Monte Carlo Simulation of the Gross-Neveu Model on the Lattice

Jesús Andrés Espinoza Valverde.

Introduction and Motivation

The GNM is simple toy model in QCD that shares very interesting properties with several Condensed Matter theories, specially in the context of superconductors. In this sense, the calculations on the Gross-Neveu model can be ingeniously interpreted and used for the study of condensed matter systems. This fact is the main motivation for the present work.

This repository contains an algorithmic framework to approximate vacuum expectation values of observables of the Gross-Neveu model for the $(1+1)$-dimensional case, and with just one fermion flavor. The simulations are done on "the lattice", i.e, on a discretized space-time with periodic and anti-periodic boundary conditions. This allow us to treat the infinite-dimensional path integrals of the theory as just highly dimensional. This highly-dimensional scenario fits perfect with the use of Monte Carlo techniques: the errors obtained from the numerical calculations do not depend on the dimensionality of the equations: the central limit theorem ensures us that the error has only to do with the number of stochastic samples used, and with the quality of the random numbers generators.

The proposal is to generalized the algorithms implemented here to be compatible with 3-dimensional and 4-dimensional space-times, (and even for $(1+n)$-space-times), and for multiple fermion flavors, with degenerated and non-degenerated mass matrices. This because the condensed matter analogies that we pretend to explore are related with these dimensional-increased space-times. The Monte Carlo technique used in this implementation (importance sampling) is too rudimentary and can be replaced by much more sophisticated techniques such us Hybrid-Monte-Carlo, which is related with a high computational efficiency and specially good dimensional scalability. The current code has only two types of observables implemented, and this number could be increased as well.

Action, Partition Function and Vacuum expected values of the Gross-Neveu model in the continuum

The Lagrangian of that describes the theory is the following:

$$\mathcal{L}[\bar{\psi},\psi,\phi]=\sum_{f=1}^{N_f}\bar{\psi}(\mathbf{x})[\gamma_\mu\partial_\mu+m^{(f)}+\sqrt{g}\phi(\mathbf{x})]\psi^{(f)}(\mathbf{x})+\frac{1}{2}\phi^2(\mathbf{x}).$$

Here f runs over the different fermionic flavors, $m^{(f)}$ is mass matrix, g is the coupling constant and γ_μ are simply the Dirac matrices. If we use this Lagrangian to model QCD systems we can interpret $\bar{\psi}(\mathbf{x})$ as anti-quark field, $\psi(\mathbf{x})$ as quark field and $\phi(\mathbf{x})$ as the gloun field. The action of the theory is then:

$$S[\bar{\psi},\psi,\phi]=\int d^2x\ \mathcal{L}[\bar{\psi},\psi,\phi],$$

and therefore the partition function for the Gross-Neveu model is given by:

$$Z_{GN} = \int \mathcal{D}[\bar{\psi}, \psi, \phi] e^{-S[\bar{\psi}, \psi, \phi]},$$

with the integration defined as:

$$\mathcal{D}[\bar{\psi}, \psi, \phi] = \prod_{\mathbf{x}} d\phi(\mathbf{x}) \prod_{f=1}^{N_f} d\psi^{(f)}(\mathbf{x}) d\bar{\psi}^{(f)}(\mathbf{x}).$$

Then, the vacuum expectation values are given by:

$$\langle O \rangle = \frac{1}{Z_{GN}} \int \mathcal{D}[\bar{\psi}, \psi, \phi] e^{-S[\bar{\psi}, \psi, \phi]} O[\bar{\psi}, \psi, \phi].$$

Gross-Neveu Model on the Lattice

We associate to the continuum space-time a square like lattice Λ with spacing a, in which we specify each space-time point $\mathbf{n}$ as (in the 2D case) as $\mathbf{n} = n_t \hat{t} + n_s \hat{s}$, with $n_{t,s} = 0, \ldots, L_{t,s} - 1$. In this way the lattice fields can be represented by $\bar{\psi}(\mathbf{n}a)$, $\psi(\mathbf{n}a)$ and $\phi(\mathbf{n}a)$. The partial derivatives and the integrals are then replaced by their discrete analogues. To have periodic an anti-periodic boundary conditions we impose on every function on the lattice the condition $f(\mathbf{n} + \hat{\mu} L_\mu) = e^{2\pi i \vartheta_\mu} f(\mathbf{n})$, with $\vartheta_\mu = 0$ for p.b.c $\vartheta_\mu = 1/2$ for a.p.b.c, where $\hat{\mu} = \hat{t}\,, \hat{s}$.

