Supplementary Material for Identifying Norms from Observation using MCMC Sampling, IJCAI 2021

Stephen Cranefield[1*] and **Ashish Dhiman**[2]

[1]Department of Information Science, University of Otago, Dunedin, New Zealand
[2]Department of Aerospace Engineering, Indian Institute of Technology, Kharagpur, India
stephen.cranefield@otago.ac.nz, ashish1610dhiman@gmail.com

1 The Metropolis-Hastings algorithm

The 'vanilla' textbook version of the Metropolis-Hastings algorithm [**?**] is shown below:

1: **procedure** METROPOLIS-HASTINGS(obs, n)
2: ▷ obs: observed data; n: num. samples desired
3: Sample $\theta^0 \sim p_0(\theta)$ such that $p(\theta^0|obs) > 0$
4: **for** $i = 1, \cdots, n$ **do**
5: Sample $\theta^* \sim J(\theta^*|\theta^{i-1})$
6: $r = \dfrac{p(\theta^*|obs)/J(\theta^*|\theta^{i-1})}{p(\theta^{i-1}|obs)/J(\theta^{i-1}|\theta^*)}$
7: $\theta^i = \begin{cases} \theta^* & \text{with probability } min(r, 1) \\ \theta^{i-1} & \text{otherwise} \end{cases}$
8: **end for**
9: Return $\langle \theta^1, ..., \theta^n \rangle$
10: **end procedure**

In the context of Bayesian inference, we have from Bayes' Theorem that $p(\theta^*|obs) = (p(\theta^*)\, p(obs|\theta^*))\, /p(obs)$ and likewise for $p(\theta^{i-1}|obs)$. These equalities can be applied to line 6 of the algorithm, with the two occurrences of $p(obs)$ cancelling out:

$$r = \frac{p(\theta^*)\, p(obs|\theta^*)\, /\, J(\theta^*|\theta^{i-1})}{p(\theta^{i-1})\, p(obs|\theta^{i-1})\, /\, J(\theta^{i-1}|\theta^*)} \tag{1}$$

We use that Bayesian-specific version of the algorithm in the paper.

2 MCMC over a probabilistic grammar

Saad et al. note that if the grammar generates a countably infinite language, it is necessary to check that the recursive process of generating an expression from the grammar will terminate with probability 1, by constructing its expectation matrix and verifying that its largest eigenvalue has a modulus less than 1. This holds for our grammar in the paper.

Saad et al. do not present their version of MCMC as an instance of the Metropolis-Hastings (M-H) algorithm, but rather give a long description and complex description from basic principles. In particular, they define their acceptance ratio with no motivation and then prove that it has the desired mathematical properties. We believe that the method becomes more approachable and simpler to understand when viewed as a Metropolis-Hastings, and here show how their acceptance ratio can be derived from that in the M-H algorithm.

In the paper we describe the jumping distribution of Saad et al. as a sequence of bullet points (pages 2–3). Here we present the probability of the jump from θ to θ^*, based on the exposition of Saad et al., but using a more concise notation, e.g. our $P_G(nt, \theta)$ corresponds to their $\mathsf{Expand}[\![E]\!](N)$.

$$J(\theta^*|\theta) = 1/|NI(\theta)| \sum_{n \in NI(\theta) \cap NI(\theta^*)} \mathbb{1}(\theta^* \backslash n = \theta \backslash n)\, P_G(nt(n, \theta), \theta^*[n])$$

where:

- $NI(\theta)$ is the set of node indices in the parse tree of θ, represented as paths from the root node. For example, $(1, 2)$ represents the path from the root through the first child note of the root, and then to that node's second child node. The size of an expression θ is defined as the size of the node indices set: $|NI(\theta)|$. In our paper, we abbreviate this as $|\theta|$.d
- $\mathbb{1}$ is the indicator function, mapping a Boolean expression to 1 (for true) or 0. This allows a concise presentation where certain terms, multiplied by a indicator expression, cancel out under a given condition.
- $\theta \backslash n$ denotes θ with the subterm at index n removed and replaced with a special "hole" symbol.
- $P_G(nt, \theta)$ is the probability of the grammar generating the expression θ starting from the non-terminal symbol nt.
- $nt(n, \theta)$ denotes the non-terminal symbol in the grammar that was used to generate the subexpression of θ at node index n. This is well defined, as each production rule in a tagged PCFG generates expressions beginning with a "tag", i.e. a symbol that is unique to that production.
- $\theta[n]$ denotes the subexpression of θ that has the node at index n as its root.

