Application of Monte-Carlo methods to community detection

Sofia Blinova, Tikhon Parshikov, Ben Sturgis
École Polytechnique Fédérale de Lausanne, Switzerland

I. Introduction

The aim of this project is to address one of the simplest community detection problems. For this purpose, we apply three different Markov Chain Monte Carlo (MCMC) methods. We analyze the Metropolis, Houdayer, and mixed Metropolis-Houdayer algorithms, their convergence time and phase transition. Finally, we come up with the conclusion that it depends on the settings which algorithm performs best.

II. Background

In the given task, we are provided with an observation graph G with N nodes. Every node is assigned to one of two communities $x_i^* \in \{-1, 1\}$ with the same probability. Furthermore, we are given two positive numbers a and b that define the probability that two nodes from the same and two nodes from different communities are connected:

$$\mathbb{P}(e_{i,j} = 1 | x_i^* x_j^* = 1) = \frac{a}{N} = 1 - \mathbb{P}(e_{i,j} = 0 | x_i^* x_j^* = 1),$$

$$\mathbb{P}(e_{i,j} = 1 | x_i^* x_j^* = -1) = \frac{b}{N} = 1 - \mathbb{P}(e_{i,j} = 0 | x_i^* x_j^* = -1).$$

It is assumed that $a > b$, in other words, the probability of an edge between two nodes of the same community is greater than between two nodes from different communities. We want to find an estimator $\hat{\mathbf{x}}$ which is as close to the vector $\mathbf{x}^*$ of our observation graph as possible. The best possible estimator can be found by using the posterior distribution:

$$\mathbb{P}(x | \{e_{ij}\}) \propto \frac{\prod_{1 \le i \le j \le N} e^{h_{ij} x_i x_j}}{\sum_{x \in \{\pm 1\}^N} \prod_{1 \le i \le j \le N} e^{h_{ij} x_i x_j}},$$

$$\text{with } h_{ij} = \frac{1}{2} \left[e_{ij} ln \frac{a}{b} + (1 - e_{ij}) ln \frac{1 - \frac{a}{N}}{1 - \frac{b}{N}} \right].$$

However, since computing the denominator is too costly, we use MCMC methods to find an estimator. The closer the estimator $\hat{\mathbf{x}}$ is to the vector $\mathbf{x}^*$, the more the cost function of the posterior distribution, the Hamilitonian, is minimized. The Hamiltonian is defined as follows:

$$H(x) = - \sum_{1 \le i \le j \le N} h_{ij} x_i x_j$$

The quality of the estimator can be expressed by the overlap:

$$q_N = \frac{1}{N} | \sum_{i=1}^{N} \hat{x}_i x_i^* |.$$

It should be said that this task has no solution and communities cannot be detected if:

$$(a - b)^2 \le 2(a + b),$$

This inequality can be rewritten in terms of $d = \frac{1}{2}(a + b)$ and $r = \frac{b}{a}$. We obtain that $a = \frac{2d}{1+r}$ and $b = \frac{2dr}{1+r}$. Therefore,

$$\left(\frac{2d}{1+r} - \frac{2dr}{1+r} \right)^2 \le 2 \left(\frac{2d}{1+r} + \frac{2dr}{1+r} \right)$$

$$\Leftrightarrow \frac{2d(1-r)^2}{(1+r)^2} \le 2$$

$$\Leftrightarrow \sqrt{d}(1 - r) \le 1 + r$$

From this we can conclude that communities cannot be detected if

$$\frac{\sqrt{d} - 1}{\sqrt{d} + 1} \le r. \tag{1}$$

The ratio r_c for which we have equality in (1) is also called **phase transition point**.

