Supplementary material to "Tuning diagonal scale matrices for HMC" by Jimmy Huy Tran and Tore Selland Kleppe

A Implementation of median crossing times

In Section 3.3, the tools needed for the process of tuning the scale matrix $\mathbf{S}$ using median crossing times are introduced. Now, a summary of how the method is implemented will be presented.

First, it is stated that the median cannot be estimated continuously in a similar fashion as VARI and ISG using (10). Under the setting of GRHMC, it is convenient to start out with $\mathbf{m}$ and $\mathbf{S}$ as the zero vector and identity matrix, respectively. A small part of the burn-in period, say $c_{\text{burn}}t_{\text{burn}}$, $c_{\text{burn}} \in [0,1]$, is spent specifically to build up a first representative estimate of $\mathbf{m}$ before the optimization of $\mathbf{S}$ takes place. By defining a grid of time points with equal spacing Δ over the burn-in period, i.e. Δi, $i = 1, 2, \ldots, i^*_{\text{burn}}, \ldots, i_{\text{burn}}$ such that $\Delta i^*_{\text{burn}} \leq c_{\text{burn}}t_{\text{burn}}$ and $\Delta i_{\text{burn}} \leq t_{\text{burn}}$, $\mathbf{m}$ is recalculated at step i based on $\mathbf{q}(\Delta i)$ using the P^2-algorithm by Jain and Chlamtac (1985), a technique for estimating quantiles dynamically without the need of storing all previous observations. For high-dimensional problems and longer burn-in periods, this approach will significantly reduce the space complexity issues that occur in such settings when the standard method of estimating quantiles is applied. The reader is referred to the article for further details regarding the P^2-algorithm. After i^*_{burn} steps, $\mathbf{m}$ is assigned to the estimated median vector at this time point and the tuning of $\mathbf{S}$ can start until the burn-in period ends while still calculating a new estimate of $\mathbf{m}$ at the remaining time points $\Delta i^*_{\text{burn}}, \ldots, \Delta i_{\text{burn}}$. Note that smaller Δ will lead to more computational effort and for high dimensional setting, it might be preferable with a moderate value of Δ.

Compared to the VARI and ISG approach, it is also mentioned at the end of Section 3.3 that adaptive updates of $\mathbf{m}$ and $\mathbf{S}$ in (5)-(8) are done each time a position coordinate crosses its respective (estimated) median. Based on the described version of the dual averaging method, one has to update S_j whenever q_j crosses m_j while letting the remaining elements of $\mathbf{S}$ unchanged. For the median vector $\mathbf{m}$, one can either update only m_j at such events or the whole $\mathbf{m}$ if one favors frequently updates of the center vector across all coordinates. This might lead to estimates of $\mathbf{m}$ slightly closer to the true median vector at the end of the burn-in period, but requires more computations, especially for large d and small $\tilde{T}$. Therefore, the former approach is applied to the setting considered in Section 6 while the latter is performed in Section 4 and 5, where the performances are studied in detail and the common updates of all coordinates of $\mathbf{m}$ might be beneficial.

An important note regarding the approach of MCT is that it is much more sensitive because of the stochastic mechanism of GRHMC. Due to the momentum refresh event, it is possible to obtain either very small or very large median crossing times, which in turn affect the estimates from the dual averaging scheme. The former case could occur if a momentum refresh event takes place only a short period of time after a median crossing event and changes the direction of motion towards the median again. On the other hand, large median crossing times are

obtained whenever a momentum refresh event happens and alters the velocity in the opposite direction when the coordinate is very close to its median. Therefore, in order to obtain more stable and comparable estimates of S_j when running the same setup multiple times, one can either increase γ_j or t_{burn}. The drawback with increasing γ_j is that the shrinking effect will become more apparent, especially for small values of t_{burn}, and the estimates of $\log(S_j)$ will not be able to move far away from μ_j during this short time span. Increasing t_{burn} requires of course more computational power, which one wants to avoid if possible.

