Adaptive MCMC

Johannes Nägele

7. Mai 2022

Inhalt

Problemstellung I

Grundsätzlich liefern MCMC-Algorithmen wie MH unter relativ schwachen Bedingungen die (gesuchte) Gleichgewichtsverteilung. Problem ist dabei die Konvergenzgeschwindigkeit. Intuitiv (Goldilocks-Prinzip):

- Varianz des proposals zu hoch: Sehr wenige Werte werden gesampled, da die meisten verworfen werden.
- Varianz des proposals zu niedrig: Algorithmus stagniert und passt sich nur sehr langsam der gewünschten Verteilung an, da Zustandsraum langsam durchschritten wird.

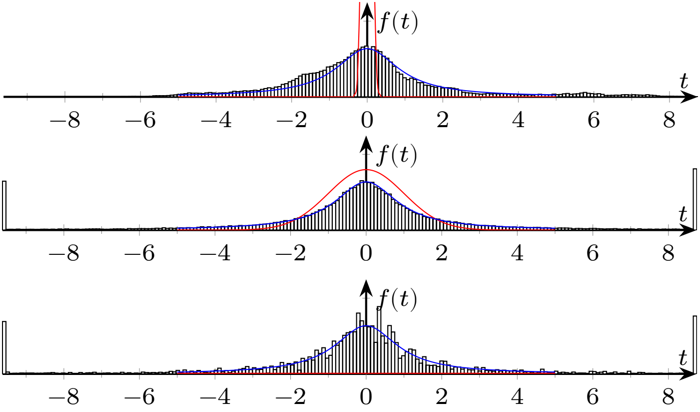
Problemstellung II

- Magische 0.234 unter technischen Annahmen als optimale acceptance rate für normalverteiltes MH, großes d und u.i.v. Komponenten mit existierender Varianz der target distribution [1]
- Für target distribution ohne u.i.v. ist die Varianz $\frac{2.38^2}{d}\Sigma_\pi$ optimal
- In der Praxis: Σ_π wird durch mehrere Probeläufe geschätzt

[1] Andrew Gelman, Walter R. Gilks und Gareth O. Roberts. „Weak convergence and optimal scaling of random walk Metropolis algorithms“. In: *The annals of applied probability* 7.1 (1997), S. 110–120.

Simulation

Abbildung 1: MCMC-Simulation für normalverteilte proposals mit $n = 100000$ und burn-in von 10000. Rot: Proposal bei $\sigma \in (0.1, 1, 100)$, blau: $\mathrm{Cauchy}(0, 1)$



Idee

- Wähle (endogen) gute Parameter für proposal distribution
- Aber: Prozess möglicherweise keine Markovkette mehr und Existenz sowie Form der stationären Verteilung nicht mehr klar
- Möglicherweise blödsinnige Werte (Varianz)
- Ähnlich zu Markov-Decision Processes

Adaptives Metropolis-Hastings

Definiere den AM-Algorithmus mit normalverteiltem Kern nach [2].

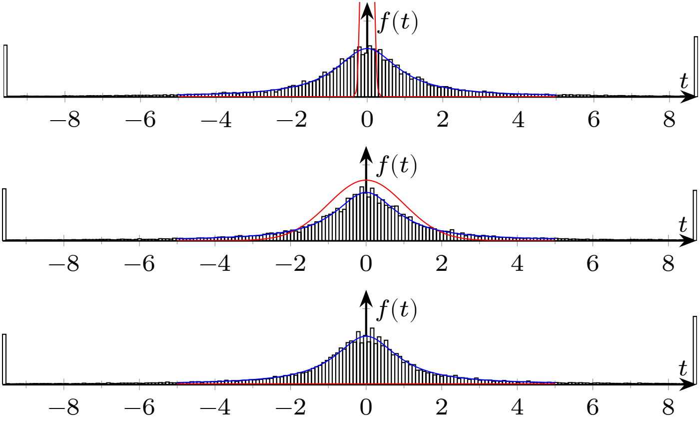
Wähle für die Zielverteilung π bei $t = 0$ die Startwerte X_0, μ_0 und Σ_0 mit $\pi(X_0) > 0$ sowie $\beta_{t+1} = 1/(t+1)$. Bestimme X_n mit

while $t < n$ **do**
  Ziehe X^* aus $Q = \mathcal{N}(X_t, \lambda\Sigma_t)$ und α aus $\mathcal{U}([0, 1])$.
  if $\pi(X^*)/\pi(X_t) > \alpha$ **then**
    $X_{t+1} \leftarrow X^*$
  else
    $X_{t+1} \leftarrow X_t$
  end
  $\mu_{t+1} \leftarrow \mu_t + \beta_{t+1}(X_{t+1} - \mu_t)$
  $\Sigma_{t+1} \leftarrow \Sigma_t + \beta_{t+1}[(X_{t+1} - \mu_t)(X_{t+1} - \mu_t)^T - \Sigma_t]$
end

[2] Geof H. Givens und Jennifer A. Hoeting. Computational Statistics. New York: John Wiley & Sons, 2012, S. 247–248.

