Monte Carlo methods for quantum many-body calculations

Markus Holzmann
LPMMC, CNRS and UGA Grenoble
markus.holzmann@grenoble.cnrs.fr
https://wiki.lpmmc.cnrs.fr
(Dated: March 26, 2024)

I will provide a rough overview of Quantum Monte Carlo calculations, and introduce the basics of Monte Carlo sampling, Markov chains, Metropolis algorithm, and error analysis.

Monte Carlo integration methods are well established in theoretical physics, broadly employed to attack many-body problems in low or high energy, classical or quantum. I will present a basic introduction on classical and quantum Monte Carlo methods, focusing on some "typical" condensed matter (statistical physics) examples. Here and in the following "quantum Monte Carlo methods" refer to algorithms for solving quantum problems running on classical computer (hardware).

The scope of employing computational methods can be quite different ranging from high precision calculations to essentially provide exact benchmarks with well controlled systematic and stochastic errors (e.g. critical exponents, critical temperatures), over exploring solutions to models in parameter space with tolarable systematc bias (e.g. possible phases and phase transitions), to phenomenological modelling of experimental observations by the algorithm itself, where outcomes may be affected by algorithmic modifications (e.g. identifying Monte Carlo steps with time evolution of a system). The choice of algorithms and mathematical rigor underlying the computational approach depend on the actual scope.

I. CLASSICAL SYSTEMS

Basic Monte Carlo integration methods are part of most statistical physics courses. Some classic examples are (apart from the calculation of pi):

- Configurations of N hard spheres in two or three spatial dimensions d.
  The partition function Z_N consists of all configurations of non overlapping hard spheres in a (periodic) box of volume $V = L^d$. Direct sampling of trying out random configurations will not work (why?), one rather proposes a new configuration from a given legal one by deplacing one (or a few) particles once a time.

- Liquids and solids.
  Assuming a pair-wise interaction, $V(\mathbf{R}) = \sum_{i<j} v(r_{ij})$, $\mathbf{R} \equiv (\mathbf{r}_1, \ldots, \mathbf{r}_N)$, the configuration integral $Z = \int d\mathbf{R} e^{-\beta V(\mathbf{R})}$ can be sampled similarly, based on a very general strategy from Metropolis: Proposing some displacements of particles in a given configuration $\mathbf{R}$, $\mathbf{R} \to \mathbf{R}' = \mathbf{R} + \delta\mathbf{R}$, the move is accepted either if the potential energy is lowered, $V(\mathbf{R}') < V(\mathbf{R})$, or with some proability according to the Boltzmann like weight $e^{-\beta[V(\mathbf{R}')-V(\mathbf{R})]}$. Otherwise one remains with the original configuration.

- Spin models: classical Heisenberg or Ising model,

$$E = -J \sum_{<ij>} \mathbf{S}_i \cdot \mathbf{S}_j - \sum_i \mathbf{H} \cdot \mathbf{S}_i$$

- Lattice field theory, e.g. ϕ^4 theory

$$S[\phi(\mathbf{r})] = \sum_{\mathbf{r}} \left[\frac{1}{2} |\nabla_L \phi(\mathbf{r})|^2 + r|\phi(\mathbf{r})|^2 + \frac{g}{2}|\phi(\mathbf{r})|^4 \right]$$

  where the summation extends over all lattice sites $\mathbf{r}$, and ∇_L is the gradient operator on the lattice (finite difference expression involving only neareast neighbor sites).

For all of these quite different systems, the Metropolis algorithm (see below), provides a quite simple Monte Carlo strategy for calculation of observables of interest. Although it works, simulations of large systems are usuallly done with improved algorithms.

- For classical particles, molecular dynamics (MD) simulations are based on Newton's equation, discretizing time to obtain an approximate solution. MD are deterministic, e.g. without intrinsic stochastic error, and usually approach equilibrium faster than simple MC simulations, but suffer from systematic errors, like violation of energy conservation. Smart Monte Carlo methods [1, 2] combine MD steps with a Metropolis acceptance to avoid such bias to combine the best of two worlds. Additionally to the configuration space $\mathbf{R}$, momentum variables $\mathbf{P} = (\mathbf{p}_1, \ldots, \mathbf{p}_N)$ are kept as degrees of freedom in the partition function (and not integrated out).
- In the context of lattice models, smart Monte Carlo methods is called hybrid Monte Carlo [3, 4]. In contrast to classical particles where momentum variables are naturally occuring, momentum and mass of lattice field models are artificially added and do not necessarily have any physical origin/interpretation. Since those momenta occur quadratically and do not interact, one can determine their effect on the partition function by analytic integration.
- For spin systems, cluster algorithms provide an important speed-up in many cases [4, 5].

II. BAYESIAN INFERENCE

In all examples above, Monte Carlo methods are used to address and solve model problems based on statistical physics. However, in experiments, and more general in applied science and engineering applications, building the model may require not only physical insight, but also parameter estimation. For a given model and a given set of parameters θ, one can deduce the outcome of an experiment, y, by assigning and computing $p(y|\theta)$, e.g. a gaussian distibution of variance σ^2 for experimental outcomes y around the model prediction y_θ.

In order to calculate the compatibility of the model θ with our experimental outcomes, we want to calculate $p(\theta|y)$ which is unknown. However, we can use Bayes rule

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)} \propto p(y|\theta)p(\theta) \tag{1}$$

where $p(\theta)$ denotes the prior density of the parameter. The probability of the outcome $p(y)$ can be determined by normalization, $p(y) = \int d\theta p(y|\theta)p(\theta)$, similar to the role of the partition function above. Monte Carlo methods are used for parameter inference and model validation allowing for large set of parameters [8].

