Monte Carlo evaluation of path integrals.

Framework: Lattice QCD simulations

Student:

Elena Gramellini

September 2025

Theoretical and Numerical Aspects of Nuclear Physics

Theoretical Physics, University of Bologna

Motivation

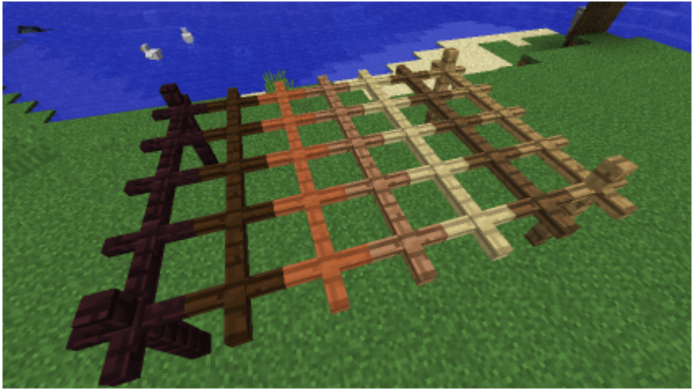


Figure: Source: https://www.9minecraft.net/lattice-mod/

1st Exercise - *theory*

Goal:

- Numerical integration of the euclidean path integral, of harmonic and quartic oscillator, to evaluate the ground-state

From standard theory of (1D euclidean) path integrals:

$$\langle x_f | e^{-\tilde{H}(t_f - t_i)} | x_i \rangle = \int \mathcal{D}x(t)\, e^{-S[x]}$$

where $S[x] \equiv \int_{t_i}^{t_f} dt\, L(x, \dot{x}) = \int_{t_i}^{t_f} dt\, \left(\frac{m\dot{x}^2(t)}{2} + V(x(t)) \right)$ (classical action).

If the propagator is known $\Rightarrow$ determine ground-state energy $(x_i = x_f = x$, $t_f - t_i = T$ and resolution of identity for energy eigenstates):

$$\langle x | e^{-\tilde{H}T} | x \rangle = \sum_n \langle x | E_n \rangle e^{-E_n T} \langle E_n | x \rangle \xrightarrow[T \to +\infty]{} e^{-E_0 T} \left| \langle x | E_0 \rangle \right|^2$$

1st Exercise - *theory*

Integrating over x one finds the ground-state energy:

$$\int dx \, \langle x | e^{-\tilde{H}T} | x \rangle \xrightarrow[T\to+\infty]{} e^{-E_0 T} \equiv \epsilon \Rightarrow E_0 = -\frac{1}{T} \log \epsilon$$

How to develop a numerical procedure to evaluate these path integrals?

1. Represent particle paths in the computer: $\{x(t),\ t_i \leq t \leq t_f\}$ are evaluated only at the nodes of a discretized t-axis $t_j = t_i + ja$, with $j = 0, ..., N$ and $a = \frac{t_f - t_i}{N}$ (grid spacing).
   $\boldsymbol{\Rightarrow \{x(t),\ t = t_0, \ldots, t_N\}}$
2. Realization of integrations over all paths $x(t)$: they are standard integral over all possible values of each $x(t_j)$, keeping the boundaries $x(t_0)$ and $x(t_N)$ fixed.
   $\boldsymbol{\Rightarrow \int \mathcal{D}x(t) \ \longrightarrow\ A \int_{-\infty}^{\infty} dx_1 \, dx_2 \, \ldots \, dx_{N-1}}$

1st Exercise - *theory*

3. Discretization of the action (the most "natural" way to do it!)
$\Rightarrow \int_{t_j}^{t_{j+1}} dt\, L \;\approx\; a\left[\frac{m}{2}\left(\frac{x_{j+1}-x_j}{a}\right)^2 + \frac{1}{2}\left(V(x_{j+1})+V(x_j)\right)\right]$, with $x_j \equiv x(t_j)$

1. + 2. + 3. = Numerical representation of the QM propagator

$$\langle x|e^{-\tilde{H}T}|x\rangle \approx A\int_{-\infty}^{\infty} dx_1 \cdots dx_{N-1}\, e^{-S_{\text{lat}}[x]}$$

where $S_{\text{lat}}[x] \equiv \sum_{j=0}^{N-1}\left[\frac{m}{2a}(x_{j+1}-x_j)^2 + a\,V(x_j)\right]$

Setting $N = 8$ I evaluate the 7D integral with **vegas** a standard Monte-Carlo routine.

