Probability Bounds on Large Language Models

High Dimensional Probability

Samuel P.C. Zegers, Lourens E. Touwen
Supervisor: Prof.dr. Frank H.J. Redig

Delft University of Technology
Delft Institute of Applied Mathematics
August 1, 2023

1 Introduction

Transformer models have become hugely popular in Natural Language Processing (NLP). They were introduced in the well-known paper "Attention is all you need" in 2017, and are the fundamental technology to Large Language Models (LLM) such as ChatGPT (*OpenAI*) , LLaMa (*Meta*) and Bard (*Google*).

LLMs are *next-token predictors*. Simply put: given a sequence of words (the prompt), the model predicts, with high accuracy, the next word. LLMs are often described (or criticized) as "Stochastic Parrots". This is not without reason: given the same sequence of words, different words may be predicted. Let's investigate how this randomness arises.

Let T be the so-called *temperature*, a hyperparameter of the model. When $T = 0$ the model is deterministic, and when $T = 1$, it is most random. We want to investigate the relation between the choice of T and the stochasticity in the output. Given token-sequence x, the model computes so-called logits $\rho_1, \ldots, \rho_k$, corresponding to the k most likely tokens. These logits are deterministic: the same token-sequence will always yield the same logits. The probability of predicting token i is then normalized through *Softmax*:

$$p_i = \frac{e^{\rho_i/T}}{\sum_{j=1}^{k} e^{\rho_j/T}}, \quad i \in \{1, \ldots, k\}.$$

Our focus for this project is on the effect of the temperature hyperparameter T. Can we find concentration bounds on certain metrics over different model evaluations? We can then answer questions such as: "Given T, what is the probability of a certain token appearing in X?". This is a relevant question for picking the right T for the right application.

We have noticed that the p_i are closely related to the Boltzmann distribution (which explains the name of higher parameter T), which arises in statistical mechanics (energy states). There is even a relation with the Ising model.

2 LLM as Finite state Markov chain

As mentioned before, LLMs predict new outputs given an input based on weights that have been generated by training on a dataset. In particular, given an input sequence of tokens, e.g. zeros and ones, it predicts the next token to be concatenated. If you then want to create longer sequences of generated data, you recursively enter the outputs back into the system.

Of course, computers have finite memory. Therefore, to make sure that we do not create a memory overflow, we introduce a *context length*. This context length is simply a window that determines how many of the previous tokens in the sequence we take into account when predicting the next token. One could think of this as the model's memory.

With this characterization of an LLM we can reformulate the setup as follows. Given a set of tokens T, context length n and input X_{i-1} consisting of n tokens, an LLM generates a new random variable X_i of token length n. In other words, it is am irreducible finite state Markov chain with state space T^n and transition probabilities generated by the neural network.

To illustrate this fact, consider a GPT with $T = 0, 1$, $n = 3$ and trained on the dataset 111101111011110. After 50 steps of training it produces the Markov chain in figure 1. As expected, the transitions that are in the training data have a higher probability of occurring. For example, the transition from 011

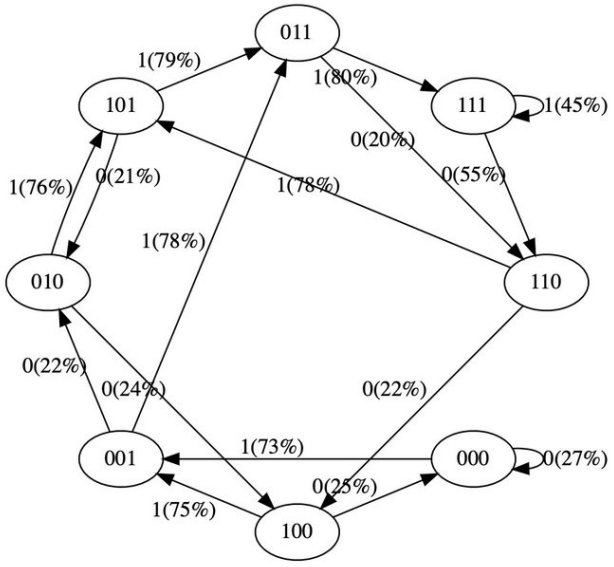

Figure 1 – The GPT Markov chain generated by dataset 111101111011110.

into 111 is much higher than into 010. Another interesting observation is that the states that are not encountered in the training data have non uniform transition probabilities. In particular, we see that there is a bias towards adding ones. A peculiar property, but very desirable for when our input is a state that is not in the training data. After all, we want to move towards the trained vocabulary with high probability.