The equation above calculates the *average probability* of jumping from θ to θ^* across all node indices of θ, as a subterm substitution could be made at any of them. For a given index n, only θ^*s containing the same index are possible results, hence the intersection in the range of the summation

*Contact author

index. The argument to the indicator function asserts that θ and θ^* must be identical except for any differences in their subtrees at index n. The final term expresses the probability of the grammar expanding the non-terminal associated with $\theta[n]$ to produce the replacement subterm $\theta^*[n]$.

Finally, in Figure 1 we show how the analysis of Saad et al. [**?**] can be adapted to derive the acceptance rate $\frac{|NI(\theta^{i-1})|\,p(obs|\theta^*)}{|NI(\theta^*)|\,p(obs|\theta^{i-1})}$ from the formula for r in the Bayesian application of the Metropolis-Hastings algorithm. This results in the following equation:

$$r = \frac{|NI(\theta^{i-1})|\,p(obs|\theta^*)}{|NI(\theta^*)|\,p(obs|\theta^{i-1})} \tag{2}$$

3 Comparison with the approach of Cranefield et al. (2016)

Due to lack of space, the paper omitted details of the six norm types in the scenario considered by Cranefield et al. (2016). Informally, they are as follows: it may be obligatory or prohibited for an agent to pass through a given node (either unconditionally or conditionally after a given node is reached), or it may be obligatory or prohibited to traverse between a specified pair of adjacent nodes.