Partition function on the lattice

On the lattice is easy to split the action on its fermionic and scalar parts:

$$S_F[\hat{\psi}, \psi, \phi] = \sum_{\mathbf{n},\mathbf{m} \in \Lambda} \hat{\psi}(\mathbf{n}) M(\mathbf{n}, \mathbf{m}) \psi(\mathbf{m}), \quad S_S[\phi] = \frac{1}{2} \sum_{\mathbf{n} \in \Lambda} \phi^2(\mathbf{n}),$$

where M is the Dirac Matrix given by:

$$M(\mathbf{n}, \mathbf{m})_{\alpha\beta} = [m + \sqrt{g} Q(\mathbf{n})] \delta_{\mathbf{n},\mathbf{m}} \delta_{\alpha\beta} - (2a)^{-1} \sum_{\mu=1}^{2} (\gamma_\mu)_{\alpha\beta} [\delta_{\mathbf{n}+\hat{\mu},\mathbf{m}} - \delta_{\mathbf{n}-\hat{\mu},\mathbf{m}}].$$

The partition function is then written as:

$$Z_{GN} = \int \mathcal{D}[\bar{\psi}, \psi, \phi] e^{-S_F[\bar{\psi}, \psi, \phi] - S_S[\phi]},$$

The fermionic part of this partition function is then

$$Z_{GN}^{(F)} = \int \mathcal{D}[\bar{\psi}, \psi] \ \exp\left(- \sum_{\mathbf{n},\mathbf{m} \in \Lambda} \hat{\psi}(\mathbf{n}) M(\mathbf{n}, \mathbf{m}) \psi(\mathbf{m}) \right),$$

This part can be integrated out easily recognizing the fermion fields as Grassmann numbers (they obey Pauli exclusion principle), being their underlying algebra characterize by the anti-commutation relation $\{\eta_i, \eta_j\} = \eta_i \eta_j + \eta_j \eta_i = 0$, which implies their nilpotency property: $\eta_i^2 = 0$. Using this we get:

$$Z_{GN}^{(F)} = \det(M) \int \prod_{\mathbf{n}} d\hat{\psi}(\mathbf{n}) d\psi'(\mathbf{n}) \left[1 + \psi'(\mathbf{n}) \bar{\psi}(\mathbf{n})\right]; \quad \psi'(\mathbf{n}) = \sum_{\mathbf{m}} M(\mathbf{n}, \mathbf{m}) \psi(\mathbf{m}),$$

$$Z_{GN}^{(F)} = \det(M).$$

Using this result, we express the total partition function as:

$$Z_{GN} = \int \prod_{\mathbf{n} \in \Lambda} \frac{d\phi(\mathbf{n})}{\sqrt{2\pi}} e^{-\phi^2(\mathbf{n})/2} \det(M[\phi]),$$

Vacuum Expectation Values on the Lattice

Using the previous results we transform the expression for vacuum expectation values from the continuum to the lattice as:

$$\langle O \rangle = \frac{1}{Z_{GN}} \int \mathcal{D}[\bar{\psi}, \psi, \phi] e^{-S[\bar{\psi}, \psi, \phi]} O[\bar{\psi}, \psi, \phi].$$

$$\rightarrow$$

$$\langle O \rangle = \frac{\int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} \prod_{\mathbf{n} \in \Lambda} \frac{d\phi(\mathbf{n})}{\sqrt{2\pi}} e^{-\phi^2(\mathbf{n})/2} \det(M[\phi]) \; O[M[\phi], \phi]}{\int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} \prod_{\mathbf{n} \in \Lambda} \frac{d\phi(\mathbf{n})}{\sqrt{2\pi}} e^{-\phi^2(\mathbf{n})/2} \det(M[\phi])} \tag{1}$$

We compute the required expectation values computing this expression numerically. The integrals contained (1) are extremely high dimensional: each point in the lattice contributes with one complete integral.

