Uniwersytet Wrocławski
Wydział Matematyki i Informatyki
Instytut Matematyczny
specjalność: Analiza danych

Bartosz Chmiela

Locally-informed proposals in Metropolis-Hastings algorithm with applications

Praca magisterska
napisana pod kierunkiem
dr hab. Pawła Lorka

Wrocław 2022

Abstract

The Markov Chain Monte Carlo methods (abbrv. MCMC) are a family of algorithms used for sampling from a given probability distribution. They prove very effective when the state space is large. This fact can be used to solve many hard deterministic problems – one of them being *traveling salesmen problem*. It will be used in this thesis to test a new approach of *locally-informed propolsals* as a modification of well known *Metropolis-Hastings* algorithm. In this thesis we will present the implementation of modified algorithm, experiments based on it, results and a comparison of to previous MCMC methods.

Metody próbkowania Monte Carlo łańcuchami Markowa są rodziną algorytmow używanych do próbkowania z danego rozkładu prawdopodobieństwa. Okazują się efektywne zwłaszcza gdy przestrzeń stańw jest wielka. Ten fakt może być wykorzystany przy rozwiązywaniu wielu deterministycznych problemów – jednym z nich jest *problem komiwojażera*. Zostanie on użyty w tej pracy do przetestowania nowego podejścia *lokalnie poinformowawnego?*, jako modyfikacji dobrze znanego algorytmu *Metropolisa-Hastinigsa*. W tej pracy zaprezentujemy implementacje zmodyfikowanego algorytmu, eksperymentów bezujących na nim, wyników oraz porównania z poprzednimi metodami próbkowania Monte Carlo.

Contents

List of Tables

List of Figures

1 Introduction

The Markov Chain Monte Carlo methods (abbrv. MCMC) are a family of algorithms used for sampling from a given probability distribution. At first they do not seem useful for solving practical deterministic problems, but with some tweaks they can become a powerful tool. It happens especially when space of possible solutions is enormous and computing becomes infeasible for machines. These offer a shortcut for obtaining "close enough" answers.

At their core, MCMC methods generate a Markov Chain (abbrv. MC) with a defined distribution and sample using it. The convergence of the chain is assured by ergodic theorems. The most known of them is *Metropolis-Hastings* algorithm, which constructs a MC using another set of distributions, maybe simpler ones.

In this thesis we work on *locally-informed proposals*, which involve determining *local* distribution – which comes down to finding transition probabilities of the state. They are a bit more complex and computationally heavy, but offer better results with less iterations.

To test this method we will need a deterministic problem which quickly becomes infeasible for machines to compute – one of them is a well-known traveling salesman problem. The testing is carried out using its benchmark training set *tsplib95* and implementation is provided in *Python3*.

2 Markov chains

Markov chains are the very basic building blocks of the theory used within this thesis. They are a natural extension of independent stochastic processes, that assume a weak dependence between the presence and the past.

In this thesis we will focus only on stochastic processes with discrete time steps and finite state space, which satisfies the Markov property. These are the ones that, we are able to simulate in computers.

2.1 Basic terminology and assumptions

We assume that the reader has a basic probabilistic background, so that we can freely use terminology from probability theory, like random or independent variables, stochastic processes, measure or σ-algebra.

A Markov chain needs to be defined with discrete state space and index set.

Definition 2.1. *A state space of a Markov chain is a countable set S.*

A state space defines values over which a Markov chain is iterating. In our case it is finite so we can associate it with a subset of natural numbers like $\{1, 2, \ldots, N\}$ (for some N depending on a number of states) instead of states.

Definition 2.2. *An index set of a Markov chain is a countable set T.*

An index set represents time in which Markov chain moves. For a chain we assume discrete time steps and again in our case it will be a finite set, so we can associate it with a subset of natural numbers like $\{1, 2, \ldots, T\}$ for some T depending on a length of time interval.

To work with any probabilistic construct such as Markov chain, we need a probability space in which it resides and can be measured.