III. QUANTUM SYSTEMS

A. Systems in continuous space.

In the following I'm mainly interested in the properties of many-body systems in continuous space (2 or 3 dimensions) at very low (zero) temperature and assume that the system is described by a general non-relativistic Hamiltonian

$$H = \sum_{i=1}^{N} \left[-\frac{\nabla_i^2}{2m} + v_{ext}(\mathbf{r}_i) \right] + \sum_{i<j} v(|\mathbf{r}_i - \mathbf{r}_j|) \tag{2}$$

where v is the inter particle interaction (Coulomb potential for electrons) and v_{ext} is an external potential, e.g. the potential created by the ions in electronic structure or chemical physics [6]. Ideally, one would like to know the eigen values/functions of the Hamiltonian for specific boundary conditions which include also the statistics of the particles (symmetric for bosons, anti-symmetric for fermions)

$$H\Psi_i(\mathbf{R}) = E_i\Psi_i(\mathbf{R}) \tag{3}$$

where from now on $\mathbf{R} = \{\mathbf{r}_1, \mathbf{r}_2, \ldots \mathbf{r}_N\}$ collects all state variables, e.g. spin degrees of freedom, although we will not explicitly consider them. In general, the wavefunctions $\Psi_i(\mathbf{R})$ will be complicated functions of dN variables, $\mathbf{R}$, where d is the spatial dimension, and the exact solution of the Schrödinger equation, Eq. (3), in $d = 2, 3$ dimensions and $N \gg 4$ is infeasible. Therefore, one either simplifies the Hamiltonian or tries to map it to a systems whose solution is accessible (e.g. density functional methods), or one works on approximative wave functions. Let us first discuss zero temperature Quantum Monte Carlo methods to obtain the (approximative) ground state wavefunction, $\Psi_0(\mathbf{R})$.

Variational principle. The ground state energy of any quantum system can be bounded by above using the variational principle: The energy expectation value of any *reasonable* (C_2) wave function, $\Psi_T(\mathbf{R})$, is always higher

than the ground state energy, E_0:

$$E_0 \leq E_T \equiv \frac{\int d\mathbf{R}\Psi_T^*(\mathbf{R})H\Psi_T(\mathbf{R})}{\int d\mathbf{R}|\Psi_T(\mathbf{R})|^2} \tag{4}$$

Note, that $\Psi_T(\mathbf{R})$ does not need to be normalized.

The proof is elementary using the eigenstates, Eq. (3),

$$\frac{\int d\mathbf{R}\Psi_T^*(\mathbf{R})H\Psi_T(\mathbf{R})}{\int d\mathbf{R}|\Psi_T(\mathbf{R})|^2} = \frac{\sum_i |c_i|^2 E_i}{\sum_i |c_i|^2} \geq E_0, \tag{5}$$

where

$$c_i = \int d\mathbf{R}\Psi_i^*(\mathbf{R})\Psi_T(\mathbf{R}) \tag{6}$$

is the overlap of the trial wave function with the exact eigenstates. The variational principle is of fundamental importance in the following, since it allows us to compare and quantify the "quality" of different wave functions without invoking the comparison with experiment. The variational principle can be extended to excited states considering only trial states which are orthogonal to all states with lower energies. In practice, only different symmetries guarantee the orthogonality, so that the variational principle can only be applied to the lowest energy state within the symmetry of Ψ_T.

Variational Monte Carlo (VMC). In variational Monte Carlo, the expectation value of an observables, A, e.g. the energy, are calculated for a given trial wave function $\Psi_T(\mathbf{R})$, noting that

$$\langle \mathbf{R}|A|\Psi_T\rangle = \int d\mathbf{R}'\langle \mathbf{R}|A|\mathbf{R}'\rangle\Psi_T(\mathbf{R}') \tag{7}$$

Introducing a "local observable" $A_L(\mathbf{R})$ via

$$A_L(\mathbf{R}) \equiv \frac{\langle \mathbf{R}|A|\Psi_T\rangle}{\Psi_T(\mathbf{R})} \tag{8}$$

we have

$$\frac{\langle \Psi_T|A|\Psi_T\rangle}{\langle \Psi_T|\Psi_T\rangle} = \frac{1}{Z}\int d\mathbf{R}\pi(\mathbf{R})A_L(\mathbf{R}) \equiv \mathbb{E}_{\mathbf{R}\sim\Psi_T^2}[A_L(\mathbf{R})] \tag{9}$$

where $\pi(\mathbf{R}) \propto |\Psi_T(\mathbf{R})|^2$ is a non-negative weight determined by the expression of the trial wave function, and the normalization $Z = \int d\mathbf{R}\pi(\mathbf{R})$ plays the role of the partition function and we can use the same Monte Carlo algorithms for sampling as in our classical systems.

Let us rapidely discuss, some generic forms of the trial wave function

- Pair-product trial wave function.
  The simplest wave function which captures the basic features of the ground state of bosonic quantum systems is a pair-product (Jastrow-type) wavefunction

  $$\Psi(\mathbf{R}) = e^{-\sum_i u_1(\mathbf{r}_i) - \sum_{ij} u_2(r_{ij}) - \sum_{ijk} u_3 \ldots} \tag{10}$$

  where $u_n(r)$ are one dimensional function which are either given by some approximate theory (Quantum Cluster expansions, Correlated Basis Function) or or parametrized in terms of basis functions.

  Notice that the form (10) is formally just the same as the integrand of the configuration integral of a classical fluid with a corresponding inter-particle potential given by $u(r) = \beta v(r)$. Therefore we have mapped our quantum problem with a pair-product wave function to a system of classical pair-wise interacting particles. However, the form the pair-wise correlation function $u(r)$ may not be fully fixed, and determined by using the variational principle.