4 Progression of Posterior in MCMC chains

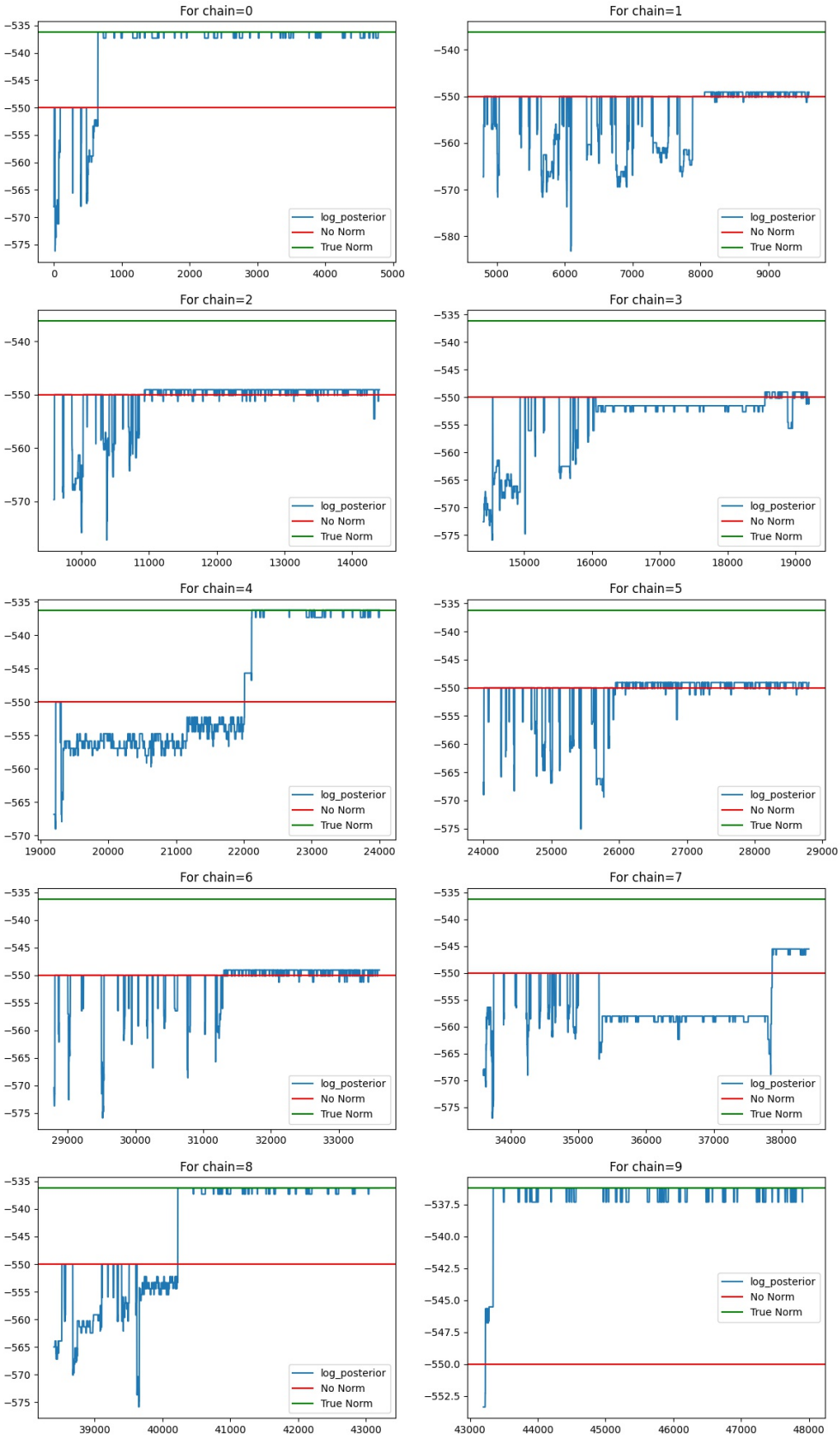
In the next three pages we present three plots (for different p_{nn} values), with subplots depicting progression of log posterior in the individual chains. The first is an experimental trial where $p_{nn} = 0.0$, the second for a trial where $p_{nn} = 0.3$ and finally for $p_{nn} = 0.55$. These plots illustrate how, for low values of p_{nn}, the no-norm hypothesis (red line) has a lower log posterior than the true norm (green line), but this trend is reversed for a high value of p_{nn}. This is primarily because with an increase in p_{nn}, the likelihood $p(obs|\theta)$ decreases (due to a decrease in the normative component of Likelihood), while the prior remains constant. At a certain threshold of p_{nn}, the gap in likelihoods will no longer be able to cover the gap in priors of the no-norm hypothesis (which has a higher prior according to the grammar). This is also the same p_{nn} threshold above which, True Norm ceases to be Norm with the highest log posterior in chains. This threshold, is intrinsic to the specific instances of 'observed behaviour', and the constraints affected by norm ('true') on such behaviour.

In many cases, we can see that not all chains find the true norm (or an equivalent expression with the same or very close log posterior). This is beacuse some chains might get stuck in certain locales of the search space. To alleviate this, we have used over-dispersed expressions (found using the vector representation detailed), such that each chain potentially initially explores a different region of search space, thereby helping us get a more accurate and robust approximation of the posterior distribution of expressions. The overdisperesed start points, thus help prevent the the worst case, i.e. where most chains are stuck, without finding True (or 'equivalent norm).

