Langevin Algorithm and Hybrid Monte Carlo

PHY989 Final Presentation

Giovanni Pederiva

11 December 2018

Michigan State University

Basics of Importance Sampling

The expectation value of a quantity A evaluated with path integrals is:

$$\langle A \rangle = \frac{1}{Z} \int \mathcal{D}[\phi] A[\phi] e^{-S[\phi]}$$

One can sample this with Monte Carlo integration by considering $e^{-S[\phi]}$ as a Boltzmann weight and use it as a *probability measure*. One than chooses a set of random configurations $\{\phi_i\}$ according to the probability distribution:

$$dP[\phi] = \frac{e^{-S[\phi]} \mathcal{D}[\phi]}{\int \mathcal{D}[\phi] e^{-S[\phi]}}$$

The expectation value is then approximated as:

$$\langle A \rangle \approx \frac{1}{N_{conf}} \sum_{i=1}^{N_{conf}} A[\phi_i]$$

Markov Chains

- Markov processes satisfy the *balance equation*:

$$\sum_U T(U'|U)P(U) = \sum_{U'} T(U|U')P(U') \Rightarrow \sum_U T(U'|U)P(U) = P(U')$$

  the equilibrium distribution is a fixed point of the Markov process.

- A more stringent requirement is that of *detailed balance*:

$$T(U'|U)P(U) = T(U|U')P(U')$$

Langevin Equation

The Langevin equation can be used to model *Brownian motion*. In its simplest form it reads:

$$\frac{dx}{dt} = -\frac{\partial V(x)}{\partial x} + \eta(t)$$

where $V(x)$ is some potential and $\eta(t)$ are random noise variables distributed according to a Gaussian PDF.
It generates a time-dependent probability distribution for the vector x, from which an observable $O[x]$ can be evaluated as:

$$\langle O[x(t)] \rangle = \int P(x,t) O(x) dx$$

Fokker-Planck Equation

The probability distribution of Langevin equation has an associated Fokker-Planck equation:

$$\frac{\partial P(x,t)}{\partial t} = \frac{1}{2}\frac{\partial}{\partial x_i}\left[\Omega\frac{\partial P}{\partial x_i} + V(x)P\right]$$

This is a deterministic equation for the time dependent $P(x,t)$. It is possible to relate this to the eucildean transporter in quantum mechanics.

Numerical Approach to Langevin Equation

Using the same trick for the HMC, see $S[\phi]$ as a potential of some fictitious hamiltonian, so that:

$$\frac{d\phi(x)}{dt} = -\frac{\partial S[\phi(x)]}{\partial x} + \eta(t)$$

Numerical algorithm is straightforward:

$$\phi_{t+1}(x) = \phi_t(x) - \epsilon \frac{\partial S[\phi_t(x)]}{\partial x} + \sqrt{2\epsilon}\eta(t)$$

Huge error in discretization. Can be improved with Metropolis Adjusted Langevin Dynamics (MALA), by adding some metropolis tests during the integration.

Hybrid Monte Carlo

The HMC is based upon considering the Hamiltonian

$$H[\phi, \pi] = \frac{\pi^2}{2} + S[\phi]$$

and integrating numerically the equations of motion:

$$\frac{\partial \phi}{\partial t} = \pi, \qquad \frac{\partial \pi}{\partial t} = -\frac{\partial S[\phi]}{\partial \phi}$$

Then to correct for integration errors one performs metropolis tests using the weight $e^{-\Delta H}$ to fulfill ergodicity and make the algorithm exact. Note that it is a reversible algorithm, satisfies detailed balance (we can use $\pi \to -\pi$ as it only comes squared in the hamiltonian)

