1 HMC for the Schwinger model without fermions

We describe the Hybrid Monte Carlo algorithm for pure U(1) gauge theory. The general idea is to introduce a fictitious Hamiltonian, $H[\pi, U]$, with momenta π and to use the equations of motion to update the gauge configuraions along a trajectory defined by a parameter τ

$$\frac{\partial H}{\partial \pi} = \frac{dU}{d\tau}, \quad \frac{\partial H}{\partial U} = -\frac{d\pi}{d\tau}. \tag{1}$$

The Hamiltonian is given by

$$H = \frac{1}{2}\pi^2 + S[U], \tag{2}$$

where

$$S[U] = \beta \sum_{x\in V} \sum_{\mu<\nu} \mathrm{Re}(1 - U_{\mu\nu}(x)), \quad \pi^2 = \sum_{x\in V} \sum_{\mu} \pi_\mu^2(x). \tag{3}$$

The $\pi_\mu(x)$ are randomly generated with a normal distribution $p[\pi] \propto \exp\big(-\pi^2/2\big)$. The new configuration $[\pi,U]$ is accepted with probability $\exp(-\Delta H)$, where $\Delta H = H[\pi', U'] - H[\pi, U]$.

In order for the HMC to fulfill detailed balance, the solver for the Hamilton equations has to be *reversible* and preserve the measure (check Refs. [1, 2] for details regarding this statement). The former means that if the solver for eqs. (1) is described by a function that maps $f[\pi, U] = [\pi', U']$, then

$$f[\pi, U] = [\pi', U'] \quad \Longleftrightarrow \quad f[-\pi', U'] = [-\pi, U]. \tag{4}$$

Preserving the measure means that the Jacobian of f is equal to one. An integrator that satisfies both conditions is the *leapfrog algorithm*, which updates the values of U_μ and $\pi_\mu(x)$ along a trajectory of length $N\Delta\tau$, $N \in \mathbb{N}$, by evolving $\pi_\mu(x)$ and $U_\mu(x)$ in N steps of size $\Delta\tau$. For a classical mechanics problem, with positions p and q, the leapfrog method is described by[1]

$$\begin{aligned} p\left(n + \frac{1}{2}\right) &= p(n) + \frac{\Delta\tau}{2} F(n), \\ q(n+1) &= q(n) + \Delta\tau p\left(n + \frac{1}{2}\right), \\ p(n) &= p\left(n + \frac{1}{2}\right) + \frac{\Delta\tau}{2} F(n+1), \end{aligned} \tag{5}$$

where n goes from zero to $N-1$ and

$$F(p(n), q(n)) = F(n) = -\frac{\partial H}{\partial q}(n) = \frac{dp}{d\tau}(n). \tag{6}$$

Check Algorithm 1 for the pseudocode.

In order to implement leapfrog for the U(1) theory we first need the force $F = -\partial H/\partial U = -\partial S/\partial U$, *i.e.* for every $x \in V$ and μ we have to compute

$$F(U_\mu(x)) = -\frac{\partial H}{\partial U_\mu}(x) = -\frac{\partial S}{\partial U_\mu}(x). \tag{7}$$

[1] There are alternative ways of implementing the leapfrog algorithm.

Algorithm 1 Leapfrog algorithm

```
$\Delta\tau = trajLength/N$
$p = p_0$ (initial momentum)
$q = q_0$ (initial position)
**for** $n = 0, \dots, N-2$ **do**
    $p+ = 0.5\Delta\tau F(p, q)$
    $q+ = \Delta\tau\, p$
    $p+ = 0.5\Delta\tau F(p, q)$
**end for**
    **return** $p, q$
```

In group theory, we can compute the derivative with respect to a group element according to

$$\frac{df(U)}{dU} = \sum_{a=1}^{N} \left.\frac{df(e^{i\omega T^a}U)}{d\omega}\right|_{\omega=0}, \tag{8}$$

where T^a are the group generators. For U(1) this is just

$$\frac{df(U)}{dU} = \left.\frac{df(e^{i\phi}U)}{d\phi}\right|_{\phi=0}. \tag{9}$$

Derivating the action results in

$$F[U_\mu(x)] = -\beta\, \mathrm{Im}\left[U_\mu(x)K_\mu^\dagger(x)\right] \tag{10}$$

where

$$K_\mu(x) = \sum_{\nu\neq\mu}\left[U_\nu(x)U_\mu(x+\hat{\nu})U_\nu^\dagger(x+\hat{\mu}) + U_\nu^\dagger(x-\hat{\nu})U_\mu(x-\hat{\nu})U_\nu(x+\hat{\mu}-\hat{\nu})\right] \tag{11}$$

is the staple.

