{x_i\}_{i\in\{1,\dots,I\}}$ and weigths $\{w_i\}_{i\in\{1,\dots,I\}}$.

In terms of a Hopfield network, we consider discrete inputs $x_i \in \{-1, 1\}$. The activation function of neuron i is given by

$$a_i = \sum_j w_{ij} a_j, \tag{1}$$

where we sum over the inputs. The weights fulfill $w_{ij} = w_{ji}$ and $w_{ii} = 0$.

Similarly to the Ising model, the associated energy is given by

$$E = -\frac{1}{2} \sum_{i,j} w_{ij} x_i x_j - \sum_i b_i x_i, \tag{2}$$

where b_i is bias term.

The dynamics of a Hopfield network is

$$x_i(a_i) = \begin{cases} 1 & \text{if } a_i \geq 0, \\ -1 & \text{otherwise.} \end{cases} \tag{3}$$

The energy difference ΔE_i after neuron i has been updated is

$$\Delta E_i = E(x_i = -1) - E(x_i = 1) = 2 \left(b_i + \sum_j w_{ij} x_j \right) \tag{4}$$

We can absorb the bias b_i in the sum by having an extra active unit at every node in the network. We thus showed that the activation defined by Eq. (1) amounts to the half of the energy difference ΔE_i.

Boltzmann machine learning

Molecular dynamics

To model interacting particle systems, we use generalized coordinates

$$\mathbf{q}_i = \left(q_i^1, \ldots, q_i^d\right) \quad \text{and} \quad \mathbf{p}_i = \left(p_i^1, \ldots, p_i^d\right). \tag{1}$$

in a system where each particle has d degrees of freedom.

The system of N particles is then described by

$$Q = (\mathbf{q}_1, \ldots, \mathbf{q}_N) \quad \text{and} \quad P = (\mathbf{p}_1, \ldots, \mathbf{p}_N), \tag{2}$$

using the Hamiltonian

$$\mathcal{H}(P, Q) = K(P) + V(Q) \tag{3}$$

with $K(P) = \sum_{i,k} \frac{\left(p_i^k\right)^2}{2m_i}$ being the kinetic energy, m_i the mass of the i^{th} particle and $V(Q)$ the potential energy. The sum over $k \in \{1, ..., d\}$ accounts for the d degrees of freedom.

The potential (e.g., an attractive or repulsive electromagnetic potential) determines the mutual interactions of all particles and therefore their dynamics. An expansion of the potential energy yields:

$$V(Q) = \sum_i v_1(q_i) + \sum_i \sum_{j>i} v_2(q_i, q_j) + \sum_i \sum_{j>i} \sum_{k>j} v_3(q_i, q_j, q_k) + \ldots \tag{4}$$

Typically three or more body interactions are neglected and their effect is considered in an effective two body interaction described by

$$v_2^{\text{eff}}(q_i, q_j) = v^{\text{attr}}(r) + v^{\text{rep}}(r) \quad \text{with} \quad r = |\mathbf{q}_i - \mathbf{q}_j|, \tag{5}$$

where $v^{\text{attr}}(r)$ and $v^{\text{rep}}(r)$ represent attractive and repulsive part of the effective potential, respectively.

For now, we only consider potentials that depend on distance, not particle orientation. Analytically, the simplest potential is the hard sphere interaction potential

$$v^{\text{rep}}(r) = \begin{cases} \infty & \text{if } r < \sigma, \\ 0 & \text{if } r \geq \sigma. \end{cases} \tag{6}$$

Equation of motion

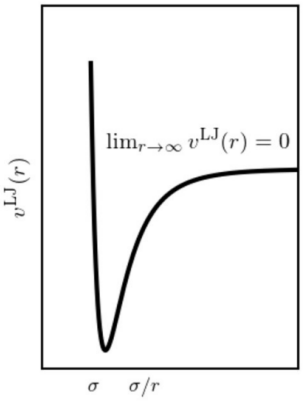
The first order Taylor approximation of a symmetric attractive or repulsive potential is given by an elastic potential. For two particles with radii R_1 and R_2, the potential is given by

$$v^{\text{rep}}(r) = \begin{cases} \frac{k}{2}(R-r)^2 & \text{if } r < R \\ 0 & \text{if } r > R \end{cases} \quad \text{with} \quad R = R_1 + R_2, \tag{7}$$

where k is the elastic spring constant.