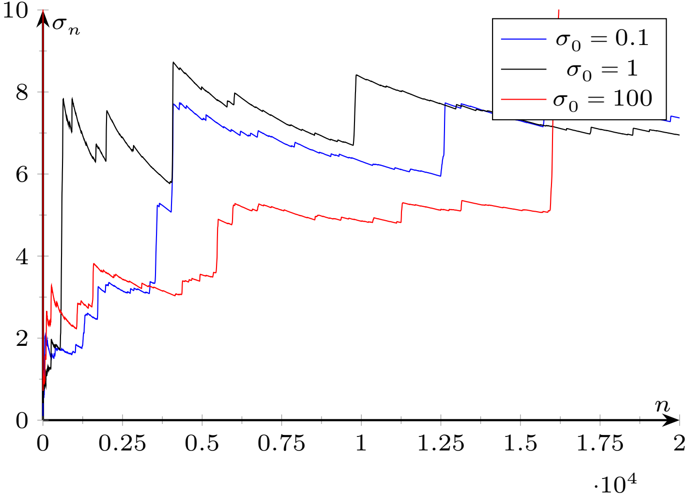
Simulation mit AM I

Abbildung 2: Adaptive MCMC-Simulation für normalverteilte proposals mit $n = 100000$ und burn-in von 10000. Rot: Proposal bei $\sigma_0 \in (0.1, 1, 100)$, blau: $\mathrm{Cauchy}(0, 1)$



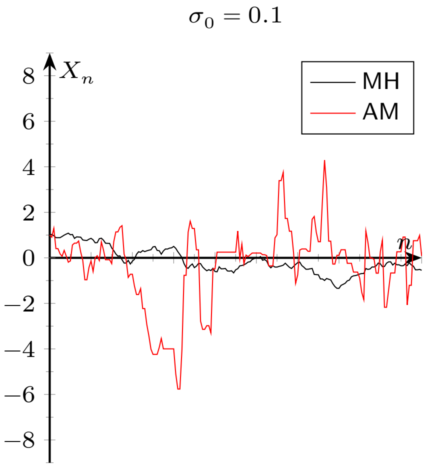
Simulation mit AM II

Abbildung 3: Entwicklung von σ_n



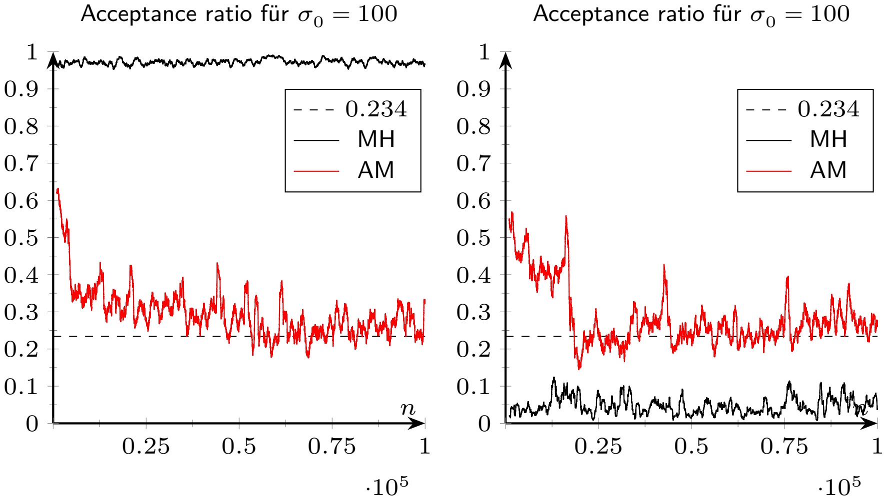
Simulation mit AM III

Abbildung 4: Gesamplete Werte X_n für $n < 200$



Simulation mit AM IV

Abbildung 5: Moving average der acceptance ratios



Ein pathologisches Beispiel I

Beispiel 1

Sei X eine uniform verteilte Zufallsvariable auf $\{1, 3, 4\}$ mit Maß π und sei $Q_i(Y, \cdot)$ eine uniform verteilte Zufallsvariable auf $\{Y - i, Y + i\}$, $i \in \{1, 2\}$. Definiere $R_i(Y, \cdot) = 1/2 \cdot Q_i(Y, \cdot) + 1/2 \cdot \pi(\cdot)$ als transition kernel. Bestimme X_n mit

while $t < n$ **do**
  Ziehe X^* aus $R_i(X_t, \cdot)$ und α aus $\mathcal{U}([0, 1])$.
  if $[\pi(X^*)/\pi(X_t)]/[R_i(X_t, X^*)/R_i(X^*, X_t)] > \alpha$ **then**
    $X_{t+1} \leftarrow X^*$
    $i \leftarrow 1$
  else
    $X_{t+1} \leftarrow X_t$
    $i \leftarrow 2$
  end
end