1st Exercise - *code*

```
def action(x_path, x0, potential_choice):
    x_full = np.concatenate(([x0], x_path, [x0]))
    kinetic = np.sum((m / (2 * a)) * (x_full[1:] - x_full[:-1])**2)
    potential = a * np.sum(potential_choice(x_full[:-1]))
    return kinetic + potential

def integrand(x_path, x0, potential_choice):
    prefactor = (m / (2 * np.pi * a)) ** (N / 2)
    return prefactor * np.exp(- action(np.array(x_path), x0, potential_choice))

# Numerical integration of the propagator
def evaluate_propagator(x0, potential_choice):
    integrator = vegas.Integrator([[-5, 5]] * (N - 1)) #interval for the (N - 1) integrations
    def f(x_path):
        return integrand(x_path, x0, potential_choice)
    result = integrator(f, nitn=10, neval=100000) #nint = n. of bins, neval = n. of evaluations
    return result.mean
```

1st Exercise - *plots*

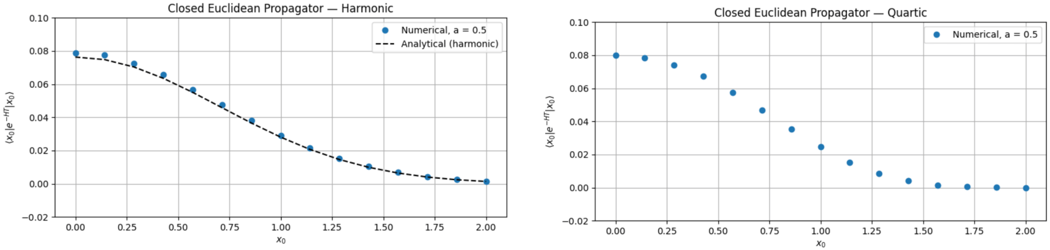


Figure: Harmonic (on the left) and quartic (on the right) propagator for lattice spacing $a = 0.5$.

Exercise 2 - *theory*

Goal:

- Monte Carlo evaluation of path integrals, going beyond the ground-state
- Evaluation of $\Delta E = E_1 - E_0$

Consider a general 2-point correlation function:

$$G(t) \equiv \langle\langle x(t_1)x(t_2)\rangle\rangle = \frac{\displaystyle\int \mathcal{D}x(t)\; x(t_2)x(t_1)\, e^{-S[x]}}{\displaystyle\int \mathcal{D}x(t)\; e^{-S[x]}}$$

Rewrite the numerator as $\int dx\, \langle x|e^{-\tilde{H}(t_f-t_2)}\,\tilde{x}\,e^{-\tilde{H}(t_2-t_1)}\,\tilde{x}\;e^{-\tilde{H}(t_1-t_i)}|x\rangle$, and go to the basis of energy eigenstates: $G(t) = \dfrac{\displaystyle\sum_n e^{-E_nT}\,\langle E_n|x\,e^{-(\tilde{H}-E_0)t}\,x|E_n\rangle}{\displaystyle\sum_n e^{-E_nT}}$ $(t \equiv t_2 - t_1)$.

Exercise 2 - *theory*

In large t limit ($1 \ll t \ll T$): $G(t) \xrightarrow{t \text{ large}} |\langle E_0|\tilde{x}|E_1\rangle|^2 \, e^{-(E_1-E_0)t}$
$\Rightarrow$ We extract the energy difference $\boldsymbol{\Delta E \equiv E_1 - E_0 = \frac{1}{a} \ln\left(\frac{G(t)}{G(t+a)}\right)}$

How to develop a numerical procedure to evaluate these weighted avarages?