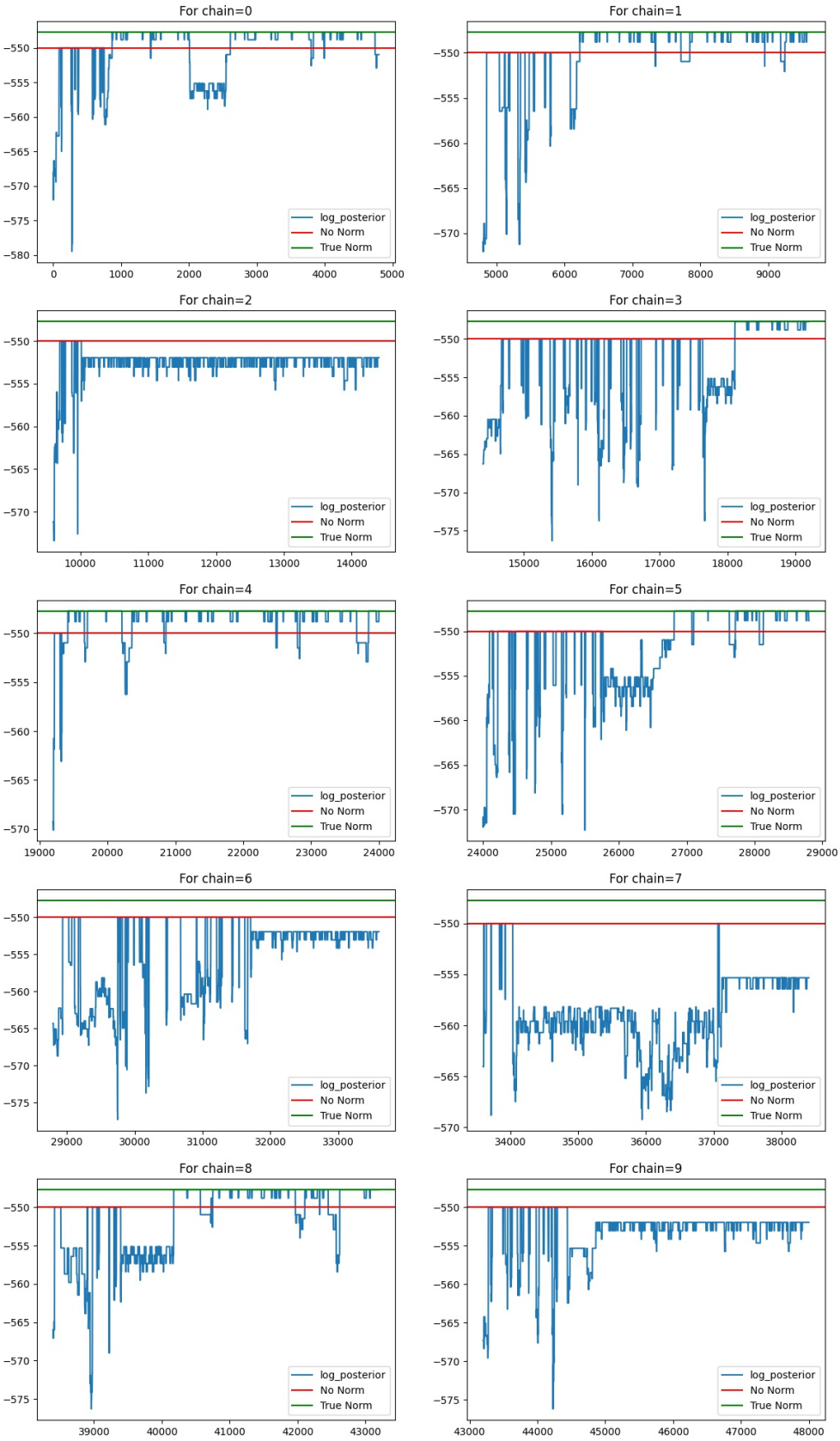
$$
\begin{aligned}
r &= \frac{p(\theta^*)\,p(obs|\theta^*)\,1/|NI(\theta^*)|\sum_{n\in NI(\theta^*)\cap NI(\theta^{i-1})}\mathbb{1}(\theta^{i-1}\backslash n=\theta^*\backslash n)\,P_G(nt(n,\theta^*),\theta^{i-1}[n])}{p(\theta^{i-1})\,p(obs|\theta^{i-1})\,1/|NI(\theta^{i-1})|\sum_{n\in NI(\theta^{i-1})\cap NI(\theta^*)}\mathbb{1}(\theta^*\backslash n=\theta^{i-1}\backslash n)\,P_G(nt(n,\theta^{i-1}),\theta^*[n])} \\
&= \frac{p(obs|\theta^*)\,1/|NI(\theta^*)|\sum_{n\in NI(\theta^*)\cap NI(\theta^{i-1})}\mathbb{1}(\theta^{i-1}\backslash n=\theta^*\backslash n)\,p(\theta^*)\,P_G(nt(n,\theta^*),\theta^{i-1}[n])}{p(obs|\theta^{i-1})\,1/|NI(\theta^{i-1})|\sum_{n\in NI(\theta^{i-1})\cap NI(\theta^*)}\mathbb{1}(\theta^*\backslash n=\theta^{i-1}\backslash n)\,p(\theta^{i-1})\,P_G(nt(n,\theta^{i-1}),\theta^*[n])} \quad \text{[Move priors inside summation]} \\
