Final project

Inversion of a Dirac matrix

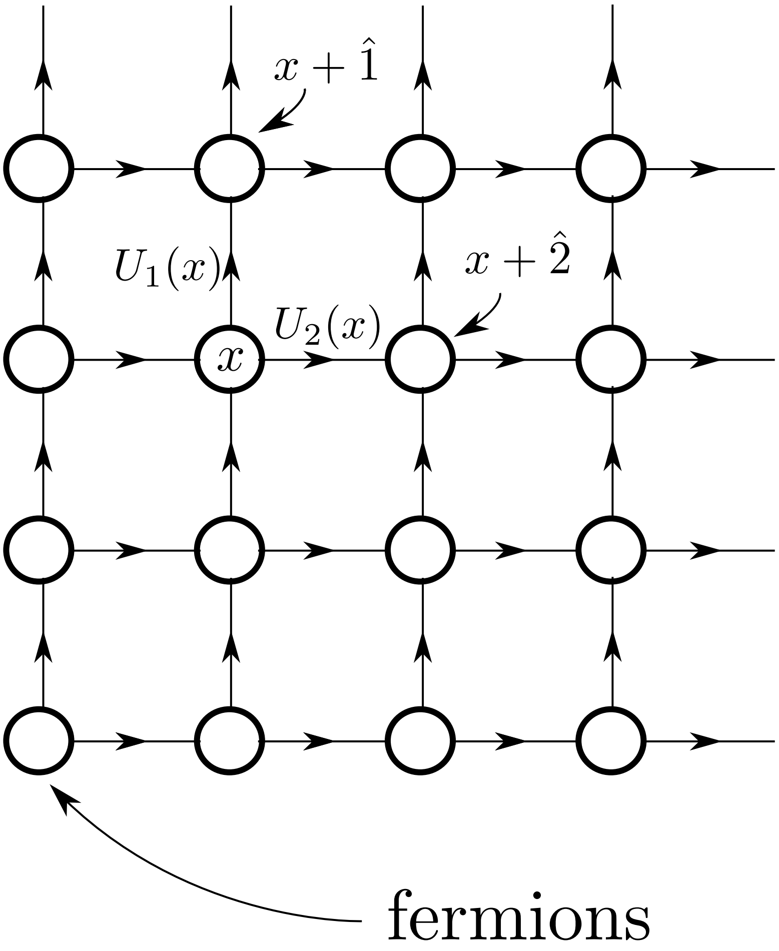
Dr. Lukas Varnhorst

BERGISCHE
UNIVERSITÄT
WUPPERTAL

In lattice calculations, often the following problem appears: A ensamble of configurations of gauge field $U_\mu(x)$ have been generated. To calculate an observable of interest, a certain matrix, which depends on the gauge fields, needs to be inverted and applied to a given vector. Here, we will look at the following simpliefied setting:

Consider a 2 dimensional square lattice of size $N_1 \times N_2$ as a discretization of a 1+1 dimensional space time. A fermion field is discretized in such a way that the field values are specified at each lattice point. Furthermore, a abelian gauge field is introduced at the links between neighboring lattice sites.

This abelian gauge field consists of complex number with absolute square 1. That is, the link between the lattice site x and the neighboring site in the direction μ, $x+\hat{\mu}$, is called $U_\mu(x)$ and has the form $U_\mu(x) = \exp(\mathrm{i}a(x))$ where $a(x)$ is a real number.



The task is now to consider the following matrix:

$$D_{xy} = m\delta_{x,y} + \frac{U_1(x)\delta_{x+\hat{1},y} - U_1^\dagger(x-\hat{1})\delta_{x-\hat{1},y}}{2} + (-1)^{x_1}\frac{U_2(x)\delta_{x+\hat{2},y} - U_2^\dagger(x-\hat{2})\delta_{x-\hat{2},y}}{2} \tag{1}$$

Here, x and y are indices that denote lattice points and $x+\hat{\mu}$ is the index of the neighboring lattice point to x in the μ direction. Furthermore, we assume antiperiodic boundary. That means that we assume periodic boundary conditions but multiply each term in which the periodicity is used by -1.

a) (4 points) Construct a neighbour table, i.e. a $4 \times (N_1N_2)$ dimension array $n_{\mu x}$. Here μ corresponds to the four possible directions (forward in directions 1 or 2 or backwards in direction 1 or 2) and x is the index of a lattice point. Then $n_{\mu x}$ should contain the index of the neighbor of the site with index x in the μ direction.

b) (4 points) Using the result of a), write a function that applies the above matrix (with $U_\mu(x) = 1$ for simplicity and $m = 0.4$) to a arbitrary vector $\vec{v} \in \mathbb{C}^{N_1N_2}$, i.e. that calculates $D\vec{v}$. Note that the function should not explicitly construct D.

c) (4 points) For a small 8×8 lattice, apply the function from b) to each of the unit vectors to construct the full matrix. Plot the magnitude of the matrix elements in a $2d$ plot, so that the structure of the matrix can be observed.

1st intermediate goal

d) (3 points) Implement the CG algorithm discussed in the lecture and use it to calulate $D^{-1}\vec{e}_i$ where $\vec{e}_i$ is the i-th unit vector. Noth that D is not symmetric and positive definite, so you have to solve the equation $(D^\dagger D)\vec{r} = D^\dagger\vec{e}_i$ for r. Construct the matrix D^{-1} from the result and plot it like in c).

e) (3 points) Let us define the correlation function

$$C(t) = \frac{1}{N_1N_2}\sum_x |[D^{-1}\vec{e}_x]_{x+(t\hat{1})}|^2, \tag{2}$$

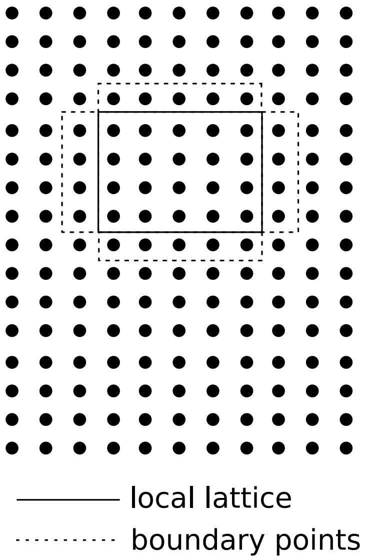
i.e. the absolute value of the inverse matrix between all points separated by a distance t in the 1-direction. Plot the result in a log-plot as a function of t

f) (3 points) Modify the function that applies D such that it can take arbitrary gauge fields $U_\mu(x)$ as additional input.

g) (3 points) Repeat the above for all the gauge fields provided and calculate the average $C(t)$ over all gauge configurations. Plot it again as before.

2nd intermediate goal

h) (3 points) Now, we want to parallelize our code using several processes in the case of $U_\mu(x) = 1$. Each process will have access only to a rectangular part of the lattice of size $N_1^{\text{loc}} \times N_2^{\text{loc}}$. In addition, each process will also need to save one layer of boundary points. Adjust your neighbor table to reflect that structure.



i) (3 points) Adjust the routine that applies D such that it calculates the results for all points in the inner local $N_1^{\text{loc}} \times N_2^{\text{loc}}$ lattice (not the boundarys).

j) (3 points) Now, adjust the CG algorithm such that it will run in parallel. This means that at each iteration previous to applying D to a vector, the contents of the boundarys for that vector have to be communicated between the processes.

k) (3 points) Experiment with different splittings of the full volume into subvolumes of different size processes and measure the timing for each splitting.