To implement the leapfrog algorithm in a gauge theory we have to be careful to keep the link variables in the group. Therefore, a slight modification to eqs. (6) is needed. One way to do it is shown in Algorithm 2.

With this method, the HMC is implemented as follows:

- Generate a random momentum configuration $p[\pi] \propto \exp\bigl(-\pi^2/2\bigr)$.
- Use the leapfrog algorithm to evolve the configuration $[\pi, U] \to [\pi', U']$ along a trajectory of fixed size and N steps (usually trajLength = 1 and $N = 10$ works well, although this has to be tuned depending on the value of β).
- Accept or reject the new configuration $[\pi', U']$ with probability $\exp(-\Delta H)$, where $\Delta H = H[\pi', U'] - H[\pi, U]$. The acceptance rate depends highly on the trajectory and the number of steps.

These three steps are repeated to generate a set of configurations $[U]$. Each time the momenta are randomly generated once again.

Some results are shown in Fig. 1.

Algorithm 2 Leapfrog for U(1) theory

```
$\Delta\tau = trajLength/N$
$\pi = \pi_0$ (random momentum sampled from a normal distribution)
$U = U_0$ (initial gauge configuration)
Forces = Force($U$) (compute the force for the initial gauge configuration)
**for** $n = 0, \dots, N-2$ **do**
    (Loops over the volume)
    **for** $x = 0, \dots, N_x$ **do**
        **for** $t = 0, \dots, N_t$ **do**
            **for** $\mu = 0, \dots, 1$ **do**
                $\pi[x,t,\mu] += 0.5\Delta\tau \text{Forces}[x,t,\mu]$
                $U[x,t,\mu] *= \exp(i\Delta\tau\, \pi[x,t,\mu])$
                $\pi[x,t,\mu] += 0.5\Delta\tau \text{Forces}[x,t,\mu])$
            **end for**
        **end for**
    **end for**
    Forces = Force($U$) (recompute forces)
**end for**
    **return** $U, \pi$
```