- Generate a large number (N_{cf}) of random paths: $x^{(\alpha)} \equiv \{x_0^{(\alpha)} x_1^{(\alpha)} ... x_{N-1}^{(\alpha)}\}$
  $\Rightarrow$ **Metropolis algorithm**

- Assign to each path the probability: $P[x^{(\alpha)}] \propto e^{-S[x^{(\alpha)}]}$
- Having a general weighted avarage $\langle\langle \Gamma[x] \rangle\rangle = \frac{\int \mathcal{D}x(t)\, \Gamma[x]\, e^{-S[x]}}{\int \mathcal{D}x(t)\, e^{-S[x]}}$, this is approximated with the unweighted avarage (estimator) $\overline{\Gamma} = \frac{1}{N_{\text{cf}}} \sum_{\alpha=1}^{N_{\text{cf}}} \Gamma[x^{(\alpha)}]$ with $\sigma_\Gamma^2 = \frac{\langle\langle \Gamma^2 \rangle\rangle - \langle\langle \Gamma \rangle\rangle^2}{N_{cf}}$

$$\boldsymbol{\Rightarrow G(t) \approx \overline{G} = \frac{1}{N_{cf}} \sum_{\alpha=1}^{N_{\text{cf}}} \; G[x^{(\alpha)}], \alpha = 1, ..., N_{cf}}$$

Exercise 2 - *Metropolis algorithm*

Code for the generation of N_{cf} random paths:

```python
#Metropolis update of the path: randomizing x[j] at the jth site
@njit
def update(x):
    for j in range(0, N):
        old_x = x[j]
        old_action = action(j, x)
        x[j] += np.random.uniform(-eps, eps)  #add a random number in (-eps, eps) to x[j]
        d_action = action(j, x) - old_action  #compute delta_S
        if d_action > 0 and np.exp(-d_action) < np.random.uniform(0, 1):  #conditions to keep the old x[j]
            x[j] = old_x
```

Exercise 2 - *Correlations and thermalization*

- Successive paths generated by Metropolis algorithm are correlated $\Rightarrow$ consider only statistically independent configurations (sweeps erase correlations): keep only every $N_{cor}th$ correlated path
- The first configuration is usually atypical $\Rightarrow$ thermalization: discard the first $5N_{cor}$-$10N_{cor}$ before collecting $x^{(\alpha)}$s

```python
def MCaverage(x, G, N_config):
    for j in range(0, N):
        x[j] = 0 #initialization
    for j in range(0, 5 * N_cor): #thermalize
        update(x)
    for alpha in range(0, N_config):
        for j in range(0, N_cor): #erase correlations
            update(x)
        for n in range(0, N):
            G[alpha][n] = compute_G(x, n) #compute G and save it
    return G
```

Exercise 2 - *Bootstrap*

Code for "statistical bootstrap", i.e. chosen method to estimate errors:

```
def bootstrap(G, nbstrap=100):
    N_cf, N = G.shape
    deltaEs = np.zeros((nbstrap, N - 1)) #initialization of nbstrap rows and N-1 columns

    for b in range(nbstrap):
        idx = np.random.choice(N_cf, N_cf, replace=True)  #resampling
        sample = G[idx]
        avg = np.mean(sample, axis=0)
        for t in range(N - 1):
            if avg[t] > 0 and avg[t + 1] > 0:  #condition to have a valid logarithm
                deltaEs[b, t] = np.log(avg[t] / avg[t + 1]) / a
            else:
                deltaEs[b, t] = 0.0

    avgE = np.mean(deltaEs, axis=0)
    sdevE = np.std(deltaEs, axis=0, ddof=1)
    return avgE, sdevE
```