Dirac Matrix Explicit Construction

On the lattice, the fermionic part of the action is given by

$$S_F[\bar{\psi}, \psi, Q] = \sum_{\mathbf{n}, \mathbf{m} \in \Lambda} \bar{\psi}(\mathbf{n}) M(\mathbf{n}, \mathbf{m}) \psi(\mathbf{m}).$$

Where, in index form:

$$M(\mathbf{n}, \mathbf{m})_{\alpha\beta} = [m + \sqrt{g} Q(\mathbf{n})] \delta_{\mathbf{n},\mathbf{m}} \delta_{\alpha\beta} - (2a)^{-1} \sum_{\mu=1}^{2} (\gamma_\mu)_{\alpha\beta} [\delta_{\mathbf{n}+\hat{\mu},\mathbf{m}} - \delta_{\mathbf{n}-\hat{\mu},\mathbf{m}}].$$

Now, in order to avoid the doubling problem, we add to the last expression the Wilson term in the real space representation (omitting spin indices):

$$M(\mathbf{n}, \mathbf{m}) - \frac{a}{2} \sum_{\mu=1}^{2} \frac{\delta_{\mathbf{n}+\hat{\mu},\mathbf{m}} - 2\delta_{\mathbf{n},\mathbf{m}} + \delta_{\mathbf{n}-\hat{\mu},\mathbf{m}}}{a^2} \longrightarrow M(\mathbf{n}, \mathbf{m}),$$

Which implies:

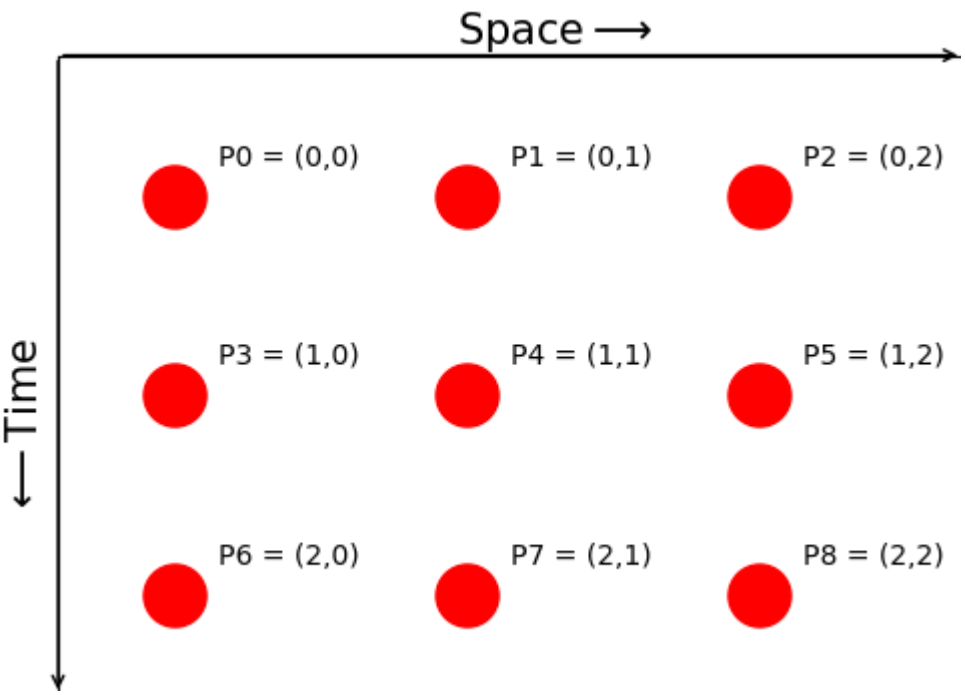
$$M(\mathbf{n}, \mathbf{m})_{\alpha\beta} = [m + \sqrt{g} Q(\mathbf{n})] \delta_{\mathbf{n},\mathbf{m}} \delta_{\alpha\beta} - (2a)^{-1} \sum_{\mu=1}^{2} \{-2\delta_{\mathbf{n},\mathbf{m}} + (I + \gamma_\mu) \, \delta_{\mathbf{n}+\hat{\mu},\mathbf{m}} + (I - \gamma_\mu) \, \delta_{\mathbf{n}-\hat{\mu},\mathbf{m}}\}$$