Another very important form of potential typically used to describe the interaction between molecules is the *Lennard-Jones* potential

$$v^{\text{LJ}}(r) = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right], \tag{8}$$



where ϵ is the attractive energy and σ the interaction range.

LJ potential approximates the spherical symmetric interaction between a pair of neutral atoms or molecules.

Once the interaction potential has been defined, we can easily derive the equations of motion using the Hamilton equations

$$\dot{q}_i^k = \frac{\partial \mathcal{H}}{\partial p_i^k}, \quad \dot{p}_i^k = -\frac{\partial \mathcal{H}}{\partial q_i^k}, \tag{9}$$

where $k \in \{1, \ldots, d\}$ and $i \in \{1, \ldots, N\}$.

For every particle, we identify q_i with the position vector x_i and $\dot{\mathbf{q}}_i = \dot{\mathbf{x}}_i$ with the velocity vector $\dot{\mathbf{v}}_i$. Due to $\dot{\mathbf{x}}_i = \mathbf{v}_i = \frac{\mathbf{p}_i}{m}$ and $\dot{\mathbf{p}}_i = -\nabla V(Q) = \mathbf{f}_i$, the equations of motion are:

$$m_i \ddot{\mathbf{x}}_i = \mathbf{f}_i = \sum_j \mathbf{f}_{ij}, \tag{10}$$

where $\mathbf{f}_{ij}$ is the force exerted by particle j on particle i.

Contact time

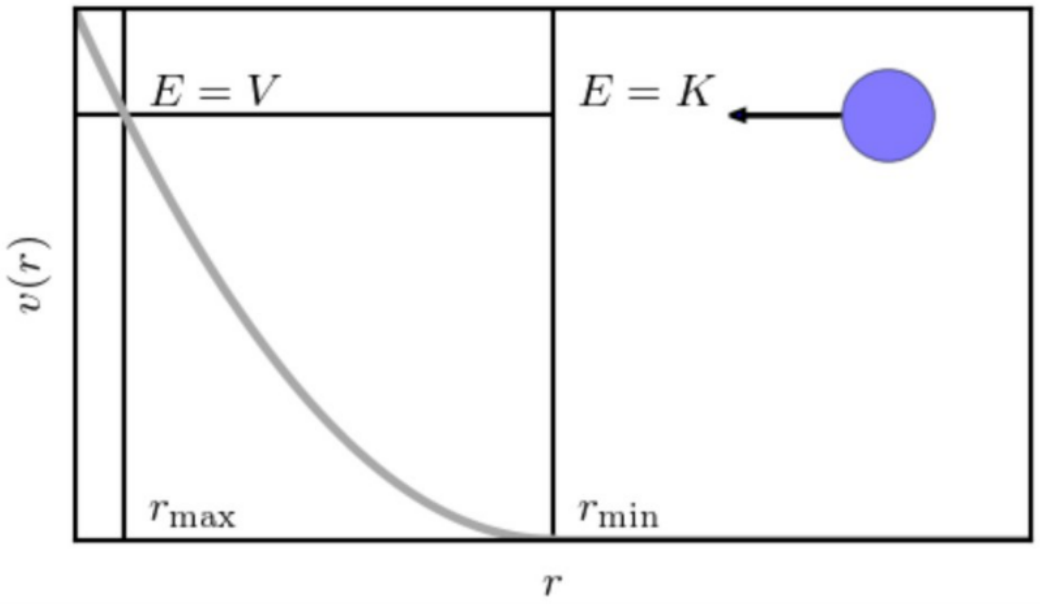


Figure 5: Derivation of the contact time.

