Computational Statistics
Project Report

18th March 2020

1 Introduction

In diesem Projekt reimplementiere ich den vorgeschlagenen Markov Chain Monte Carlo (MCMC) Algorithmus aus dem Paper [Lau and Krumscheid, 2019] und versuche die Ergebnisse der Experimente zu reproduzieren. Der untersuchte Paper baut auf dem *Metropolis-Hastings* MCMC Algorithmus [Metropolis et al., 1953] auf. [Liu et al., 2000] Der source code unter MIT Lizenz zu diesem Projekt findet sich in dem GitHub repository

`github.com/rinkwitz/Adaptive_Plateau_MCMC`.

2 Adaptive Component-wise Multiple-Try Metropolis Algorithm

Der Kernalgorithmus des Papers [Lau and Krumscheid, 2019] besteht aus einem MCMC Algorithmus der drei Eigenschaften erfüllt. Der Algorithmus schlägt beim sampling mehrere Vorschläge aus verschiedenen Plateauverteilungen vor. Dies passiert unabhängig für alle Komponenten eines samples. Die Plateauverteilungen adaptieren ihre Form in Abhängigkeit von der Frequenz der akzeptierten sample Vorschläge.

2.1 Plateau Proposal Distributions

Der MCMC Algorithmus in [Lau and Krumscheid, 2019] verwendet zum sampling von Vorschlägen *non-overlapping plateau proposal distributions*. Die dabei verwendete grundlegende probability distribution function f ist eine Kombination einer uniform distribution mit exponentiell decaying tails. Dabei ist die Verteilung f konstant um einen Mittelwert μ in dem abgeschlossenen Intervall $[\mu-\delta, \mu+\delta]$ mit $\delta > 0$. Ausserhalb dieses Intervall folgt die Verteilung einem exponentiellen Verfall dessen tail Breite durch ein seitenabhängiges

$\sigma_i > 0$ bestimmt wird, je nachdem ob man sich auf der linken oder rechten Seite des Intervalls $[\mu - \delta, \mu + \delta]$ befindet. Definiert man eine unnormalisierte Verteilungsfunktion

$$\tilde{f}(y; \mu, \delta, \sigma_1, \sigma_2) = \begin{cases} \exp\left(-\frac{1}{2\sigma_1^2}[y - (\mu - \delta)]^2\right) & , y < \mu - \delta \\ 1 & , \mu - \delta \leq y \leq \mu + \delta \\ \exp\left(-\frac{1}{2\sigma_2^2}[y - (\mu + \delta)]^2\right) & , y > \mu + \delta \end{cases}$$

und berechnet das folgende Integral

$$\begin{aligned} C(\delta, \sigma_1, \sigma_2) &= \int_{-\infty}^{\infty} \tilde{f}(y; \mu, \delta, \sigma_1, \sigma_2) dy \\ &= \int_{-\infty}^{\mu-\delta} \exp\left(-\frac{1}{2\sigma_1^2}[y - (\mu - \delta)]^2\right) dy + \int_{\mu-\delta}^{\mu+\delta} 1 dy + \int_{\mu+\delta}^{\infty} \exp\left(-\frac{1}{2\sigma_2^2}[y - (\mu + \delta)]^2\right) dy \\ &= \frac{\sqrt{2\pi\sigma_1^2}}{2} + 2\delta + \frac{\sqrt{2\pi\sigma_1^2}}{2} \end{aligned}$$

als die Summen von 2 halben Gausschen Integralen und einem Integral über eine konstatne Funktion, dann ergibt sich die normalisierte Verteilungsfunktion $f(y; \mu, \delta, \sigma_1, \sigma_2) = C(\delta, \sigma_1, \sigma_2)^{-1} \tilde{f}(y; \mu, \delta, \sigma_1, \sigma_2)$ [Lau and Krumscheid, 2019]. Mithilfe von f kann man die Plateau probability density distributions $T_{j,k}$, $j \in \{1, \ldots, M\}$ für die trial proposals der k-ten Komponente definieren als

$$T_{j,k}(x, y) = \begin{cases} f(y; x, \delta_1, \sigma, \sigma) & , j = 1 \\ \frac{1}{2} f(y; x - \delta_1 - \delta, \delta, \sigma, \sigma) + \frac{1}{2} f(y; x + \delta_1 + \delta, \delta, \sigma, \sigma) & , j = 2, \ldots M - 1 \\ \frac{1}{2} f(y; x - \delta_1 - \delta, \delta, \sigma_0, \sigma) + \frac{1}{2} f(y; x + \delta_1 + \delta, \delta, \sigma, \sigma_1) & , j = M \end{cases}$$