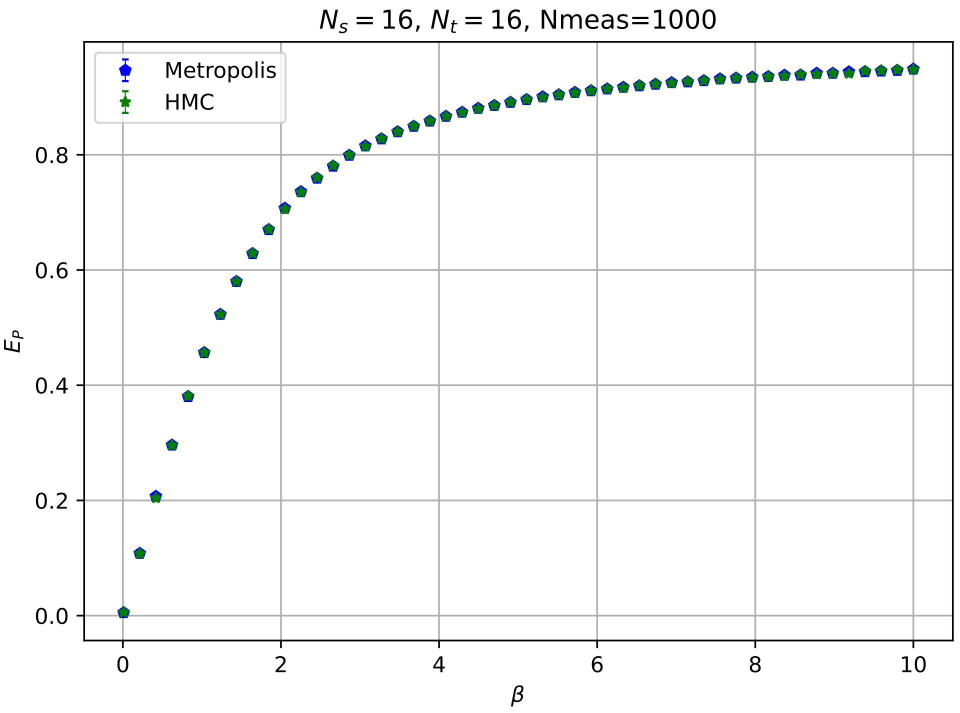

Figure 1: Average plaquette value. We compare the results of the Metropolis and HMC algorithms. For $\beta < 5$ the trajectory length is 1 and leapfrog performs 10 steps. For $\beta \geq 5$ the update is along a trajectory of length 2 with 40 steps

2 Simulation of the 2-flavor Schwinger model

The Hybrid Monte Carlo algorithm for a U(1) gauge theory with fermions is essentially the same as the quenched case. This means that the leapfrog algorithm and the molecular dynamics step are not modified. However, computing the force and the action is more complicated. The Hamiltonian is given by

$$H = \frac{1}{2}\pi^2 + S[U, \psi, \psi^\dagger], \tag{12}$$

where the action is

$$S[U, \psi, \psi^\dagger] = \beta \sum_{\mathbf{n}} \sum_{\mu<\nu} \mathrm{Re}\,(1 - U_{\mu\nu}(\mathbf{n})) + \sum_{\mathbf{n},\mathbf{n'}} \sum_{\alpha,\beta=0}^{1} \psi^\dagger_\alpha(\mathbf{n'})(D)^{\alpha\beta}_{\mathbf{n'},\mathbf{n}}\psi_\beta(\mathbf{n}), \tag{13}$$

where D is a discretization of the Dirac operator. For instance, with Wilson fermions (in lattice units) we have

$$D\left[\mathbf{n'}, \mathbf{n}\right]^{\alpha\beta} = (m_0 + 2)\,\delta^{\alpha\beta}\delta_{\mathbf{n'},\mathbf{n}} - \frac{1}{2}\sum_{\mu=0}^{1}\Big[(1-\sigma_\mu)^{\alpha\beta}\,U_\mu(\mathbf{n'})\delta_{\mathbf{n'}+\hat{\mu},\mathbf{n}} + (1+\sigma_\mu)^{\alpha\beta}\,U^\dagger_\mu(\mathbf{n'}-\hat{\mu})\delta_{\mathbf{n'}-\hat{\mu},\mathbf{n}}\Big]. \tag{14}$$

The index $\mu = 0$ refers to time and $\mu = 1$ to space and m_0 is the bare mass. We choose the following σ matrices

$$\sigma_0 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_1 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}. \tag{15}$$

The partition function is

$$Z = \int \mathcal{D}[U]\mathcal{D}[\psi^\dagger]\mathcal{D}[\psi]\exp(-S_G - S_F) = \int \mathcal{D}[U]\exp(-S_G)\det(D), \tag{16}$$

where S_G corresponds to the gauge term and S_F to the fermions. The probability distribution of having a configuration $[U, \psi, \psi^\dagger]$ is

$$P[U] = \frac{1}{Z}e^{-S_G[U]}\det(D), \tag{17}$$

which involves calculating the fermion determinant. This is computationally expensive. A way of replacing this step by the inversion of $DD^\dagger$ is with the introduction of *pseudofermions*. These are complex scalar random fields $\Phi = D\chi$, where χ is a real vector sampled from a random normal distribution. Their utility is due to the following property, which can be proved for two degenerate fermions (check Ref. [1])

$$\begin{aligned}
\det(D_u)\det(D_d) = \det(D)\det(D) &= \det(D)\det(D^\dagger) \\
&= \det(DD^\dagger) \\
&= \int \mathcal{D}[\psi_u]\mathcal{D}[\psi^\dagger_u]\mathcal{D}[\psi_d]\mathcal{D}[\psi^\dagger_d]\exp\left(-\psi^\dagger_u D\psi_u - \psi^\dagger_d D\psi_d\right) \\
&= \pi^{-N}\int \mathcal{D}[\Phi_R]\mathcal{D}[\Phi_L]\exp\left(-\Phi^\dagger(DD^\dagger)^{-1}\Phi\right) \\
&= \det(DD^\dagger).
\end{aligned} \tag{18}$$