Then, defining $\Gamma_{\pm\mu} = \frac{1}{2}(I \pm \gamma_\mu)$, and suppressing the parameter a, we obtain:

$$M(\mathbf{n},\mathbf{m}) = [2 + m + \sqrt{g}Q(\mathbf{n})]\delta_{\mathbf{n},\mathbf{m}} - \sum_{\mu=1}^{2}\{\Gamma_\mu \delta_{\mathbf{n}+\hat{\mu},\mathbf{m}} + \Gamma_{-\mu}\delta_{\mathbf{n}-\hat{\mu},\mathbf{m}}\}$$

Labeling the lattice

For a $N_1 \mathrm{x} N_2$ lattice, we associate a label to each of the points applying the following fashion: if the point have coordinates given by (n_1, n_2); $n_1 = 0, \ldots, N_1 - 1$; $n_2 = 0, \ldots, N_2 - 1$, then we associate to it the label $N = n_2 N_2 + n_1$. For example, for a $3 \mathrm{x} 3$ lattice, we obtain:



With the above labeling it is possible to associate a single N number to each vector $\mathbf{n}$, this simplifies the index specification of the matrix $M(\mathbf{n},\mathbf{m})$ entries.

Explicit Construction of M(n,m)

Let's start with finding the explicit form of $\delta_{\mathbf{n},\mathbf{m}}$, $\delta_{\mathbf{n}+\hat{\mu},\mathbf{m}}$, $\delta_{\mathbf{n}-\hat{\mu},\mathbf{m}}$. Taking into account that $\hat{\mu}$ can take two values $\hat{t} = (1,0)$ and $\hat{s} = (0,1)$, we obtain:

$$\delta_{\mathbf{n},\mathbf{m}} = \delta_{[n_1,n_2],[m_1,m_2]} = \delta_{N,N'}; \quad \begin{cases} N = n_2 N_2 + n_1 \\ N' = m_2 N_2 + m_1 \end{cases}$$

$$\delta_{\mathbf{n}+\hat{1},\mathbf{m}} = \delta_{[n_1+1,n_2],[m_1,m_2]} = \delta_{N,N'}; \quad \begin{cases} r_1 = n_1 + 1 \mod N_2 \\ N = n_2 N_2 + r_1 \\ N' = m_2 N_2 + m_1 \end{cases}$$

$$\delta_{\mathbf{n}-\hat{1},\mathbf{m}} = \delta_{[n_1-1,n_2],[m_1,m_2]} = \delta_{N,N'}; \quad \begin{cases} r_1 = n_1 - 1 \mod N_2 \\ N = n_2 N_2 + r_1 \\ N' = m_2 N_2 + m_1 \end{cases}$$

$$\delta_{\mathbf{n}+\hat{2},\mathbf{m}} = \delta_{[n_1,n_2+1],[m_1,m_2]} = \delta_{N,N'}; \quad \begin{cases} r_2 = n_2 + 1 \mod N_1 \\ N = r_2 N_2 + n_1 \\ N' = m_2 N_2 + m_1 \end{cases}$$

$$\delta_{\mathbf{n}-\hat{2},\mathbf{m}} = \delta_{[n_1,n_2-1],[m_1,m_2]} = \delta_{N,N'}; \quad \begin{cases} r_2 = n_2 - 1 \mod N_1 \\ N = r_2 N_2 + n_1 \\ N' = m_2 N_2 + m_1 \end{cases}$$