Generalized HMC

We can extend the HMC algorithm by considering the stochastic evolution equations:

$$\frac{\partial \phi}{\partial t} = \pi$$

$$\frac{\partial \pi}{\partial t} = -\frac{\partial S[\phi]}{\partial \phi} - 2\mu_0 \pi + \eta(t)$$

where $\eta(t)$ is again white noise and $\mu_0 > 0$ is a "mass term". This reduces to Stochastic Molecular Dynamics (a variation of HMC) for $\mu_0 \to 0$. In the second order form:

$$\frac{\partial^2 \phi}{\partial t^2} + 2\mu_0 \frac{\partial \phi}{\partial t} = -\frac{\partial S[\phi]}{\partial \phi} + \eta(t)$$

for large μ_0 (after redefining time as $' = 2\mu_0 t$) this form recovers Langevin equation.

A Connection Between Algorithms

Why is this interesting?

- Langevin methods have been proven to be *renormalizable* a long time ago
- HMC has been proven to be *not renormalizable*, but SMD is.
- But the existence of a parameter μ_0 that interpolates between the two could suggest that they are in the *same universality class*, hence have similar behavior when approaching the continuum limit.

Implication of Renormalizability

- For an algorithm to be renormalizable we mean that the autocorrelation function of the Markov Process has a finite scaling exponent when approaching the continuum theory (smaller lattice spacing).
- For example the Metropolis algorithm scales as d^2 (d is the dimensionality of the system, degrees of freedom), while Langevin as $d^{4/3}$.
- For very long trajectories the HMC scales as $d^{5/4}$, better than Langevin, but this is not true in general

Potentially renormalizable algorithms could be more efficient than the HMC near the continuum limit.

Some Real Data for Free-Theory

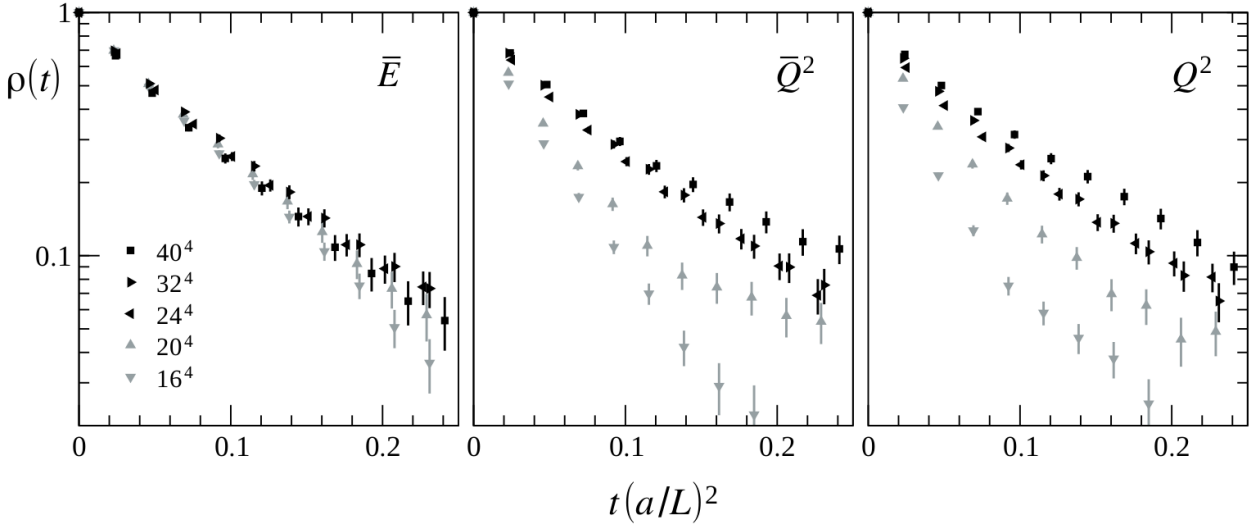


Fig. 2. Normalized autocorrelation functions of the observables $\bar{E}(L/2)$, $\bar{Q}(L/2)^2$ and Q^2 at flow time t_0, plotted as a function of the simulation time lag t given in units of $(L/a)^2$. The $\mathrm{SMD}_{0.3}$ algorithm was used all cases shown here. For better legibility, the data points obtained on the coarsest lattices (16^4 and 20^4) are coloured in grey, while the black points are those from the other lattices (24^4, 32^4 and 40^4).