Exercise 2 - *plots*

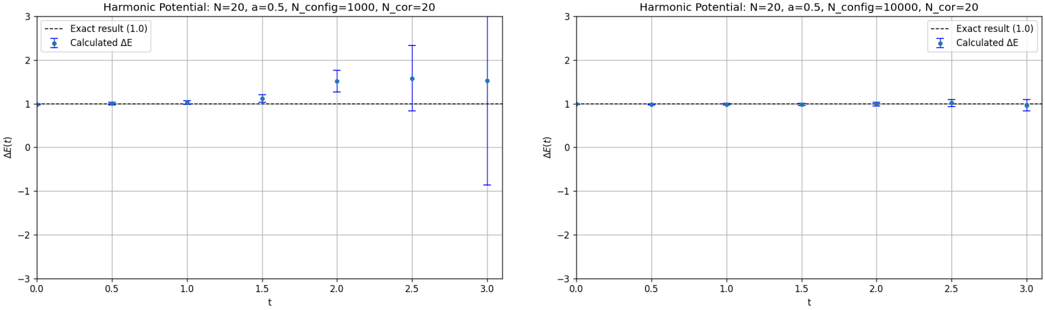


Figure: Monte Carlo values $\Delta E(t) = \frac{1}{a} \log \frac{G(t)}{G(t+a)}$ plotted versus t for an harmonic oscillator. The exact asymptotic result, $\Delta E(\infty) = 1$, is indicated by a line. Results are for a 1D lattice with N $=$ 20 sites, lattice spacing a $=$ 1/2, and Ncf $=$ 1000/10000 configurations, keeping configurations only every Ncor $=$ 20 sweeps.

Exercise 2 - *plots*

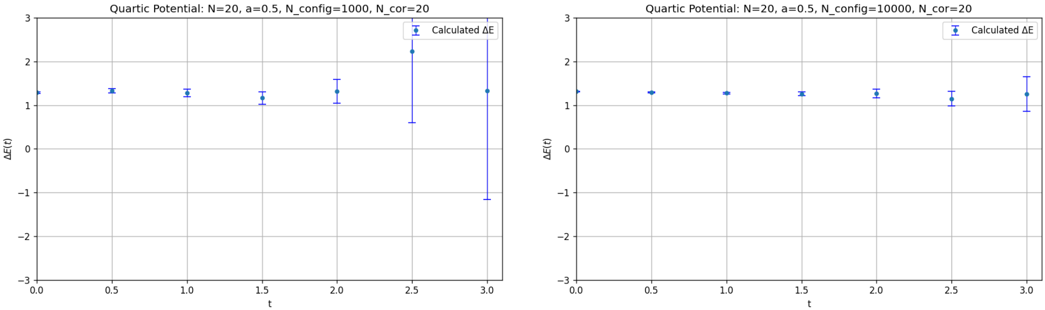
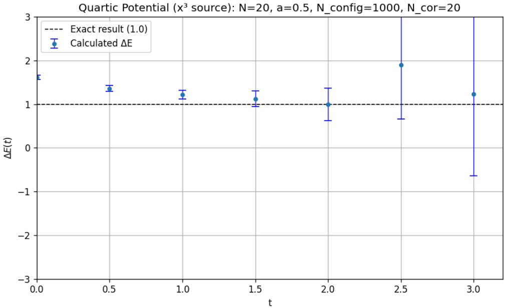


Figure: Monte Carlo values $\Delta E(t) = \frac{1}{a} \log \frac{G(t)}{G(t+a)}$ plotted versus t for a quartic oscillator. Results are for a 1D lattice with N $=$ 20 sites, lattice spacing a $= 1/2$, and Ncf $=$ 1000/10000 configurations, keeping configurations only every Ncor $=$ 20 sweeps.

Exercise 3 - *plots*

Goal:

- Repeat the same method, but for the propagator $G(t) = \frac{1}{N_{cf}} \sum_j \langle\langle x^3(t_j + t)\, x^3(t_j)\rangle\rangle$



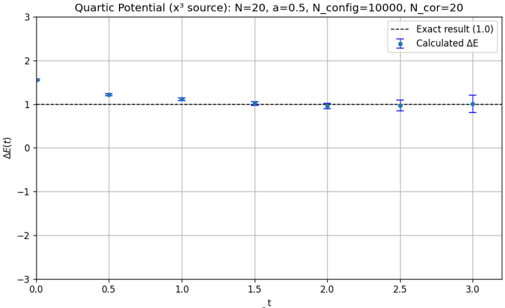


Figure: Monte Carlo values $\Delta E(t)$ for an harmonic oscillator, this time using x^3 as the source and sink; parameters are the same as before. The energies take longer to reach their asymptotic value.