Ein pathologisches Beispiel II

Obiges Beispiel aus [3].
Zu zeigen: X_n hat invariantes Maß $\mu \neq \pi$.

Beweis.

Siehe Tafel. □

[3] Yves F. Atchadé und Jeffrey S. Rosenthal. „On adaptive Markov chain Monte Carlo algorithms". In: *Bernoulli* 11.5 (2005), S. 815–828, S. 816–817.

Ein pathologisches Beispiel III

```julia
using LinearAlgebra # for eigenvalues
using Pipe:@pipe # for cleaner code

function Q(n, j, i) # variable part of the transition kernel
    return (n-j) in [-i, i] ? 1/2 : 0 # uniform distribution
end

function P(m, n, j, β) # assumes all values have probability > 0
    i = (m == n) ? 1 : 2
    ψ = (m == n == 1) | (m != n == 4) ? (1-β) : (1-β)/2
    return (j == n) ? β*1/3 + ψ : (1-β)*Q(n, j, i) + β*1/3
end
```

Ein pathologisches Beispiel III

```
β = 1/2 # mixing parameter
state = [1,3,4] # state space
cartesian = Iterators.product(state, state) |>
    collect |> permutedims
transition = zeros(9,9)
# fill the transition matrix
for n in 1:9
    for j in 1:9
        value_n = cartesian[n]
        value_j = cartesian[j]
        transition[n,j] = (value_n[2] == value_j[1]) ?
        P(value_n[1], value_n[2], value_j[2], β) : 0
    end
end
```

Ein pathologisches Beispiel III

```
julia> round.(transition, digits=2)
9×9 Array{Float64,2}:
0.67  0.17  0.17  0.0   0.0   0.0   0.0   0.0   0.0
0.0   0.0   0.0   0.42  0.42  0.17  0.0   0.0   0.0
0.0   0.0   0.0   0.0   0.0   0.0   0.17  0.17  0.67
0.42  0.42  0.17  0.0   0.0   0.0   0.0   0.0   0.0
0.0   0.0   0.0   0.17  0.42  0.42  0.0   0.0   0.0
0.0   0.0   0.0   0.0   0.0   0.0   0.17  0.17  0.67
0.42  0.42  0.17  0.0   0.0   0.0   0.0   0.0   0.0
0.0   0.0   0.0   0.42  0.42  0.17  0.0   0.0   0.0
0.0   0.0   0.0   0.0   0.0   0.0   0.17  0.42  0.42
```

Ein pathologisches Beispiel III

```julia
# find eigenvectors
equilibrium = transpose(transition) |> eigvecs |> transpose
# find eigenvector with eigenvalue 1
eig = @pipe transpose(transition) |> eigvals |>
    isapprox.(_, 1) |> findfirst(x -> x == true, _)
μ = @pipe equilibrium[eig,:] |>
    convert(Vector{Real}, _) |> transpose
# normalize
μ = μ/sum(μ)
```

```
julia> sum(μ[1:3]) # I use Julia btw
0.35166240409207217
```

Andere adaptive MCMC-Algorithmen

- Adaptiver Metropolis-within-Gibbs-Algorithmus
- Bisher internal MCMC, außerdem (vgl. [4])
  - External MCMC
  - Interacting MCMC
- In der Praxis viele Möglichkeiten (z. B. batches)

[4] Atchadé und Rosenthal, „On adaptive Markov chain Monte Carlo algorithms“, S. 1–4.

Voraussetzungen I

Die nachfolgende Darstellung orientiert sich an [5].
Sei X_n eine Zufallsvariable auf $\mathcal{A}$, die den Zustand des Algorithmus repräsentiert, und θ_n eine Zufallsvariable auf dem Parameterraum Θ, welche die Wahl der proposal distribution beschreibt, $n \in \mathbb{N}_0$.

Definition

Für A_n, B_n Zufallsvariablen auf $\mathcal{A}$ ist $D_n(A_n, B_n) = \sup_{X \in \mathcal{A}} |P_{A_n} - P_{B_n}|$ die *total variational distance* zwischen A_n und B_n zum Zeitpunkt $n \in \mathbb{N}_0$.