Some Real Data for Free-Theory

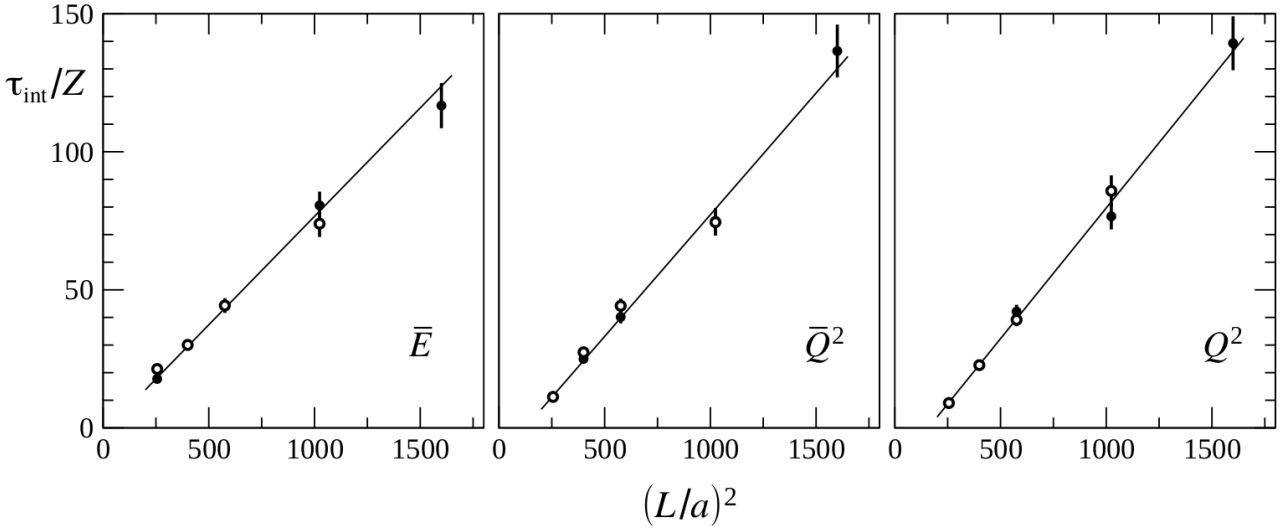


Fig. 3. Integrated autocorrelation times of the observables $\overline{E}(L/2)$, $\overline{Q}(L/2)^2$ and Q^2 at flow time t_0, as obtained on the $(L/a)^4$ lattices using the HMC algorithm (open circles, scale factor $Z = 1.32$) and the $\mathrm{SMD}_{0.3}$ algorithm (full circles, $Z = 1$). Many HMC points lie on top of the $\mathrm{SMD}_{0.3}$ points and thus mask the latter. The curves are straight-line fits of the $\mathrm{SMD}_{0.3}$ data.

A Small Test

Let's consider the harmonic oscillator in one dimension with action:

$$S[x] = \sum_{i=1}^{N-1} \left[\frac{m}{2a}(x_i - x_{i+1})^2 + \frac{a}{2}\left(V(x_i) + V(x_{i+1})\right) \right]$$

and try to sample the ground state energy defined as $E = \langle x^2 \rangle$ using the Metropolis, the HMC and Langevin algorithms.