We used the following propery

$$D^\dagger = \gamma_5 D\gamma_5 \quad \Rightarrow \det(D^\dagger) = \det(\gamma_5 D\gamma_5) = \det(\gamma_5^2)\det(D) = \det(D). \tag{19}$$

Therefore, for two degenerate flavors one can work with an effective action

$$S[U,\Phi] = \beta \sum_{\mathbf{n}} \sum_{\mu<\nu} \mathrm{Re}\,(1 - U_{\mu\nu}(\mathbf{n})) + \sum_{\mathbf{n},\mathbf{n}'} \sum_{\alpha,\beta=0}^{1} \Phi^{\dagger}_{\alpha}(\mathbf{n}')(DD^{\dagger})^{-1\,\alpha\beta}_{\mathbf{n}',\mathbf{n}} \Phi_{\beta}(\mathbf{n}), \tag{20}$$

which together with property (18) give an alternative for computing the determinant.

The Dirac matrix for the Schwinger model has $(2N_xN_t)^2$ entries. Assembling the matrix is possible, but it might be not a good idea for large lattices. Instead, it is convenient to define the matrix operations

$$\begin{aligned}(D\psi)_{\alpha}(\mathbf{n}') &= \sum_{\mathbf{n}} \sum_{\beta=0}^{1} D\left[\mathbf{n}',\mathbf{n}\right]^{\alpha\beta} \psi_{\beta}(\mathbf{n}) \\ &= (m_0+2)\psi_{\alpha}(\mathbf{n}') \\ &- \frac{1}{2}\sum_{\beta=0}^{1}\sum_{\mu=0}^{1} \Big[(1-\sigma_{\mu})^{\alpha\beta} U_{\mu}(\mathbf{n}')\psi_{\beta}(\mathbf{n}'+\hat{\mu}) + (1+\sigma_{\mu})^{\alpha\beta} U^{\dagger}_{\mu}(\mathbf{n}'-\hat{\mu})\psi_{\beta}(\mathbf{n}'-\hat{\mu})\Big]\end{aligned} \tag{21}$$

and

$$\begin{aligned}(D^{\dagger}\psi)_{\alpha}(\mathbf{n}') &= \sum_{\mathbf{n}} \sum_{\beta=0}^{1} D^{\dagger}\left[\mathbf{n}',\mathbf{n}\right]^{\alpha\beta} \psi_{\beta}(\mathbf{n}) \\ &= (m_0+2)\psi_{\alpha}(\mathbf{n}') \\ &- \frac{1}{2}\sum_{\beta=0}^{1}\sum_{\mu=0}^{1} \Big[(1-\sigma_{\mu})^{\alpha\beta} U^{\dagger}_{\mu}(\mathbf{n}'-\hat{\mu})\psi_{\beta}(\mathbf{n}'-\hat{\mu}) + (1+\sigma_{\mu})^{\alpha\beta} U_{\mu}(\mathbf{n}')\psi_{\beta}(\mathbf{n}'+\hat{\mu})\Big].\end{aligned} \tag{22}$$

The fields ψ_{β} have anti-periodic boundary conditions in the time direction, while the gauge links have periodic boundary in every direction. This has to be considered when implementing eqs. (21) and (22).

To invert the Hermitian matrix $DD^{\dagger}$ the easiest method is *conjugate gradient.* This approach works for symmetric positive-definite matrices, which is the case of $DD^{\dagger}$. Close the chiral limit this method is quite inefficient and takes many iterations to converge. In such cases better methods, such as multigrid, are convenient. We will stick to conjugate gradient for simplicity. However, we remark that this method is not useful for inverting D, because its eigenvalues are complex (not a positive-definite matrix). Still, just for the simulation it is enough to invert the product $DD^{\dagger}$. For computing hadronic correlation functions, the inverse of D (the propagator) is needed. A matrix inversion method for that case is Bi-CGR, a generalization of conjugate gradient.

