# Markov-Chain-Monte-Carlo-Verfahren und der Metropolis-Hastings Algorithmus

Elise Wolf

18. November 2023

# Inhaltsverzeichnis

# Kapitel 1

# Einführung in MCMC

Um den Einstieg zu erleichtern, wird zu Beginn ein kurzer Überblick über die MCMC umfassenden Themenbereiche gegeben. Anhand von Beispielen wird das Problem der Anwendung von Monte-Carlo-Simulation in Spezialfällen erläutert, ehe eine Einführung in MCMC eine Begründung für die Sinnhaftigkeit der Methode liefern soll.

## 1.1 Zentrale Idee hinter MCMC

Das Markov-Chain-Monte-Carlo-Verfahren (MCMC) ist eine Methode, um Stichproben aus hoch dimensionierten Wahrscheinlichkeitsverteilungen zu erzeugen.

Das Verfahren setzt sich zusammen aus der Monte-Carlo-Simulation und Markovketten.

Die Monte-Carlo-Simulation nutzt unabhängige Zufallszahlen zur Simulation von Prozessen und dem Finden von approximativen Lösungen. Markovketten finden in MCMC wiefolgt Anwendung.: Es wird rekursiv eine Folge von Zufallsvariablen $(X_t)_{t\in\mathbb{N}}$ erzeugt, die für große t ungefähr die Zielverteilung $\pi$ annimmt. Die Wahrscheinlichkeit für $X_{t+1}$ hängt dabei spezifisch nur von der Wahrscheinlichkeit von $X_t$ ab und nicht von denen von $(X_0, X_1, ..., X_{t-1})$. Diese Eigenschaft wird in der Definition der Markovkette zusammengefasst. Die Markovkette hat die invariante Verteilung $\pi$. Invariant heißt hier, dass, sobald die Markovkette im Zustand $X_m$ die Verteilung $\pi$ annimmt, sie fortan auch für alle $m+1, m+2, ...$ die Verteilung $\pi$ beibehält.

Werden beide Methoden, Markovketten und Monte Carlo kombiniert, können Stichproben von Zufallszahlen für hochdimensionierte Wahrscheinlichkeitsverteilungen erzeugt werden, in denen auch die stochastische Abhängigkeit zwischen den Stichproben berücksichtigt wird.

MCMC kann vielfältig zur Schätzung komplexer Wahrscheinlichkeitsverteilungen, zum Beispiel in der Bayes'scher Statistik, Machine Learning, Bayesian Networks, Finanzwesen und vielen anderen Bereichen eingesetzt werden. Als illustratives Beispiel wird die Anwendung von MCMC in der Bayesianischen (Bayes'schen) Statistik erläutert. Sie beruht auf dem folgenden Theorem:

**Theorem 1.1** (Bayes'sches Theorem in der Bayesschen Statistik)**.** *Für die a-posteriori Verteilung der Parameter $\theta$ nach den beobachteten Daten $X$ gilt:*

$$P(\theta|X) = \frac{P(X|\theta) \cdot P(\theta)}{P(X)}$$

*wobei:*

- $P(\theta|X)$ *ist die a-posteriori Verteilung der Parameter $\theta$ nach den beobachteten Daten $X$.*

- *$P(X|\theta)$ ist die Likelihood-Funktion, die die Wahrscheinlichkeit der Beobachtungen $X$ gegeben den Parametern $\theta$ beschreibt.*
- *$P(\theta)$ ist die a-priori Verteilung der Parameter, basierend auf vorherigem Wissen oder Annahmen.*
- *$P(X)$ ist die Evidenz, die die Wahrscheinlichkeit der Beobachtungen $X$ beschreibt. Sie ist eine Normalisierungskonstante.*

Für die Berechnung der a-posteriori Verteilung $P(\theta|X)$ werden $P(X|\theta)$, $P(\theta)$ und $P(X)$ benötigt. Oftmals sind $P(X|\theta)$ und $P(\theta)$ explizit bekannt. Dies ist für die Normalisierungskonstante $P(X)$ nicht immer der Fall, stattdessen kann sie als das Integral über das Produkt der Likelihood-Funktion und der a-priori-Verteilung integriert werden.

$$P(X) = \int_{\theta} P(X|\theta) \cdot P(\theta)\, d\theta$$

Dieses ist jedoch oft keine bekannte Verteilung und die hohe Dimensionalität vom Parameter $\theta$ erschwert die Integration.

MCMC bietet die Möglichkeit, über Markovketten eine Folge von Zuständen $\{\theta^{(1)}, \ldots, \theta^{(n)}\}$ zu erzeugen, wobei die Wahrscheinlichkeit des Eintritts der Zustände langfristig gegen die a-posteriori-Verteilung konvergiert.

$$\theta^{(i)} \sim P(\theta|X) \quad \text{für } i = 1, \ldots, n$$

Der *Metropolis-Hastings-Algorithmus* ist das bekannteste MCMC-Verfahren und entwickelte sich aus dem von Nicholas Metropolis, Marshall Rosenbluth Edward Teller, Augusta H. Teller und Arianna W. Rosenbluth 1953 publizierten *Metropolis-Algorithmus*.

## 1.2 Probleme in der Anwendung der Monte-Carlo-Simulation

Im Folgenden werfen wir einen Blick auf das Beispiel *Importance Sampling* in der Monte-Carlo-Simulation, um zu verstehen, wie MCMC die Effektivität in der Simulation und Berechnung von Wahrscheinlichkeitsverteilungen steigert.

Die Monte-Carlo Simulation beschreibt Methoden, dessen Ziel die Bestimmung der Wahrscheinlichkeitsverteilung über das wiederholte Ziehen von Stichproben durch Zufallsexperimente ist. Sie basiert auf dem Gesetz der großen Zahlen, was besagt, dass der Durchschnitt über viele unabhängige Zufallsvariablen gegen ihren Erwartungswert konvergiert.

Die Stichprobe, also eine Folge von unabhängig identisch verteilten Zufallsvariablen $X_1, X_2, \ldots, X_n$, ist gemäß einer Wahrscheinlichkeitsverteilung $P$ verteilt. Der Monte-Carlo-Schätzer $T_n$ is definiert als ein erwartungstreuer Schätzer für den Erwartungswert von $f(X)$ auf dem hochdimensionierten Raum $\Omega$:

$$T_n(f) = \frac{1}{n} \sum_{i=1}^{n} f(X_i) \xrightarrow[n\to\infty]{} \int_{\Omega} f(x)p(x)dx = \mathbb{E}_p[f(X)]$$

In einigen Anwendungen kann die Monte-Carlo-Simulation bereits gut eingesetzt werden. Ein einfaches Beispiel ist die Approximation von $\pi$ durch die Berechung der Fläche eines Einheitskreises auf dem ensprechenden Quadrat definiert durch die Bedingung $x^2 + y^2 < 1$. Dabei werden uniformverteilte Paare von Zufallszahlen $(x, y)$ erzeugt, die Punkte auf dem Koordinatensystem darstellen. Durch anteiliges Zählen der Punkte in der grauen Fläche kann eine Approximation an diese Fläche erzeugt werden.

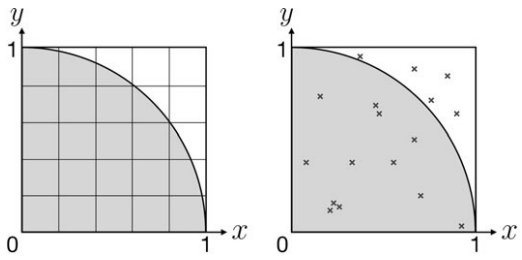



Abbildung 1.1: Approximation von $\pi$ durch randomisierte Paare uniformverteilter Zufallszahlen. Quelle: [11] Hanada, MCMC from Scratch

Nun zu einem Beispiel, bei welchem die Monte-Carlo-Simulation ineffektiv ist. Wir betrachten das Integral über die Gauß-Funktion $\sqrt{\frac{1}{2\pi}}e^{-\frac{x^2}{2}}$ auf einem Intervall $[-a, a]$. Zur approximativen Berechnung werden uniformverteilte Paare von Zufallsvariablen auf dem Intervall $[-a, a]$ erzeugt und wie bereits im Beispiel zuvor gesehen, ein Wert für den Anteil der sich unter dem Integral befindenden Punkte erzeugt. Wird dieser Wert mit $2a$ multipliziert, erhält man den Flächeninhalt des Integrals. In Abbildung 1.2 ist der berechnete Wert des Integrals für verschiedene Längen $a$ in Abhängigkeit der Anzahl der Paare der Zufallszahlen $K$ dargestellt. Es ist bekannt, dass es sich bei der Gauß-Funktion um eine Dichtefunktion handelt und dementsprechend das Integral über die Gauß-Funktion für $a \to \infty$ gegen 1 konvergieren sollte. Aufgrund der Verteilung der Masse der Standardnormalverteilung rund um 0 und der Lokalisation von wenig Masse an den Rändern, ändert sich die Fläche des Integral über die Gauß-Funktion bei Veränderung des Intervalls $[-a, a]$ für große $a$ kaum.