mit Werten $\delta_1, \delta, \sigma, \sigma_0, \sigma_1 > 0$ [Lau and Krumscheid, 2019]. In Figure 1 sieht man die trial proposal propability density distributions für die Parameter $M = 5, \delta_1 = \delta = 1, \sigma = 0.05, \sigma_0 = \sigma_1 = 0.5$. Man erkennt, dass sich die Verteilungen nur an ihren exponential decaying tails überschneiden. Die äußeren tails fallen durch die größeren σ_0, σ_1 Werte flacher ab als die restlichen tails. In dem Paper [Lau and Krumscheid, 2019] verwenden die Autoren durchgehend die Werte $\delta = \delta_1 = 2, \sigma = 0.05, \sigma_0 = \sigma_1 = 3$.

2.2 Component-wise Multiple-Try Metropolis

Der *Component-wise Multiple-Try Metropolis* Algorithmus, welcher die Grundlage für den Paper [Lau and Krumscheid, 2019] bildet, startet mit seinem MCMC sampling Verfahren von einer Startposition $x_0 \in \mathbb{R}^d$. Für jede MCMC Realisationen x_n mit $n \in \{1, \ldots, N\}$ wird für jede der d Komponenten das folgende Verfahren angewendet. Sei $x = (x_1, \ldots, x_d)$ der letzte gesampelte Kandidat des MCMC Algorithmus, dann schlägt der Algorithmus trials z_j für $i = 1, \ldots, M$ in dem er diese aus den Verteilungen $T_{j,k}(x_k, \cdot)$

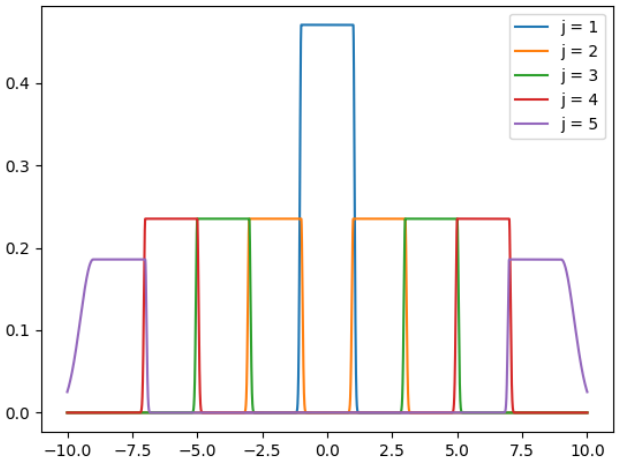

Figure 1: Trial proposal propability density distributions for $M = 5$

sampelt. In meiner Reeimplementierung des Papers verwende ich dazu ein rejection sampling Verfahren [Peng, 2018]. Dabei verwende ich eine gleichförmige Verteilung zum Erzeugen der samples über den folgenden Intervallen

- $j = 1: \quad I_1 = [x_k - \delta_1 - t_1, x_k + \delta_1 + t_1]$ mit $t_1 = \sqrt{-2\sigma^2 \log(0.0001C(\delta_1, \sigma, \sigma))}$,
- $j = 2, \ldots, M-1: \quad I_2 = [x_k - 2(j+1)\delta - \delta_1 - t_2, x - 2j\delta - \delta_1 + t_2] \cup [x_k + 2j\delta + \delta_1 - t_2, x + 2(j+1)\delta - \delta_1 + t_2]$ mit $t_2 = \sqrt{-2\sigma^2 \log(0.0002C(\delta, \sigma, \sigma))}$, und
- $j = M: \quad I_3 = [x_k - 2(M+1)\delta - \delta_1 - t_{32}, x_k - 2M\delta - \delta_1 + t_{31}] \cup [x_k + 2M\delta + \delta_1 - t_{31}, x_k + 2(M+1)\delta + \delta_1 + t_{32}]$ mit $t_{31} = \sqrt{-2\sigma^2 \log(0.0002C(\delta, \sigma, \sigma_0))}, t_{32} = \sqrt{-2\sigma_0^2 \log(0.0002C(\delta, \sigma, \sigma_0))}$.