Using the equations for energy

$$E = \frac{1}{2} m \dot{r}^2 + V(r) = \text{const.} \tag{11}$$

and radial velocity

$$\frac{\mathrm{d}r}{\mathrm{d}t} = \left[\frac{2}{m} \left(E - V(r) \right) \right]^{\frac{1}{2}}, \tag{12}$$

we derive the contact time

$$t_c = 2 \int_0^{\frac{1}{2} t_c} \mathrm{d}t = 2 \int_{r_{\min}}^{r_{\max}} \frac{\mathrm{d}t}{\mathrm{d}r} \mathrm{d}r = 2 \int_{r_{\min}}^{r_{\max}} \left[\frac{2}{m} \left(E - V(r) \right) \right]^{-\frac{1}{2}} \mathrm{d}r, \tag{13}$$

where $r_{\min}$ and $r_{\max}$ are the range of the potential and the turning point of a colliding particle, respectively.

We expect reasonable results only if the time step is not larger than the smallest contact time. The time integration of the equations of motion is then possible using an integration method such as

- Euler's method,
- Runge-Kutta methods,
- Predictor-corrector methods,
- Verlet methods,
- Leap-frog methods.

Verlet method

We begin with a Taylor expansion of $x(t+\Delta t)$ for sufficiently small time steps Δt so that

$$\begin{aligned}\mathbf{x}(t+\Delta t) &= \mathbf{x}(t) + \Delta t\,\mathbf{v}(t) + \frac{1}{2}\Delta t^2\,\dot{\mathbf{v}} + \mathcal{O}(\Delta t^3),\\ \mathbf{x}(t-\Delta t) &= \mathbf{x}(t) - \Delta t\,\mathbf{v}(t) + \frac{1}{2}\Delta t^2\,\dot{\mathbf{v}} - \mathcal{O}(\Delta t^3).\end{aligned} \tag{14}$$

Adding the latter two expressions yields

$$\mathbf{x}(t+\Delta t) = 2\mathbf{x}(t) - \mathbf{x}(t-\Delta t) + \Delta t^2\,\ddot{\mathbf{x}}(t) + \mathcal{O}(\Delta t^4). \tag{15}$$

Newton's second law enables us to express $\ddot{\mathbf{x}}(t)$ as

$$\ddot{\mathbf{x}}_i(t) = \frac{1}{m_i}\sum_j \mathbf{f}_{ij}(t) \quad \text{with} \quad \mathbf{f}_{ij}(t) = -\nabla V\left(r_{ij}(t)\right). \tag{16}$$

The particle trajectories are then computed by plugging in the latter results in Eq. (15). Typically, we use a time step of approximately $\Delta t \approx t_c/20$, with t_c a contact time defined in Eq. (13).

Some general remarks about the Verlet method:

- Two time steps need to be stored (t and $t-\Delta t$).
- Velocities can be computed with $\mathbf{v}(t) = \frac{\mathbf{x}(t+\Delta t)-\mathbf{x}(t-\Delta t)}{2\Delta t}$.
- The local numerical error is of order $\mathcal{O}\left(\Delta t^4\right)$, i.e. it is globally a third order algorithm.
- The numbers which are added are of order $\mathcal{O}\left(\Delta t^0\right)$ and $\mathcal{O}\left(\Delta t^2\right)$.
- Improvable by systematical inclusion of higher orders (very inefficient).
- The method is time reversible, which allows to estimate the error accumulation by reversing the process and comparing it to the initial conditions.

Leapfrog method

For the derivation of the Leapfrog method, we consider velocities at intermediate steps:

$$\mathbf{v}\left(t+\frac{1}{2}\Delta t\right)=v(t)+\frac{1}{2}\Delta t\,\dot{\mathbf{v}}(t)+\mathcal{O}(\Delta t^2), \tag{17}$$

$$\mathbf{v}\left(t-\frac{1}{2}\Delta t\right)=v(t)-\frac{1}{2}\Delta t\,\dot{\mathbf{v}}(t)+\mathcal{O}(\Delta t^2). \tag{18}$$

Taking the difference of the two equations leads to

$$\mathbf{v}\left(t+\frac{1}{2}\Delta t\right)=\mathbf{v}\left(t-\frac{1}{2}\Delta t\right)+\Delta t\,\ddot{\mathbf{x}}\,(t)+\mathcal{O}(\Delta t^3) \tag{19}$$

and we then update the positions according to

$$\mathbf{x}\,(t+\Delta t)=\mathbf{x}\,(t)+\Delta t\,\mathbf{v}\left(t+\frac{1}{2}\Delta t\right)+\mathcal{O}(\Delta t^4). \tag{20}$$