3 Hoeffding Bound on Markov Chain

Since we have seen that we can reformulate the GPT model to a finite state Markov chain, we can try to derive concentration inequalities for the states of the LLM.

We are interested in the behavior of the partial sums $\sum_{i=1}^{n} f(X_i)$, where $\{X_i\}_{i\in\mathbb{Z}}$ is an irreducible Markov chain generated from tokens T and context length n, and $f : T^n \longrightarrow [a, b]$ is real-valued.

We will show that the following theorem [4] holds

Theorem 3.1 (Hoeffding for Markov Chains)**.** *Let $\{X_i\}_{i\in\mathbb{Z}}$ be an irreducible Markov chain on a finite state space S, driven by an initial distribution q, and an irreducible probability matrix P. Let $f : S \longrightarrow [a, b]$ be a real-valued function. Then for any $t > 0$*

$$\mathbb{P}_q\left(\left|\sum_{i=1}^{n}\left(f(X_i) - \mathbb{E}_q[f(X_k)]\right)\right| \geq t\right) \leq 2\exp\left\{-\frac{t^2}{2\nu^2}\right\},$$

where $\nu^2 = \max_{x,y\in S} \frac{1}{4}n(b-a)^2\mathbb{E}[T_y|X_1 = x]^2$

Proof. We start in a similar vein to the proof of McDiarmid's inequality by defining the filtration $\mathcal{F}_k = \sigma\{X_1, ..., X_k\}$, where we set $\mathcal{F}_0 = \sigma\{\emptyset\}$. Furthermore, we also take the same martingale differences $\Delta_k = \mathbb{E}\left[S_{1,n}|\mathcal{F}_k\right] - \mathbb{E}\left[S_{1,n}|\mathcal{F}_{k-1}\right]$, where $S_{1,n} = \sum_{i=1}^{n} f(X_i)$. Next, we want to control these martingale differences. We start off by noting that

$$\min_{y\in S} \mathbb{E}\left[S_{1,n}|\mathcal{F}_{k-1}, X_k = y\right] - \mathbb{E}\left[S_{1,n}|\mathcal{F}_{k-1}\right] \leq \Delta_k,$$

and

$$\Delta_k \geq \max_{x\in S} \mathbb{E}\left[S_{1,n}|\mathcal{F}_{k-1}, X_k = x\right] - \mathbb{E}\left[S_{1,n}|\mathcal{F}_{k-1}\right].$$

This means that to control Δ_k, we need to bound

$$\begin{aligned}
&\max_{x\in S} \mathbb{E}\left[S_{1,n}|\mathcal{F}_{k-1}, X_k = x\right] - \min_{y\in S} \mathbb{E}\left[S_{1,n}|\mathcal{F}_{k-1}X_k = y\right] \\
&\quad = \max_{x,y\in S}\{\mathbb{E}\left[S_{n,k}|X_k = x\right] - \mathbb{E}\left[S_{n,k}|X_k = y\right]\} \\
&= \max_{x,y\in S}\{\mathbb{E}\left[S_{1,n-k+1}|X_k = x\right] - \mathbb{E}\left[S_{1,n-k+1}|X_k = y\right]\},
\end{aligned}$$

where the Markov property was used in the second step and the homogeneity of the transition matrix in the last step. Since we are in an irreducible finite state Markov chain, we can use a hitting time argument. Note that if we set $T_y = \inf\{n \geq 0 : X_{n+1} = y\}$, we can find the bounds on $S_{1,n-k+1}$ in the forms of

$$S_{1,n-k+1} \leq bT_y + S_{T_y+1,n-k+1}, \;\; aT_y + S_{T_y+1,n-k+1} \leq S_{T_y+1,T_y+n-k+1},$$

where we used the fact that $f : S \longrightarrow [a, b]$. Combining the two inequalities gives that

$$S_{1,n-k+1} \leq T_y(b-a) + S_{T_y+1,T_y+n-k+1}.$$

By taking conditional expectations on $X_1 = x$ and using the Markov property we get

$$\mathbb{E}[S_{1,n-k+1}|X_1 = x] \leq (b-a)\mathbb{E}[T_y|X_1 = x] + \mathbb{E}[S_{1,n-k+1}|X_1 = x].$$