Diese Intervalle ermöglichen es in dem Bereich aus $T_{j,k}(x_k, \cdot)$ effektiv zu samplen wo die probability density function größer als 0.0001 ist. Dazu sampelt man solange ein $u \sim U(0,1)$ und ein $y \sim U(I_i)$ bis

$$u < \frac{T_{j,k}(x_k, y)}{c|I_i|}$$

wobei $|I_i|$ die Breite des verwendeten Intervalls I_i ist, g_j die probability density function der gleichmäßogen Verteilung über I_j ist und c situationsabhängig folgende Werte hat

- $j = 1: \quad c = \sup_{y \in I_1} \frac{T_{1,k}(x_k, y)}{g_1(y)} = \frac{|I_1|}{C(\delta_1, \sigma, \sigma)}$,
- $j = 2: \quad c = \sup_{y \in I_j} \frac{T_{j,k}(x_k, y)}{g_j(y)} = \frac{|I_j|}{2C(\delta, \sigma, \sigma)}$, und

- $j = M:$ $\quad c = \sup_{y \in I_M} \frac{T_{M,k}(x_k, y)}{g_M(y)} = \frac{|I_M|}{2C(\delta, \sigma, \sigma_0)}$.

2.3 Adaption of Proposal Distributions

TO DO ...

3 Experiments and Results

3.1 Performance Measures

Die Autoren in dem Paper [Lau and Krumscheid, 2019] verwenden zwei performance measures, um die Wirksamkeit des implementierten Algorithmus zu untersuchen. Dies ist zum einen die integrated autocorrelation times (ACT), welche in R MCMC Simulationen mit jeweils N Schritten für K Komponenten berechnet wird. Sei im Folgenden $X_t^{(r)} = (X_{t,1}^{(r)}, \ldots, X_{t,K}^{(r)})$ das Ergebnis der r-ten MCMC Simulation bei Schritt t, wobei $r \in \{1, \ldots, R\}$ und $t \in \{1, \ldots, N\}$. Die Implementierung benutzt einen *initial positive sequence estimator* wie ihn [Geyer, 1992] verwendet. Dafür wird zunächst die komponentenweise die empirische Autokovarianz bestimmt um die lagged autocovariance $\gamma_{i,k}^{(r)}$ mit $k \in \{1, \ldots, K\}$ zu schätzen. Dabei ist

$$\hat{\gamma}_{t,k}^{(r)} = \frac{1}{N} \sum_{i=1}^{N-t} (X_{i,k}^{(r)} - \bar{X_k}^{(r)})(X_{i+t,k}^{(r)} - \bar{X_k}^{(r)})$$

wobei $\bar{X_k}^{(r)} = 1/N \sum_{i=1}^{N} X_{i,k}^{(r)}$ das arithmetische Mittel k-ten Komponente in der r-ten Simulation bezeichnet. Danach schauen wir uns die Summen

$$\hat{\Gamma}_{m,k}^{(r)} = \hat{\gamma}_{2m,k}^{(r)} + \hat{\gamma}_{2m+1,k}^{(r)}$$

von benachbarten Autokovarianzen Paaren an. Schließlich ergibt sich die integrated autocorrelation times als

$$\mathrm{ACT}_k^{(r)} = -\hat{\gamma}_{0,k}^{(r)} + 2 \sum_{i=0}^{m} \hat{\Gamma}_{m,k}^{(r)}$$

wobei m die größte natürliche Zahl ist, sodass $\hat{\Gamma}_{i,k}^{(r)} > 0$ für alle $i \in \{1, \ldots, m\}$ gilt [Geyer, 1992].

3.2 Experiments

TO DO ...

3.3 Technical Details of Implementation

TO DO ...

3.4 Results

TO DO ...

4 Discussion

TO DO ...

References

[Geyer, 1992] Geyer, C. J. (1992). Practical markov chain monte carlo. *Statist. Sci.*, 7(4):473–483.

[Lau and Krumscheid, 2019] Lau, F. D.-H. and Krumscheid, S. (2019). Plateau proposal distributions for adaptive component-wise multiple-try metropolis.

[Liu et al., 2000] Liu, J., Liang, F., and Wong, W. (2000). The multiple-try method and local optimization in metropolis sampling. *Journal of The American Statistical Association - J AMER STATIST ASSN*, 95:121–134.

[Metropolis et al., 1953] Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

[Peng, 2018] Peng, R. D. (2018). Advanced statistical computing. `https://bookdown.org/rdpeng/advstatcomp/rejection-sampling.html`. Accessed: 2020-03-17.