To balance the issues above without increasing the demand of computational effort, the following strategy regarding the MCT approach is proposed:

- Instead of one single run with MCT, a two-step strategy of burn-in using MCT is applied. Let t_{burn_1} and t_{burn_2} denote the duration of the simulation in the first and second step of burn-in. The first burn-in period will be considerably shorter as the intention of this part is to give a better initial configuration of the second burn-in period. Unless noted, $t_{\text{burn}_1}/t_{\text{burn}_2} = 0.2$. The default duration of the longer burn-in period t_{burn_2} is 5000 time units. The total burn-in duration is therefore $t_{\text{burn}} = t_{\text{burn}_1} + t_{\text{burn}_2} = 6000$ for the default setting. For both runs, κ_j and $\kappa_j^{(0)}$ from the dual averaging algorithm are set to 0.75 and 10 for all coordinates, respectively, and c_{burn} is set to 0.05. The desired median crossing time for all coordinates is equal to $\tilde{T} = \pi$ by default, a constant momentum refresh event rate of $\lambda = 0.2$ is considered and $\Delta = 1$ in the P^2-algorithm. On the other hand, for the Stock and Watson model considered in Section 6, Δ is set to 10 due to the reasons regarding computational effort mentioned above.

- For the first run, there is no prior information available, and $\mathbf{m}$ is therefore initialized as the zero vector and $\mathbf{S}$ as the identity matrix. This leads to the choice of $\mu_j = 0$ for all j, i.e. the estimates of S_j are shrunken towards unity. The shrinkage parameter for the first run is chosen to be $\gamma_j = 10$. The resulting estimate of S_j obtained should at least be closer to the ideal value if it is indeed significantly different from unity. Therefore, the final value of S_j and m_j from this first step of burn-in can be used to initialize the second and longer burn-in period, which in theory should lead to faster convergence.

- For the second step of burn-in, the estimates from the previous step are used as starting point for $\mathbf{m}$, $\mathbf{S}$ and μ_j in the dual averaging method by letting $\mu_j = \log(S_j)$. However, since the first run tends to underestimate when the true value is larger than 1, it can be beneficial to consider $\mu_j = \alpha_j \log(S_j)$ with $\alpha_j > 1$ to compensate for this fact. By default, α_j is set to 1.1 for all j. Finally, in contrast to the first step, γ_j is set to a larger value of 25 as the procedure has now received a head start when initializing it with the results from the first run. This should reduce the variation in the results for different iterations while still obtaining final estimates that are close to the true value without increasing the duration of the burn-in period.

B Further details related to implementation

In order to get a sense of how the performance varies between the approaches given a specific model/situation, the following strategy is adopted for all three methods when simulating samples in Section 4, 5 and 6:

- The burn-in period is now split into two parts such that $t_{\text{burn}} = t_{\text{burn}_{\mathbf{S}}} + t_{\text{burn}_\lambda}$: The first period of $t_{\text{burn}_{\mathbf{S}}}$ time units deals with the tuning of $\mathbf{m}$ and $\mathbf{S}$ given a specified and fixed constant event rate λ_{initial}. After this is done, $\mathbf{m}$ and $\mathbf{S}$ are fixed using the estimates from the recent run and the remaining t_{burn_λ} time units of the burn-in period is spent to select a suitable value of λ. In practice, both $\mathbf{m}$, $\mathbf{S}$ and λ are tuned simultaneously. However, in order to get a better understanding of the presented methods themselves, this splitting is easier for quantifying the marginal effect of each scale matrix tuning approach. For the experiments to come, $t_{\text{burn}_{\mathbf{S}}} = 6000$ (same time length mentioned in the implementation of MCT in Appendix A) while $t_{\text{burn}_\lambda} = 5000$. In addition, λ_{initial} is chosen to be 0.2 as default if it is not mentioned otherwise.

- Regarding the tuning process of the momentum refresh event rate λ, the methodology presented in Section 4.3.2 by Kleppe (2022) is applied. The default value of the smoothing parameter in the exponential smoothing process is 0.99.

- After t_{burn}, $\mathbf{m}$, $\mathbf{S}$ and λ are completely fixed and N samples is generated for the remaining $t_{\text{sample}} = T - t_{\text{burn}}$ time units with equal spacing.

- As a final comment, $\mathbf{m}$ and $\mathbf{S}$ are also initialized as the zero vector and identity matrix by default for the ISG and VARI approach just like in the description of the median crossing time method in Appendix A.