Thus, we find that

$$0 \leq \Delta_k \leq \max_{x,y\in S} \mathbb{E}[T_y|X_1 = x].$$

Using Hoeffding's lemma, we now find that

$$\mathbb{E}\left[e^{\theta\Delta_k}|\mathcal{F}\right] \leq \exp\left\{\frac{\max_{x,y\in S}\mathbb{E}[T_y|X_1 = x]^2\theta^2(b-a)^2}{8}\right\}.$$

The result then follows by using the method of McDiarmid's inequality, i.e. applying Markov and then succesively conditioning. □

The challenge is then to calculate the maximum expected hitting time. Due to the irriducibility and finite state space, it is rather straightforward to calculate $\max_{x,y\in S}\mathbb{E}[T_y|X_1 = x]$. For this, we use Lemma 2.12 from [1], which gives a relation between the mean hitting time $\mathbb{E}[T_y|X_1 = x]$, the fundamental matrix Z and the stationary distribution π:

$$\pi_y\mathbb{E}[T_y|X_1 = x] = Z_{y,y} - Z_{x,y}. \tag{1}$$

This gives the following procedure:

1. The first step is to calculate the stationary distribution π. Because the Markov chain generated by GPT is irreducible with a finite state space, each state is non-null persistent, therefore a stationary state is guaranteed to exist.

2. Next, the fundamental matrix $Z = (I - P + D)^{-1}$ needs to be found, where I identity, P transition matrix, D stationary distribution matrix. This matrix is also guaranteed to exist, due to the fact that the chain is ergodic.

3. Since the entries of the fundamental matrix, $Z_{i,j}$, tell us the expected number of times the chain hits state i when started in j before to returning to j, we can calculate $\mathbb{E}\left[T_j|X_1 = i\right] = (Z_{jj} - Z_{ij})/\pi_j$.

4. So by calculating this value for each combination of $x, y \in S$, we obtain $\max_{i,j}\mathbb{E}\left[T_i|X_1 = j\right]$.

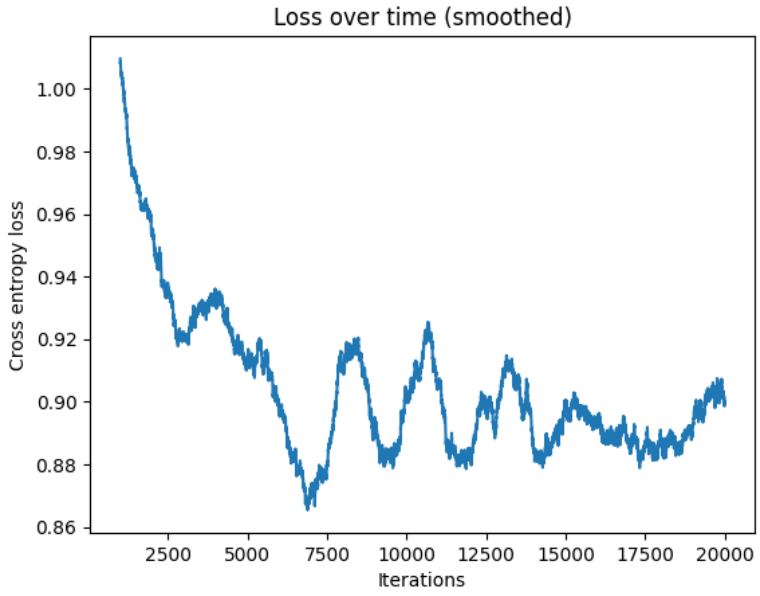

Figure 2 – Training of the larger model

4 Experiment

We want to apply this to a practical setting. We could experiment in a synthetic language model, setting the state-transition probabilities manually. Since we'll consider small statespaces, this is possible. However, the transitition probabilities should be representative for those that could come from a LLM. That is not trivial to manually do, because LLMs, as their name suggests, have a large amount of parameters. So, we do not go the manual route, but instead train real LLM models. We use a tiny, decoder-only, implementation of GPT-2, [3]. This is trained with the AdamW descent algorithm. For this reason, we consider two examples, both with a small statespace, but transition probabilities generated by relatively large LLMs. First, we consider the small example mentioned in a previous sector, with tokens $0, 1$ and context size 3. The model is trained on all possible transition in the small dataset of 111101111011110. Secondly, we do an experiment using 40.000 lines of the Shakespeare corpus. Regrettably, for the interesting setting of 27 tokens (the alphabet + space), our application is computationally challenging. Instead we consider the vowels, consonants and spaces as three different tokens. Although a limited case, we hope that this encapsulates the underlying structure of language in the transition probability. Then, training batches are randomly generated from the dataset.