Exercise 3 - *plots*

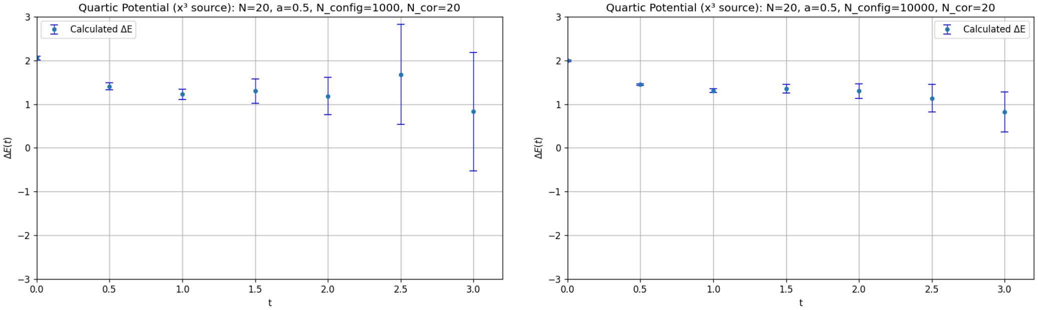


Figure: Monte Carlo values $\Delta E(t) = \frac{1}{a} \log \frac{G(t)}{G(t+a)}$ for a quartic oscillator, this time using x^3 as the source and sink; parameters are the same as before.

Exercise 4 - *theory*

Goal:

- Improve Exercise 2 by implementing a binning procedure

Conceptually, it follows the same reasoning as coarse graining in SFT:

- Improves the estimates of statistical errors by grouping (binning) together some ("binsize") configurations, and considering their averages
- There are no correlations if the statistical errors stop depending on "binsize"

Exercise 4 - *code*

```
def binning(data, bin_size):
    N_cf, N = data.shape
    N_bin = N_cf // bin_size #some configurations might be discarded
    binned = np.zeros((N_bin, N))  #initialization of the array that will contain the avarages
    for i in range(N_bin):
        binned[i] = np.mean(data[i * bin_size:(i + 1) * bin_size], axis=0)  #for each block finds avarages
    return binned
```

Exercise 4 - *plots*

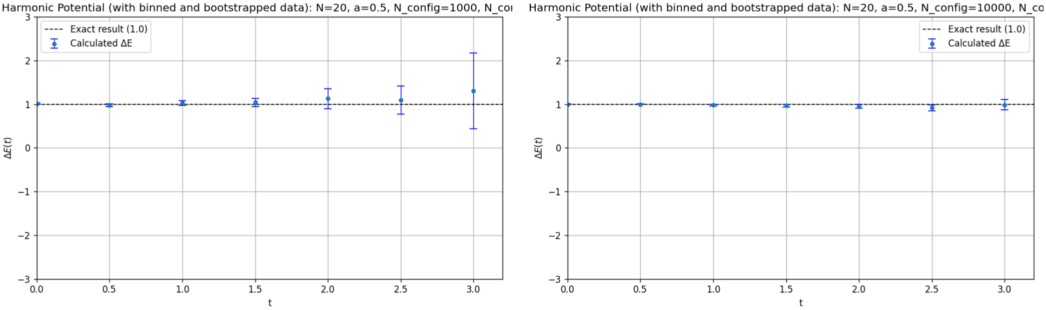
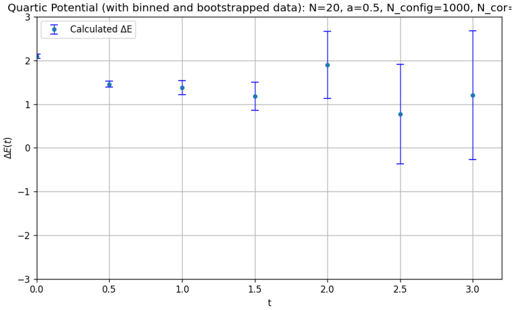


Figure: Monte Carlo values $\Delta E(t) = \frac{1}{a} \log \frac{G(t)}{G(t+a)}$ plotted versus t for an harmonic oscillator, this time implementing also a binning procedure.