A. The Metropolis Algorithm

The Metropolis algorithm is a MCMC method to sample from a probability distribution π. The procedure consists of several steps:

1. Choose an aperiodic, irreducible Markov chain with transition probabilities ψ_{ij} on the given state-space S such that $\psi_{ij} \ge 0$ if and only if $\psi_{ji} \ge 0$. This chain is also called base chain.
2. Define acceptance probabilities $a_{ij} = \min(1, \frac{\pi_j \psi_{ji}}{\pi_i \psi_{ij}})$.
3. Define the transition matrix P as follows:

$$P = \begin{cases} p_{ij} = \psi_{ij} a_{ij}, i \ne j \\ p_{ii} = 1 - \sum_{k \ne i} \psi_{ik} a_{ik} \end{cases}$$

For these transition probabilities $p_{ij}(n) \to \pi_j$ for $n \to \infty$ and P satisfies detailed balance. The other two algorithms are based on this method, but with some modifications.

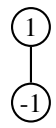
B. The Houdayer Algorithm

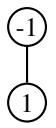
In this algorithm, compared to the previous one, we have two Markov chains $\mathbf{x}(\mathbf{1})$ and $\mathbf{x}^{(2)}$. After performing a Metropolis step one each Markov chain, we make a Houdayer step that can be described as follows:

1. Pick at random one of the nodes $i \in \{1..N\}$ such that $x_i^{(1)} \ne x_i^{(2)}$.
2. Define a set *M* with all nodes $j \in \{1..N\}$ such that each *j* is connected to *i* and $x_j^{(1)} \ne x_j^{(2)}$.

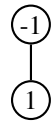
3. For all the nodes in *M* and *i* change its community to the opposite one.

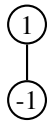
It is known that the Markov chains we construct as described above are ergodic. In the Houdayer algorithm each state is a pair of configurations $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)})$. Let us consider an example for which the chain on the pair configurations is not ergodic. Define one simple chain with two nodes which are connected by an edge. We start with the following pair of configurations:

$\mathbf{x}^{(1)}$: 

$\mathbf{x}^{(2)}$: 

Let's take the houdayer step. Each pair of nodes in the two graphs is different, i.e. $x_i^{(1)} \neq x_i^{(2)}$ for $i \in 1, 2$, thus we pick at random one of them. Then, we define a cluster *M* of the nodes that are connected with the picked one. In this case, our cluster consists of the whole chain. For all the nodes in this cluster we flip the values. As a result, we obtain:

$\mathbf{x}^{(1)}$: 

$\mathbf{x}^{(2)}$: 

Hence, if we do more such iterations we stay in the same situation where one node of the graph is -1 and the other one is 1. Finally, these chains will never come, for example, to the configuration:

$$(\begin{pmatrix}1\\1\end{pmatrix}, \begin{pmatrix}1\\1\end{pmatrix})$$

The chain which consists of pairs of the configurations is not irreducible and therefore not ergodic.

C. The Mixed Metropolis-Houdayer Algorithm

The mixed Metropolis-Houdayer algorithm is almost the same as the Houdayer algorithm. The only difference is that we perform the Houdayer step not every iteration but every n_0th step. This decreases the computation time while we still have the advantage of doing non-local moves.

III. Base Chain

States in our base chain are a configuration in which each of the N nodes of the observation graph is assigned to one of the two communities $\{-1, 1\}$. Each state of the base chain can thus be understood as a vector in $\mathbf{x} = \{-1, 1\}^N$. Every state $\mathbf{x^{(i)}}$ has N adjacent states $\mathbf{x^{(j)}}$ that differ by only one element, i.e. $x_k^{(i)} \neq x_k^{(j)}$ and a self-loop. Since we choose uniform transition probabilities $\psi_{i,j} = \frac{1}{N+1}$, if $i = j$ or $\mathbf{x^{(i)}}$ and $\mathbf{x^{(j)}}$ differ by only one element (N neighbors and a self-loop). Otherwise $\psi_{i,j}$ is 0. Since this chain has self-loops, it is aperiodic. Moreover, the chain is irreducible, since any state can be reached from any other state by simply changing, one after the other, the elements in which the two states differ. Moreover, it is obvious that $\psi_{ij} \geq 0$ if and only if $\psi_{ji} \geq 0$. In our base chain, it is even the case that $\psi_{ij} = \psi_{ji}$. Consequently, we can use this chain for the Metropolis algorithm.