- Slater-Jastrow wave function.
  Our pair-product ansatz, Eq. (10), is symmetric with respect to particle exchanges and therefore can approximate bosonic wavefunctions. In order to treat Fermions, we have to use an antisymmetric ansatz for the wavefunction. This is conviniently done using an additional term - a Slater determinant of single particle orbitals

  $$D(\mathbf{R}) = \det_{ki} \phi_k(\mathbf{r}_i). \tag{11}$$

For homogeneous systems we are forced to use plane wave orbitals $\phi_k(\mathbf{r}) = e^{i\mathbf{k}\cdot\mathbf{r}}$ inside the determinant, typically the first N plane-waves with energies $\hbar^2k^2/2m$ less than the Fermi-energy ϵ_F. The simplest non-trivial ansatz for the fermionic wavefunction, the Slater-Jastrow form, is thus

$$\Psi_F(\mathbf{R}) = \det_{ki} \phi_k(\mathbf{r}_i) e^{-\sum_{ij} u(r_{ij})} \tag{12}$$

and is still one of the most used forms for precise Fermion calculations up to date. For inhomogeneous systems, Kohn-Sham DFT orbitals, or orbitals from quantum chemistry, in particular Hartree-Fock, are frequently used to build the Slater determinant of the wave function.

- Iterated backflow and similar deep neural networks give access to very flexible functional forms allowing to systematically approach the ground state energy to high precision.

Stochastic optimization. Our trial wave function $\Psi_\theta(\mathbf{R})$ contains some parameters θ, potentially very a very large number. The variational principle allows us to fix them by searching for the minimum of the energy expectation value

$$E(\theta) = \mathbb{E}_{\mathbf{R}\sim\Psi_\theta^2}[E_L(\mathbf{R}] = \frac{\int d\mathbf{R}\Psi_\theta^2(\mathbf{R})E_L(\mathbf{R})}{\int d\mathbf{R}\Psi_\theta^2(\mathbf{R})} \tag{13}$$

where $E_L(\mathbf{R}) = [H\Psi_\theta(\mathbf{R})]/\Psi_\theta(\mathbf{R})$ is the local energy which depends on the parameters θ. This is almost a standard minimization problem, however, it is in general non-linear, and affected by stochastic noise, since $E(\theta)$ is evaluated by MC.

Ignoring the problem of noise for the moment, one would expand $E(\theta+\delta\theta)$ up to second order in small changes $\delta\theta$ of our parameters, and use an iterative methods based on Newton's method

$$\theta_{n+1} = \theta_n - [\partial_\theta^2 E(\theta)]^{-1}\partial_\theta E(\theta) \tag{14}$$

where $\partial_\theta^2 E(\theta)$ denotes the second derivative (hessian) matrix. Apart that the hessian might be quite costly to calculate for large number of parameters, the inversion is likely to be corruped by the stochastic noise. Instead of trying to reduce the noise as much as possible by using long MC runs, a simple gradient decent

$$\theta_{n+1} = \theta_n - \epsilon_n\partial_\theta E(\theta) \tag{15}$$

where ϵ_n is some learning rate is likely to be more efficient (Robbins-Monro). Despite the stochastic error affecting the gradient $\partial_\theta E(\theta)$, the algorithm is guaranteed to converge for any sequence of positive learning rates ϵ_n with $\epsilon_n \to 0$ and $\sum_n \epsilon_n = \infty$, e.g. $\epsilon_n \propto n^{-1}$ or $n^{-1/2}$. Instead of taking the gradient of expression (13), an estimater of $\partial_\theta E(\theta)$ is easier obtained by using

$$\partial_\theta E(\theta) = 2\frac{\int d\mathbf{R}\Psi_\theta(\mathbf{R})H\partial_\theta\Psi_\theta(\mathbf{R})}{\int d\mathbf{R}\Psi_\theta^2(\mathbf{R})} - 2\frac{\int d\mathbf{R}\Psi_\theta(\mathbf{R})H_\theta\Psi_\theta(\mathbf{R})}{\int d\mathbf{R}\Psi_\theta^2(\mathbf{R})}\frac{\int d\mathbf{R}\Psi_\theta(\mathbf{R})\partial_\theta\Psi_\theta(\mathbf{R})}{\int d\mathbf{R}\Psi_\theta^2(\mathbf{R})} \tag{16}$$

and noting that, since H is a Hermitan operator, we can apply it to the left, such that we get

$$\partial_\theta E(\theta) = 2\mathbb{E}_{\mathbf{R}\sim\Psi_\alpha^2}[(E_L(\mathbf{R}) - E(\alpha)\partial_\alpha \log\Psi_\alpha] \tag{17}$$

Strategies based on stochastic gradient decent, also known as stochastic reconfiguration [7], have been successfully used for minimizing very large set of variational parameters. Optimization methods of VMC wave functions have become very similar to machine learning approaches.

Imaginary time projection. The proof of the variational principle also suggests how any trial wave function can be improved in principle considering

$$\Psi_\beta(\mathbf{R}) = \langle\mathbf{R}|e^{-\beta H}|\Psi_T\rangle = \int d\mathbf{R}' G(\mathbf{R},\mathbf{R}';\beta)\Psi_T(\mathbf{R}') \tag{18}$$

where we have defined the propagator

$$G(\mathbf{R},\mathbf{R}';\beta) = \langle\mathbf{R}|e^{-\beta H}|\mathbf{R}'\rangle \tag{19}$$

We get

$$E_\beta \equiv \frac{\int d\mathbf{R}\Psi_\beta^*(\mathbf{R})H_N\Psi_\beta(\mathbf{R})}{\int d\mathbf{R}|\Psi_\beta(\mathbf{R})|^2} = \frac{\sum_i |c_i|^2 e^{-2\beta E_i}E_i}{\sum_i |c_i|^2 e^{-2\beta E_i}} = E_0 + \frac{\sum_{i>0}|c_i/c_0|^2 e^{-2\beta(E_i-E_0)}(E_i-E_0)}{1+\sum_{i>0}|c_i/c_0|^2 e^{-2\beta(E_i-E_0)}} \tag{20}$$

which converges exponentially to the true ground state from above for large projection time β.