Exercise 4 - *plots*



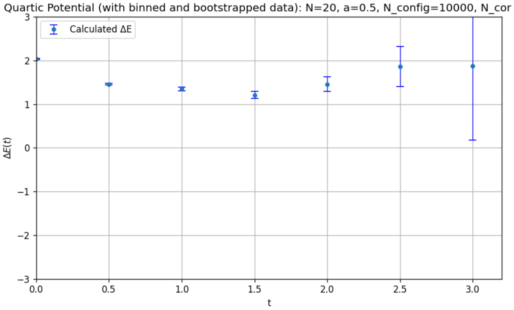


Figure: Monte Carlo values $\Delta E(t) = \frac{1}{a} \log \frac{G(t)}{G(t+a)}$ plotted versus t for a quartic oscillator, this time implementing also a binning procedure.

Exercise 5 - *theory*

Goal:

- Estimate ΔE using an improved form for the action

Integrate by parts the classical action $\Rightarrow S[x] = \int_{t_i}^{t_f} dt \left(-\frac{1}{2} m x(t) \ddot{x}(t) - V(x(t))\right)$.
Improve it by discretizing better $\ddot{x}(t)$, with an error of order a^4 instead of a^2:

$$\ddot{x}(t) = \Delta^{(2)} x_j - \frac{a^2}{12} \left(\Delta^{(2)}\right)^2 x_j \qquad \left(\text{where} \quad \Delta^{(2)} x_j \equiv \frac{x_{j+1} - 2x_j + x_{j-1}}{a^2}\right)$$

$$\Rightarrow S_{\text{imp}}[x] \equiv \sum_{j=0}^{N-1} a \left[-\frac{1}{2m} x_j \left(\Delta^{(2)} x_j - \frac{a^2}{12} \left(\Delta^{(2)}\right)^2\right) x_j + V(x_j)\right] =$$

$$= a \sum_{j=0}^{N-1} \left[-\frac{1}{2} m x_j \left(\frac{-x_{j+2} + 16 x_{j+1} - 30 x_j + 16 x_{j-1} - x_{j-2}}{12 a^2}\right) + V(x_j)\right]$$

Exercise 5 - *code*

```python
@njit
def action(j, x, use_improved_action):
    jp1 = (j + 1) % N
    jm1 = (j - 1) % N
    jp2 = (j + 2) % N
    jm2 = (j - 2) % N

    if use_improved_action:
        # Improved kinetic term
        kinetic = (1 / (12 * a)) * x[j] * (x[jm2] - 16 * x[jp1] + 15 * x[j] - 16 * x[jm1] + x[jp2])

    else:
        # Standard kinetic term
        kinetic = x[j] * (x[j] - x[jp1] - x[jm1]) / a

    potential = a * harmonic_potential(x[j])
    return kinetic + potential
```

Exercise 5 - *plots*

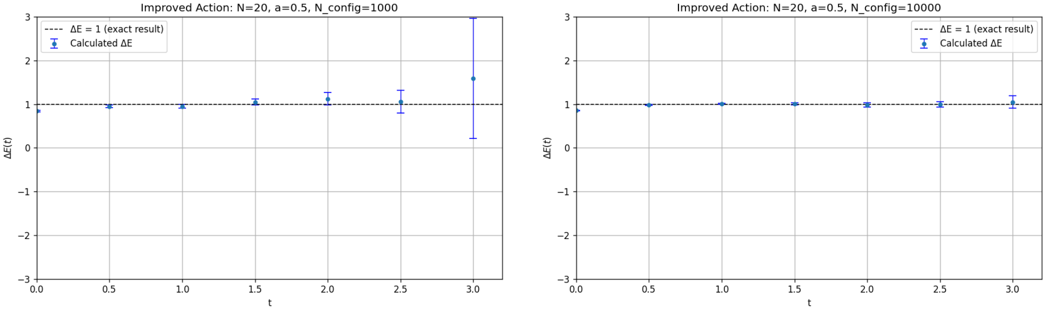


Figure: Monte Carlo values $\Delta E(t) = \frac{1}{a} \log \frac{G(t)}{G(t+a)}$ plotted versus t for an harmonic oscillator, this time improving the action.