We will not explain the intuition behind conjugate gradient here and instead we only write the steps to implement the method. If we want to solve $A\mathbf{x} = \mathbf{b}$ we follow these steps:

- Propose an initial solution $\mathbf{x}_0$.
- Compute $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$ and define the initial searching direction for the solution as $\mathbf{d}_0 = \mathbf{r}_0$
- Calculate[2]

$$\alpha_i = \frac{\langle \mathbf{r}_i, \mathbf{r}_i\rangle}{\langle \mathbf{d}_i, A\mathbf{d}_i\rangle}$$

[2]For complex vectors, the internal product is $\langle \mathbf{x}, \mathbf{y}\rangle = \sum_i x_i y_i^*$.

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i \mathbf{d}_i$$
$$\mathbf{r}_{i+1} = \mathbf{r}_i - \alpha_i A \mathbf{d}_i$$
$$\beta_{i+1} = \frac{\langle \mathbf{r}_{i+1}, \mathbf{r}_{i+1} \rangle}{\langle \mathbf{r}_i, \mathbf{r}_i \rangle}$$
$$\mathbf{d}_{i+1} = \mathbf{r}_{i+1} + \beta_{i+1} \mathbf{d}_i$$

- Repeat until err= $\langle \mathbf{r}_{i+1}, \mathbf{r}_{i+1} \rangle$ <tol, where we have the freedom to choose the tolerance.

For the simulation $A = DD^\dagger$. In order to avoid storing the matrix, we work with eqs. (21) and (22).

Now we proceed to compute the force, which is used in the leapfrog algorithm. We use the following equation for the derivative

$$\frac{\partial f(U)}{\partial U} \equiv \left. \frac{\partial f\left(e^{i\omega_\mu(z)} U\right)}{\partial \omega_\mu(z)} \right|_{\omega_\mu(z)=0} . \tag{23}$$

Then

$$F_\mu(\mathbf{n}) \equiv - \left. \frac{\partial \mathcal{H}(U,\pi)}{\partial \omega_\mu(\mathbf{n})} \right|_{\omega_\mu(\mathbf{n})=0} = - \left. \frac{\partial S[U,\psi,\psi^\dagger]}{\partial \omega_\mu(\mathbf{n})} \right|_{\omega_\mu(\mathbf{n})=0} \tag{24}$$
$$= F_\mu^{gauge}(\mathbf{n}) - \left. \frac{\partial \left[\Phi^\dagger (DD^\dagger)^{-1} \Phi\right]}{\partial \omega_\mu(\mathbf{n})} \right|_{\omega_\mu(\mathbf{n})=0}$$
$$= F_\mu^{gauge}(\mathbf{n}) - \Phi^\dagger \left. \frac{\partial (DD^\dagger)^{-1}}{\partial \omega_\mu(\mathbf{n})} \right|_{\omega_\mu(\mathbf{n})=0} \Phi. \tag{25}$$

To perform the derivative consider $A = DD^\dagger$ and note that

$$\frac{\partial A^{-1}(x)}{\partial x} = \frac{\partial \left[A^{-1} A A^{-1}(x)\right]}{\partial x} = 2 \frac{\partial A^{-1}(x)}{\partial x} + A^{-1}(x) \frac{\partial A(x)}{\partial x} A^{-1}(x). \tag{26}$$

Thus

$$\frac{\partial A^{-1}(x)}{\partial x} = -A^{-1}(x) \frac{\partial A(x)}{\partial x} A^{-1}(x). \tag{27}$$

Then

$$\Phi^\dagger \frac{\partial (DD^\dagger)^{-1}}{\partial \omega_\mu(\mathbf{n})} \Phi = - \left(\left(DD^\dagger\right)^{-1} \Phi \right)^\dagger \cdot \left[\frac{\partial D}{\partial \omega_\mu(\mathbf{n})} D^\dagger + D \frac{\partial D^\dagger}{\partial \omega_\mu(\mathbf{n})} \right] \cdot \left(\left(DD^\dagger\right)^{-1} \Phi \right). \tag{28}$$