To calculate the acceptance probabilities $a_{ij} = \min(1, \frac{\pi_j \psi_{ji}}{\pi_i \psi_{ij}})$ in this base chain, we can make some simplifications that speed up the calculation from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$. If $\mathbf{x^{(k)}}$ and $\mathbf{x^{(l)}}$ differ in one element $x_m^{(k)} \neq x_m^{(l)}$, it follows:

$$\begin{aligned}\frac{\pi_j \psi_{ji}}{\pi_i \psi_{ij}} &= \frac{\prod_{1\leq i<j\leq N} \exp\left(h_{ij} x_i^{(l)} x_j^{(l)}\right)}{\prod_{1\leq i<j\leq N} \exp\left(h_{ij} x_i^{(k)} x_j^{(k)}\right)} \\ &= \frac{\exp\left(\sum_{\substack{1\leq i\leq N \\ i\neq m}} h_{ij} x_i^{(l)} x_j^{(l)}\right)}{\exp\left(\sum_{\substack{1\leq i\leq N \\ i\neq m}} h_{ij} x_i^{(k)} x_j^{(k)}\right)} \\ &= \exp\left(2 x_j^{(l)} \sum_{\substack{1\leq i\leq N \\ i\neq m}} h_{ij} x_i^{(l)}\right)\end{aligned}$$

IV. Analysis of the Algorithms

A. Convergence time

Influence of $\frac{b}{a}$ ratio and d: For these experiments we set $N = 250$ and $\#iterations = 10000$. Also, for each plot we make 100 experiments and then take average between the results.

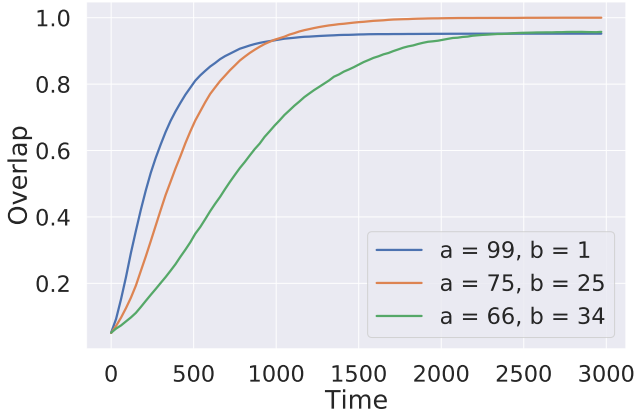


Figure 1: Overlap as a function of time for $N = 250$ and different $\frac{b}{a}$ ratios of Metropolis algorithm (fixed average degree d)

Figures 1 and 2 represent convergence of the Metropolis algorithm with different a, b combinations. It shows that the convergence time of the overlap as a function of time depends on the parameters $r = \frac{b}{a}$ and $d = \frac{1}{2}(a + b)$, the average degree of the observation graph. Similar behavior can also be observed for the Houdayer algorithm and the Mixed Metropolis-Houdayer algorithm. The number of steps executed by the algorithm is used as a measure of time. To analyze the influence of the parameters r and d, we fix in each graph the parameter that we do not observe.

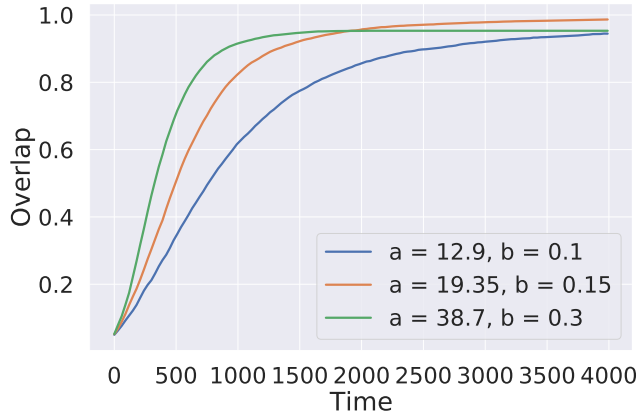


Figure 2: Overlap as a function of time for $N = 250$ and different average degrees d of Metropolis algorithm (fixed $\frac{b}{a}$ ratio)