The analogies and differences between the Leapfrog method

$$\begin{aligned}\dot{\mathbf{v}}\,(t+\Delta t)&=\frac{f\,(\mathbf{x}\,(t))}{m},\\ \mathbf{v}\,(t+\Delta t)&=\mathbf{v}\,(t)+\Delta t\,\dot{\mathbf{v}}\,(t+\Delta t)\,,\\ \mathbf{x}\,(t+\Delta t)&=\mathbf{x}\,(t)+\Delta t\,\mathbf{v}\,(t+\Delta t)\end{aligned} \tag{21}$$

and the forward Euler integration

$$\begin{aligned}\dot{\mathbf{v}}\,(t+\Delta t)&=\frac{f\,(\mathbf{x}\,(t))}{m},\\ \mathbf{x}\,(t+\Delta t)&=\mathbf{x}\,(t)+\Delta t\,\mathbf{v}\,(t)\,,\\ \mathbf{v}\,(t+\Delta t)&=\mathbf{v}\,(t)+\Delta t\,\dot{\mathbf{v}}\,(t+\Delta t)\end{aligned} \tag{22}$$

are the following: The update of the variables is done in a different order (both methods rely on explicit forward integration). In the case of the Leapfrog method, the position is not updated using the previous velocity, as it is done in the usual Euler method.

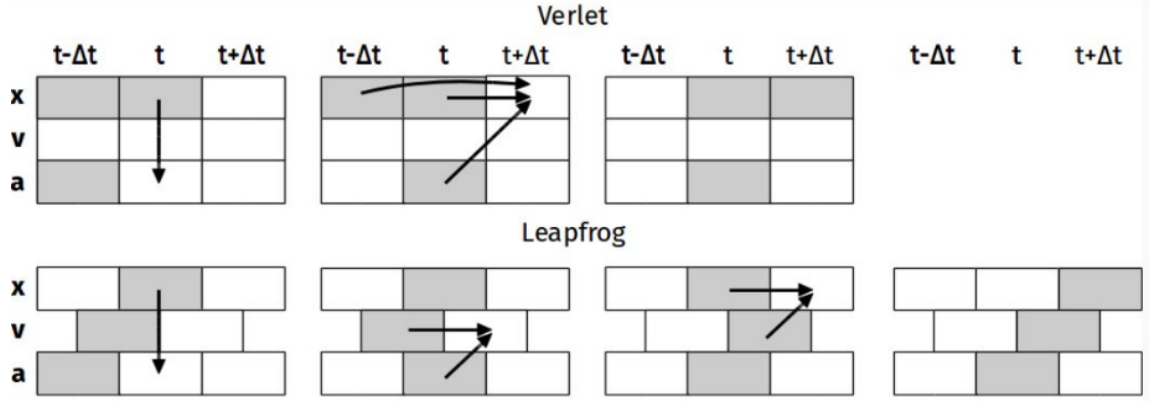


Figure 6: A comparison between Verlet and Leapfrog update schemes.

Velocity Verlet

$$\mathbf{a}^n = \frac{1}{m}\mathbf{F}(\mathbf{x}^n, \mathbf{v}^{n-1/2})$$
$$\mathbf{v}^{n+1/2} = \mathbf{v}^{n-1/2} + \mathbf{a}^n \Delta t$$
$$\mathbf{x}^{n+1} = \mathbf{x}^n + \mathbf{v}^{n+1/2} \Delta t$$

```
#verlet step
"""
r_next = 2.0*r_current - r_previous + dt2_m*F

v_current = (r_next - r_previous)*dt2_inv

for i in range(N):
    if any((r_current[i,:] > L) & (r_next[i,:] > L)):
    #if any(r_current[i,:] > L):
            r_current[i,:] %= L
            r_next[i,:] %= L
"""

#velocity verlet
v_current += dt_m*F
r_next = r_current + v_current*dt

r_next %= L
```

VV doesn’t need information about r(t-dt)! Simpler than the Verlet method.