We can simplify this expression since

$$\left[\frac{\partial D}{\partial \omega_\mu(\mathbf{n})} D^\dagger + D \frac{\partial D^\dagger}{\partial \omega_\mu(\mathbf{n})} \right] = 2\mathrm{Re} \left[\frac{\partial D}{\partial \omega_\mu(\mathbf{n})} D^\dagger \right]. \tag{29}$$

Taking into account that $\psi^\dagger A \psi$ is real as long as A is real (which is the case of Re $\left[\frac{\partial D}{\partial \omega_\mu(\mathbf{n})} D^\dagger\right]$) we have

$$\Phi^\dagger \frac{\partial (DD^\dagger)^{-1}}{\partial \omega_\mu(\mathbf{n})} \Phi = -2\mathrm{Re} \left[\left(\left(DD^\dagger\right)^{-1} \Phi \right)^\dagger \cdot \left[\frac{\partial D}{\partial \omega_\mu(\mathbf{n})} D^\dagger \right] \cdot \left(\left(DD^\dagger\right)^{-1} \Phi \right) \right]. \tag{30}$$

Therefore, we only need one derivative

$$\begin{aligned}\frac{\partial D\left[\mathbf{n}',\mathbf{n}\right]^{\alpha\beta}}{\partial\omega_\mu(\mathbf{z})} &= \frac{\partial}{\omega_\mu(\mathbf{z})}\Bigg((m_0+2)\,\delta^{\alpha\beta}\delta_{\mathbf{n}',\mathbf{n}} \\ &\quad - \frac{1}{2}\sum_{\nu=0}^{1}\Big[(1-\sigma_\nu)^{\alpha\beta}\,e^{i\omega_\nu(\mathbf{n}')}U_\nu(\mathbf{n}')\delta_{\mathbf{n}'+\hat{\nu},\mathbf{n}} + (1+\sigma_\nu)^{\alpha\beta}\,U_\nu^\dagger(\mathbf{n}'-\hat{\nu})\delta_{\mathbf{n}'-\hat{\nu},\mathbf{n}}\Big]\Bigg)\Bigg|_{\omega_\mu(\mathbf{z})=0} \\ &= -\frac{\mathrm{i}}{2}\Big[(1-\sigma_\mu)^{\alpha\beta}\,U_\mu(\mathbf{z})\delta_{\mathbf{n}'+\hat{\mu},\mathbf{n}}\delta_{\mathbf{n}'\mathbf{z}} - (1+\sigma_\mu)^{\alpha\beta}\,U_\mu^\dagger(\mathbf{z})\delta_{\mathbf{n}'-\hat{\mu},\mathbf{n}}\delta_{\mathbf{n}\mathbf{z}}\Big]\,. \end{aligned} \tag{31}$$

If we apply a multiplication from both sides by arbitrary fields we get

$$\begin{aligned}\left(\psi^\dagger\frac{\partial D}{\partial\omega_\mu}\chi\right)(\mathbf{z}) &= \sum_{\mathbf{n},\mathbf{n}'}\sum_{\alpha\beta}\psi_\alpha^\dagger(\mathbf{n}')\frac{\partial D\left[\mathbf{n}',\mathbf{n}\right]^{\alpha\beta}}{\partial\omega_\mu(\mathbf{z})}\chi_\beta(\mathbf{n}) \\ &= -\frac{i}{2}\sum_{\alpha\beta}\Big[\psi_\alpha^\dagger(\mathbf{z})(1-\sigma_\mu)^{\alpha\beta}U_\mu(\mathbf{z})\chi_\beta(\mathbf{z}+\hat{\mu}) - \psi_\alpha^\dagger(\mathbf{z}+\hat{\mu})(1+\sigma_\mu)^{\alpha\beta}U_\mu^\dagger(\mathbf{z})\chi_\beta(\mathbf{z})\Big]\,. \end{aligned} \tag{32}$$