Definition 2.3. *A probability space is a triplet* $(\Omega, \mathcal{F}, P)$*, where* Ω *is some abstract sample space,* $\mathcal{F}$ *is a* σ*-algebra (event space) and* P *is a probability measure.*

In case of Markov chain sample space is a space of all possible Markov chains and an event space is a space of all events with Markov chains(?). Probability measure is not explicitly given, because it is often not possible to find a probability of a chain.

Most of the time a Markov chain will be associated with a stochastic transition matrix $\mathbf{P}$, that represents probabilities of transitions between states.

Definition 2.4. *A stochastic matrix* $\mathbf{P}$ *is a matrix, which rows sum to* 1*.*

Now with the notation and the background we are able to define a Markov chain.

2.2 Definition and basic properties

In this subsection we will formally define a Markov chain and list some of its basic properties.

Definition 2.5. *A Markov chain (abbrv. MC) is a sequence of random variables* $(X_n)_{n\in T}$ *defined on a common probability space* $(\Omega, \mathcal{F}, P)$*, that take values in* S*, such that it satisfies Markov property:*

$$P(X_{n+m} = j | X_n = i, X_{l_{k-1}} = i_{l_{k-1}}, X_{l_{k-2}} = i_{l_{k-2}}, \ldots, X_{l_1} = i_1) = P(X_{n+m} = j | X_n = i).$$

For all indices $l_1 < \ldots < l_{k-1} < n < n+m,\ 1 \leq k < n$ *and all states* $j, i, i_{n-1}, i_{n-2}, \ldots, i_0 \in S$*.*

This definition indicates independence of the past of a MC. The probabilities depend on current state and the number of steps.

Definition 2.6. *A Markov chain is homogeneous if additionally:*

$$P(X_{n+m} = j | X_m = i) = P(X_n = j | X_0 = i).$$

In this case we define:

$$p_{i,j}(n) \stackrel{df}{=} P(X_n = j | X_0 = i).$$

Homogeneous Markov chains (abbrv. HMC) are more natural for us and easier to study. These eliminate the dependence on the number of steps that a Markov chain went through. From now on, whenever we use a term *markov chain* we will mean a *homogeneous markov chain.*

The probabilities $p_{i,j}(n)$ give us a probability of transition between states i and j in n steps. We can use them to form a special matrix that will be linked with a MC.

Definition 2.7. *A transition matrix in* n *steps* $\mathbf{P}(n)$ *for a HMC is a stochastic matrix constructed using transition probabilities:*

$$\mathbf{P}_{i,j}(n) = p_{i,j}(n),\ \mathbf{P}_{i,j}(0) = \mathbb{I},\ \mathbf{P} \stackrel{df}{=} \mathbf{P}_{i,j}(1).$$

Definition 2.8. *A initial distribution of a Markov chain is a vector* $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_N) \in \mathbb{R}^N$ *such that* $\sum_{i=1}^{N} \mu_i = 1$. *This is a distribution of a random variable* X_0, *which is an initial state of a Markov chain.*

A transition matrix together with initial distribution define a HMC, so these are the only objects that need to be analyzed if one wants to study those chains.

Theorem 2.1. *Let* $\boldsymbol{\mu}^{(n)}$ *be a distribution of a HMC at n-th step, then for all n:*

$$\boldsymbol{\mu}^{(n)} = P(X_n = j) = \boldsymbol{\mu}\mathbf{P}^n.$$

Proof of this theorem involves unfolding vectorized equation and using basic induction so it will be left. It also shows that given some knowledge of matrix $\mathbf{P}$ one can easily work with HMC.

2.2.1 Irreducibility

Irreducibility guarantees that all states of a MC have the same properties, so that we do not need to analyze every state separately. Moreover an irreducible MC cannot be split into more chains.

Definition 2.9 (Irreducibility)**.** *A Markov chain with transition matrix* $\mathbf{P}$ *is called irreducible if and only if for every pair of states* i *and* j *there exists a positive probability of transition between them i.e.,*

$$\exists\, n\, \mathbf{P}_{i,j}(n) > 0.$$

2.2.2 Periodicity

Periodicity tells us something about the structure of the transition matrix. It especially indicates when there is a possibility of a chain staying in one state.