For the small example, with 8 states, we configure the GPT model to have 4 heads of 4 layers with an embedding dimension of 16, so the model has in total 12656 parameters. For the larger example, with 81 states, we configure the GPT model to have 4 heads of 4 layers with an embedding dimension of 64, so the model has in total 198208 parameters. T For reference, the full GPT-2 implementation has 12 heads of 12 layers, with embedding dimension 768. This comes to a total of 1.5 billion parameters. Consider that, with a vocabulary size of 50304 and a context length of 1024, if we'd consider GPT-2 as a finite state markov chain, the statespace would be of size 2.7610^{4814}.

See figure 2, to see how the cross-entropy, the considered loss metric, decreases as the parameters are fitted. As the loss stops increasing, the remaining entropy is inherent to the data.

We now discuss the results of the maximum hitting over the choice of the Temperature hyperparameter. In figure 3, we see that the maximum hitting time decreases at a double exponential rate

as the temperature gets higher. In fact, as we see in figure 6 there appears to be some sort of *phase-transition* at around $T = 0.11$, for both models, where the hitting time absolutely explodes. Another interesting observation is that the maximum hitting time is significantly lower for the larger model. This could be inherent to the data, or a consequence of a larger state offering different paths from state to state, thus alleviating improbable bottlenecks.