Metropolis Code

```
def update(x):
    x_new = x + np.random.uniform(-step_size,step_size, len(x))
    dS = action(x_new) - action(x)
    if np.exp(-dS) > np.random.random():
        x = x_new
        return x, 1
    return x, 0

def metropolis_sampling(x, N_conf, N_corr):
    acceptance = 0
    energy = np.zeros(N_conf)
    for i in trange(N_conf):
        for k in range(N_corr):
            x, accepted = update(x)
            acceptance += accepted
        energy[i] = np.average(x*x)
```

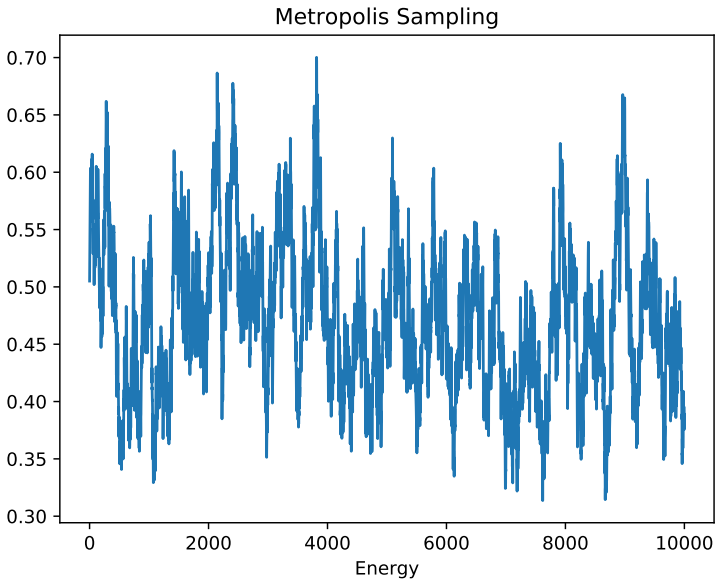
HMC Code

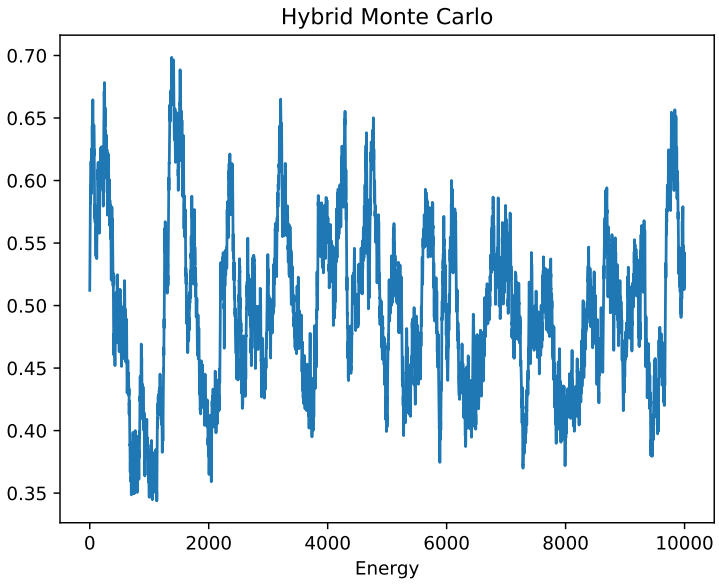
```
def leap_frog(x,p,n_step):
    p = p - step_size*0.5 * action_derivative(x)
    for i in range(1,n_step):
        x = x + step_size*p
        p = p - step_size * action_derivative(x)
    x = x + step_size*p
    p = p - step_size*0.5 * action_derivative(x)
    return x, p

def hmc(x, N_conf, N_corr):
    acceptance = 0
    energy = np.zeros(N_conf)
    for i in trange(N_conf):
        for k in range(N_corr):
            p = np.random.normal(0, 1, N)
            x_new, p_new = leap_frog(x, p, n_step_traj)
            dK = np.sum(p_new*p_new)/2 - np.sum(p*p)/2
            dS = action(x_new) - action(x)
            if np.exp(-(dS+dK)) > np.random.random():
                x = x_new
                acceptance+=1
        energy[i] = np.average(x*x)
```