Definition 2.10 (Periodicity)**.** *Let* d_i *be a greatest common divisor of those* n *such that* $\mathbf{P}_{i,i}(n) > 0$ *i.e.,*

$$d_i = gcd\{n \geq: \mathbf{P}_{i,i}(n) > 0\}$$

If $d_i > 1$ *then state* i *is periodic. If* $d_i = 1$ *then state* i *is aperiodic.*

Definition 2.11. *A Markov chain with transition matrix* $\mathbf{P}$ *is called periodic with a period* d *when all states are periodic with a period* d. *In particular, when MC is irreducible and there is a state with a period* d, *then all the states are with period* d *and chain is periodic.*

2.3 Stationarity and ergodicity

In this subsection we will cover asymptotics for the long-term behavior of a MC.

Definition 2.12 (Stationarity)**.** *A probability distribution* $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_N)^T$ *is called stationary if it satisfies*

$$\pi_j = \sum_{i \in S} \pi_i p_{ij},$$

or equivalently in vector form:

$$\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{P}.$$

This equation is often described as the balance equation.

From this definition it is easy to see that if a MC gets to stationary distribution it will not leave it.

Definition 2.13 (Ergodicity). *When a MC is ergodic?*

Theorem 2.2. *For any irreducible and aperiodic Markov chain, there exist at least one stationary distribution.*

The proof of the theorem 2.2 can be found in [2].

Theorem 2.3. *Let (X_n) be a irreducible and aperiodic HMC, then: ergodic???*

$$\lim_{n\to\infty} p_i j(n) = \pi_j.$$

Proof. content... □

Those theorems tell us for some MC we are able to predict their behaviors after enough time passes. It also gives us a tool to sample from a given distribution $\boldsymbol{\pi}$ using Markov chains.

2.4 Reversibility

A reversible MC has a property of having the same distribution in the past and in the future.

Definition 2.14 (Reversibility). *A Markov chain (X_n) is reversible if random vectors*

$$(X_{i_1}, X_{i_2}, \ldots, X_{i_k}) \text{ and } (X_{m-i_1}, X_{m-i_2}, \ldots, X_{m-i_k}),$$

have the same distribution for all m and $i_j, m - i_j \in T$.

This condition is just a mathematical representation of the previous statement and it is not practical, so we need another way of describing it.

Definition 2.15 (Detailed balance). *A probability distribution $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_N)^T$ satisfies detailed balance for a Markov chain with transition matrix $\mathbf{P}$ if*

$$\forall\, i, j \in S\ \pi_i p_{i,j} = \pi_j p_{j,i}$$

Definition 2.16. *A Markov chain is said to be reversible if there exists a reversible distribution for it.*

This definition is much more practical, because the condition can be calculated for each proposed distribution.

Theorem 2.4. *If a probability distribution $\boldsymbol{\pi}$ satisfies detailed balance for a Markov chain with transition matrix $\mathbf{P}$, then $\boldsymbol{\pi}$ is a stationary distribution for this chain.*

Proof. Summing over all $i \in S$ we get

$$\sum_{i\in S} \pi_i p_{i,j} = \pi_j \sum_{i\in S} p_{j,i} = \pi_j,$$

which is a balance equation. □

The detailed balance will be often used in as it is somewhat easier to check than balance equation.

3 Markov chain Monte Carlo methods

In this section we will show how to use aforementioned properties of Markov chains for solving problems. The idea is to use a Markov chain to simulate complicated models and estimate relevant parameters.

The classical example is measuring the area under a curve or a figure, with counting how many randomly generated points are inside or outside the figure and estimating area as a mean. Such a sequence of points is a sequence of random independent variables, which is also a MC, but without any dependence.

If we could generate a MC with a stationary distribution proportional to some function of our interest we could find its maximum, by counting frequencies of states. That is the core idea of Metropolis algorithm.

3.1 Metropolis-Hastings algorithm

We seek an algorithm constructing a MC, which has a stationary distribution of our given probability distribution $\boldsymbol{\pi}$ ($\pi_i > 0$). One could of course find such a transition matrix $\mathbf{P}$ that has stationary distribution $\boldsymbol{\pi}$ but this does not avoid the problem of enormous state space – the matrix $\mathbf{P}$ would also be enormous. So it would be a feasible idea, when the state space is small and also deterministic algorithms are able to find solutions.