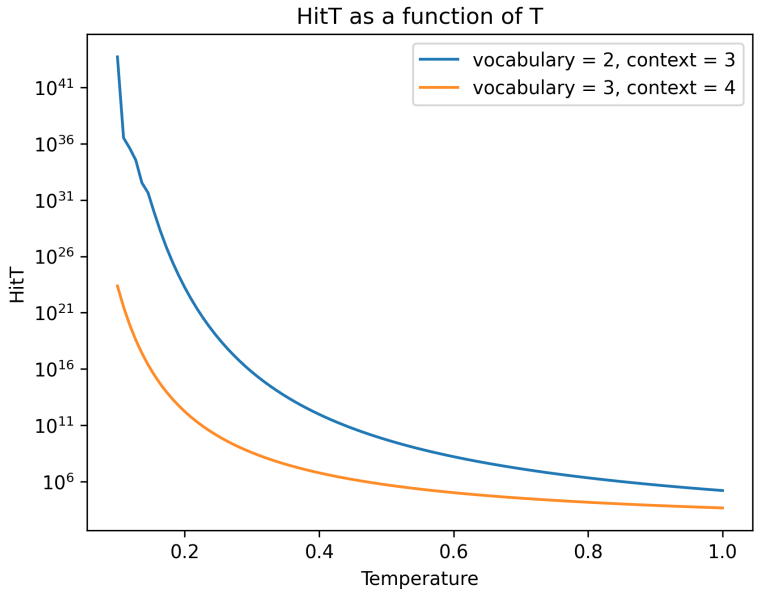

Figure 3 – The maximum hitting time decreases with higher temperature

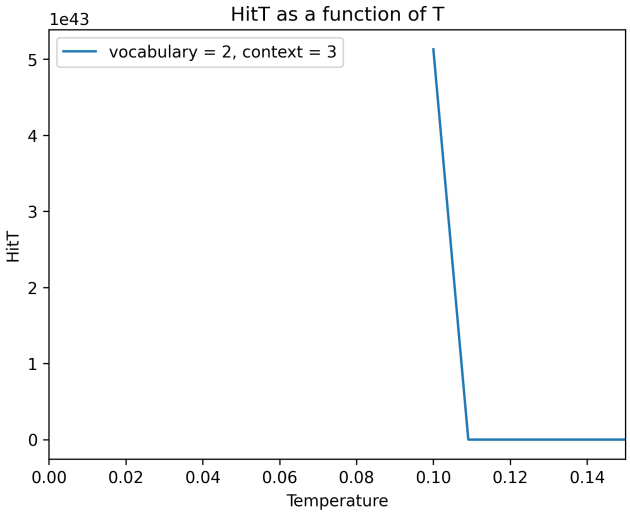

Figure 4 – Statespace 8

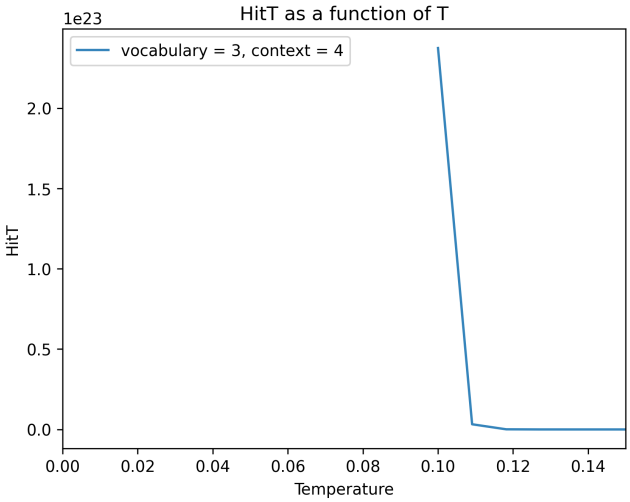

Figure 5 – Statespace 81

Figure 6 – Closeup of phase-transition for both experiments

4.1 Ising model connection

The softmax probabilities and empirical phase transition brings to mind a model that is ubiquitous in physics: the Ising model. This model is used to study magnetization in ferromagnetic materials. In the Ising model, the material is modeled as a d-dimensional lattice, where each lattice site is occupied

by a particle that either has spin up (+1) or spin down (-1). The probability of the particle being in either configuration is then determined by the interaction Hamiltonian of the system. An example of such a Hamiltonian would be the case in 2D, where the particle wants to copy the orientation of its neighbors. The probability of the system being in a certain state $s_i \in \mathbf{s}$ is then

$$\mathbb{P}_T(s_i) = \frac{e^{-H(s_i)/T}}{\sum_{\mathbf{s}} e^{-H(\mathbf{s})/T}}, \tag{2}$$

where H is the interaction Hamiltonian. This is of course remarkably similar to the definition of our LLM, the differences being that the Hamiltonian of our state is now determined by a complex function that is obtained by training the neural network and that we have more states. However, we can say something about the inner workings of the Hamiltonian of our neural network, by reminding ourselves of the fact the transition probabilities are dependent on the *context length*. In particular, it tells us that the probability of choosing a certain token is like a particle choosing its spin in an Ising model, while interacting with four previous neighbors. However, characterizing the entire system is this manner seems to be problematic, since the recursive nature of the LLM prohibits us to calculate the probabilities of sequences of tokens. This is because the interactions change based on the context tokens. However, such systems with changing interactions have also been modelled and are known as *spin glass* models. Research into connections between machine learning and spin glass models have been studied and show a promising future [2]. Concentration results for these types of probability measures, based on a Hamiltonian as stochastic process, could prove to be a framework to analyze LLMs and similar systems.

5 Avoiding Naughty States

Here we give an example of an application for concentration results. Imagine a company wants to deploy a large language model. It may very well be that some output is undesirable. The company decides to test the large language model before deployment. Can we develop a statistical test, that can guarantee with a certain confidence, some states will not arise? We attempt to answer this through the markov chain interpretation of the model. Then, undesirable output can be seen as a set of states to be avoided, which we call the *naughty states* $N \subset S$, where S is the full state space. A tester would manual evaluate each state that the LLM generates, and decide whether it is naughty or not. In our notation, this is the function $f : S \mapsto \{0, 1\}$, with $f(x) = \mathbb{I}_{x \in N}$. Then, this is a bounded function defined on the statespace of the markov chain, meaning we can apply the previously derived inequality.

Consider a statistical test. We first formulate a null hypothesis. $H_0 : \mathbb{E}_q[f(X_k)] = X_0$

Then let $\bar{X} := \frac{1}{n}\sum_{i=1}^{n} f(X_i)$. We now view $U = |X_0 - \bar{X}|$ as a test statistic. Then $\mathbb{P}\left(|X_0 - \bar{X}| \geq T\right)$ gives the p-value under the null-hypothesis, for which we have an upper bound through the Hoeffding inequality.