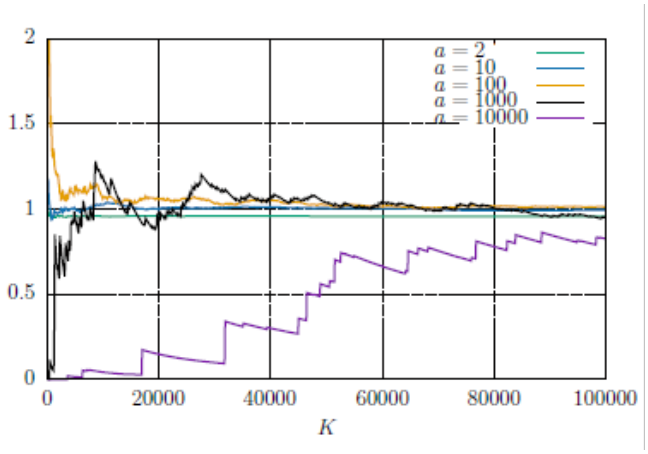



Abbildung 1.2: Durchschnittliche Werte des Integrals über die Anzahl der Zufallszahlen $K$. Quelle: [11] Hanada, MCMC from Scratch

Im Wesentlichen sind bei der Berechnung zwei Punkte zu beachten.:

1. **Langsame Konvergenz für große $a$:** In Abbildung 1.2 ist eine Konvergenz des zu approximierenden Integrals gegen 1 zu erkennen, jedoch dauert die Konvergenz gegen 1 für große $a$ deutlich länger an. Grund dafür ist, dass effektiv nur Punkte im Intervall $[-2, 2]$ die Größe des Integrals wesentlich beeinflussen. Man beachte $\int_{-2}^{2} \sqrt{\frac{1}{2\pi}}e^{-\frac{x^2}{2}}\,dx \approx 0.95$. Umgekehrt ist der Anteil der uniformverteilten Zufallszahlen auf dem Intervall $[-10000, 10000]$, die im Intervall $|x| \leq 2$ liegen, 99.98%. Das erklärt auch die vielen Sprünge, da nur einmal alle paar tausend generierten Zufallszahlen ein genügend kleines $x$ auftritt, welches die größe des approximierten Integrals beeinflusst. Das bedeutet aber auch eine unnötige Berechnung von etwa 99.98% der Zufallszahlen. In diesem einfachen Fall kommt es im Rechenaufwand zu keinen Problemen.

2. **Multivariate Verteilungen:** Haben wir jedoch stattdessen eine multivariate Verteilung mit mehreren Variablen, sieht der Rechenaufwand schon anders aus. Dieser kann verringert werden, indem die Zufallszahlen nur in einem Intervall erzeugt werden, welches signifikant das zu approximierende Integral beeinflusst. Eine Methode um herauszufinden, welche Intervalle zum Integral beiträgt, ist es, zufällige Intervalle zu erzeugen und deren Beitrag zum Integral stückweise zu berechnen. Wenn das Hinzufügen eines Intervalls keinen wesentlichen Unterschied in der Größe des zu approximierenden Integrals auslöst, kann diese Region bei der Berechnung künftig ignoriert werden. Im Beispiel der Gauß-Funktion wäre $x_1^2+\ldots+x_n^2 < 3$ ein guter Ansatz. Man wähle also Intervalle $R_1 \leq x_1^2+\ldots+x_n^2 < R_2$ mit $[R_1, R_2) = [0, 1), [R_1, R_2) = [1, 2)$ und $[R_1, R_2) = [2, 3)$. Schwierig wird es dann, wenn der Integrand eine kompliziertere Funktion ist und die Intervalle, die die Größe des Integrals beeinflussen, nicht auf einen Blick erkennbar sind.

## 1.3 Effizienzsteigerung durch MCMC

MCMC läuft im Gegensatz zur Monte-Carlo-Simulation nicht zufällig über den Parameterraum. Stattdessen werden Vorschläge für nachfolgende Punkte $x_{t+1}$ in der Nähe von $x_t$ durch Markovketten generiert. So können Regionen mit viel Einfluss auf die Zielfunktion höher gewichtet werden. Wie der Zusammenhang zu vorherigen Punkten entsteht, ist leicht in einer vereinfachten Darstellung der wesentlichen Schritte des bekannten Metropolis-Algorithmus erkennbar:

**Algorithm 1** Metropolis-Algorithmus

1: Wähle einen Startwert $x_0$.
2: **for** $t = 0, 1, 2, ...$ **do**
3: Wähle eine uniformverteilte Zufallszahl $\delta_t$.
4: Schlage einen Kandidaten $x_{t*} = x_t + \delta_t$ vor.
5: Führe den Metropolis-Test durch:
6: Berechne die Akzeptanzwahrscheinlichkeit $p(x_t, x_{t*}) = \min\left(1, \frac{f(x_{t*})}{f(x_t)}\right) \in [0, 1]$.
7: Generiere eine uniformverteilte Zufallszahl $u$ in $[0, 1]$.
8: **if** $u \leq p(x_t, x_{t*})$ **then**
9: Akzeptiere den Kandidaten: $x_{t+1} \leftarrow x_{t*}$.
10: **else**
11: Lehne den Kandidaten ab: $x_{t+1} \leftarrow x_t$.
12: **end if**
13: **end for**

## 1.4 zentrale Aspekte von Markovketten in MCMC

Folgende Definitionen werden zur Vorbereitung der Definition der Markovkette benötigt.:

**Definition 1.2.** *Ein **stochastischer Prozess** auf $(\Omega, \mathcal{F}, \mathbb{P})$ mit Werten im Zustandsraum und der Sigmaalgebra $(E, \epsilon)$ ist eine Familie $(x_t)_{t\in I}$ von $E$-wertigen Zufallsvariablen. Für festes $\omega \in \Omega$ heißt die Abbildung $t \mapsto x_t(\omega)$ eine **Trajektorie/Pfad/Realisierung** von $(x_t)_{t\in I}$. Falls $I = \mathbb{N}_0$ oder $I = [0, \infty)$, so heißt die Verteilung von $x_0$ die **Startverteilung** des stochastischen Prozesses.*

**Definition 1.3.** *Ein stochastischer Prozess $(x_n)_{n\in\mathbb{N}_0}$ auf $(\Omega, \mathcal{F}, \mathbb{P})$ mit Werten in $E$ besitzt die **(elementare) Markoveigenschaft**, wenn für jedes $n \in \mathbb{N}_0$ und alle $x_0, \ldots, x_{n+1} \in E$ mit $\mathbb{P}(X_0 = x_0, \ldots, X_n = x_n) > 0$ gilt:*

$$\mathbb{P}(X_{n+1} = x_{n+1}|X_0 = x_0, \ldots, X_n = x_n) = \mathbb{P}(X_{n+1} = x_{n+1}|X_n = x_n)$$

*Falls zudem für alle $n \in \mathbb{N}_0$ und $x, y \in E$ gilt, dass $\mathbb{P}(X_{n+1} = y|X_n = x) = \mathbb{P}(X_1 = y|X_0 = x)$, so besitzt $(x_n)_{n\in\mathbb{N}_0}$ die **zeithomogene Markoveigenschaft**.*

**Definition 1.4.** *Eine Matrix $P = (p(x,y))_{x,y\in E}$ heißt **stochastisch** oder **Übergangsmatrix**, falls:*

*(i)* $p(x,y) \geq 0 \ \forall \ x,y \in E$,

*(ii)* $\sum_{y\in E} p(x,y) = 1 \ \forall \ x \in E$.

MCMC unterscheidet sich durch die Berücksichtigung von wesentlichen Aspekten in der Simulation von der Monte-Carlo-Simulation. Diese stammen aus dem Gebiet der Markovketten und werden in den folgenden vier Definitionen erläutert.:

- Definition der Markovkette
- Definition der Irreduzibilität der Markovkette
- Definition der Aperiodizität
- Definition der Invarianz oder Detailed balance condition

### 1.4.1 Definition der Markovkette

**Definition 1.5.** *Sei $P = (p(x,y))_{x,y\in E}$ eine stochastische Matrix und $v\colon E \to [0,1]$ ein Wahrscheinlichkeitsvektor. Ein stochastischer Prozess $(x_n)_{n\in\mathbb{N}_0}$ auf $(\Omega, \mathcal{F}, \mathbb{P})$ mit Zustandsraum $E$ heißt **(zeitinhomogene) Markovkette** mit Übergangsmatrix $P$ und Startverteilung $v$ (kurz: $(v,P)$-Markovkette), falls:*

*(i)* $\mathbb{P}[X_{n+1} = x_{n+1} | X_0 = x_0, \ldots, X_n = x_n] = p(x_n, x_{n+1}) \ \forall \ n \in \mathbb{N}_0$ *und* $x_0, \ldots, x_{n+1} \in E$ *mit* $\mathbb{P}[X_0 = x_0, \ldots, X_n = x_n] > 0$.

*(ii)* $\mathbb{P}[X_0 = x_0] = v(x_0) \ \forall \ x_0 \in E$.

**Beispiel 1.6.** *Sei $\mu$ eine Wahrscheinlichkeitsverteilung auf $\mathbb{Z}^d$. Setze $p(x,y) = \mu(y-x)$ für alle $x,y \in \mathbb{Z}^d$. Offensichtlich ist $P = (p(x,y))_{x,y\in\mathbb{Z}^d}$ eine stochastische Matrix. Dann nennt man die $(\mathbb{1}_x, \mathbb{P})$-Markovkette $(X_n)_{n\in\mathbb{N}_0}$ eine Irrfahrt (Random Walk) auf $\mathbb{Z}^d$ mit Start in $x \in \mathbb{Z}^d$. Im Spezialfall, dass*

$$\mu(x) = \begin{cases} \frac{1}{2d}, & \text{für } |x| = 1, \\ 0, & \text{sonst.} \end{cases}$$

*nennt man $(X_n)_{n\in\mathbb{N}_0}$ eine einfache symmetrische Irrfahrt.*

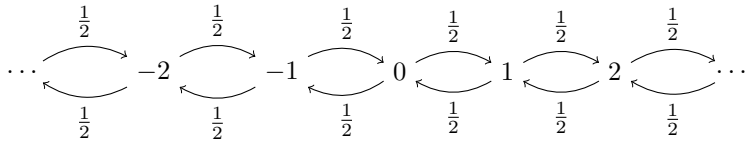



Abbildung 1.3: Graph einer einfachen symmetrischen Irrfahrt auf $\mathbb{Z}$

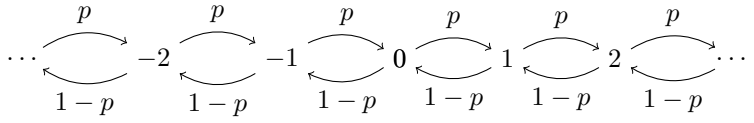



Abbildung 1.4: Graph einer einfachen asymmetrischen Irrfahrt auf $\mathbb{Z}$

Ein wichtiger Satz, der die Markovketten mit der Markoveigenschaft in Verbindung setzt, ist der Folgende.

