# Dynamic temperature ladder Quick Burst

Richard Mitchell

December 2025

## 1 Introduction

The purpose of this document is to explain the implementation of the dynamic temperature ladder in the Quick Burst code. The implementation of the dynamic temperature ladder is based on this paper [1]. The equation used in the paper which actually updates the tempeartures is equation 13. However, this equation does not directly tell you how to update the temperature ladder since it is in terms of S which is defined in equation 11. Also it is worth noting that $\kappa(t)$ is defined equation 14. Because the implementation of the dynamic temperature ladder is left kind of ambiguous I have made this document to explain what is actually done and how to think about it. Before reading on please go read section 2.0, 2.1, 3.0, and 3.1 of the paper. You need this background. Its not important that you understand everything, but it is important you understand section 3.0 and 3.1.

## 2 The computation of the new temperatures

### 2.1 Updating the betas

Equation 13 in the paper is the following

$$S_i(t+1) - S_i(t) = \kappa(t)[A_i(t) - A_{i+1}(t)] \tag{1}$$

Lets put everything that depends on t on one side and everything that depends on $t+1$ on the other.

$$S_i(t+1) = S_i(t) + \kappa(t)[A_i(t) - A_{i+1}(t)] \tag{2}$$

Now in Quickburst the array which we have access to is the betas array which essentially holds the temperatures ($\beta = \frac{1}{T}$). So in order to actually update the temperature ladder we need to get equation 2 in terms of the new betas. But before we do that, just for sake of simplicity, we are going to define the following.

$$N = S_i(t) + \kappa(t)[A_i(t) - A_{i+1}(t)] \tag{3}$$

So that we can write

$$S(t+1) = N \tag{4}$$

And this will make our computaitons we do simpler. Anyway we use the definiton of S in equation 4.

$$ln(T_i(t+1) - T_{i-1}(t+1)) = N \tag{5}$$

Then we exponentiate and bring the $T_{i-1}$ over to the other side

$$T_i(t+1) = T_{i-1}(t+1) + e^N \tag{6}$$

This then is the equation we will use to update our temperature ladder. Now as you read in the paper, you know that the highest temperature and the lowest temperature are never updated. So, for example, if we have 5 chains, we would leave chain 0 and 4 alone and update chain 1, 2, and 3 (note here I am using the 0 indexing which is used in code). So how this algorithm for updating works is that you need to start with $i = 1$ and make the for loop up to $n\_chains - 1$ index.

### 2.2 Note on $A_i(t)$

Notice in the paper defines that $A_i(t) = A_{i,i-1}(t)$ which is the swap probability between chain i and i-1. However, Quickburst already had an array of acceptance rates which was defined differently. In Quickburst the chain index 0 indicates the swap acceptance probability between chain 0 and chain 1. And likewise chain index 1 keeps track of the swap probability of chains 1 and 2. This is different from the paper which would say that chain index 1 should keep track of the swap probability of chains 0 and 1. Just keep this in mind when understanding the code.

### 2.3 Alterations from the original algorithm

There was some stuff in the original algorithm I thought didn't make much sense so I changed some of the implementation. The first thing I changed was having the highest temperature be infinite. I thought this was a poor choice as it allows the highest temperature to basically go anywhere without bound. It would allow the highest temperature to go places which makes no sense. This is going to cause the acceptance probability between the highest chain and the chain below it to be super small since the highest chain could be at some nonsense place. This would seem to perpetually raise the temperature of all the other chains to acheive uniform acceptance probability. Therefore the highest temperature I use is the highest temperature set by the user. I figure the user should just set this temperature to be very high, and the dynamic temperature ladder will make adjustments to bring the temperatures to sensible values.

The other thing I didn't keep the same was when the temperature is updated. I think the original paper called for the temperature to be updated every single step. I didn't think that made much sense since the only time the acceptance ratio array is updated is at a parallel tempering swap. Therefore, temperatures will be monotomically increasing or decreasing if there are no parallel tempering swaps happening. This didn't make sense to me. It seemed much better to only update the temperature ladder every few parallel tempering steps(I chose 10 in the program). Therefore, program at least has some time to let the effect of the new temperatures set in.

The last thing is that, in $\kappa(t)$ notice that it has the variable t. I use the iteration of the current noise step as t.

## References

[1] I. Mandel W. D. Vousden W. M. Farr. "Dynamic temperature selection for parallel-tempering in Markov chain Monte Carlo simulations". In: *Arxiv* (2016). DOI: `https://doi.org/10.1093/mnras/stv2422`.