Projection Monte Carlo methods are based on imaginary time projection, based on a path integral representation for the propagator $G(\mathbf{R},\mathbf{R}';\beta)$. Based on the identity $e^{-\beta H} = [e^{-\beta H/M}]^M = [e^{-\tau H}]^M$ with $\tau = \beta/M$ we have

$$G(\mathbf{R},\mathbf{R}';\beta) = \int d\mathbf{R}_1 \cdots \int d\mathbf{R}_{M-1} G(\mathbf{R},\mathbf{R}_1;\tau)\cdots G(\mathbf{R}_{M-1},\mathbf{R}';\tau) \tag{21}$$

and we can use an approximation of $G(\mathbf{R},\mathbf{R}';\tau) = \langle \mathbf{R}|e^{-\tau H}|\mathbf{R}'\rangle$ valid for small τ. The simplest one is the primitive approximation based on Trotters theorem

$$e^{-\tau(T+V)+\tau^2[T,V]/2+\mathcal{O}(\tau^3)} = e^{-\tau T}e^{-\tau V} \tag{22}$$

which leads to a simple expression

$$G(\mathbf{R},\mathbf{R}';\tau) \simeq G_0(\mathbf{R},\mathbf{R}';\tau)e^{-\tau[V(\mathbf{R})+V(\mathbf{R}')]/2} \tag{23}$$

where $G_0(\mathbf{R},\mathbf{R}';\tau) = \langle \mathbf{R}|e^{-\tau T}|\mathbf{R}'\rangle$ is the propagator of non-interacting particles

$$G_0(\mathbf{R},\mathbf{R}',\tau) = \Big(\frac{m}{2\pi\hbar^2\tau}\Big)^{dN/2} e^{-m(\mathbf{R}-\mathbf{R}')^2/2\hbar^2\tau} \tag{24}$$

We have thus obtained a representation for the propagator $G(\mathbf{R},\mathbf{R}';\beta)$ which we can feed into a computer at the prize to perform many more integrations.

Path Integral Monte Carlo methods are based on the same idea to represent the propagator $G(\mathbf{R},\mathbf{R}',\beta)$ as a path-integral to perform finite temperature calculations

$$\langle A(\mathbf{R})\rangle = \frac{\int d\mathbf{R} G(\mathbf{R},\mathbf{R};\beta)A(\mathbf{R})}{Z} \tag{25}$$

$$Z = \int d\mathbf{R} G(\mathbf{R},\mathbf{R};\beta) = \int d\mathbf{R}_0 \int d\mathbf{R}_1 \cdots \int d\mathbf{R}_{M-1} G_0(\mathbf{R}_0,\mathbf{R}_1)\cdots G_0(\mathbf{R}_{M-1},\mathbf{R}_0) e^{-\tau\sum_m V(\mathbf{R}_m)} \tag{26}$$

(setting $\mathbf{R}_0 = \mathbf{R}$). This is very similar to zero temperature projection Monte Carlo, however, it is periodic in imaginary time, instead of having a trial wave function at both ends.

B. Spin systems.

All of the above methods also apply to quantum spin systems however we have to sample the discrete space of spin configurations expressed in some basis, e.g. in the basis of eigenvectors $|S\rangle$ of the σ_z operator. A general wave function can be written as

$$|\Psi\rangle = \sum_{S_1,S_2,\ldots S_N} |S_1,\ldots S_N\rangle \Psi(S_1,\ldots S_N) \tag{27}$$

and we can use VMC to sample some trial wave function $\Psi_T(S_1,\ldots S_N)$. Projection in imaginary time can be done based on Trotter's theorem, however, it is not evident to find always representations which are non negative [4].

C. Summary of (Q)MC methods

Variational Monte Carlo (VMC) uses Monte Carlo methods for the integrations needed to evaluate the energy or other observables of a trial wave function, $\Psi_T(\mathbf{R})$, which is given explicitly. **Projector Monte Carlo** methods, e.g. Diffusion, Reptation, or Variational Path Integral Monte Carlo, are based on the additional stochastic sampling of the imaginary-time propagator to sample the true ground state wave function. All methods are based on representations of the quantum problem, which allows us to use classical Monte Carlo simulation methods.

We are interested in calculating expectation values of a given operator, O,

$$\langle O\rangle = \frac{\int d\mathbf{R} O(R)\pi(\mathbf{R})}{Z}, \quad Z = \int d\mathbf{R}\pi(\mathbf{R}) \tag{28}$$

where the statistical weight $\pi(\mathbf{R})$ is given by

$$\begin{aligned}
\pi(\mathbf{R}) &= \exp[-\beta V(\mathbf{R})], \quad \text{Classical MC} \\
\pi(\mathbf{R}) &= |\Psi_T(\mathbf{R})|^2 \quad \text{Variational MC} \\
\pi(\mathbf{R}, \mathbf{R}_1, \ldots, \mathbf{R}_{M-1}, \mathbf{R}') &= |\Psi_T|(\mathbf{R})|G(\mathbf{R}, \mathbf{R}_1, \tau)G(\mathbf{R}_1, \mathbf{R}_2; \tau) \cdots G(\mathbf{R}_{M-1}, \mathbf{R}'; \tau)|\Psi_T(\mathbf{R}')| \quad \text{Variational Path Integral MC,} \\
\pi(\mathbf{R}_0, \mathbf{R}_1, \ldots, \mathbf{R}_{M-1}, \mathbf{R}_0) &= G(\mathbf{R}_0, \mathbf{R}_1, \tau)G(\mathbf{R}_1, \mathbf{R}_2; \tau) \cdots G(\mathbf{R}_{M-1}, \mathbf{R}_0; \tau) \quad \text{Path Integral MC}
\end{aligned}$$

and Z plays the role of a partition function.

IV. MARKOV CHAIN MONTE CARLO

Deterministic integration. Calculating observables involves high dimensional integrations, $D = dN$ dimensions in VMC, and even more in projection methods. For simplicity, we will just consider the general integral

$$I = \int d\mathbf{R} \tilde{f}(\mathbf{R}) \tag{29}$$

Conventional integration is based on discretization of space in finite intervals, let us use P points in each dimension. The integration error, ϵ will scale as a power of $h = 1/P$ ($\epsilon \sim h^2$ for trapezoidal rule) if we can reach the asymptotic region $P \gg 1$. However, the CPU time, T, scales with the total number of points, $T \sim P^D \sim \epsilon^{-D/2}$. Therefore, reducing the error by a factor of two, we need $2^{D/2}$ more computer time.