First of all, we can observe that all curves have the form of a part of logistic function independent of the parameters r and d. The smaller the ratio $\frac{b}{a}$, the faster the curve converge. For example, comparing the curves in Figure 1 after 500 iterations, the Metropolis algorithm achieves an overlap of about 0.38 for $a = 66$ and $b = 34$, an overlap of about 0.7 for $a = 75$ and $b = 25$, and an overlap of less than 0.8 for $a = 99$ and $b = 1$. For the average degree d, we observe that the curves converge faster the larger d is. After 1000 iterations, the Metropolis algorithm achieves an overlap of about 0.61 on an observation graph with $a = 12.9$ and $b = 0.1$, an overlap of about 0.82 with $a = 19.35$ and $b = 0.15$, and an overlap of about 0.91 with $a = 38.7$ and $b = 0.3$.

From Equation (1) we know, that for $N \to \infty$ communities cannot be detected if $\frac{\sqrt{d}-1}{\sqrt{d}+1} \leq r$. By decreasing the ratio r and increasing the average degree d, we increase the term $\frac{\sqrt{d}-1}{\sqrt{d}+1} - r$. We can see that it is not only possible to detect communities for the Monte-Carlo methods when $\frac{\sqrt{d}-1}{\sqrt{d}+1} - r > 0$, but that the communities can also be found faster the larger the term is.

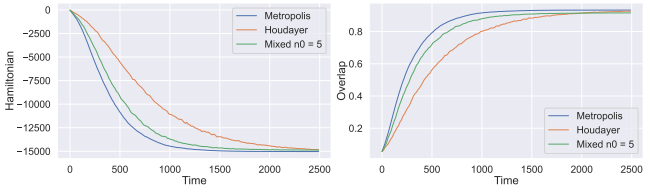


Figure 3: Hamiltonian and overlap as a function of time for $N = 250$, $a = 99$, $b = 1$ and different algorithms

Comparison of different algorithms: Figure 3 shows the average Hamiltonian and overlap over 100 experiments as a function of time for the standard Metropolis, Houdayer, and mixed Metropolis-Houdayer algorithm with $n_0 = 5$ for an observation graph with 250 nodes generated with parameters $a = 500$ and $b = 20$. For the Houdayer and the mixed Metropolis-Houdayer algorithms, the Hamiltonian and the overlap of one of the two spin configurations $(\mathbf{x^{(1)}}, \mathbf{x^{(2)}})$ are mapped respectively. Again, the number of steps executed by the algorithms is used as a measure of time. However, it must be taken into account that one step of the Houdayer algorithm takes more time than one step of the Metropolis algorithm.

Regardless of the specific algorithm the curves on the right side of Figure 3, which map the average overlap as a function of time, again have the form of a part of logistic function. However, the concrete convergence behavior depends on the algorithm. For example, in the 500th iteration, the Metropolis algorithm already reaches an overlap of about 0.8, the mixed Metropolis-Houdayer algorithm reaches an overlap of about 0.72, while the Houdayer algorithm has only reached an overlap of about 0.48. After about 1250 iterations, both the Metropolis and Mixed Metropolis-Houdayer algorithms have reached an overlap of about 1, while the Houdayer algorithm has an overlap of about 1 only after 2000 iterations.

The curves of the Hamiltonian function as a function of time on the left side of Figure 3 confirm the behavior described above. The curves also correspond to a logistic function mirrored on the x-axis, but in this case they do not converge to -1, but to about -15000. The Hamiltonian function also reflects the superiority of the Metropolis algorithm over the mixed Metropolis-Houdayer algorithm, and the superiority of both algorithms over the Houdayer algorithm, as the Hamiltonian is minimized fastest in the Metropolis algorithm, second fastest in the mixed Metropolis-Houdayer algorithm, and slowest in the Houdayer algorithm.