The force can be computed according to

$$\begin{aligned}F_\mu(\mathbf{n}) &= F_\mu^{gauge}(\mathrm{n}) \\ &\quad + \mathrm{Im}\left(\sum_{\alpha\beta}\Big[\psi_\alpha^\dagger(\mathbf{n})(1-\sigma_\mu)^{\alpha\beta}U_\mu(\mathbf{n})\chi_\beta(\mathbf{n}+\hat{\mu}) - \psi_\alpha^\dagger(\mathbf{n}+\hat{\mu})(1+\sigma_\mu)^{\alpha\beta}U_\mu^\dagger(\mathbf{n})\chi_\beta(\mathbf{n})\Big]\right), \end{aligned} \tag{33}$$

where $\psi = (DD^\dagger)^{-1}\Phi$ and $\chi = D^\dagger\psi$.

With these ingredients the rest of the HMC is the same as in the pure gauge case. A C++ implementation can be found in Ref. [3]. In Fig. 2 we show the average value of the plaquette but including the fermions. As the bare mass grows bigger, the HMC with fermions converges to the quenched simulation, as it should do.

References

[1] C. Gattringer and C. B. Lang, *Quantum Chromodynamics on the Lattice: An Introductory Presentation*, Lect. Notes. Phys. 788 Springer, (2010).

[2] J. F. Nieto Castellanos, “The 2-flavor Schwinger model at finite temperature and in the delta-regime”, B. Sc. thesis, Universidad Nacional Autónoma de México, (2021).

[3] J. F. Nieto Castellanos. SchwingerModel. https://github.com/Fabian2598/SchwingerModel (2025).

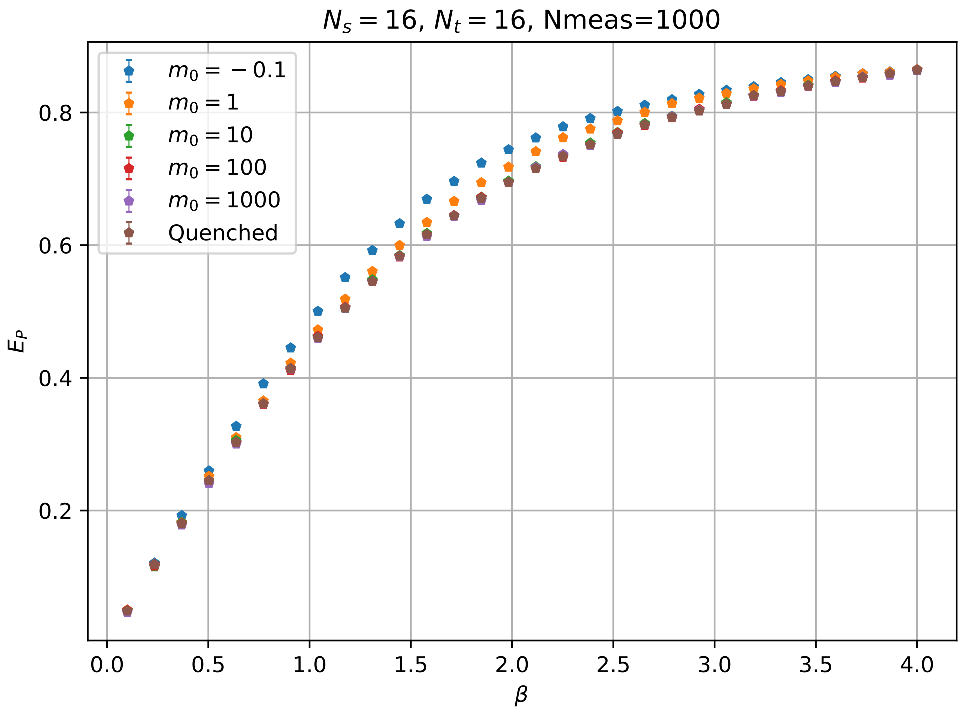

Figure 2: Average plaquette value. The trajectory length is 1 and leapfrog performs 10 steps. We used 1000 configurations with 200 thermalization steps and 1 decorrelation step.