&= \frac{p(obs|\theta^*)\,1/|NI(\theta^*)|\sum_{n\in NI(\theta^*)\cap NI(\theta^{i-1})}\mathbb{1}(\theta^{i-1}\backslash n=\theta^*\backslash n)\left[\frac{\cancel{p(\theta^{i-1})}}{\cancel{p(\theta^{i-1})}}\frac{\cancel{P_G(nt(n,\theta^*),\theta^*[n])}}{\cancel{P_G(nt(n,\theta^*),\theta^{i-1}[n])}}\right]P_G(nt(n,\theta^*),\theta^{i-1}[n])}{p(obs|\theta^{i-1})\,1/|NI(\theta^{i-1})|\sum_{n\in NI(\theta^{i-1})\cap NI(\theta^*)}\mathbb{1}(\theta^*\backslash n=\theta^{i-1}\backslash n)\,\cancel{p(\theta^{i-1})}\,\cancel{P_G(nt(n,\theta^{i-1}),\theta^*[n])}} \quad \begin{array}{l}\text{[Saad et al. (Lemma 4.7) \&}\\ nt(n,\theta^*)=nt(n,\theta^{i-1})]\end{array} \\
&= \frac{|NI(\theta^{i-1})|\,p(obs|\theta^*)}{|NI(\theta^*)|\,p(obs|\theta^{i-1})} \quad \text{[The generation of } \theta^* \text{ guarantees that } \theta^*\backslash n=\theta^{i-1}\backslash n]
\end{aligned}
$$