Regardless of this fact let us start with constructing such a matrix. Assume that we have another stochastic matrix $\mathbf{Q}$ which is irreducible and aperiodic. Let us consider a matrix defined us:

$$\mathbf{P}_{i,j} = \begin{cases} \mathbf{Q}_{i,j} \min\left(1, \frac{\pi_j \mathbf{Q}_{j,i}}{\pi_i \mathbf{Q}_{i,j}}\right) & \text{if } i \neq j, \\ 1 - \sum_{j \in S \setminus \{i\}} \mathbf{P}_{i,j} & \text{if } i = j. \end{cases} \tag{3.1.1}$$

Theorem 3.1. *A matrix defined in 3.1.1 is stochastic, irreducible, aperiodic and has a stationary distribution $\boldsymbol{\pi}$.*

Proof. The matrix $\mathbf{P}$ is stochastic from a definition – one entry in a row is just a sum of every other and subtracted from 1. For $\mathbf{Q}_{i,j} > 0$ we have $\mathbf{P}_{i,j} > 0$ and irreducibility and aperiodicity are inherited from $\mathbf{Q}$. Let us look at the detailed balance:

$$\pi_j \mathbf{P}_{j,i} = \pi_j \mathbf{Q}_{j,i} \min\left(1, \frac{\pi_i \mathbf{Q}_{i,j}}{\pi_j \mathbf{Q}_{j,i}}\right) = \min\left(\pi_i \mathbf{Q}_{i,j}, \pi_j \mathbf{Q}_{j,i}\right) = \pi_i \mathbf{Q}_{i,j} \min\left(1, \frac{\pi_j \mathbf{Q}_{j,i}}{\pi_i \mathbf{Q}_{i,j}}\right) = \pi_i \mathbf{P}_{i,j}.$$

We see that this matrix satisfies detailed balance, co the distribution $\boldsymbol{\pi}$ is stationary. □

The matrix $\mathbf{Q}$ is called *candidate matrix* as it will be later proposing candidates for a MC. We proved more general case, but if matrix $\mathbf{Q}$ is symmetric some terms cancel and the proof becomes easier.

The *Metropolis-Hastings algorithm* utilizes this construction to generate irreducible, aperiodic MC with a stationary distribution $\boldsymbol{\pi}$. When candidate matrix $\mathbf{Q}$ is symmetric we call this a *Metropolis algorithm.*

In reality one does not need to create whole candidate matrix or a transition matrix. We just need to know how to sample at one step, so how to choose a candidate given a current state. If the procedure is symmetric it simplifies drastically, we need only to know the quotient of distribution $\boldsymbol{\pi}$ at this step. The next constructions use this as an advantage to reduce number of computations.

Algorithm 1 Metropolis-Hastings algorithm

Choose a state $i \in S$.
$X_0 \leftarrow i$
for $n = 1, 2, \ldots$ **do**
    Sample $j \sim \mathbf{Q}_i = (\mathbf{Q}_{i,1}, \mathbf{Q}_{i,2}, \ldots, \mathbf{Q}_{i,N})$.
    Sample $U \sim Unif(0, 1)$.
    if $U \leq \min\left(1, \frac{\pi_j \mathbf{Q}_{j,i}}{\pi_i \mathbf{Q}_{i,j}}\right)$ **then**
        $X_{n+1} \leftarrow j$
    else
        $X_{n+1} \leftarrow X_n$
    end if
end for

4 Traveling salesman problem

Traveling salesman problem (abbrv. TSP) is a well known for being hard to solve and this is why many researchers, including us, use it as a benchmark problem for testing new methods. It is an old problem, with no solution, only with methods that try to achieve the best answer. It is proved to be a NP-hard problem, so deterministic algorithms cannot be reasonably used. This is why probabilistic methods like MCMC become interesting as they eliminate the need of computing all the steps or states of the problem.