Along the same line, we can formulate a subset of the critical region for U, i.e. the range of values for which we reject the null-hypothesis. For a test with significance $100(1-\alpha)\%$ and test statistic T, the critical region is given by a t^* such that

$$\mathbb{P}\left(U \geq t^*\right) = \alpha.$$

By the Hoeffding Inequality we know that

$$\mathbb{P}\left(U \geq t^*\right) = \alpha \leq 2\exp\left\{-\frac{2(t^*)^2}{n\text{HIT}^2}\right\}.$$

Hence $t^* \leq \sqrt{-\frac{1}{2}\text{HitT}^2 \log \alpha/2}\sqrt{n}$. In other words, we find a more extreme critical region, lets call it the supercritical region, such that we can reject the null-hypothesis.

Numerical example

Let's return to our numeral example with the 81 states, with $1-\alpha = 0.95$. We find $\sqrt{-\frac{1}{2}\text{HitT}^2 \log \alpha/2} = \sqrt{-\frac{1}{2} \cdot 105595^2 \cdot -3.6889} \approx 1.43 \times 10^5$. So, we then find the supercritical region given by $t^{**} \approx (1.43 \times 10^5)\sqrt{n}$. To give an extreme example. Say that under the null-hypothesis $X_0 = 0.2$, then the earliest point for which the null-hypothesis can be rejected is then $0.8n = 440\sqrt{n} \implies n = 3.21 \times 10^{10}$. So, the model would have to enter naughty states for more than 300 thousand iterations in a row, to reject with 95%-certainty that the expectation of encountering a naughty state is not 0.2.

6 Conclusion

Markov Chains offer a toolkit to study LLms

The Markov Chain characterization offers a great backlog of research that can immediately be applied to concentration problems with regards to LLMs. Especially because for these models, the practical applications are much more advanced than the theoretical understanding, it useful to be able to apply well-understood theory to smaller LLM systems.

State space grows very quickly

GPT4 has 50k tokens, a context length of 2048 and thus a state space of 3*109623, which means that this approach is computationally infeasible to analyze real world systems.

Hitting time explodes at low temperature

Like other Gibbs measure models with sufficient interaction length, there is a phase transition between low and high temperature.

7 Discussion

Does the hitting time keep decreasing with larger state space?

We see a significant drop in the hitting time when moving from 8 states to 81 states. This is promising, but we cannot yet conclude whether this would hold as the state space increases.

Alternatives to softmax transformation of logits

Current state of the art LLMs use the softmax transformation to introduce stochasticity, as we do here. As mentioned earlier, these statistics are like working with the Boltzmann distribution. In particular, it is like we are treating each state as bosons. An interesting question to ask is what would happen if a different partition function was used. After all, it seems like fermions would be a better analogue to words, since they adhere to the exclusion principle.

Different quantity of interest than Hitting Time

A promising direction might to find a bound in terms of Markov Chain couplings or spectral gaps as described in [5].

8 Code Availability

The code used in the project has been made available on GitHub: https://github.com/LourensT/GPTasMC.

References

[1] David Aldous and James Allen Fill. *Reversible Markov Chains and Random Walks on Graphs*. 2002. URL: https://www.stat.berkeley.edu/~aldous/RWG/book.pdf.

[2] Giuseppe Carleo et al. "Machine learning and the physical sciences". In: *Reviews of Modern Physics* 91.4 (Dec. 2019). DOI: 10.1103/revmodphys.91.045002. URL: https://doi.org/10.11032Frevmodphys.91.045002.

[3] Andrej Karpathy. *gpt-finite-state.ipynb*. 2023. URL: https://colab.research.google.com/drive/1SiF0KZJp75rUeetKOWqpsA8clmHP6jMg?usp=sharing#scrollTo=mGHwSuHQuTXI.

[4] Vrettos Moulos. "A Hoeffding Inequality for Finite State Markov Chains and its Applications to Markovian Bandits". In: *IEEE International Symposium on Information Theory - Proceedings* 2020-June (Jan. 2020), pp. 2777–2782. ISSN: 21578095. DOI: 10.1109/ISIT44484.2020.9173931. URL: https://arxiv.org/abs/2001.01199v2.

[5] Daniel Paulin. "Concentration inequalities for Markov chains by Marton couplings and spectral methods". In: *Electronic Journal of Probability* 20.none (2015), pp. 1–32. DOI: 10.1214/EJP.v20-4039. URL: https://doi.org/10.1214/EJP.v20-4039.