1. Background on Parallel Tempering Markov chain Monte Carlo

1.1 Parallel Tempering Markov chain Monte Carlo

Parallel Tempering Markov chain Monte Carlo runs M chains simultaneously, each of which is updated via the Metropolis-Hastings algorithm, where the metropolis ratio for a target distribution π of a Markov chain θ_i^m for chain m at iteration i is given by

$$\alpha(\theta_i^m, \theta^*) = \left(\frac{\pi(\theta^*)}{\pi(\theta_i^m)}\right)^{1/T^m}$$

where θ^* is a proposed position, and T^m is such that $T^0 = 1$, and $T^{m+1} > T^m$ (monotonicity). When T^m is small (cold) the chains run similar to a random-walk Metropolis-Hastings algorithm. However then T^m is large (hot) the target distribution is flattened and the Markov chain can explore the parameter space more freely, meaning they are less likely to get stuck in local modal points. After each time step, adjacent chains can swap Markov chain positions according to a probability given by the metropolis ratio:

$$\alpha_{m,m+1}(\theta_i^m, \theta_i^{m+1}) = \left(\frac{\pi(\theta_i^{m+1})}{\pi(\theta_i^m)}\right)^{1/T^m - 1/T^{m+1}}$$

This means that local modal points found by the explorative warmer chains can be adjacently passed down to the colder chains which estimate the posterior distributions (in this algorithm the posterior distribution is estimated using samples from the coldest ($T_0 = 1$) chain only). Therefore, this algorithm allows for efficient mixing for disconnected and multi-modal target distributions.

The `ptmc` R package also allows for two contemporary extensions of standard parallel tempering Markov chain Monte Carlo; an adaptive covariance matrix and adaptive temperature ladder.

1.1.1 Adaptive covariance matrix for each chain

The proposal distribution is given by

$$q(.|\theta_i^m) \sim \begin{cases} \mathcal{N}(\theta_i^m, \exp(\lambda_i) I_d), & i \leq i_{covar} || m > 4 || p < 0.05 \\ \mathcal{N}(\theta_i^m, \exp(M_i)\Sigma_d), & i > i_{covar} || p > 0.05 \end{cases}$$

- $\mathcal{N}(\theta_i^m, \exp(\lambda_i) I_d)$ is a multivariate normal distribution with covariance matrix given by the identity (I_d) multiplied by an adaptive scalar value (λ_i) updated at each time step according to a stochastic approximation equation $\lambda_{i+1} = \lambda_i + \gamma_i(\alpha(\theta_i^m, \theta^*) - 0.234)$, $\gamma_i = (1+i)^{0.5}$)
- $\mathcal{N}(\theta_i^m, \exp(M_i)\Sigma_d)$ is a multivariate normal with the covariance matrix given by the the previous samples in the chain and the formula for M_i is as given for λ_i but with $\gamma_i = (1 + i - i_{covar})^{0.5}$

- i_{covar} is the number of iterations to run before the covariance matrix starts estimating. Having a large value here allows the Markov chain to find a sensible place before the covariance matrix is estimated.

For warmer chains ($m > 4$) the adaptive covariance matrix is not used, as its use leads to very poor mixing. In the colder chains, after i_{covar} steps, to ensure that the Markov chains converge to the target distribution, a sample from $\mathcal{N}(\theta_i^m, M_i\Sigma_d)$ occurs 95% of the time, and a sample from $\mathcal{N}(\theta_i^m, \lambda_t I_d)$ occurs 5% of the time (i.e. $p \sim \mathcal{U}(0,1)$, as proposed in Sherlock et al. 2010.

In the `ptmc` R package, the `onAdaptiveCov` setting triggers the above proposal, if FALSE then only $\mathcal{N}(\theta_i^m, \lambda_t I_d)$ is used (default = TRUE). `burninAdaptiveCov` is the value of i_{covar} and `updatesAdaptiveTemp` is the frequency they update (default = 1, every step).

1.1.2 Adaptive tempering ladder

The values of the temperature ladder (T^m) influence the rate at which adjacent Markov chains are swapped. By allowing temperature values to change at each time step (T_i^m) according to the stochastic approximation as used for the proposal scaling parameters (λ_i and M_i), it is possible force inter-chain swapping rates of 0.234 (as is optimal). The algorithm works by defining $T_0^m := 10^{7m/(M-1)}$, $S_0^m := \log(T_0^{m+1} - T_0^m)$ and updating according to the equations:

$$S_{i+1}^m = S_i^m + \gamma_i(\alpha_{m,m+1} - 0.234)$$

$$T_i^{m+1} = T_i^m + \exp(S_i^m)$$

In the `ptmc` R package, the `onAdaptiveCov` boolean variable in settings toggles whether adaptive temperature parameters are on (default = TRUE), with the `updatesAdaptiveTemp` variable stating how often the adaptive temperature ladder updates (default = 1, every step).

1.2 Further reading

For a fuller understanding of parallel tempering MCMC algorithms I recommend reading:

- Chapter 6 of Advance MCMC provides a good introduction to all population type chains, including parallel tempering MCMC. Also Syed et al. 2019
- Adaptive covariance PTMC guide
- Adaptive Temperature Ladder PTMC: Miasojedow et al. 2012, Vousden et al. 2015