It is worth mentioning that for the Mixed Metropolis-Houdayer algorithm and especially for the Houdayer algorithm in the plots for the not averaged overlap and Hamiltonian as a function of time, there are sometimes large jumps and the curves oscillate around the average curves. The not averaged plots of the Hamiltonian algorithm, on the other hand, have only a small variance and a smoother curve.

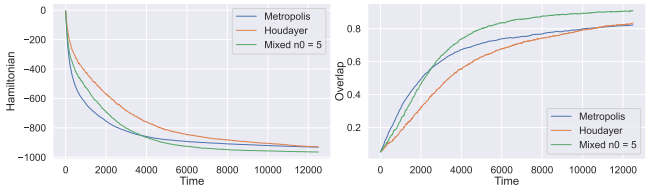


Figure 4: Hamiltonian and overlap as a function of time for $N = 250$, $a = 7.4$, $b = 0.1$ and different algorithms

However, if we choose a smaller average degree d and a slightly larger ratio r by choosing $a = 7.4$ and $b = 0.1$ so that it becomes more difficult to detect communities, the situation described above changes. In this case, the average Hamiltonian and overlap converge faster for the

mixed Metropolis-Houdayer (green) algorithm than for the Metropolis (blue) and Houdayer (orange) algorithm. The reason for this behavior is that the Metropolis algorithm performs worse the closer we get to the critical point r_c of the phase transition, as we will see later. In this case, Metropolis algorithm often gets stuck in a local minimum of the Hamilitonian and it is beneficial to take a cluster step after a certain number of steps instead of a local step. This way a local minimum can be left faster and the landscape can be explored better. However, in the Houdayer algorithm, the cluster steps are executed very frequently, which slows down convergence. A cluster move can also undo the progress of the Metropolis algorithm to a minimum of the Hamiltonian.

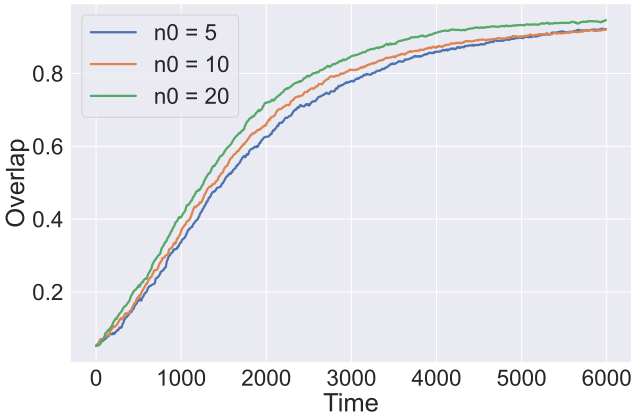


Figure 5: Overlap as a function of time with different values of n_0

Influence of n_0: In this section, we analyze how parameter n_0 affects the convergence behavior of the mixed Metropolis-Houdayer algorithm. The parameter n_0 indicates after how many Metropolis steps the Houdayer step is executed. Figure 5 shows the curves of the mixed Metropolis-Houdayer algorithm with different values n_0 for the overlap as a function of time. The observation graph with 250 nodes was generated with parameters $a = 9.9$ and $b = 0.1$. The mixed Metropolis-Houdayer algorithm corresponds to the Houdayer algorithm for $n_0 = 1$ and to the Metropolis algorithm for $\lim_{n_0 \to +\infty}$. According to the results from the first paragraph of this section, the larger n_0 and thus the closer the mixed Metropolis-Houdayer algorithm is to the Metropolis algorithm, the faster the curves for the overlap and the Hamiltonian converge.

B. Phase transition

For these experiments we set $N = 1000$ and $\#iterations = 40000$.

All algorithms: Figure 7 shows the relationships between the overlap and the $\frac{b}{a}$ ratio for all three algorithms. Here we fix $b = 0.1$ and change a to obtain the ratio given on the x-axis. We examine the ratio in the interval $(0, 0.1]$ to pinpoint the phase transition. Also, for each point, we run 10 experiments and then take average overlap over them.