Stochastic sampling methods. Let us now assume that we can create P configurations $\mathbf{R}(t)$ at different times $t_1, t_2, \ldots, t_P$, and each configuration is distributed by the (normalized) probability $p(\mathbf{R})$. Let us rewrite the integrals under consideration as

$$I = \int d\mathbf{R} f(\mathbf{R}) p(\mathbf{R}), \quad f(\mathbf{R}) = \frac{\tilde{f}(\mathbf{R})}{p(\mathbf{R})} \tag{30}$$

Our best estimate of the integral will be

$$I \approx \bar{f} \equiv \frac{1}{P} \sum_{i=1}^{P} f(\mathbf{R}_i) \tag{31}$$

where all configurations, $\mathbf{R}_i$ (called walkers), are distributed by the probability $p(\mathbf{R})$. But do we have any control of the error?

To estimate the error, we can use the central limit theorem, which states that our estimate $\bar{f}$ for large sample sizes $P \gg 1$ is a gaussian distribution around the true mean value $f_1 \equiv I$ (our desired integral) with variance $\sigma^2 = f_2/P$,

$$c_{\bar{f}}(x) = \left(\frac{P}{2\pi f_2} \right)^{1/2} \exp\left[-\frac{(x - f_1)^2}{2f_2/P} \right], \quad f_1 = \int d\mathbf{R} f(\mathbf{R}) p(\mathbf{R}), \quad f_2 = \int d\mathbf{R} \left[f(\mathbf{R}) - f_1 \right]^2 p(\mathbf{R}), \tag{32}$$

The asymptotic sampling error using $P \gg 1$ points is therefore given by $\sigma = (f_2/P)^{1/2}$, if the variance f_2 exists and is finite. The important consequence is that the error of stochastic sampling methods decreases as $\epsilon \sim P^{-1/2}$ independent of the dimensions of the integral, and this, under fairly large conditions (finite variance independent of D, P are number of *independent* points). The CPU time of this so-called Monte Carlo methods thus scales as $T \sim \epsilon^{-2}$, and becomes favourable compared to conventional integration for $D > 4$ dimensions. Reducing the Monte Carlo error by a factor of two, we only need to wait 4 times longer!

Direct Sampling. Unfortunately, we will not be able to directly create configurations according to our probability distribution $p(\mathbf{R}) = \pi(\mathbf{R})/Z$, Eq. (28) and Eq.(29), for most of our problems. Only a few distributions, uniform, gaussian, exponential..., can be created directly. However, using for example a uniform distribution, one can convince oneself (e.g. looking at classical hard spheres), that the variance of the probability distribution in generally increases exponentially with the dimension D, so that *direct sampling* methods will not work either for many-body systems with $D = dN \gg 1$.

Markov chains. At this point we introduce an (artificial) Monte Carlo dynamics in the system, with the aim of creating configurations in time distributed according to $p(\mathbf{R}) = \pi(\mathbf{R})/Z$, at least asymptotically. Therefore, we consider the conditional probability that the system is in state $\mathbf{R}_{t_n}$ at time t_n

$$P(\mathbf{R}_{t_n} | \mathbf{R}_{t_{n-1}}, \mathbf{R}_{t_{n-2}}, \ldots, \mathbf{R}_{t_1}) \tag{33}$$

given that at the preceding time it was in state $\mathbf{R}_{t_{n-1}}$ at t_{n-1}, $\mathbf{R}_{t_{n-2}}$ at t_{n-2}, etc. In the most simplest case this conditional probability is just given statistically uncorreclated samples, or

$$P_{\text{direct sampling}}(\mathbf{R}_{t_n}|\mathbf{R}_{t_{n-1}},\mathbf{R}_{t_{n-2}},\ldots,\mathbf{R}_{t_1}) = p(\mathbf{R}_{t_n})p(\mathbf{R}_{t_{n-1}})\cdots p(\mathbf{R}_{t_1}) \tag{34}$$

However this reduces direct sampling algorithm above which was not efficient enough.

A particular dynamics which is still simple to analyse are obtained by Markov chains: we choose certain transition probabilities from one state to another which do not depend on the particular state

$$P_n(\mathbf{R}_{t_n}|\mathbf{R}_{t_{n-1}},\mathbf{R}_{t_{n-2}},\ldots,\mathbf{R}_{t_1}) = p(\mathbf{R}_{t_1})T(\mathbf{R}_{t_1}\to\mathbf{R}_{t_2})T(\mathbf{R}_{t_2}\to\mathbf{R}_{t_3})\cdots T(\mathbf{R}_{t_{n-1}}\to\mathbf{R}_{t_n}) \tag{35}$$

The basic object in Markov chains is the transition probability $T(\mathbf{R}\to\mathbf{R}')$ which depends on both states $\mathbf{R}$ and $\mathbf{R}'$ but is independant of time.

The transition probability must be normalized

$$\sum_{\mathbf{R}'} T(\mathbf{R}\to\mathbf{R}') = \sum_{\mathbf{R}} T(\mathbf{R}\to\mathbf{R}') = 1 \tag{36}$$

Detailed balance. We can write down a master equation for the probability, $p(\mathbf{R},t+1)$, to find $\mathbf{R}$ at $t+1$ knowing the distribution at t by considering all possible processes

$$p(\mathbf{R},t+1) = \sum_{\mathbf{R}'\neq\mathbf{R}} p(\mathbf{R}',t)T(\mathbf{R}'\to\mathbf{R}) + p(\mathbf{R})T(\mathbf{R}\to\mathbf{R}) \tag{37}$$

or using the normalization (36)

$$\begin{aligned} p(\mathbf{R},t+1) - p(\mathbf{R},t) &= \sum_{\mathbf{R}'\neq\mathbf{R}} p(\mathbf{R}',t)T(\mathbf{R}'\to\mathbf{R}) - \sum_{\mathbf{R}'\neq\mathbf{R}} p(\mathbf{R},t)T(\mathbf{R}\to\mathbf{R}') &(38)\\ &= \sum_{\mathbf{R}'} p(\mathbf{R}',t)T(\mathbf{R}'\to\mathbf{R}) - \sum_{\mathbf{R}'} p(\mathbf{R},t)T(\mathbf{R}\to\mathbf{R}') &(39)\end{aligned}$$