**Satz 1.7.** *Sei $(X_n)_{n\in\mathbb{N}_0}$ eine Folge von $E$-wertigen Zufallsvariablen auf $(\Omega,\mathcal{F},\mathbb{P})$, $\nu$ eine Verteilung auf $E$ und $P=(p(x,y))_{x,y\in E}$ eine stochastische Matrix. $(X_n)_{n\in\mathbb{N}_0}$ ist genau dann eine $(\nu,P)$-Markovkette, wenn für alle $n\in\mathbb{N}_0$ und $x_0,\dots,x_n\in E$ gilt:*

$$\mathbb{P}[X_0=x_0,\dots,X_n=x_n]=\nu(x_0)\cdot p(x_0,x_1)\cdot\ldots\cdot p(x_{n-1},x_n).$$

*Beweis.* Zeige, dass $(X_n)_{n\in\mathbb{N}_0}$ eine Markovkette ist, d.h., zeige die Eigenschaften einer Markovkette.: Sei $n\in\mathbb{N}_0$, $x_0,\dots,x_n\in E$ mit $\mathbb{P}[X_0=x_0,\dots,X_n=x_n]>0$. Dann gilt

$$\begin{aligned}\mathbb{P}[X_{n+1}=x_{n+1}|X_0=x_0,\dots,X_n=x_n]&=\frac{\mathbb{P}[X_0=x_0,\dots,X_{n+1}=x_{n+1}]}{\mathbb{P}[X_0=x_0,\dots,X_n=x_n]}\\&=\frac{v(x_0)\cdot p(x_0,x_1)\cdot\ldots\cdot p(x_n,x_{n+1})}{v(x_0)\cdot p(x_0,x_1)\cdot\ldots\cdot p(x_{n-1},x_n)}=v(x_0)\cdot p(x_0,x_1)\cdot\ldots\cdot p(x_n,x_{n+1}).\end{aligned}$$

und $\mathbb{P}[X_0=x_0]=v(x_0)$ nach Definition. $\Longrightarrow$ $(X_n)_{n\in\mathbb{N}_0}$ ist eine Markovkette.

Hinrichtung durch Induktion.

**IA:**

$$\begin{aligned}n=0:\mathbb{P}[X_0=x_0]&=v(x_0).\\n=1:\mathbb{P}[X_1=x_1,X_0=x_0]&=v(x_0)\cdot p(x_0,x_1).\end{aligned}$$

**IV:** Die Behauptung gelte für beliebiges, aber festes $n\in\mathbb{N}_0$.

**IS:** Sei $\mathbb{P}[X_0=x_0,\dots,X_n=x_n]>0$. Wir zeigen die Behauptung für $n+1$. Betrachte

$$\begin{aligned}&\mathbb{P}[X_0=x_0,\dots,X_{n+1}=x_{n+1}]\\&=\mathbb{P}[X_0=x_0,\dots,X_n=x_n]\cdot\mathbb{P}[X_{n+1}=x_{n+1}|X_0=x_0,,X_n=x_n]\\&\overset{\text{IV}}{=}v(x_0)\cdot p(x_0,x_1)\cdot\ldots\cdot p(x_{n-1},x_n)\cdot\mathbb{P}[X_{n+1}=x_{n+1}|X_0=x_0,,X_n=x_n]\\&\overset{\text{Eig. MK}}{=}v(x_0)\cdot p(x_0,x_1)\cdot\ldots\cdot p(x_n,x_{n+1})\end{aligned}$$

wobei die letzte Gleichung die Induktionsannahme und die gegebene Bedingung nutzt. Daher ist die Behauptung für $n+1$ bewiesen. □

**Definition 1.8.** *Sei $X=(X_n)_{n\in\mathbb{N}}$ ein stochastischer Prozess auf $(\Omega,\mathcal{F},\mathbb{P})$ mit Werten in $(E,\epsilon)$. Setze*

$$\mathcal{F}_n^x:=\sigma\left(x_k:k\in\{0,\dots,n\}\right)=\left\{(X_0,\dots,X_n)^{-1}(B):B\in\epsilon^{\times(n+1)}\right\},$$

*für $n\in\mathbb{N}_0$. Dann heißt $(\mathcal{F}_n^x)_{n\in\mathbb{N}_0}$ die **von $X$ erzeugte $\sigma$-Algebra**.*

**Definition 1.9** (Stoppzeit, Treffzeit, Eintrittszeit, $T$-Vergangenheit)**.** *Sei $X=(X_n)_{n\in\mathbb{N}_0}$ ein stochastischer Prozess auf $(\Omega,\mathcal{F},\mathbb{P})$ mit Werten in $E$. Eine Abbildung $T:\Omega\to\mathbb{N}_0\cup\{\infty\}$ heißt **Stoppzeit** bezüglich $X$, wenn gilt $\{T=n\}\in\sigma(x_0,\dots,x_n)=\mathcal{F}_n^x\forall n\in\mathbb{N}_0$.*

*Die erste Rückkehr- bzw. Treffzeit $S_A$ und die **Eintrittszeit** $T_A$ in $A\subseteq E$ sind gegeben durch:*

$$S_A(\omega):=\inf\{n\in\mathbb{N}:X_n(\omega)\in A\}$$

$$T_A(\omega):=\inf\{n\in\mathbb{N}_0:X_n(\omega)\in A\}$$

*Ist $T:\Omega\to\mathbb{N}_0\cup\{\infty\}$ eine Stoppzeit bezüglich eines stochastischen Prozesses $X=(X_n)_{n\in\mathbb{N}_0}$ auf $(\Omega,\mathcal{F},\mathbb{P})$, so bezeichnet*

$$\mathcal{F}_T^x:=\{A\in\mathcal{F}:A\cap\{T=n\}\in\mathcal{F}_n^x\forall n\in\mathbb{N}_0\}$$

*die Menge der **$T$-Vergangenheit**.*

**Bemerkung 1.10.** *1. Ist $\{T=\infty\}$, so tritt die Stoppzeit nie ein.*

*2. $S_A$ und $T_A$ sind Stoppzeiten, denn z.B. $\{S_A=n\}=\{X_1\notin A,\dots,X_{n-1}\notin A,X_n\in A\}\in\mathcal{F}_n^x$.*

3. *Die $T$-Vergangenheit ist eine $\sigma$-Algebra. Die Eigenschaften können nachgewiesen werden.*

**Satz 1.11** (Starke Markoveigenschaft)**.** *Ist $(X_n)_{n\in\mathbb{N}_0}$ eine $(\nu, P)$-Markovkette mit Zustandsraum $E$ und $T$ eine Stoppzeit, so gilt für alle $A \in \epsilon^{\times\mathbb{N}_0}$, $F \in \mathcal{F}_T^x$ und $x \in E$ mit $\mathbb{P}_\nu[F, X_T = x, T < \infty] > 0$:*

$$P_\nu\left[(X_T, X_{T+1}, \ldots) \in A \mid F, X_T = x, T < \infty\right] = P_x\left[(X_0, X_1, \ldots) \in A\right]$$

*wobei $X_T$ auf $\{T < \infty\}$ definiert ist durch $X_T(\omega) := X_{T(\omega)}(\omega)$.*

*Beweis.* Sei $A \in \epsilon^{\times\mathbb{N}_0}$, $F \in \mathcal{F}_T^x$ und $x \in E$ mit $P_\nu[F, X_T = x, T < \infty] > 0$. Dann gibt es zu jedem $n \in \mathbb{N}_0$ ein $B \subseteq E^n$ mit

$$F \cap \{T = n\} \cap \{X_T = x\} = \{(X_0, \ldots, X_{n-1}) \in B, X_n = x\}$$

Falls zusätzlich

$$P_\nu[(X_0, \ldots, X_{n-1}) \in B, X_n = x] > 0$$

ist, folgt aus

$$P_\nu[(X_n, X_{n+1}, \ldots) \in A \mid (X_0, \ldots, X_{n-1}) \in B, X_n = x] = P_x\left[(X_0, X_1, \ldots) \in A\right]$$

Das ist die Verallgemeinerung der Markoveigenschaft auf unendlich viele zukünftige Ereignisse.

$$\begin{aligned}
&P_\nu[(X_T, X_{T+1}, \ldots) \in A \mid F, X_T = x, T = n] \\
&= P_\nu[(X_n, X_{n+1}, \ldots) \in A \mid (X_0, \ldots, X_{n-1}) \in B, X_n = x] \\
&= P_x\left[(X_0, X_1, \ldots) \in A\right]
\end{aligned}$$

Also:

$$\begin{aligned}
&P_\nu[(X_T, X_{T+1}, \ldots) \in A \mid F, X_T = x, T < \infty] \\
&= \sum_{n=0}^{\infty} \frac{P_\nu[(X_T, X_{T+1}, \ldots) \in A \mid F, X_T = x, T = n] \cdot P_\nu[F, X_T = x, T = n]}{P_\nu[F, X_T = x, T < \infty]} \\
&= \sum_{n=0}^{\infty} \frac{P_x\left[(X_0, X_1, \ldots) \in A\right] \cdot P_\nu[F, X_T = x, T = n]}{P_\nu[F, X_T = x, T < \infty]} \\
&= P_x\left[(X_0, X_1, ) \in A\right]
\end{aligned}$$

Die letzte Umformung folgt aus: $\sum_k P[A \mid B_k] \cdot P[B_k] = P[A \mid \bigcup_k B]$ und Sigmaadditivität. □

### 1.4.2 Definition der Irreduzibilität

Zustände in Markovketten können verschieden klassifiziert werden, abhängig von ihren Übergangswahrscheinlichkeiten.