The Modulo operations are introduced to take into account periodic boundary conditions. To make the p.b.c in the $\hat{t}$ anti-periodic we simply make the following substitutions:

$$\delta_{\mathbf{n}+\hat{1},\mathbf{m}} \longrightarrow \alpha\delta_{\mathbf{n}+\hat{1},\mathbf{m}}; \quad \begin{cases} \alpha = -1 \text{ if } \mathbf{n}\cdot\hat{1} = N_2 - 1 \\ \alpha = 1 \text{ else} \end{cases}$$

$$\delta_{\mathbf{n}-\hat{1},\mathbf{m}} \longrightarrow \alpha\delta_{\mathbf{n}-\hat{1},\mathbf{m}}; \quad \begin{cases} \alpha = -1 \text{ if } \mathbf{n}\cdot\hat{1} = 0 \\ \alpha = 1 \text{ else} \end{cases}$$

Making the definition $Q'(\mathbf{n}) = 2 + m + \sqrt{g}Q(\mathbf{n})$, and considering a 2x2 lattice, we get:

$$M(\mathbf{n},\mathbf{m}) = \begin{bmatrix} Q'(0,0) & 0 & 0 & 0 \\ 0 & Q'(1,0) & 0 & 0 \\ 0 & 0 & Q'(0,1) & 0 \\ 0 & 0 & 0 & Q'(1,1) \end{bmatrix}$$

$$- \Gamma_1 \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{bmatrix} - \Gamma_{-1} \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix} - \Gamma_2 \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} - \Gamma_{-2} \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} Q'(0,0) & \Gamma_{-1} - \Gamma_1 & -(\Gamma_2 + \Gamma_{-2}) & 0 \\ \Gamma_1 - \Gamma_{-1} & Q'(1,0) & 0 & -(\Gamma_2 + \Gamma_{-2}) \\ -(\Gamma_2 + \Gamma_{-2}) & 0 & Q'(0,1) & \Gamma_{-1} - \Gamma_1) \\ 0 & -(\Gamma_2 + \Gamma_{-2}) & \Gamma_1 - \Gamma_{-1} & Q'(1,1) \end{bmatrix}$$

Now, taking into account the definitions of $\Gamma_{\pm\mu}$:

$$\Gamma_1 = \frac{1}{2}(I + \gamma_1) = \frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$$\Gamma_{-1} = \frac{1}{2}(I - \gamma_1) = \frac{1}{2}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

$$\Gamma_2 = \frac{1}{2}(I + \gamma_2) = \frac{1}{2}\begin{bmatrix} 1 & -i \\ i & 1 \end{bmatrix}$$

$$\Gamma_{-2} = \frac{1}{2}(I - \gamma_2) = \frac{1}{2}\begin{bmatrix} 1 & i \\ -i & 1 \end{bmatrix}$$

And then:

$$\Gamma_1 - \Gamma_{-1} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\Gamma_2 + \Gamma_{-2} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

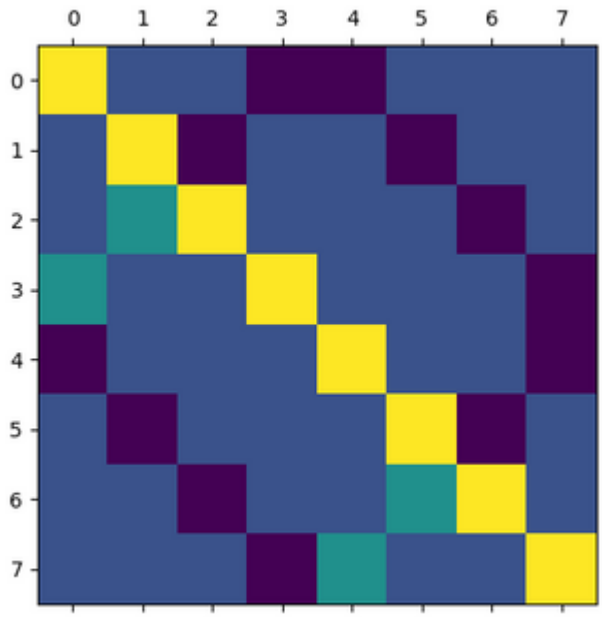
And therefore:

$$M(\mathbf{n},\mathbf{m}) = \begin{bmatrix} \begin{bmatrix} Q'(0,0) & 0 \\ 0 & Q'(0,0) \end{bmatrix} & \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} & \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} & \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} & \begin{bmatrix} Q'(1,0) & 0 \\ 0 & Q'(1,0) \end{bmatrix} & \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} & \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \\ \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} & \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} & \begin{bmatrix} Q'(0,1) & 0 \\ 0 & Q'(0,1) \end{bmatrix} & \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} & \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} & \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} & \begin{bmatrix} Q'(1,1) & 0 \\ 0 & Q'(1,1) \end{bmatrix} \end{bmatrix}$$