Figure 1: Derivation of the formula for computing the acceptance rate r

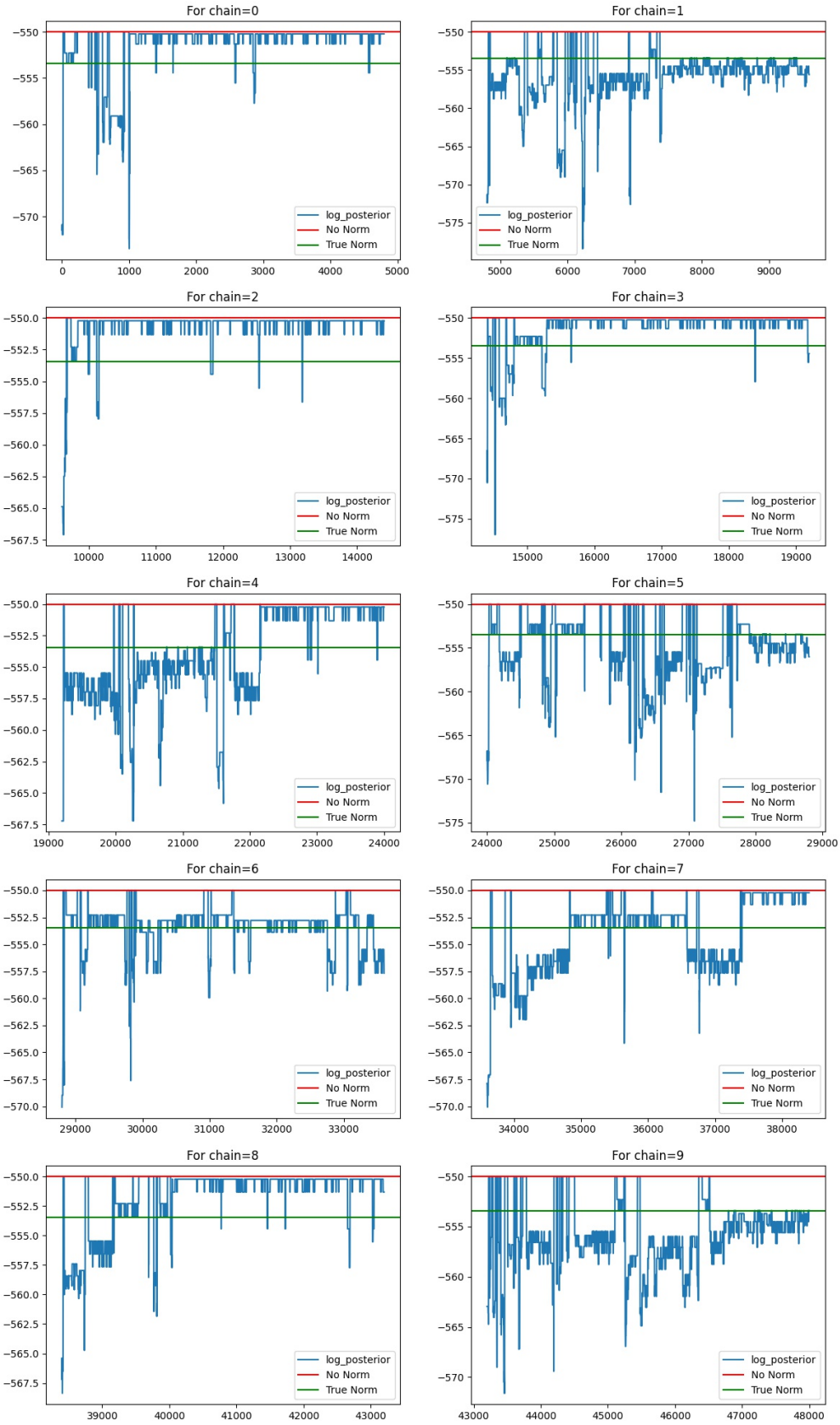
4.1 $p_{nn} = 0.0$


For chain=0
For chain=1
For chain=2
For chain=3
For chain=4
For chain=5
For chain=6
For chain=7
For chain=8
For chain=9
log_posterior
No Norm
True Norm

4.2 $p_{nn} = 0.3$


For chain=0
For chain=1
For chain=2
For chain=3
For chain=4
For chain=5
For chain=6
For chain=7
For chain=8
For chain=9
log_posterior
No Norm
True Norm

4.3 $p_{nn} = 0.55$


For chain=0
For chain=1
For chain=2
For chain=3
For chain=4
For chain=5
For chain=6
For chain=7
For chain=8
For chain=9
log_posterior
No Norm
True Norm

5 Prior Calculation from Grammar

In this section, we want to present an example of how Prior is calculated for a given Norm Expression. To this end we use the grammar described in figure: 1 of the main paper text.

And to illustrate the Prior Calculation, we use the True Norm used for experiments in main text, i.e.

```
['Norms', ['Obl', ['Moved', ['Colour', 'r'],
['Shape', 'any'], ['Zone', '1'], ['Next-Move',
['Colour', 'any'], ['Shape', 'any']]], ['Zone',
'2']], ['Per', ['Action', 'putdown'], ['Colour',
'any'], ['Shape', 'square'], ['PerZone', '3']]].
```

The prior calculation essentially requires following the tags in the expression, and tracing the path from the source symbol of Grammer to the Terminal symbols, while keeping a tab of the probablities associated with each production rule. This approach levied on the above Expression yeilds the folowing list of probablities:

```
[0.25, [0.5, [0.333, 0.166, 0.5, 0.333, [0.6666,
0.5, 0.5]], 0.333], [1, 1, 0.5, 0.166, 0.166]].
```

The prior then, is just a multiplication of all the above tag probablities, or 8.5787×10^{-07} in the case above. This is evidently equivalent to parsing down a tree, from source to end, and remebering the path, so one could trace back the steps. And in the figure:2 below, we have tried to set an intution for the same.

Figure 2: Schematic description calulation of Prior