**Definition 1.12** (Absorbierender, rekurrenter, transienter Zustand)**.**

*(i) Ein Zustand $x$ ist **absorbierend**, falls $p(x, x) = 1$.*

*(ii) Ein Zustand $x$ ist **rekurrent**, falls $\mathbb{P}_x[S_x < \infty] = 1$.*

*(iii) Ein Zustand $x$ ist **transient**, falls $\mathbb{P}_x[S_x < \infty] < 1$.*

*Hierbei ist $S_x := \inf\{n \in \mathbb{N} : X_n = x\}$ die erste Treffzeit.*

**Definition 1.13** (Positiv rekurrenter, nullrekurrenter Zustand)**.** *Ein rekurrenter Zustand $x$ aus $E$ ist*

1. ***positiv rekurrent***, *falls* $\mathbb{E}_x[S_x] < \infty$.

2. ***nullrekurrent***, *falls* $\mathbb{E}_x[S_x] = \infty$.

Irreduzibilät ist ein Konzept, welches besagt, dass für Paare $(x, y)$ ein Verbindung zwischen beiden Knoten in einer endlichen Anzahl von Schritten möglich ist. Formal wird diese Eigenschaft durch kommunizierende Klassen ausgedrückt.

**Definition 1.14** (Kommunizierende Klasse)**.** *Eine nichtleere Teilmenge $K \subseteq E$ heißt* ***kommunizierende Klasse****, falls:*

*(i) $\forall x, y \in K$ gilt $x \leftrightarrow y$, wobei $x \to y$ und $y \to x$ heißt, dass $x$ und $y$ miteinander kommunizieren, also $\exists n \in \mathbb{N}_0$, sodass $p_n(x, y) > 0$ und umgekehrt.*

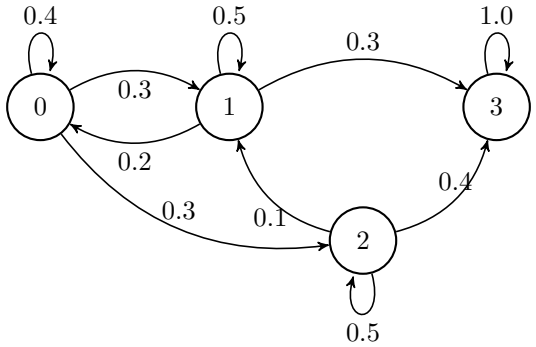
*(ii) $\forall x \in K$ und $y \in E$ mit $x \leftrightarrow y$ gilt $y \in K$ (Abgeschlossenheit).*

**Definition 1.15** (Irreduzibel)**.** *Eine stochastische Matrix $P$ auf $E$ heißt* ***irreduzibel****, falls $E$ nur aus einer kommunizierenden Klasse besteht. Eine $(\nu, P)$-Markovkette $(X_n)_{n\in\mathbb{N}_0}$ heißt irreduzibel, falls $P$ irreduzibel ist.*

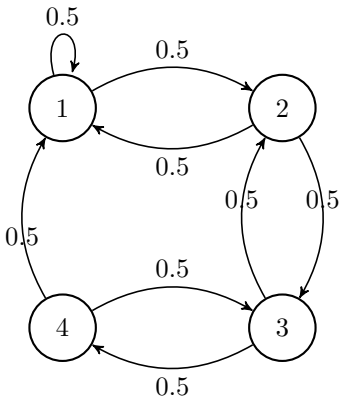
Folgende beiden Sätze helfen uns, um im Ergodensatz Rückschlüsse auf die Konvergenz der Markovkette zu ziehen. Aufgrund der Länge ihres Beweises, werden sie nicht beweisen - die Beweise finden sich im Skript der Markovketten-Vorlesung.

**Satz 1.16.** *Ist $(X_n)_{n\in\mathbb{N}_0}$ eine irreduzible $(\nu, P)$-Markovkette auf einem endlichen Zustandsraum $E$, so ist $x \in E$ positiv rekurrent.*

**Satz 1.17.** *Ist $(X_n)_{n\in\mathbb{N}_0}$ eine irreduzible $(v, P)$-Markovkette mit Zustandsraum $E$, dann gilt: Wenn $y \in E$ rekurrent ist, dann $\mathbb{P}_x[S_y < \infty] = 1$ $\forall x, y \in E$.*




Reduzible Markovkette
Nach Eintritt in Zustand 3 ist keine Rückkehr aus diesem mehr möglich.
Dementsprechend werden die anderen Zustände nicht mehr in endlicher Zeit erreicht.

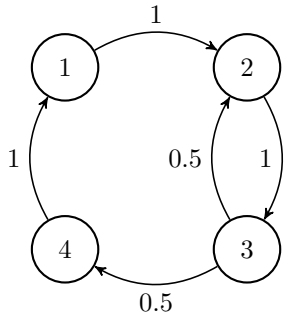



Irreduzible Markovkette
Jeder Zustand kann in endlicher Zahl von Schritten von einem anderen aus erreicht werden.

### 1.4.3 Definition der Aperiodizität

**Definition 1.18** (Periode)**.** *$\forall x \in E$ heißt $d(x) = ggT\{n \in \mathbb{N}_0 : p_n(x,x) > 0\}$ die **Periode** des Zustands $x$. Ist $d(x) = 1$, so heißt der Zustand $x$ **aperiodisch**.*

**Bemerkung 1.19.** *Über die Chapman-Kolmogorov-Gleichung kann gezeigt werden, dass $x$ genau dann aperiodisch ist, wenn $\exists N(x) \in \mathbb{N}$ gibt, sodass $p_n(x,x) > 0$ $\forall n \geq N(x)$.*




Periodische Markovkette
Von einem Zustand in einen anderen zu gelangen, benötigt $2n$ Schritte,
z.B. $n \geq 2$ für Zustand 1 und $n \geq 1$ für Zustand 3

### 1.4.4 Definition der Detailed Balance Bedinung

**Definition 1.20** (Invariantes Maß, Gleichgewichtsverteilung)**.** *Sei $P = (p(x,y))_{x,y \in E}$ eine stochastische Matrix. Ein Maß $\pi$ auf $E$ heißt **invariants Maß** bezüglich $P$, falls*

$$\pi(x) = (\pi P)(x) = \sum_{y \in E} \pi(y) p(y,x) \quad \forall x \in E.$$

*Falls $\pi$ invariant und eine Verteilung ist, d.h., $\pi[E] = 1$, so nennt man $\pi$ eine **Gleichgewichtsverteilung** oder invariante Verteilung.*

**Satz 1.21.** *Sei $(X_n)_{n \in \mathbb{N}_0}$ eine $(v,P)$-Markovkette mit Zustandsraum $E$. Falls $(X_n)$ irreduzibel ist, existiert höchstens eine Gleichgewichtsverteilung.*

**Definition 1.22** (Reversibel, Detailed Balance Bedingung)**.** *Ein Maß $\pi$ auf $E$ heißt **reversibel** bezüglich einer stochastischen Matrix $P = (p(x,y))_{x,y\in E}$, falls die sogenannte **Detailed Balance Bedingung** erfüllt ist:*

$$\pi(x) \cdot p(x,y) = \pi(y) \cdot p(y,x) \quad \forall x, y \in E.$$

*Eine stochastische Matrix $P$ nennt man reversibel, falls ein bezüglich $P$ reversibles Maß existiert.*

**Bemerkung 1.23.** *Wenn $\pi$ reversibel bezüglich $P$ ist, dann ist $\pi$ invariant bezüglich $P$.*

**Satz 1.24.** *Sei $(X_n)_{n\in\mathbb{N}_0}$ eine $(v,P)$-Markovkette mit Zustandsraum $E$ und $\emptyset \neq K \subseteq E$ eine kommunizierende Klasse. Dann gilt: $\exists!$ invariante Maß $\pi$ mit $\sum_{x\in K}\pi(x) = 1$ genau dann, wenn $K$ rekurrent ist. Insbesondere gilt $\pi(x) = \frac{1}{\mathbb{E}_x[S_x]}$ $\forall x \in K$.*

*Beweis.* **Hinrichtung:** Sei $\pi$ ein invariantes Maß bezüglich der stochastischen Matrix $P$ mit $\sum_{x\in K}\pi(x) = 1$. Dann existiert ein $x \in K$ mit $\pi(x) > 0$. Für alle $y \in K$ gilt $x \leftrightarrow y$. Also folgt $\pi(y) > 0$ $\forall y \in K$ aus

$$\exists n \in \mathbb{N}: \quad p_n(x,z) > 0 \implies \pi(x) = (\pi P^n)(x) = \sum_{y\in K}\pi(y)p_n(y,x) \geq \pi(z)p_n(x,z) > 0 \quad .$$

$\forall x \in E \setminus \{z\}$. Betrachte nun das invariante Maß $\lambda = \frac{\pi}{\pi(x)}$. Dann folgt

$$\lambda(y) = \mu_x(y) := \mathbb{E}_x\left[\sum_{n=0}^{S_x-1}\mathbb{1}_{\{X_n=y\}}\right]$$

aus Satz 3.4 im Skript über Markovketten. Insbesondere ist $\mathbb{E}_x[S_x] = \sum_{y\in K}\lambda(y) = \frac{1}{\pi(x)}\sum_{y\in K}\pi(y) = \frac{1}{\pi(x)} < \infty$. Folglich ist $x$ positiv rekurrent, und es gilt $\pi(x) = \frac{1}{\mathbb{E}_x[S_x]}$ für alle $x \in K$, d.h., $\pi$ ist eindeutig bestimmt.