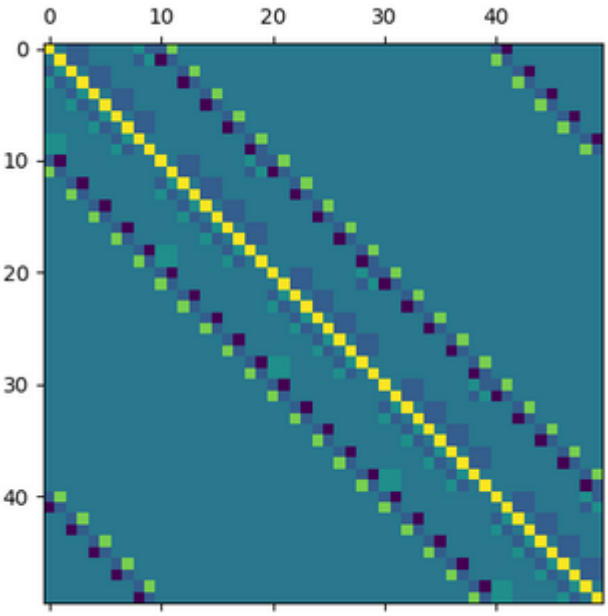
$$= \begin{bmatrix} Q'(0,0) & 0 & 0 & -1 & -1 & 0 & 0 & 0 \\ 0 & Q'(0,0) & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & Q'(1,0) & 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & Q'(1,0) & 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 & Q'(0,1) & 0 & 0 & -1 \\ 0 & -1 & 0 & 0 & 0 & Q'(0,1) & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & Q'(1,1) & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & Q'(1,1) \end{bmatrix}$$

The explicit forms of matrices M (n, m) corresponding to larger lattices are too extensive to be shown here. Because of this we will present examples of these bigger matrices using pictorial representations.

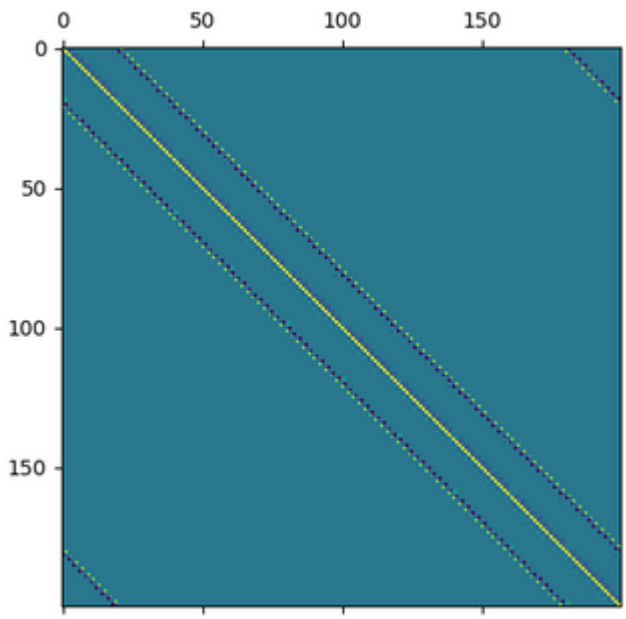
Case 2 x 2: ($4\text{x}2^4$ matrix elements)