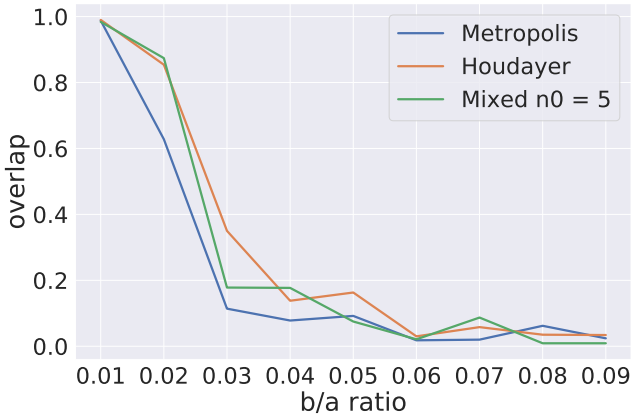


Figure 6: Overlap as a function of ratio for different algorithms

In Figure 6 it can be seen that $r_c \in (0.04, 0.06)$. Let us take $r = 0.06$, then $a = 1.67$ ($b = 0.1$). Using Equation (1) we obtain:

$$r = 0.06 \geq \frac{\sqrt{d} - 1}{\sqrt{d} + 1} = -0.03$$

Also, let us consider $r = 0.04$:

$$r = 0.04 \leq \frac{\sqrt{d} - 1}{\sqrt{d} + 1} = 0.065$$

In the point with ratio $r = 0.04$, all algorithms still converge and the overlaps are not equal to zero. But in the points with the ratio between 0.04 and 0.06 all overlap functions are decreasing and after 0.06 they are close to zero. Thus, our experiments confirm the theory.

Figure 7 also shows that the curve of the Metropolis algorithm lays below the Houdayer and mixed Metropolice-Houdayer curves. It means that often the accuracy of the Metropolis algorithm is lower. Thus, for the task settings with r close to phase transition it makes sense to use the Houdayer or Mixed algorithm.

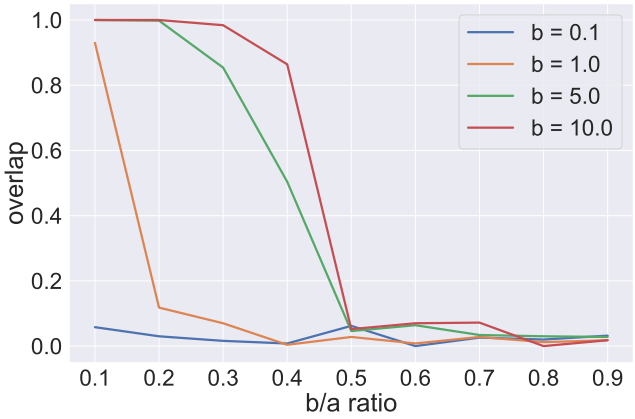


Figure 7: Overlap as a function of ratio for Metropolis algorithm

Phase transition for different d: In Figure 7 we compare the phase transition for the Metropolis algorithm with different d. It can be seen that the bigger b we choose, the bigger r_c we obtain. The difference is especially noticeable at the examples with $b = 0.1$ and $b = 10$. The Phase transition for $b = 0.1$ is less than 0.1, in the same time r_c is close to 0.5 for $b = 0.1$.

V. Conclusion

In this project, three algorithms were implemented and analyzed. We noticed that the Metropolis converges faster than the other two algorithms in a setting that is far from the phase transition point. This is the case when the $\frac{b}{a}$ ratio r is small and the average degree d is large (see (1)). On the other side the limiting performance of the other two algorithms (Houdayer and mixed Metropolis-Houdayer) is better. However, we must take into account that these algorithms are more time consuming because of the Houdayer step. For this reason, there is a tradeoff between the computation time and the quality of the results. In the mixed Metropolis-Houdayer algorithm, this complexity is reduced by not performing a Houdayer step at each iteration.