**Rückrichtung:** Sei $K$ eine positiv rekurrente Klasse. Dann gilt für ein $x \in K$: $\sum_{y\in K}\lambda(y) = \mathbb{E}_x[S_x] < \infty$. Folglich ist $\pi(y) := \frac{\lambda(y)}{\mathbb{E}_x[S_x]}$ eine Gleichgewichtsverteilung mit $\pi(y) = 0$ für alle $y \in K^c$. Also gilt $\sum_{y\in K}\pi(y) = 1$. □

**Beispiel 1.25** (stationäre Verteilung, Aperiodizität und Irreduzibilität)**.** *Sei die Übergangsmatrix $P$ gegeben durch:*

$$P = \begin{bmatrix} 1-p & p/2 & p/2 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix},$$

*mit $p \in [0,1]$. Die stationäre Verteilung berechnet sich aus*

$$\pi \cdot P = \pi \Leftrightarrow \begin{bmatrix}\pi_1 & \pi_2 & \pi_3\end{bmatrix}\begin{bmatrix} 1-p & p/2 & p/2 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix}\pi_1 & \pi_2 & \pi_3\end{bmatrix},$$

$$\Leftrightarrow \begin{cases} (1-p)\pi_1 + \pi_2 + \pi_3 & = \pi_1, \\ p/2\pi_1 & = \pi_2, \\ p/2\pi_1 & = \pi_3, \\ \pi_1 + \pi_2 + \pi_3 & = 1, \end{cases}$$

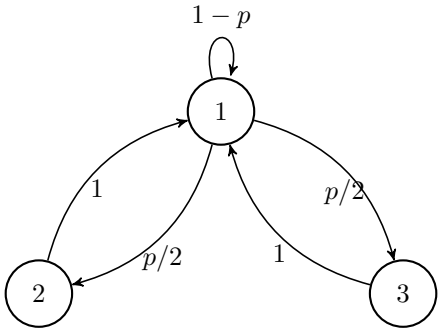
$$\Rightarrow \begin{cases} \pi_1 - p\pi_1 + \pi_2 + \pi_3 & = \pi_1, \\ \pi_1 + \pi_2 + \pi_3 & = \pi_1 + p\pi_1, \\ 1 & = (1+p)\pi_1, \end{cases}$$

$$\Rightarrow \begin{cases} \pi_1 & = \frac{1}{1+p}, \\ \pi_2 & = \frac{p}{2+2p}, \\ \pi_3 & = \frac{p}{2+2p}, \end{cases}$$

*daraus folgt die stationäre Verteilung*

$$\pi = \begin{bmatrix}\pi_1 & \pi_2 & \pi_3\end{bmatrix} = \begin{bmatrix}\frac{1}{1+p} & \frac{p}{2+2p} & \frac{p}{2+2p}\end{bmatrix}, \quad p \in [0,1].$$

*Um zu entscheiden, für welche p die Markovkette irreduzibel bzw. aperiodisch ist, betrachten wir die Abbildung*




*Für $p \in (0,1]$ ist die Markovkette irreduzibel. Für $p \in [0,1)$ ist die Markovkette aperiodisch.*

### 1.4.5 Ergodizität als zusammenfassende Eigenschaft

Irreduzible, aperiodische Markovketten heißen ergodisch. In MCMC wird für beliebige Startverteilungen $v(x_0)$ eine erdogische Markovkette erzeugt. Dabei wird die Übergangsmatrix $P$ so gewählt, dass die $(v,P)$-Markovkette gegen eine invariante (stationäre) Verteilung $\pi$, die Zielverteilung konvergiert.

Die Konvergenzaussage beschreibt die Konvergenz der Verteilung der Markovkette gegen die stationäre Verteilung im positiv rekurrenten Fall. Der Fundamentalsatz für ergodische Markovketten wird oft nur auf irreduziblen, aperiodischen Markovketten und endlichem Zustandsraum definiert. Dann lässt die Folgerung von positiver Rekurrenz aus der Irreduzibilität der Markovkette auf endlichem Zustandsraum nur den ersten Fall im Fundamentalsatz zu. Dann ist aber auch wieder die Ausgangssituation für den Fundamentalsatz mit der zusätzlichen Einschränkung auf endlichen Zustandsräumen erfüllt und somit der Satz auch in diesem Speziallfall gültig.

Der Ergodensatz trifft eine Aussage darüber, wie lange die Markovkette im Durchschnitt Zeit in einem Zustand entsprechend der stationären Verteilung $\mu_x$ verbringt, wann die Markovkette also tatsächlich gegen die stationäre Verteilung konvergiert.

**Satz 1.26** (Fundamentalsatz für ergodische Markovketten, Konvergenzaussage)**.** *Sei $(X_n)_{n\in\mathbb{N}_0}$ eine irreduzible, aperiodische, rekurrente Markovkette mit Zustandsraum $E$ und Übergangsmatrix $P = (p(x,y))_{x,y\in E}$. Dann gilt für alle $x, y \in E$:*

$$\lim_{n\to\infty} p_n(x,y) = \begin{cases} \frac{1}{\mathbb{E}_y[S_y]}, & \text{falls } \mathbb{E}_y[S_y] < \infty, \\ 0, & \text{falls } \mathbb{E}_y[S_y] = \infty. \end{cases}$$

**Bemerkung 1.27.** *Im positiv rekurrenten Fall konvergiert somit $p_n(x,y)$ gegen die (eindeutig bestimmte) Gleichgewichtsverteilung $\pi(y) = \frac{1}{\mathbb{E}_y[S_y]}$.*

Zur Vorbereitung des Beweises führen wir eine Schreibweise für Markovketten ein, die sich aus zwei Markovketten zusammensetzen.

**Definition 1.28.** *Eine bivariate Markovkette $((X_n, Y_n))_{n\in\mathbb{N}_0}$ mit Werten im Zustandsraum $E \times E$ heißt (markovsche) Kopplung der $(\mu,P)$-Markovkette $(X_n)_{n\in\mathbb{N}_0}$ und der $(\nu,P)$-Markovkette $(Y_n)_{n\in\mathbb{N}_0}$ auf $E$, falls $\forall n \in \mathbb{N}_0, (x,y), (x',y') \in E \times E$ gilt:*

$$\mathbb{P}[X_{n+1} = x' \mid (X_n, Y_n) = (x, y)] = p(x, x')$$
$$\mathbb{P}[Y_{n+1} = y' \mid (X_n, Y_n) = (x, y)] = p(y, y')$$

**Bemerkung 1.29.** *Man schreibt die Markovkette $((X_n, Y_n))_{n\in\mathbb{N}_0}$ mit Werten in $E \times E$, Startverteilungen $\mu$ und $\nu$, sowie Übergangsmatrix $\mathbf{P}'$ mit $p'((x, y), (x', y')) := p(x, x') \cdot p(y, y')$ als eine unabhängige Kopplung.*

*Beweis von Satz zur Konvergenzaussage.* Wir beweisen nur den positiv rekurrenten Fall. **Schritt 1:** Sei $((X_n, Y_n))_{n\in\mathbb{N}_0}$ die Markovkette der unabhängigen Kopplung. Da $P$ irreduzibel und aperiodisch nach Definition ist, $\forall x, x', y, y' \; \exists N^* := N^*(x, x', y, y') \in \mathbb{N}$, sodass $p'_n((x, x'), (y, y')) = p_n(x, x') \cdot p_n(y, y') > 0$ für alle $n \geq N0$. Folglich ist die Markovkette irreduzibel.

Für ein beliebiges $x_0 \in E$ definiere $S := S_{(x_0,x_0)} := \inf\{n \in \mathbb{N} : (X_n, Y_n) = (x_0, x_0)\}$.

Zu zeigen ist, dass $\forall n \in \mathbb{N}$:

$$|p_n(x, y) - p_n(z, y)| \leq \mathbb{P}_{(x,z)}[S > n]$$

Es gilt aber

$$\begin{aligned}
\mathbb{P}_{(x,z)}[X_n = y, S \leq n] &= \sum_{m=1}^{n} \mathbb{P}_{(x,z)}[X_n = y, S = m] \\
&= \sum_{m=1}^{n} \mathbb{P}_{(x,z)}[S = m] \cdot \mathbb{P}_{(x,z)}[X_n = y | (X_s, Y_s) = (x_0, x_0), S = m] \\
&= \sum_{m=1}^{n} \mathbb{P}_{(x,z)}[S = m] \cdot \mathbb{P}_{(x_0,x_0)}[X_{n-m} = y].
\end{aligned}$$

Weiterhin gilt

$$\begin{aligned}
\mathbb{P}_{(x_0,x_0)}[X_{n-m} = y] &= \sum_{y'\in E} p'_{n-m}((x_0, x_0), (y, y')) \\
&= \sum_{y'\in E} p_{n-m}(x_0, y) p_{n-m}(x_0, y') \\
&= \sum_{y'\in E} p_{n-m}(x_0, y') p_{n-m}(x_0, y) \\
&= \sum_{y'\in E} p'_{n-m}((x_0, x_0), (y', y)) \\
&= \mathbb{P}_{(x_0,x_0)}[Y_{n-m} = y].
\end{aligned}$$

Daraus folgt

$$\begin{aligned}
\mathbb{P}_{(x,z)}[X_n = y, S \leq n] &= \sum_{m=1}^{n} \mathbb{P}_{(x,z)}[S = m] \cdot \mathbb{P}_{(x_0,x_0)}[X_{n-m} = y] \\
&= \sum_{m=1}^{n} \mathbb{P}_{(x,z)}[S = m] \cdot \mathbb{P}_{(x_0,x_0)}[Y_{n-m} = y] \\
&= \mathbb{P}_{(x,z)}[Y_n = y, S \leq n].
\end{aligned}$$

Somit ergibt sich

$$\begin{aligned}
|p_n(x, y) - p_n(z, y)| &= |\mathbb{P}_{(x,z)}[X_n = y] - \mathbb{P}_{(x,z)}[Y_n = y]| \\
&= |\mathbb{P}_{(x,z)}[X_n = y, S > n] - \mathbb{P}_{(x,z)}[Y_n = y, S > n]| \\
&= |\mathbb{P}_{(x,z)}[X_n = y | S > n] - \mathbb{P}_{(x,z)}[Y_n = y | S > n]| \cdot \mathbb{P}_{(x,z)}[S > n] \\
&\leq \mathbb{P}_{(x,z)}[S > n].
\end{aligned}$$

**Schritt 2:** Betrachte $(X_n)_{n\in\mathbb{N}_0}$ positiv rekurrent. Dann existiert eine eindeutig bestimmte Gleichgewichtsverteilung $\pi$. Da aber

$$\sum_{(x,y)\in E\times E}(\pi\otimes\pi)(x,y)p'((x,y),(x',y')) = (\pi P)(x')(\pi P)(y') = \pi(x')\pi(y') = (\pi\otimes\pi)(x',y'),$$

folgt $\pi\otimes\pi$ ist eine Gleichgewichtsverteilung von $((X_n,Y_n))_{n\in\mathbb{N}_0}$. Insbesondere ist diese Markovkette positiv rekurrent und damit auch rekurrent. Daraus folgt $\mathbb{P}_{(x,z)}[S_{(x_0,x_0)} < \infty] = 1$.

$$\Rightarrow \limsup_{n\to\infty}|p_n(x,y) - p_n(z,y)| \le \mathbb{P}_{(x,z)}[S_{(x_0,x_0)} = \infty] = 0.$$

Also $\lim_{n\to\infty}|p_n(x,y) - p_n(z,y)| = 0$.