Case 5 x 5: ($4x5^4$ **matrix elements)**



Case 10 x 10: ($4x10^4$ **matrix elements)**



Calculation details

The observables are computed through Monte Carlo simulations using the Importance Sampling technique. Due to the extremely high dimensionality of the problem and the large amount of Monte Carlo steps required to obtain good results, a really high quality pseudo random number generator is needed. We've chosen Ranlux - Martin Lüscher - CERN , which exhibits despicable correlation and astronomical periods in the sequences it produces. This algorithm generates uniform random numbers between 0 and 1, so a further technique was applied to construct the Gaussian random numbers needed. The Box-Muller algorithm was implemented and used for this purpose:

```
void gaussian_random(double* gauss_rands,
                     const int M_dim,
                     const int seed)
{
    int                     i{0};
    double                  uniform_rands[2];
    double                  u1, u2;
    double                  gamma0, gamma1;

    rlxd_init(1,seed);

    while(i < M_dim/2)
    {
        ranlxd(uniform_rands, 2);
        u1 = uniform_rands[0];
        u2 = uniform_rands[1];
        u1 = 2*u1-1;
        u2 = 2*u2-1;
        gamma0 = pow(u1, 2) + pow(u2, 2);

        if (gamma0 < 1 && gamma0 != 0)
```

```
        {
            gamma1 = sqrt(-2*log(gamma0)/gamma0);
            u1 = gamma1*u1;
            u2 = gamma1*u2;
            gauss_rands[2*i] = u1;
            gauss_rands[2*i+1] = u2;
            i += 1;
        }
    }
}
```

It is clear that this piece of code is crying for being optimized, specially because it is used inside the main Monte Carlo for-loop.

To calculate the determinants of equation (1) the `umfpack` library of [suitesparse] was used. This suite is specialized on sparse-matrix linear algebra.

Comparison of numerical calculations with analytical results

Luckily there are a couple of cases with can solve exactly by analytical means. This is extremely useful because it allow us to test our numerical results. For the free case $g = 0$, we have that the scalar expected value for $\phi = \frac{1}{N_p}\sum_{\mathbf{m}\in\Lambda}\phi(\mathbf{m})$ and $\phi^2 = \frac{1}{N_p}\sum_{\mathbf{m}\in\Lambda}\phi^2(\mathbf{m})$ are calculated as:

$$\langle\phi\rangle_\phi = \frac{\int_{-\infty}^{+\infty}\cdots\int_{-\infty}^{+\infty}\prod_{\mathbf{n}\in\Lambda}\frac{d\phi(\mathbf{n})}{\sqrt{2\pi}}e^{-\phi^2(\mathbf{n})/2}\left(\frac{1}{N_p}\sum_{\mathbf{m}\in\Lambda}\phi(\mathbf{m})\right)}{\int_{-\infty}^{+\infty}\cdots\int_{-\infty}^{+\infty}\prod_{\mathbf{n}\in\Lambda}\frac{d\phi(\mathbf{n})}{\sqrt{2\pi}}e^{-\phi^2(\mathbf{n})/2}} = 0,$$

$$\langle\phi\rangle_\phi = \frac{\int_{-\infty}^{+\infty}\cdots\int_{-\infty}^{+\infty}\prod_{\mathbf{n}\in\Lambda}\frac{d\phi(\mathbf{n})}{\sqrt{2\pi}}e^{-\phi^2(\mathbf{n})/2}\left(\frac{1}{N_p}\sum_{\mathbf{m}\in\Lambda}\phi^2(\mathbf{m})\right)}{\int_{-\infty}^{+\infty}\cdots\int_{-\infty}^{+\infty}\prod_{\mathbf{n}\in\Lambda}\frac{d\phi(\mathbf{n})}{\sqrt{2\pi}}e^{-\phi^2(\mathbf{n})/2}} = \frac{\frac{2^{\frac{3}{2}}\Gamma(\frac{3}{2})}{\sqrt{2\pi}}}{\frac{2^{\frac{1}{2}}\Gamma(\frac{1}{2})}{\sqrt{2\pi}}}$$

$$= \frac{(2\,2^{1/2})(\frac{1}{2}\Gamma(\frac{1}{2}))}{(\,2^{1/2})(\Gamma(\frac{1}{2}))} = \frac{\sqrt{2\pi}}{\sqrt{2\pi}} = 1,$$

the first result is simply because the integration argument is a odd function. To use our program to calculate the mean scalar field expected value for the free case , we use a input file of the form:

```
10              # Number of columns of the lattice
10              # Number of rows of the lattice
0               # Fermion mass
0               # g coupling constant
mean_scalar     # Observable name
2000            # Number of Monte Carlo Steps
0               # print matrix (Not recommended)
```