Exercise 5 - *plots*

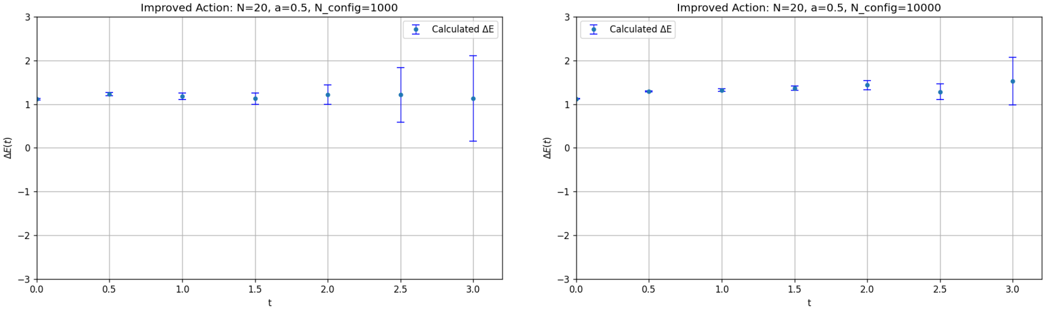


Figure: Monte Carlo values $\Delta E(t) = \frac{1}{a} \log \frac{G(t)}{G(t+a)}$ plotted versus t for a quartic oscillator, this time improving the action.

Exercise 6 - *theory*

Goal:

- Improve ΔE estimation by removing the "ghost-modes", for the harmonic case

Deriving the equation of motion via the variational principle and considering the ansatz $x_j = e^{-\omega t_j}$, one finds solutions for ω (set explicitly now $V(x_j) = \frac{1}{2} m \omega_0^2 x_j^2$): For the unimproved action:

$$\omega^2 = \omega_0^2 \left[1 - \frac{(a\omega_0)^2}{12} + \mathcal{O}((a\omega)^4) \right]$$

For the improved action there are two solutions:

$$\omega^2 = \omega_0^2 \left[1 - \frac{(a\omega_0)^4}{90} + \mathcal{O}((a\omega)^6) \right]$$

$$\omega^2 \approx \left(\frac{2,6}{a} \right)^2$$

Exercise 6 - *theory*

The last solution, i.e. the ghost mode, is an artifact of the improved lattice theory $\rightarrow$ let's remove it!

Trick: change of variables $x_j \rightarrow \tilde{x}_j + \delta\tilde{x}_j$, where $\delta\tilde{x}_j = \xi_1 a^2 \Delta^{(2)}\tilde{x}_j + \xi_2 a^2 \omega_0^2 \tilde{x}_j$

$\Rightarrow \tilde{S}_{imp}(\tilde{x}) = \frac{1}{2} m\tilde{x}_j + \tilde{V}_{imp}(\tilde{x}_j)$, with $\boldsymbol{\tilde{V}_{\textbf{imp}}(\tilde{x}) = \frac{1}{2} m\omega_0^2 \tilde{x}_j^2 \left(1 + \frac{(a\omega_0)^2}{12}\right)}$

Exercise 6 - *plots*

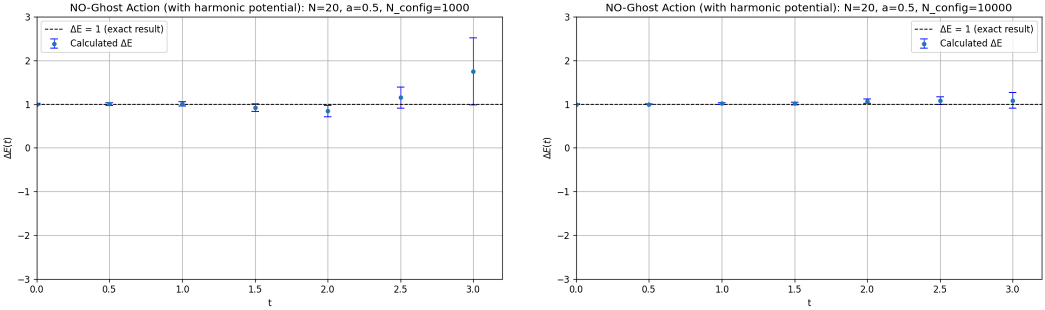


Figure: Monte Carlo values $\Delta E(t) = \frac{1}{a} \log \frac{G(t)}{G(t+a)}$ plotted versus t for an harmonic oscillator, this time removing ghost modes coming from the improved action.