Weiterhin gilt

$$|p_n(x,y) - \pi(y)| = \left|\sum_{z\in E}\pi(z)(p_n(x,y) - p_n(z,y))\right| \le \sum_{z\in E}\pi(z)|p_n(x,y) - p_n(z,y)|.$$

Also folgt aus dem Satz von Lebesgue

$$\limsup_{n\to\infty}|p_n(x,y) - \pi(y)| \le 0.$$

Da $\pi(y) = 1/\mathbb{E}_y[S_y]$ nach Definition der eindeutigen Gleichgewichtsverteilung im rekurrenten Fall, folgt die Behauptung.

□

**Satz 1.30** (Ergodensatz)**.** *Sei $(X_n)_{n\in\mathbb{N}_0}$ eine irreduzible, rekurrente Markov-Kette auf dem Zustandsraum $E$ mit Übergangsmatrix $P$. Definiere $V_x(N) := \sum_{n=0}^{N-1}\mathbb{I}_{\{X_n=x\}}$, wobei $x\in E$. Für ein $x\in E$ bezeichne mit $\mu_x$ das invariante Maß $\mu_x(y) := \mathbb{E}_x\left[\sum_{n=0}^{S_x-1}\mathbb{I}_{\{X_n=y\}}\right]$, $y\in E$. Dann gilt für alle $f\in L^1(\mu_x)$:*

$$\lim_{N\to\infty}\frac{1}{V_x(N)}\sum_{n=0}^{N-1}f(X_n) = \mathbb{E}_{\mu_x}[f(X_0)] = \sum_{y\in E}f(y)\cdot\mu_x(y) \quad \mathbb{P}_x\text{-fast sicher.}$$

*Beweis.* Bezeichne mit $S_x^k$ die $k$-te Treffzeit des Zustandes $x$ in $E$, d.h., $S_x^0 = 0$ und $S_x^k = \inf\{n > S_x^{k-1} : X_n = x\}$. Da $x$ rekurrent ist und damit $\mathbb{P}_x[S_A < \infty] = 1\ \forall x\in A$ gilt $\forall n\in\mathbb{N}$

$$\begin{aligned}\mathbb{P}_x[S_A^n < \infty] &= \sum_{y\in A}\mathbb{P}_x[S_A^n < \infty | S_A^{n-1} < \infty, X_{S_A^{n-1}} = y]\cdot\mathbb{P}_x[S_A^{n-1} < \infty, X_{S_A^{n-1}} = y]\\ &= \sum_{y\in A}\mathbb{P}_y[S_A < \infty]\cdot\mathbb{P}_x[S_A^{n-1} < \infty, X_{S_A^{n-1}} = y]\\ &= \mathbb{P}_x[S_A^{n-1} < \infty].\end{aligned}$$

Induktiv ergibt sich $\mathbb{P}_x[S_x^k < \infty] = 1$ für alle $k\in\mathbb{N}_0$.

Für ein $f : E\to[0,\infty)$ und $x\in E$ definiere $I_x^f(k) := \sum_{n=S_x^{k-1}}^{S_x^k-1}f(X_n)$ für $k\in\mathbb{N}$. Dann ist $\mathbb{P}_x$-fast sicher $I_x^f(k) < \infty$ für alle $k\in\mathbb{N}$.

Zu zeigen ist, $(I_x^f(k))_{k\in\mathbb{N}}$ ist unabhängig identisch verteilt unter $\mathbb{P}_x$.

$$\begin{aligned}&\mathbb{P}_x(I_x^f(1)\le t_1,\dots,I_x^f(k)\le t_k]\\ &= \mathbb{P}_x[I_x^f(1)\le t_1,\dots,I_x^f(k-1)\le t_{k-1}]\cdot\mathbb{P}_x[I_x^f(k)\le t_k | I_x^f(1)\le t_1,\dots,I_x^f(k-1)\le t_{k-1}, X_{S_x^{k-1}} = x, S_x^{k-1} < \infty]\\ &= \mathbb{P}_x[I_x^f(1)\le t_1,\dots,I_x^f(k-1)\le t_{k-1}]\cdot\mathbb{P}_x[I_x^f(1)\le t_k]\\ &= \dots = \prod_{i=1}^{k}\mathbb{P}_x[I_x^f(1)\le t_i]\end{aligned}$$

Folglich sind $(I_x^f(k))$ für $k \in \mathbb{N}$ unabhängig und identisch verteilt unter $\mathbb{P}_x$.
**Schritt 1:** Sei $f \in L^1(\mu_x)$ mit $f \geq 0$. Dann gilt

$$\mathbb{E}_x[I_x^f(1)] = \mathbb{E}_x\left[\sum_{n=0}^{S_x-1} f(X_n)\right] = \sum_{y\in E} f(y)\mathbb{E}_x\left[\sum_{n=0}^{S_x-1} \mathbb{I}_{\{X_n=y\}}\right] = \sum_{y\in E} f(y)\mu_x(y) < \infty.$$

Also folgt aus dem starken Gesetz der großen Zahlen

$$\lim_{N\to\infty} \frac{1}{N}\sum_{k=1}^{N} I_x^f(k) = \mathbb{E}_x[I_x^f(1)] = \mathbb{E}_{\mu_x}[f(X_0)] \quad \mathbb{P}_x\text{-fast sicher.}$$

Weiterhin gilt $\mathbb{P}_x$-fast sicher $S_x^{V_x(N)-1} \leq N-1 < S_x^{V_x(N)}$ für alle $N \in \mathbb{N}$.
$\Longrightarrow \sum_{k=1}^{V_x(N)-1} I_x^f(k) = \sum_{n=0}^{S_x^{V_x(N)-1}-1} f(X_n) \leq \sum_{n=0}^{N-1} f(X_n) \leq \sum_{n=0}^{S_{x-1}^{V_x(N)}} f(X_n) = \sum_{k=1}^{V_x(N)} I_x^f(k)$.
Aus der Rekurrenz von $x$ folgt zudem, dass $\lim_{n\to\infty} V_x(N) = \sum_{n=0}^{\infty} \mathbb{1}_{\{X_n=x\}} = \infty$ $\mathbb{P}_x$-fast sicher. Also

$$\lim_{N\to\infty} \frac{1}{V_x(N)}\sum_{n=0}^{N-1} f(X_n) = \mathbb{E}_{\mu_x}[f(X_0)] \quad .$$

**Schritt 2:** Für ein beliebiges $f \in L^1(\mu_x)$ betrachte $f := f_+ - f_-$ und wende Schritt 1 auf $f_+, f_-$ an. □

# Kapitel 2

# Metropolis-Hastings Algorithmus

1953 publizierten **Nicholas Metropolis, Marshall Rosenbluth, Edward Teller, Augusta H. Teller und Arianna W. Rosenbluth** erstmals den *Metropolis-Algorithmus*. Mit diesem wird eine Markovkette erzeugt, die Zustände entsprechend einer Verteilung simuliert. Dabei besitzt die Markovkette bereits diese Verteilung als stationäre Verteilung.

1970 wurde der Algorithmus von **W. Keith Hastings** verallgemeinert, um Zustände für beliebige Wahrscheinlichkeitsverteilungen zu simulieren. Wir beginnen zunächst mit einer Einführung des Metropolis-Algorithmus und dessen Eigenschaften, ehe wir uns dem Metropolis-Hastings Algorithmus zuwenden.

## 2.1 Einführung über die Vorgängervarianten

MCMC läuft im Gegensatz zur Monte-Carlo-Simulation nicht zufällig über den Parameterraum. Stattdessen werden Vorschläge für nachfolgende Punkte $x_{k+1}$ in der Nähe von $x_k$ durch Markovketten generiert. So können Regionen mit viel Einfluss auf die Zielfunktion höher gewichtet werden. Der *Metropolis-Algorithmus* simuliert Zufallszahlen über eine Verteilung proportional zum Boltzmann-Faktor, vereinfacht als

$$P(x) \propto e^{-\frac{E_x}{T}}$$

dargestellt. Damit werden thermische Systeme zu modellieren. Es erscheint sinnvoll, eine Approximation an die tatsächliche Verteilung der analytischen Berechnung vorzuziehen, da Teilchen in Systemen, wie z.B. Gase, Magnetfelder oder Schwingungen in einem Molekülgitter nie unabhängig von den sie umgebenden Teilchen agieren. In der Physik ist der Ort/Zustand, an dem das Teilchen die niedrigste potenzielle Energie besitzt, als der gewünschte Zielzustand anzusehen. Der Algorithmus kombiniert Random Walk-Verhalten mit Acception-Rejection-Sampling. im Letzteren wird der vorgeschlagene Kandidat genau dann akzeptiert, wenn die Energie in diesem Zustand niedriger ist als im Vorherigen. Wie der Zusammenhang zwischen dem vorgeschlagenen und vorherigen Zustand hergestellt wird, ist leicht in einer Darstellung der wesentlichen Schritte des bekannten Metropolis-Algorithmus im Pseudocode erkennbar.:

**Algorithm 2** Metropolis-Algorithmus mit Anwendung im physikalischen Kontext

**Initialisierung:**
Wähle eine Anfangskonfiguration $x_0$ mit bekannter Energie $E_0$.
**for** $k = 0, 1, 2, ...$ **do**
    **Übergangsvorschlag:**
    Generiere eine zufällige Änderung $\delta_k$ zur aktuellen Konfiguration $x(k)$, um eine neue nahegelegene Konfiguration $x(k^*) = x(k) + \delta_k$ als Kandidaten zu erhalten.
    Die Verteilung von $\delta_x$ wird so gewählt wird, dass $-\delta_x$ mit gleicher Wahrscheinlichkeit auftritt. (uniform auf $[-c, c]$)
    **Berechne Energien:**
    Berechne die Energien $E_{x(k)}$ und $E_{x(k^*)}$.
    **Metropolis-Test und Aktualisierung:**
    **if** $E_{x(k^*)} < E_{x(k)}$ **then**
        Akzeptiere die neue Konfiguration $x(k^*)$, sie hat niedrigere Energie
    **else**
        Generiere eine uniformverteilte Zufallszahl $u$ in $[0, 1]$.
        **if** $u \leq p = e^{-\frac{E_{x(k^*)} - E_{x(k)}}{T}}$ die Akzeptanzwahrscheinlichkeit: **then**
            Akzeptiere neue Konfiguration $x(k^*) \leftarrow x(k)$, die dann höhere Energie hat
        **else**
            Behalte die aktuelle Konfiguration: $x(k + 1) \leftarrow x(k)$).
        **end if**
    **end if**
    **Wiederhole:**
    Wiederhole den Prozess.
**end for**

Man beachte, dass das Akzeptanzkriterium, welches manchmal als *Metropolis-Test* bezeichnet wird, auch wiefolgt notiert werden kann.:

$$\alpha = min(1, e^{-\frac{E_{x(k^*)} - E_{x(k)}}{T}})).$$

Wenn die Temperatur $T$ hoch ist, werden somit Schritte in die "falsche"Richtung, also Schritte in Richtung Zustände mit hoher Energie, öfter zugelassen, da T den Bruch kleiner und damit die Akzeptanzwahrscheinlichkeit größer macht. Wenn die Temperatur stattdessen abnimmt, sinkt auch die Wahrscheinlichkeit $p$. Das heißt, Regionen, für die bekannt ist, dass sie eine große Masse der Verteilung beinhalten, werden typischerweise öfter akzeptiertm wordurch langfristig eine Konvergenz zum Zustand mit niedrigster Energie sichergestellt wird ("*Exploitation*"). Dabei wird die Entscheidung der Akzeptanz randomisiert über die uniformverteilte Zufallsvariable $u$ getroffen. Durch die Akzeptanz weniger wahrscheinlicher Zustände kann die Erkundung ("*Exploration*") auf dem Raum der Verteilung sichergestellt werden.

Wie man leicht erkennen kann, wird $\delta_k$ unabhängig von den vorherigen Schritten $x_0, \ldots, x_{k-1}$ gewählt. Dementsprechend handelt es sich bei der Folge von erzeugten Zuständen um eine Markovkette.

Man betrachte die Gebiete, auf denen sich die im physikalischen Kontext stehende Markovkette bewegt, in der Realität. Diese sind verbunden, sodass $\forall x, y$ auf diesen Gebieten eine Verbindung zwischen beiden Zuständen in endlicher Anzahl von Schritten möglich ist. Daher ist die Markovkette irreduzibel.

$\forall n = 1, 2, \ldots$ und $x$ gibt es einen Pfad, der $x$ mit sich selbst verbindet. Für $n = 1$ ist das für ein $k$ mit $\delta_k = 0$ erfüllt. Dann wäre die Energie des vorgeschlagenen Zustands gleich dem des aktuellen Zustandes, wodurch die Akzeptanzwahrscheinlichkeit $p = 1$ gilt. Da $u \leq 1$, wird der Zustand immer akzeptiert. Daraus folgt, dass die Markovkette aperiodisch ist.

Die Detailed Balance Bedingung benötigt einige Überlegungen. Wir haben $\forall k \in \mathbb{N}$ $\delta_k \in [-c, c]$, $c \in \mathbb{R}$ gewählt. Dann ist die Übergangswahrscheinlichkeit null ist für alle $|x(k^*) - x(k)| = |\delta_k| \geq c$. Dementsprechend ist die Detailed Balance Bedingung gleich 0

und für diesen Fall erfüllt. Betrachte $|x(k^*) - x(k)| < c$. Aufgrund der Definition über uniforme Zufallsvariablen, tritt $\delta_k$ und $-\delta_k$ mit gleicher Wahrscheinlichkeit auf, nämlich genau $\frac{1}{2c}$. Die Übergangswahrscheinlichkeiten sind gegeben durch

$$p(x(k), x(k^*)) = \frac{1}{2c} min(1, e^{-\frac{E_{x(k^*)} - E_{x(k)}}{T}})$$
$$p(x(k^*), x(k)) = \frac{1}{2c} min(1, e^{-\frac{E_{x(k)} - E_{x(k^*)}}{T}})$$

Angenommen $E_{x(k^*)} > E_{x(k)}$. Dann $e^{-\frac{E_{x(k)} - E_{x(k^*)}}{T}} \geq 1 \ \forall T \geq 1$. Das heißt, $x(k)$ wird als Kandidat für $x(k^*)$ mit Wahrscheinlichkeit 1 angenommen, also $p(x(k^*), x(k)) = \frac{1}{2c}$. Daraus folgt für die Detailed Balance Bedingung

$$\pi(x(k^*)) \cdot p(x(k^*), x(k)) = e^{-\frac{E_{x(k^*)}}{T}} \cdot \frac{1}{2c} \quad \forall x, y \in E.$$

Umgekehrt gilt für $E_{x(k)} > E_{x(k^*)}$, dass $p(x(k), x(k^*)) = \frac{1}{2c} e^{-\frac{E_{x(k^*)} - E_{x(k)}}{T}} \ \forall T \geq 1$. Also;

$$\pi(x(k)) \cdot p(x(k), x(k^*)) = e^{-\frac{E_{x(k)}}{T}} \cdot \frac{1}{2c} e^{-\frac{E_{x(k^*)} - E_{x(k)}}{T}} = \frac{1}{2c} \cdot e^{-\frac{E_{x(k^*)}}{T}} \quad \forall x, y \in E.$$

Damit ist Detailed Balance Bedingung erfüllt.

## 2.2 Generalisiertes Verfahren: Metropolis-Hastings Algorithmus

Für den Metropolis-Hastings-Algorithmus ist ein Zustandsraum S und eine Übergangsmatrix P sowie die invariante Wahrscheinlichkeitsverteilung $\pi = (\pi_x)_{x \in S}$ gegeben. Das Ziel ist es, von der Zielverteilung $\pi$ Sampel zu erstellen, was aufgrund der Komplexität von $\pi$ nicht direkt geschehen kann. Weiterhin wähle man sich eine beliebige Übergangsmatrix $Q = (q_{x_i, x_j})_{x_i, x_j \in S}$ auf $S$. Die zugehörige Markovkette zu $Q$ ist als *Referenzkette* zu bezeichnen. Die Übergangsmatrix wird auch als *Vorschlagsverteilung* mit den *Vorschlagswahrscheinlichkeiten* $q(x_j|x_i)$ mit bezeichnet. Definiere der Vollständigkeit halber eine symmetrische Funktion $s(x_i, x_j)$, die später geeigent umdefiniert wird, um die verschiedenen Ausführungen des Algorithmus zu erhalten.

Wir definieren

**Algorithm 3** Metropolis-Hastings Algorithm

1: **Input:** Vorschlagswahrscheinlichkeit $q(x_j|x_i)$, symmetrische Funktion $s(x_i, x_j)$, Zielverteilung $\pi$ für $x_i, x_j \in S$, $k \in \mathbb{N}$, wobei $k$ eine Iteration über die Wiederholungen
2: **Output:** approximiertes Sample $X \sim \pi$
3: **Initializierung:** wähle beliebigen Startpunkt $x_0 \in S$
4: **repeat** $k = 0, 1, 2, \ldots$
5: **Übergangswahrscheinlichkeit:**
6: $x(k^*) \sim q(\cdot|x(k))$
7: **Akzeptanzwahrscheinlichkeit:**
8: $\rho(x(k), x(k^*)) = s(x(k), x(k^*)) \cdot \frac{\pi(x(k^*))q(x(k)|x(k^*))}{\pi(x(k))q(x(k^*)|x(k)) + \pi(x(k^*))q(x(k)|x(k^*))}$
9: **Test und Aktualisierung:**
10: Generiere eine uniformverteilte Zufallszahl $u$ in $[0, 1]$.
11: **if** $u \leq \rho(x(k), x(k^*))$ **then**
12: Akzeptiere den Kandidaten: $x(k+1) \leftarrow x(k^*)$.
13: **else**
14: Lehne den Kandidaten ab: $x(k+1) \leftarrow x(k)$.
15: **end if**
16: **until** Konvergenz

Es gibt verschiedene Ausführungen des Metropolis-Hastings-Algorithmus. Wir nähern uns der allgemeinen Konstruktion von Hastings von 1970, indem wir die symmetrische Funktion s(x,y) geeignet umdefinieren.

$$s(x(k), x(k^*)) = \frac{\pi(x(k))q(x(k^*)|x(k)) + \pi(x(k^*))q(x(k)|x(k^*))}{\max\left(\pi(x(k))q(x(k^*)|x(k)), \pi(x(k^*))q(x(k)|x(k^*))\right)}$$

Umformen gibt für die Akzeptanzwahrscheinlichkeit

$$\rho(x(k), x(k^*)) = \min\left(1, \frac{\pi(x(k^*))q(x(k)|x(k^*))}{\pi(x(k))q(x(k^*)|x(k))}\right)$$

$\forall \pi(x(k))q(x(k^*)|x(k))$. Diese Akzeptanzwahrscheinlichkeit ist nun die in von Hasting 1970 beschriebene.

Es gibt noch eine weitere Vereinfachung, die nur eine symmetrische Verteilung der Vorschlagswahrscheinlichkeit zulässt. Dies ist das originale Setting des Metropolis-Algorithmus von 1953. Dort wird also $q(x|y) = q(y|x)$ als symmetrisch vorausgesetzt. Damit vereinfacht sich $\rho(x(k), x(k^*)) = min(1, \frac{\pi(x(k^*))}{\pi(x(k))})$, wie es auch in der Boltzmann-Verteilung aus dem vorherigen Kapitel illustriert wurde.