The stationary state $p_\infty(\mathbf{R}) = \lim_{t\to\infty} p(\mathbf{R},t)$ then satisfied the so called detailed balance condition

$$\sum_{\mathbf{R}'} p_\infty(\mathbf{R}')T(\mathbf{R}'\to\mathbf{R}) = \sum_{\mathbf{R}'} p_\infty(\mathbf{R})T(\mathbf{R}\to\mathbf{R}') \tag{40}$$

which can be enforced for each term in the summation since all elements are non negative

$$p_\infty(\mathbf{R}')T(\mathbf{R}'\to\mathbf{R}) = p_\infty(\mathbf{R})T(\mathbf{R}\to\mathbf{R}') \tag{41}$$

This condition just ensures that once equilibrium is reached it remains unchanged, since losses and gains just compensate.

Exponential convergence. How can be assure that the Markov process is really converging to the stationary distribution $\pi(\mathbf{R})$ and what is the time scale associated to reach stationatity? Let us start from an arbitrary distribution $p(\mathbf{R}_0,t=0)$, after time $t=n$ we have

$$p(\mathbf{R}_n,t=n) = \sum_{\mathbf{R}_0}\sum_{\mathbf{R}_1}\sum_{\mathbf{R}_2}\cdots\sum_{\mathbf{R}_{n-1}} p(\mathbf{R}_0,t=0)T(\mathbf{R}_0\to\mathbf{R}_1)T(\mathbf{R}_1\to\mathbf{R}_2)\cdots T(\mathbf{R}_{n-1}\to\mathbf{R}_n) \tag{42}$$

which can be simply written as a matrix product introducing a huge vecor to represent the probabilities $p(\mathbf{R},t)$ and a matrix for $T(\mathbf{R}\to\mathbf{R}')$

$$p(\mathbf{R}_n,t=n) = \langle\mathbf{R}_n,t=n|T^n|\mathbf{R}_0,t=0\rangle p(\mathbf{R}_0,t=0) \tag{43}$$

Diagonalizing the matrix T and denoting $|\lambda_i\rangle$ the eigenvector with eigenvalue λ_i we have

$$\begin{aligned} p(\mathbf{R}_n,t=n) &= \sum_i \lambda_i^n \langle\mathbf{R}_n,t=n|\lambda_i\rangle\langle\lambda_i|\mathbf{R}_0,t=0\rangle p(\mathbf{R}_0,t=0) &(44)\\ &\simeq \lambda_0^n a_0[1+(\lambda_1/\lambda_0)^n a_1+\ldots] &(45)\end{aligned}$$

where λ_0 is the largest eigenvector. For large n we will have exponential convergence to this eigenvector since the contributions of the other eigenvectors will decay with e.g.

$$(\lambda_1/\lambda_0)^n = e^{-n\log\lambda_0/\lambda_1} \tag{46}$$

for the first eigenstate.

Using the normalization, Eq. (36), we have

$$\sum_{\mathbf{R}'} p_\infty(\mathbf{R})T(\mathbf{R}\to\mathbf{R}') = p_\infty(\mathbf{R}) \tag{47}$$

Substituting the detailed balance inside the lhs we get

$$\sum_{\mathbf{R}'} p_\infty(\mathbf{R}')T(\mathbf{R}'\to\mathbf{R}) = p_\infty(\mathbf{R}) \tag{48}$$

we see that the stationary distribution $p_\infty(\mathbf{R})$ is an eigenstate of the transition matrix with eigenvalue one. If we can show that all other eigenvalues are smaller than one we indeed reach the desired equilibrium. This can indeed be assured if it is possible to go from any of the configurations to all others in a finite number of steps.

Metropolis algorithm. Scope of our Markov chain is to reach a stationary distribution $p_\infty(\mathbf{R}) \propto \pi(\mathbf{R})$ according to our (known) weight $\pi(\mathbf{R})$. A particular solution for the transition matrix which satisfies detailed balance is the Metropolis algorithm using

$$T(\mathbf{R}\to\mathbf{R}') = \min\left[1, \frac{\pi(\mathbf{R}')}{\pi(\mathbf{R})}\right] \tag{49}$$

To prove detailed balance for general values of $\pi(\mathbf{R})$ and $\pi(\mathbf{R}')$ we write down the tranition matrix $T(\mathbf{R}\to\mathbf{R}')$ and $T(\mathbf{R}'\to\mathbf{R})$. Assume $\pi(\mathbf{R}') > \pi(\mathbf{R})$ first. In that case we have $T(\mathbf{R}\to\mathbf{R}') = 1$, but $T(\mathbf{R}'\to\mathbf{R}) = \pi(\mathbf{R})/\pi(\mathbf{R}')$. We immediately find $\pi(\mathbf{R})T(\mathbf{R}\to\mathbf{R}') = \pi(\mathbf{R})'T(\mathbf{R}'\to\mathbf{R})$ to be fullfilled. The case $\pi(\mathbf{R}) > \pi(\mathbf{R}')$ goes through just the same. Therefore we enforces detailes balance for each term in the summation of Eq. (40) (all elements there are non negative)

$$\pi(\mathbf{R}')T(\mathbf{R}'\to\mathbf{R}) = \pi(\mathbf{R})T(\mathbf{R}\to\mathbf{R}') \tag{50}$$

A priori probabilities. Sometimes it is useful to choose a-priori probabilities to propose moves from $\mathbf{R}$ to $\mathbf{R}'$ which are different from the a-priori probability for the return move $\mathbf{R}'$ to $\mathbf{R}$. In this case we write

$$T(\mathbf{R}\to\mathbf{R}') = \mathcal{A}(\mathbf{R}\to\mathbf{R}')p(\mathbf{R}\to\mathbf{R}') \tag{51}$$

where $\mathcal{A}(\mathbf{R}\to\mathbf{R}')$ is the a-priori probability and $p(\mathbf{R}\to\mathbf{R}')$ the final acceptance probability. Enforcing detailed balance we are led to a generalized Metropolis algorithm for the acceptance probability

$$p(\mathbf{R}\to\mathbf{R}') = \min\left[1, \frac{\pi(\mathbf{R}')}{\mathcal{A}(\mathbf{R}\to\mathbf{R}')}\frac{\mathcal{A}(\mathbf{R}'\to\mathbf{R})}{\pi(\mathbf{R})}\right] \tag{52}$$

The a-priori probability is choosen such that it is fast to evaluate and bigger moves can be done for a reasonable acceptance probability.