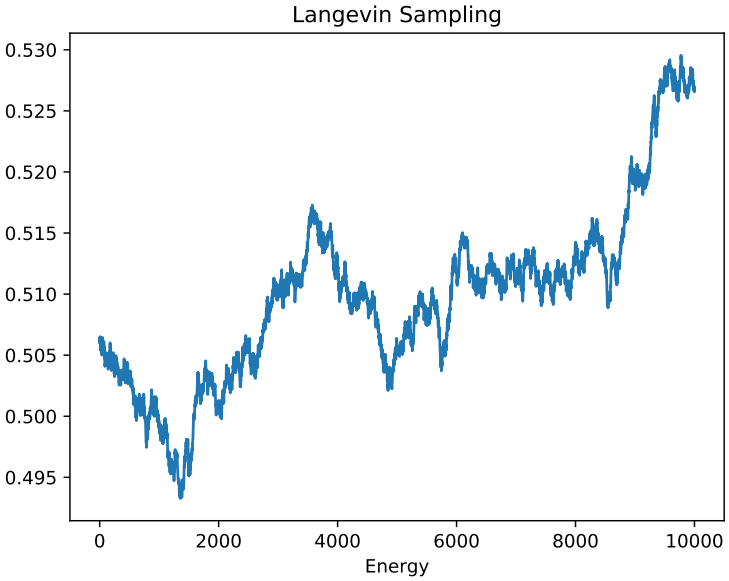
Langevin Code

```
def langevin(x, N_conf, N_corr):
    acceptance = 0
    energy = np.zeros(N_conf)
    for i in trange(N_conf):
        for k in range(N_corr):
            eta = np.random.normal(0, 1, len(x))
            x += -action_derivative(x)*step_size + eta*np.sqrt(2*step_size)
        energy[i] = np.average(x*x)
```

Sample Monte Carlo Histories







Estimation of τ_{int}

The autocorrelation function is defined as:

$$\Gamma(t) = \Gamma(-t) = \langle (x_i - \bar{x})(x_{i+t} - \bar{x}) \rangle \approx \frac{1}{N-t} \sum_{i=1}^{N-t} (x_i - \bar{x})(x_{i+t} - \bar{x}),$$

where t is the "lag" between two points. The integrated autocorrelation time is given by:

$$\tau_{int} = \frac{1}{2} \sum_{t=1}^{\infty} \frac{\Gamma(t)}{\Gamma(0)} = \frac{1}{2} \sum_{t=1}^{\infty} \rho(t).$$

In order to truncate the infinite summation one can look at the deviation squared of $\rho(t)$:

$$\langle \delta\rho(t)^2 \rangle \approx \frac{1}{N} \sum_{k=1}^{\infty} \left[\rho(k+t) + \rho(k-t) - 2\rho(k)\rho(t) \right]^2 .$$

All these terms, for a sufficiently large value of k should all vanish, hence one can choose a cutoff Λ and truncate the sum up to $t + \Lambda$. The integrated autocorrelation time, if the deviations of $\rho(t)$ become small, plateaus.

