LATTICE PRACTICE HANDS-ON

JACOB FINKENRATH

1. 2D-U(1) Model

To reduce computational costs, we will use a two-dimensional model with U(1) gauge links

1.1. Gauge action.

$$S_G(U) = -\beta \sum_x \mathrm{Re}\{P_{01}(x)\} \tag{1}$$

$$= -\beta \sum_x \mathrm{Re}\{U_0(x)U_1(x+\hat{0}U_0^\dagger(x+\hat{1})U_1^\dagger(x)\} \tag{2}$$

$$= -\beta \sum_x \cos(\theta_0(x) + \theta_1(x+\hat{0}) - \theta_0(x+\hat{1}) - \theta_1(x)) \tag{3}$$

by using the relation

$$U_\mu(x) = \exp(i\theta_\mu(x)) \tag{4}$$

1.2. Fermion action.

We consider two degenerated flavours of Wilson-fermions with action

$$S_F[U,\overline{\psi},\psi] = \sum_{f=u,d} \overline{\psi}'_f (D_W + m_0)\psi'_f \tag{5}$$

where

$$D_W = \sum_{\mu=0}^{1} \frac{1}{2}\{\gamma_\mu(\nabla^*_\mu + \nabla_\mu) - a\nabla^*_\mu\nabla_\mu\} \tag{6}$$

and

$$a\nabla_\mu\psi(x) = U^\dagger_\mu(x)\psi(x+\hat{\mu}) - \psi(x) \tag{7}$$

$$a\nabla^*_\mu\psi(x) = \psi(x) - U_\mu(x-\hat{\mu})\psi(x-\hat{\mu}). \tag{8}$$

(9)

In two dimension the γ matrices are given by the Pauli matrices, $\gamma_\mu = \sigma_\mu$, with $\mu = 0, 1$. We use the following representation

$$\sigma_0 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad \text{and} \quad \sigma_1 = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \tag{10}$$

The fermion action can be written as

$$S_F[U,\overline{\psi},\psi] = \sum_{f=u,d} \overline{\psi}'_f M(\kappa)\psi'_f \tag{11}$$

where $M = 1 - \kappa H_{hop}$ and $\kappa = 1/(2(m+D))$. The hopping matrix H_{hop} is given by

$$H_{hop} = \sum_{\mu=0}^{1} \delta_{x-\hat{mu},y}(1+\gamma_\mu)U_\mu(x-\hat{\mu}) + \delta_{x-\hat{mu},y}(1-\gamma_\mu)U^\dagger_\mu(x) \tag{12}$$

2. HMC with Fermions

Unless differently specified for the simulations in these exercises you can consider the lattice parameters: $\beta = 4.0$, $\kappa = 0.26$ and $L/a = 12$.

2.1. **HMC.**

1. Generate a thermalized configuration. (Measure the provided observables to understand when thermalization is reached.)
2. Implement a check of the reversibility of the HMC. Starting from a set of thermalized configurations for $\theta_\mu(x)$, and measure

$$\Delta_\theta = ||\theta' - \theta|| = \max_{x,\mu} |\theta'_\mu - \theta_\mu(x)| \tag{13}$$

$$\Delta_\pi = ||\pi' - \pi|| = \max_{x,\mu} |\pi'_\mu - \pi_\mu(x)| \tag{14}$$

$$\delta\delta H = |H(\pi',\theta') - H(\pi,\theta)| \tag{15}$$

   with the trajectory length $\tau = n \cdot h$ and

$$(\pi',\theta') = F \circ [I_{MD}(h)]^n \circ F \circ [I_{MD}(h)]^n(\pi,\theta) \quad F(\pi,\theta) = (-\pi,\theta). \tag{16}$$

   – Observe the behavior of Δ_θ, Δ_π and $\delta\delta H$, as τ increases at fixed h.
   – Check how the solver residuum used for the linear solver of the fermion force computations effects the reversibility.
3. A proper implementation of the HMC with a sensible choice of algorithmic parameters should satisfy (within errors)

$$\langle e^{-\delta H} \rangle = 1 \tag{17}$$

2.2. **MD integration.**

The code includes an implementation of the OMF 4th order integrator. Based on that:

- Implement the Leap Frog integrator

$$[I_{LPFR}(h)]^n = \left(e^{h/2\hat{S}} e^{h\hat{T}} e^{h/2\hat{S}}\right)^n \tag{18}$$

  For a more efficient implementation the itermediate force steps can be combined:

$$[I_{LPFR}(h)]^2 = \left(e^{h/2\hat{S}} e^{h\hat{T}} e^{h\hat{S}} e^{h\hat{T}} e^{h/2\hat{S}}\right) \tag{19}$$

2. Study the dependence of δH for the different integrators as a function of h at fixed trajectory length τ. For that, restart from a single configuration and change the number of steps. Is there another way to study it ?
3. What integrator is the most cost effective at the given physical parameters ?

2.2.1. *Topology freezing.* To study topological freezing, the continuum limit of the topological charge in pure U(1) gauge theory can be studied. To detect some non-trivial topology the physical volume should be not too small. Due to that take $R = V/\beta = 40, 80$ and study how the continuum limit is approached by taking $\beta \to \infty$ at fixed R. Plot the Monte Carlo history of the geometrical definition of the topological charge

$$Q \equiv \frac{1}{2\pi} \sum_x \mathrm{Im} \ln \left(P_{01}(x)\right) \tag{20}$$

.

2.2.2. *Forces.* Compare the effect of Hasenbusch mass preconditioning.

2.2.3. *Code.* The main program can be found in *main/schwinger.c* and use the provided *Makefile* to generate binary. Then the program can be executed via
schwinger input cnfg > output
Examples for input files for QED_2 and the pure gauge model can be found in *main/infile_qed* and *main/infile_ym*. Also the readme give some additional infos. The config file is optional, if not provided it starts from a cold configuration with $\theta_\mu(x) = 0$.

The size of the lattice is hard coded and needs to be set before the compilation in the file *include/lattice.h*. Here, also two global errors, the gauge field *gauge[ix][mu]* and the hopping matrix *hop[ix][mu]* is defined (see *modules/geometry/hopping.c* for more details).

The points are ordered via the lexicographic index $ix = x_0 + x_1 L$. The main routine for the HMC is implemented in *modules/update/hmc.c*. Check out *devel/check/check1.c*, which might be handy for developments.

The compiler flags -DMDINT_DBG active additional computations and printout of the HMC forces. It has to be added to the CFLAGS within the Makefile. *The code is a customized version for Lattice Practices 2021 of the schwinger package written by Stefan Schaefer for the Lattice Practices 2018 Edition. We thank the author of the code for shearing this with us, as well as for his useful comments on the exercises.*

3. Equivariant flows

You might be interested to compare the autocorrelation of the HMC with a newer approach, based on a proposal with gauge-equivariant flows. Here we want to refer to the introduction and the jupyter nodebook provided in *arXiv:2101.08176*. The second part is applying the normalizing flows on the 2D-U1 model. The program, which is provided is based on it, with some additional modifications and added parameters. However, the current status is that the torch-library is too new, and training of the neural networks is buggy. I didn't manage to fix this in time but can give general advice on it.