A. Basic MC simulation

program **MonteCarlo**

- call **initialize**($\mathbf{R}$)
- LOOP
  - call **MCstep**($\mathbf{R}$)
  - call **observables**($\mathbf{R}$)
- END LOOP

Inside the subroutine **MCstep** we propose a new configuration and accept or reject the new configurations according to any rule which satisfies ergodicity and detailed balance. After each move we can write our observables or average over them, independently if the previous Monte Carlo move was accepted or rejected.

A simple Monte Carlo step based on a uniform displacement of the configuration by a random vector inside a cubic volume of length Δ and subsequent Metropolis algorithm for acceptance writes

subroutine **MCstep**

- based on the old configuration $\mathbf{R}$ sample a new one $\mathbf{R}_{new}$
  - $\mathbf{R}_{new} = \mathbf{R} + \Delta(\text{rnd}() - 0.5)$
- MC acceptance/ rejection using Metropolis algorithm
  - $p = \pi(\mathbf{R}_{new})/\pi(\mathbf{R})$
  - IF $p >$rnd():
    - accept move
    - use $\mathbf{R}_{new}$ as new configuration: $\mathbf{R} = \mathbf{R}_{new}$
- return configuration $\mathbf{R}$

Note that rnd() creates random number with $0 \leq \text{rnd}() < 1$.

Acceptance ratio. In the above example, Δ was a parameter which determines the maximal displacement of the configurations and must be adapted to improve the efficiency of the Monte Carlo sampling. It is important to measure always the acceptance ratio of the MC step. In general, one should vary the step size Δ such that one obtains $0.1 \lesssim$ acceptance ratio $\lesssim 0.9$. Optimizing the acceptance probability does not lead to important improvements in the efficiency.

Optimizing a-priori probabilities. In the above example, we have chosen a uniform a-priori probability $\mathcal{A}_u(\mathbf{R} \to \mathbf{R}') = \Delta^{-D}$ if $\mathbf{R}'$ is inside the cube of length Δ centered around $\mathbf{R}$, and zero outside. However, we can use any transition rule as long as you can go anywhere in space within a finite number of steps (ergodicity). Changes in the a-priori probability can lead to important improvements in the efficiency as one may spend more time in the relevant region of phase space (importance sampling). It is clear if one can create an apriori-probability $\mathcal{A}(\mathbf{R} \to \mathbf{R}') \approx \pi(\mathbf{R}')/C$, one samples close to the exact distribution. As a consequence, the acceptance ration will get close to one. More generally, we might chose $\mathcal{A}(\mathbf{R} \to \mathbf{R}') \approx \pi(\mathbf{R}')/\pi(\mathbf{R})$ and expand $\pi(\mathbf{R}')$ for $\mathbf{R}'$ around $\mathbf{R}$. From the first order term, we obtain a drift-force (*Force-Bias MC*), expanding $\log \pi(\mathbf{R}')$ up to second order we obtain the best gaussian a-priori probability. To compare the efficiency of a simulation, we should optimize $1/(\text{CPU time} \times \text{Error}^2)$ for different algorithms.

Error estimation. The central limit theorem provides the basis for error estimations: For uncorrelated data, for large T, we have

$$\langle O \rangle = \overline{O}_T \pm \epsilon_T \tag{53}$$

$$\overline{O}_T = \frac{1}{T}\sum_t O(\mathbf{R}_t) \tag{54}$$

$$\epsilon_T = \langle \left[\overline{O}_T - \langle O \rangle\right]^2 \rangle = \sqrt{\frac{\sigma^2(O)}{T}} \approx \sqrt{\frac{1}{T(T-1)}\sum_t \left[O(\mathbf{R}_t) - \overline{O}_T\right]^2} \tag{55}$$

However, our random walk will in general create data $\mathbf{R}_t$ which are correlated in time, and the true error will be given by

$$\epsilon_T \approx \sqrt{\frac{\tau}{T(T-1)}\sum_t \left[O(\mathbf{R}_t) - \overline{O}_T\right]^2} \tag{56}$$

where τ introduces the exact correlation time (T/τ are the number of true independent configurations created).

To estimate the correlation time, we can average over M subsequent data points, or, equivalently consider the operator

$$O^{(M)} = \frac{1}{M}\sum_{i=1}^{M} O(\mathbf{R}_{(t-1)M+i}) \tag{57}$$

for $t = 1, \ldots T/M$. Whereas $\overline{O}^{(M)} = \overline{O}_T$ for all M, we can assume that for large blocking time $M \approx M^* \gg 1$, the averaged values $O_t^{(M)}$ are decorrelated and gaussian distributed

$$p(\overline{O}^{(M^*)}) = \left(\frac{P}{2\pi\sigma^2_{M^*}}\right)^{1/2} \exp\left[-\frac{(\overline{O}^{(M^*)} - \langle O\rangle)^2}{2\sigma^2_{M^*}/P}\right] \tag{58}$$

where $\sigma^2_{M^*}$ is the variance of $O^{(M^*)}$ and $P = T/M^*$ are the number of configurations we have. The error is therefore given by

$$\epsilon_T^{(M^*)} = \sqrt{M^*\sigma^2_{M*}/T} \tag{59}$$

corresponding to the correlation time $\tau = M^*\sigma^2_{M^*}/\sigma_1^2$. Since the average of two gaussian distributed variables of variance σ^2 is also a gaussian of variance $\sigma^2/2$, we see that further averaging does not change the error any more. Therefore, blocking together M data point, we can determine the true error by increasing M until the apparent error $\sim \sigma_M\sqrt{M/T}$ does not increase anymore and remains stationary.