Definition

Wir nennen X_n *ergodisch* bzgl. π, wenn $\lim_{n \to \infty} D_n(X_n, \pi) = 0$.

---

[5] Krys Latuszynski. Adaptive Markov Chain Monte Carlo: Theory, Methodology and Practice [Vortrag EBEB-Meeting]. 2018, S. 53–55.

Voraussetzungen II

Definition

Es gilt *simultaneous uniform ergodicity*, wenn $\forall\varepsilon > 0\,\exists n(\varepsilon) \in \mathbb{N}\colon ||P_\theta^n(X,\cdot) - \pi(\cdot)|| \leq \varepsilon \forall X \in \mathcal{A}, \theta \in \Theta$.

Definition

Sei $A_n = \sup_{X\in\mathcal{A}} ||P_{\theta_n}(X,\cdot) - P_{\theta_{n+1}}(X,\cdot)||$, dann hat der Prozess *diminishing adaptation*, wenn $\lim_{n\to\infty} A_n = 0$.

Definition

Sei $M_\varepsilon(x,\theta) = \inf\{n \geq 1\colon ||P_\theta^n(x,\cdot) - \pi(\cdot)|| \leq \varepsilon\}$, dann gilt *containment*, wenn $\{M_\varepsilon(X_n,\gamma_n)\}_{n=0}^\infty$ für alle $\varepsilon > 0$ in Wahrscheinlichkeit beschränkt ist.

Resultate I

In [6] werden folgende zentrale Resultate gezeigt.

Theorem 1

Gelten die Bedingungen containment und diminishing adaptation, dann folgt Ergodizität der Markovkette.

Beweis.

Siehe [6], Theorem 2. □

Theorem 2

Gelten die Bedingungen simultaneous uniform ergodicity und diminishing adaptation, dann folgt Ergodizität der Markovkette.

[6] Gareth O. Roberts und Jeffrey S. Rosenthal. „Coupling and Ergodicity of Adaptive Markov Chain Monte Carlo Algorithms“. In: *Journal of Applied Probability* 44.2 (2007), S. 458–475.

Resultate II

Beweis.

Siehe [6], Theorem 1. □

Theorem 3

Gelte simultaneous uniform ergodicity sowie diminishing adaptation und sei $g\colon \Omega \to \mathbb{R}$ eine beschränkte messbare Funktion, dann folgt unter beliebigen Startwerten

$$\frac{1}{n}\sum_{i=1}^{n} g(X_i) \to \pi(g).$$

Beweis.

Siehe [6], Theorem 5. □

Anwendung auf AM

Bemerkung

Der AM-Algorithmus erfüllt die Voraussetzungen von Theorem und .

Beweis.

Simultaneous uniform ergodicity klar, diminishing adaptation nicht. □

Für diminishing adaptation siehe [7].

[7] Heikki Haario, Eero Saksman und Johanna Tamminen. „An adaptive Metropolis algorithm". In: *Bernoulli* 7.2 (2001), S. 223–242, S. 230–231.

Zusammenfassung

- AMCMC-Algorithmen bieten einfachen Weg, um Konvergenz zu beschleunigen
- Per se können viele MCMC-Algorithmen auf diese Weise beschleunigt werden, aber Vorsicht ist trotzdem geboten
- Theoretisch vor allem Ergodensätze, LLNs und Konvergenzgeschwindigkeit (scaling)

References

- Atchadé, Yves F. und Jeffrey S. Rosenthal. „On adaptive Markov chain Monte Carlo algorithms". In: *Bernoulli* 11.5 (2005), S. 815–828.
- Gelman, Andrew, Walter R. Gilks und Gareth O. Roberts. „Weak convergence and optimal scaling of random walk Metropolis algorithms". In: *The annals of applied probability* 7.1 (1997), S. 110–120.
- Givens, Geof H. und Jennifer A. Hoeting. Computational Statistics. New York: John Wiley & Sons, 2012.
- Haario, Heikki, Eero Saksman und Johanna Tamminen. „An adaptive Metropolis algorithm". In: *Bernoulli* 7.2 (2001), S. 223–242.
- Latuszynski, Krys. Adaptive Markov Chain Monte Carlo: Theory, Methodology and Practice [Vortrag EBEB-Meeting]. 2018. URL: https://warwick.ac.uk/fac/sci/statistics/staff/academic-research/latuszynski/adap_course_rio.pdf.

References

Roberts, Gareth O. und Jeffrey S. Rosenthal. „Coupling and Ergodicity of Adaptive Markov Chain Monte Carlo Algorithms“. In: *Journal of Applied Probability* 44.2 (2007), S. 458–475.