Exercise 6 - *theory*

Goal:

- Generalize the previous reasoning for the anharmonic case

Consider an anharmonic potential (c is dimensionless): $V(x) = \frac{1}{2} m\omega_0^2 x^2 \left(1 + cm\omega_0 x^2\right)$.
Perform the change of variable $x_j \to \tilde{x} + \delta\tilde{x}_j = \tilde{x}_j + \xi_1 a^2 \Delta^{(2)}(\tilde{x}_j) + \xi_2 a^2 \omega_0^2 \tilde{x}_j + \xi_3 a^2 m\omega_0^3 \tilde{x}_j^3$
We obtain:

$$\boldsymbol{\tilde{V}_{\text{imp}}(\tilde{x}) = \frac{1}{2} m\omega_0^2 \tilde{x}^2 \left(1 + cm\omega_0 \tilde{x}^2\right) + \frac{a^2 m\omega_0^4}{24} \left(\tilde{x} + 2cm\omega_0 \tilde{x}^3\right)^2 - a\delta v(\tilde{x}) + \frac{a^3}{2} \delta v(\tilde{x}) + \frac{a^3}{2} \delta v(\tilde{x})^2}$$

where $\delta v(\tilde{x}) = cm\omega_0^3 \tilde{x}^2 / 4$

Exercise 6 - *plots*

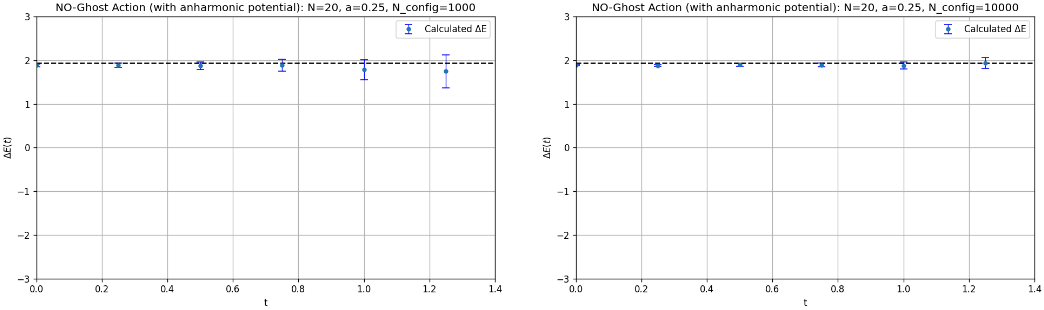


Figure: Monte Carlo values $\Delta E(t) = \frac{1}{a} \log \frac{G(t)}{G(t+a)}$ plotted versus t for an anharmonic oscillator with $a = 0,5$, this time removing ghost modes.

Exercise 7 - *plot*

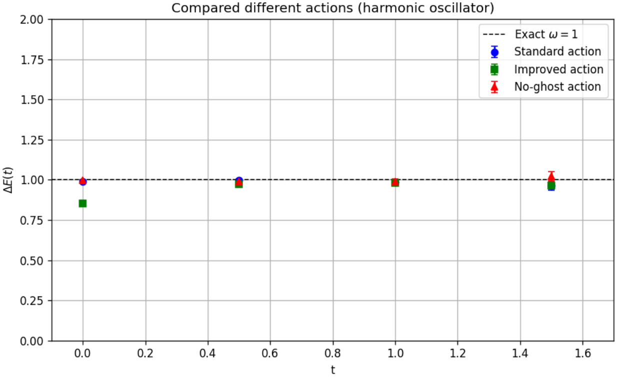


Figure: Comparison between different estimations of ΔE depending on the choice of action's discretization.

Bibliography

- My github: https://github.com/GramelliniElena/Lapage-LatticeQCD
- G. Peter Lapage's paper:https://arxiv.org/abs/hep-lat/0506036
- Another possible solution: https://github.com/Shaun252/latticeQCD_novices