Windowing Procedure for τ_{int}

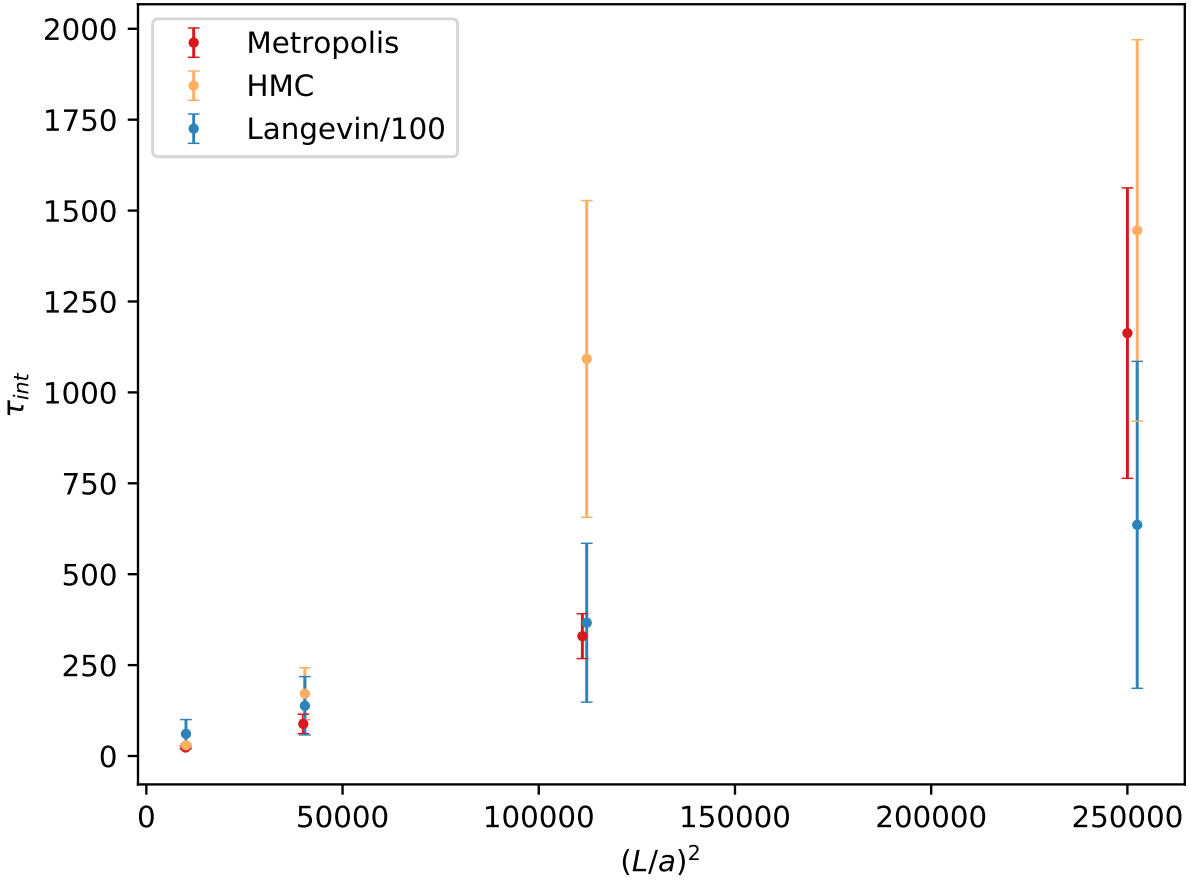
We choose a cutoff W such that:

$$\tau_{int} = \frac{1}{2}\sum_{t=1}^{W}\rho(t),$$

where W is the first lag t for which $\rho(t) < \sqrt{(\langle\delta\rho(t)^2\rangle}$, when the contribution to the integration of τ_{int} from that lag become smaller than the deviation of that same lag.
An approximate error estimate of the integrated autocorrelation time can be defined as:

$$\sigma^2(\tau_{int}) \approx \frac{2(2W+1)}{N}\tau_{int}^2$$

Results


Metropolis
HMC
Langevin/100
2000
1750
1500
1250
1000
750
500
250
0
τ_{int}
0
50000
100000
150000
200000
250000
$(L/a)^2$

Conclusions

Sources

- M. Lüscher , S. Schaefer, *Non-renormalizability of the HMC algorithm*, (2011), Journal of High Energy Physics
- M. Lüscher , S. Schaefer, *Lattice QCD without topology barriers*, (2011), Journal of High Energy Physics
- J. Zinn-Justin, *Quantum Field Theory and Critical Phenomena*, (1996),
- C.Gattringer, C.B. Lang, *Quantum Chromodynamics on the Lattice* (2010), Springer
- R. M. Neal, *MCMC using Hamiltonian dynamics*, Chapter 5 of the "Handbook of Markov Chain Monte Carlo"