And we obtain:

```
Result: -4.18852e-05 + 2.83215e-37i Processes: 4 M-constr-time: 0.000921292 Comp-time:
1.92721 Comm-time: 0.0169848 Total-time: 1.94511
```

Note $-4.18852\text{x}10^{-05} + 2.83215\text{x}10^{-37}i \sim 0$ which is in agreement with the analytical result. To obtain better precision a larger number of Monte Carlo Steps is needed.

The square scalar case will need a input as the following:

```
10              # Number of columns of the lattice
10              # Number of rows of the lattice
0               # Fermion mass
0               # g coupling constant
mean_sq_scalar  # Observable name
2000            # Number of Monte Carlo Steps
0               # print matrix (Not recommended)
```

And it produces:

```
Result: 1.00077 + 2.36228e-32i Processes: 4 M-constr-time: 0.000912439 Comp-time: 1.94237
Comm-time: 0.000912096 Total-time: 1.94419
```

With $1.00077 + 2.36228\text{x}10^{-32}i \sim 1$, also is in agreement with its corresponding analytical result. The following plot illustrates the error reduction with larger amounts of Monte Carlo Steps:

Software Structure and usage

The structure tree of the software is the following:

```
.
├── benchmarking
│   ├── plotter.sh
│   ├── runner.sh
│   └── TMP
│       ├── sdata.png
│       ├── str_avg_std_time.dat
│       ├── str_speed_up.dat
│       ├── wk_avg_std_time_mean_scalar.dat
│       ├── wk_avg_std_time_mean_sq_scalar.dat
│       ├── wk_efficiency_mean_scalar.dat
│       ├── wk_efficiency_mean_sq_scalar.dat
│       ├── wk_errors_mean_scalar.dat
│       └── wk_errors_mean_sq_scalar.dat
├── images
│   ├── 10x10.png
│   ├── 2x2.png
│   ├── 3x3.png
│   ├── 4x4.png
│   ├── 5x5.png
│   ├── 5x8.png
```

```
│   ├── 6x4.png
│   ├── error.png
│   └── space-time.png
├── include
│   ├── conditionalOStreamClass.h
│   ├── determinantWrapper.h
│   ├── gaussianRandomGenerator.h
│   ├── importanceSamplingIntegrator.h
│   ├── mpiCommManager.h
│   ├── observables.h
│   ├── ranlxd.h
│   ├── sparseDiracMatrixClass.h
│   ├── suiteSparseDet.h
│   └── utilities.h
├── input.inp
├── libref
│   ├── RANLUX_Guide.pdf
│   ├── UMFPACK_QuickStart.pdf
│   └── UMFPACK_UserGuide.pdf
├── main.cc
├── Makefile
├── python
│   ├── matrix.py
│   └── polinom_det.py
├── README.md
├── src
│   ├── determinantWrapper.cc
│   ├── gaussianRandomGenerator.cc
│   ├── importanceSamplingIntegrator.cc
│   ├── ranlxd.c
│   ├── sparseDiracMatrixClass.cc
│   ├── suiteSparseDet.c
│   └── utilities.cc
└── tests

8 directories, 49 files
```

To download the required dependencies run (only for debian systems):

```
$ sudo apt-get install libsuitesparse-dev
$ sudo apt-get install libsuitesparse-doc
```

To compile it, just run:

```
$ make
```

To run it:

```
$ mpirun -np [nprocs] gn [input file]
```

The different parameters of the simulation are specified in the input file, for example:

```
10              # Number of columns of the lattice
10              # Number of rows of the lattice
0               # Fermion mass
0               # g coupling constant
mean_sq_scalar  # Observable name
2000            # Number of Monte Carlo Steps
0               # print matrix (Not recommended)
```

Benchmarking

References