Example: Cluster/worm/loop algorithm for (classical) Ising model

Let us consider again the classical Ising model with

$$E = -J\sum_{<ij>} s_i s_j \tag{60}$$

one of the standard example for a MC calculation in statistical physics. It is straightfoward to identify our "configuration" $\mathbf{R}$ with the set of spins $\mathbf{S} = (s_1, \ldots, s_N)$, $s_i = \pm 1$, and apply Metropolis' algorithm to sample the discrete configurations (replacing above integrals by sums).

However, it is clear that such a sampling will be quite inefficient at low temperature where large clusters containing like-spins are formed. Instead of trying single spin flips, one would rather try to flip clusters of like-spins. The energy cost of flipping the sign of a whole cluster is determined by the spins on the boundary next to the cluster, and thus proportional to the surface. One might think about identifying all domains connecting like spins and flip them all together, essentially modifying the above a-priori propability and the subsequent Metropolis acceptance. In one dimensions, one can then easily conclude the absence of a long-range ordered phase at any finite temperature (for an infite chain). However, it two dimensions the surface energy will become large enough to prevent direct attempts to flip large domains.

Instead of simply flipping domains of like-spin, one would rather want to move the boundary of domains, e.g. moving "dislocations". One way to formalize this strategy is to extend the configuration space $\mathbf{R} = (\mathbf{S}, C)$, specifying a given state not only by the configuration of spins $\mathbf{S}$ but also by a selection of clusters C. Note that the weight of C for fixed spin configuration $\mathbf{S}$ is somewhat artifical, e.g. it is not determined directly by the energy which depends only on $E(\mathbf{S})$. All we need to assure is that

$$Z = \sum_{\mathbf{S}} e^{-\beta E(\mathbf{S})} = \sum_{\mathbf{S},C} \pi(\mathbf{S}, C) \tag{61}$$

for any weight $\pi(\mathbf{S}, C)$ attributed to a given selection of $\mathbf{S}$ and C. Cluster algorithms (Svendson, Wang,Wolff) construct clusters by "freezing" bonds of like spins with a certain probability such that the final probability for cluster flips gets independent of the surface.

Slightly different, one might consider explicit introduction of dislocations which will move create new sets of domains of like-spin. This is formalized in worm or loop algorithm.

Technically, for the ising model, cluster algorithms correspond to a reformulation of the partition function in terms of bonds

$$Z = \sum_{\mathbf{S}} e^{-\beta E(\mathbf{S})} = \prod_{<ij>} e^{\beta J s_i s_j} \propto \prod_{<ij>} (1 + t s_i s_j) \tag{62}$$

with $t = tanh(\beta J)$. This is based on $s_i^2 = 1$, such that $\exp K\sigma_i = \cosh K + \sinh K\sigma_i$; similar identities for spin systems are used all over the time. For a finite system the partition function is now a polynomial in t, but also in s_i, and can be written as

$$Z = \sum_{\mathbf{S}} \sum_{G} t^{|G|} \prod_i s_i^{n_i(G)} \tag{63}$$

where G indicates a graph on our lattice, build out of the vertices at i and the ("occupied") nearest neighbor bonds $< ij >$. The number of bonds in G is denoted by $|G|$, and $n_i(G)$ is the number of bonds in G connected to the vertex i.

Summing over all $\mathbf{S}$ for a given graph G, we can use that $\sum_{s_i=\pm 1} s_i^n$ is either 0 or 1, depending if n is odd or even. All graphs with odd vertices $n_i(G)$ can be ignored as they do not contribute to the partition function. It is easy to see that graphs with even vertices must be closed and we have

$$Z = \sum_{\text{closed}G} t^{|G|} \tag{64}$$

Cluster algorithms now construct Markov chains moving in the space of closed graphes.

Worm or loop algorithms circumvent the constraint of closed graphes by introducing dislocations in a well defined matter, considering an extended partition function

$$\widetilde{Z} = \sum_{h=0,1} \sum_{\mathbf{S}} e^{-\beta(E(\mathbf{S})+\epsilon h \sum_i s_i)} \tag{65}$$

The sector with $h = 0$ is our usual partition function Z, whereas $h = 1$ corresponds to a system in a (small) external field ϵh. Using again that $\exp[-\beta\epsilon h \sum_i s_i] \propto 1 + \epsilon \widetilde{K} s_i$, we see that $h = 1$ introduces an open graph with odd number of bonds connected to s_i. We only need to consider graphs up to order ϵ^2, which means to move around to open bonds. Configuration space can be sampled efficiently (and easily) by such worm/loop algorithms.

At high temperature: heat-bath algorithm, basically sample free (mean-field) spin in a thermal configuration.

---

[1] M. Allen and D. Tildesley, *Computer Simulations of Liquids.*
[2] D. Frenkel and B. Smit, *Understanding Molecular Simulations.*
[3] J. M. Thijssen, *Computational Physics.*
[4] J. Gubernatis, N. Kawashima, and P. Werner, *Quantum Monte Carlo Methods.*
[5] W. Krauth, *Statistical Mechanics: Algorithms and Computations.*
[6] R. M. Martin, L. Reining, and D. M. Ceperley, *Interacting Electrons.*
[7] F. Becca and S. Sorella, *Quantum Monte Carlo approaches for correlated systems.*
[8] P. Metha, M. Bukov, C.-H. Wang, A. G. R. Day, C. Richardson, C. K. Fisher, and D. J. Schwab, *A high bias, low variance introduction to Machine Learning for physicists*, Physics Reports **810**, 1 (2019).