TSP asks to find a shortest (the least costly) path between the vertices of a given graph, that covers all of them and is a cycle. This question becomes harder to answer with more vertices added to a graph. It started as a problem of salesman visiting all of the cities and coming back to his place.

4.1 Statement of the problem

To state this problem we need to define weighted graphs and paths as they can represent the problem.

Definition 4.1. *A undirected graph G is a pair (V, E), where V is a set of vertices and $E \subseteq V \times V$ is a set of edges.*

Definition 4.2. *A weighted undirected graph $G = (V, E)$ is a graph such that each edge e has assigned a weight $w_e \in \mathbb{R} \cup \{\infty\}$.*

Vertices could be any set, so we can think of them as a set of all cities. An edge is a pair of vertices, so it can be a connection between cities. A weight is just a function on edges, so it could be a distance between cities.

Again, because we have finite number of cities, we can work on set of indices instead. We will associate weight $w_{i,j}$ with the weight of an edge between vertices i and j.

Definition 4.3. *A path is a sequence of edges.*

Definition 4.4. *A cycle is a path $\sigma = (e_1, e_2, \ldots, e_n)$ such that $e_1 = e_n$.*

Definition 4.5. *A Hamiltonian cycle is a cycle that visits each vertex exactly once.*

Now we can express salesman path as a Hamiltonian cycle that visits a city once. Such a cycle can be thought as a permutation of vertices.

Definition 4.6 (Traveling salesman problem)**.** *Given an undirected weighted graph* $G = (V, E)$*,* $|V| = n$ *find a permutation* $\sigma_{\min}$ *of vertices such that*

$$\sigma_{\min} = \arg\min_{\sigma \in S_n} \sum_{i=1}^{n} w_{\sigma(i),\sigma(j)}.$$

Where S_n *is a set of all permutations of vertices.*

It definition exactly states the Traveling salesman problem: visit all cities once, with the least distance covered.

4.2 Complexity

At first glance, one might not think of this as a hard problem, but to understand complexity of that, it is enough to think of all the permutation of vertices. The set of n vertices S_n has $n!$ elements, a number which grows rapidly. It means that if we want to check all possible salesman tours, we need to compute distances at most $n!$ times, which becomes infeasible with only 20 cities.

Depending on a method of calculating the distance we obtain complexity of $O(n! \cdot d(n))$ where $d(n)$ is a number of steps needed to calculate distance of one path. If one just adds all weights then the complexity will be of $O(n! \cdot n)$, which is a lot more than a polynomial complexity and quickly becomes impossible to compute.

There are other methods like dynamic programming – Held-Karp algorithm, but the complexity of $O(n^2 \cdot 2^n)$ is still a lot. The problem has been shown to be NP-hard even with removing some of the contradicts or using easier metrics.

4.3 Dataset

To test our methods we have obtained data from *TSPLIB* ([1]) a site, which is a library of sample instances for TSP (not only that) from various sources and types. All the files there are of the extension *.tsp* (or alternatively *.xml*) and of following structure: *nameN.tsp*. *name* defines where does the data come from and N defines how many vertices there are. For handling this extension we use *tsplib95* package in *Python3*

All of the datasets there have an optimal solution, so we are able to compere our solutions. We have chosen only some of them:

- *berlin52* 52 locations in Berlin, with an optimal solution: 7542,
- *kroA150* 150-city problem A, with an optimal solution: 26524,
- *att532* 532 AT&T switch locations in the USA, with an optimal solution: 27686,
- *dsj1000* clustered random problem, with an optimal solution: 18659688.

References

[1] Tsplib. http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/index.html. (Accessed on 06/01/2022).

[2] O. Häggström et al. *Finite Markov chains and algorithmic applications*, volume 52. Cambridge University Press, 2002.

[3] C. Karpiński. On use of monte carlo markov chains to decode encrypted text and to solve travelling salesman problem. 2020.

[4] C. J. Maddison, D. Duvenaud, K. J. Swersky, M. Hashemi, and W. Grathwohl. Oops i took a gradient: Scalable sampling for discrete distributions. 2021.

A Source code