## 2.3 Tuningparameter untersucht

In diesem Code habe ich den Metropolis-Hastings-Algorithmus implementiert, um eine Stichprobe aus einer Gaußverteilung zu ziehen. Der Algorithmus verwendet eine Vorschlagswahrscheinlichkeit , eine Zielverteilung $\pi$ und eine Akzeptanzregel, um die nächste Position in der Kette zu bestimmen. Die Dichte der Gaußverteilung ist durch die Formel

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$

ersetzt. In diesem Fall ist die Zielverteilung $\pi$ die Wahrscheinlichkeitsdichte der Gaußverteilung mit Mittelwert 0, da wir die Verteilung über den Mittelwert betrachten. Dann habe ich den Algorithmus für verschiedene Iterationen ausgeführt und den Mittelwert der generierten Stichproben über die Iterationen hinweg berechnet. Die Ergebnisse werden mit Plotly visualisiert, wobei die Mittelwerte für verschiedene Iterationszahlen auf einer einzigen Abbildung dargestellt sind. Dies ermöglicht es, den Konvergenzverlauf des Algorithmus zu beobachten.

Listing 2.1: Python-Code zum Metropolis-Hastings-Algorithmus zur Simulation

```
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
from plotly.subplots import make_subplots
import plotly.graph_objects as go

def gaussian(x, mean=0, variance=1):
    """Gau _Funktion: Simulation der Dichte der Standardnormalverteilung

    Args:
        x: Werte f r die Verteilung
        mean (int, optional): Mittelwert. Defaults to 0.
        variance (int, optional): Varianz. Defaults to 1.

    Returns:
       Werte f r die Gr  e in Abh ngigkeit von x f r die Standardnormalverteilu
    """
```

```
    return np.exp(-0.5 * ((x - mean) / np.sqrt(variance))**2) / np.sqrt(2 * np.pi *

def metropolis_hastings(q, pi, k):
    """Algorithmus des Metropolis-Hastings-Algorithmus nach Skript
    Args:
        q (_type_): Vorschlagswahrscheinlichkeit
        pi (_type_): invariante Gleichgewichtsveretilung
        k (_type_): Anzahl der Iterationen

    Returns:
        _type_: Samples  ber  die gegebene proposal distribution
    """
    # Initialisierung
    samples = []
    x = np.random.normal()  # Startpunkt aus einer normalverteilten Zufallszahl

    # Iterationen
    for _ in range(k):
        #       bergangswahrscheinlichkeit
        x_star = np.random.normal()

        # Akzeptanzwahrscheinlichkeit
        rho = min(1, pi(x_star) * q(x_star) / (pi(x) * q(x)))

        # Test und Aktualisierung
        u = np.random.uniform(0, 1)
        if u <= rho:
            x = x_star  # Akzeptiere den Kandidaten
        samples.append(x)

    return samples

# Vorschlagswahrscheinlichkeit (   bergangsmatrix    Q)
def q(x):
    return np.random.normal(size=x.shape)

variance = 1
iterations_values = [100, 1000, 5000, 10000]

# Plotly-Darstellung
fig = make_subplots(
    rows=1, cols=len(iterations_values), subplot_titles=[f'k={iterations}' for ite
)

for i, iterations in enumerate(iterations_values):
    samples = metropolis_hastings(np.random.normal, pi, iterations)
    x_range = np.linspace(-3, 3, 100)

    fig.add_trace(
        go.Histogram(x=samples, nbinsx=30, histnorm='probability density'),

        row=1, col=i + 1
    )

    fig.add_trace(
        go.Scatter(x=x_range, y=pi(x_range), mode='lines', line=dict(color='red', w
```

```
            row=1, col=i + 1
        )

fig.update_layout(height=400, width=1000, showlegend=False)
fig.show()

def calculate_running_mean(samples):
    """laufenden Mittelwert f r eine Liste von Samples

    Args:
        samples: k siehe Wertte von obn

    Returns:
        mittelwert
    """
    return np.cumsum(samples) / np.arange(1, len(samples) + 1)

# Plot f r jeden Mittelwert von k
plt.figure(figsize=(15, 5))
for i, iterations in enumerate(iterations_values):
    samples = metropolis_hastings(np.random.normal, pi, iterations)

    # Berechne den laufenden Mittelwert
    running_mean = calculate_running_mean(samples)

    # Plotte den laufenden Mittelwert
    plt.subplot(1, len(iterations_values), i + 1)
    plt.plot(np.arange(1, iterations + 1), running_mean, label=f'k = {iterations}')
    plt.xlabel('Anzahl der Iterationen')
    plt.ylabel('Laufender Mittelwert')
    plt.legend()

plt.show()
```

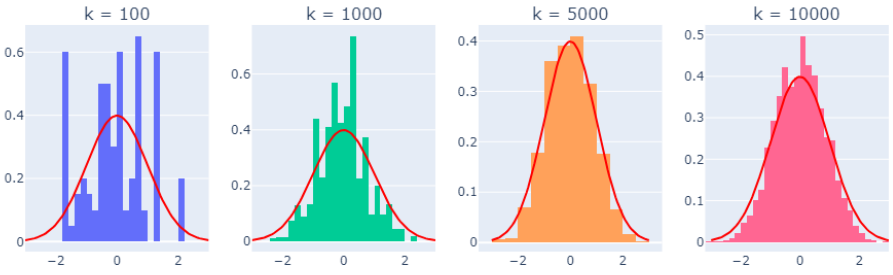



Abbildung 2.1: Metropolis-Hastings-Algorithmus für verschieden Sample-Sizes. Das Histogramm konvergiert mit wachsender Sample-Größe gegen die invariante Gleichgewichtsverteilung.

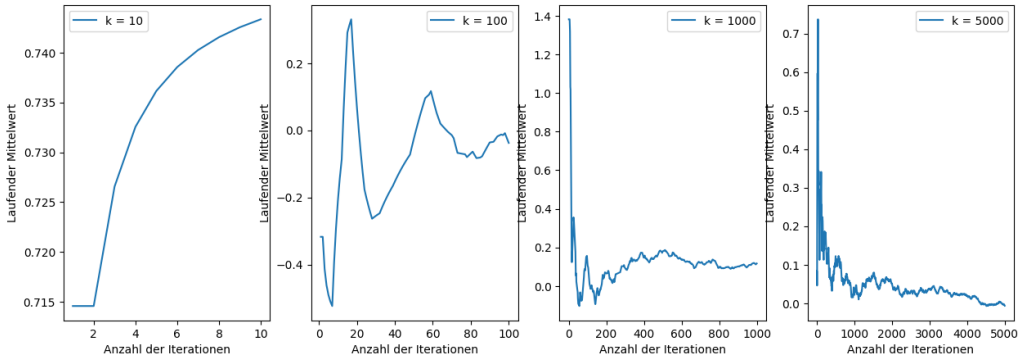



Abbildung 2.2: Mittelwert über Metropolis-Hastings-Algorithmus für verschieden Sample-Sizes. Das Histogramm konvergiert mit wachsender Sample-Größe gegen den Mittelwert der Zielverteilung, welche 0 ist.

Umm zu verhindern, dass die Simulation unendlich lange läuft, führen wir die Mischzeit als Stoppzeit ein. Dies ist die kleinste Anzahl an Schritten, die die Markovkette benötigt, um bis auf $\epsilon$ nah an der Zielverteilung $\pi$ zu sein.

**Definition 2.1.** *Sei $(X_n)_{n\in\mathbb{N}_0}$ eine Markov-Kette mit invarianter Verteilung $\{\pi_k\}_{k\in S}$. Sei $\varepsilon > 0$. Die $\varepsilon$-Mischzeit $\tau(\varepsilon)$ der Kette ist definiert als das kleinste $n > 0$, so dass für alle $k \in S$ und für beliebige Startverteilung gilt:*

$$|P(X_n = k) - \pi_k| \leq \varepsilon$$

*Das bedeutet: Für $n = \tau(\varepsilon)$ ist der Abstand von der invarianten Verteilung höchstens $\varepsilon$.*

Die Schrittweise ist ein wesentlicher Parameter, der die Konvergenzgeschwindigkeit beeinflusst. Wie im *Metropolis-Algorithmus* zu Beginn beschrieben, wurde $c > 0$ frei gewählt. Wenn das gewählte $c$ zu groß gewählt wird, wird aufgrund der geringen Differenz zwischen der Energie des neuen und der Energie des alten Zustandes die Akzeptanzwahrscheinlichkeit sehr klein. Umgekehrt ist die Akzeptanzwahrscheinlichkeit bei kleinem $c$ nahe 1, doch die Veränderung vom alten zum neuen Zustand ist so gering, dass es zu keiner wesentlichen Änderung in der Lokation von $x$ kommt.

Weiterhin ist zu beachten, dass $x(k+1)$ durch eine Veränderung von $x(k)$ erzeugt wird. Also sind $x(k+1)$ und $x(k)$ miteinander korrelliert. Man spricht auch ovn *Autokorrelation*. Je geringer die Schrittgröße oder die Akzeptanzrate, desto höher die Autokorrelation. Dadurch dauert nicht nur die Konvergenz gegen die Zielverteilung länger, auch die Samples werden schlechter.

Es gibt grundsätzlich zwei Wege, die Autokorrelation zu verringern. Einerseits durch Untersuchen der Korrelation mit dem Startwert und andererseits durch die Untersuchung der Burn-In-Phase. Die **Burn-In-Phase** bezeichnet den Zeitraum zu Beginn der Simulation, in welchem die vorgeschlagenen und auch akzeptierten Werte noch stark vom Startwert abhängen. Nach einer gewissen Zeit sinkt die Korrelation zum Initial Value und die Markovkette konvergiert gegen die Zielverteilung. Die *Jacknife-Methode* wird nicht weiter beschrieben, jedoch ist auf diese zur Schätzung der Größe der Autokorrelation verwiesen.

### 2.3.1 komponentenweise Metropolis-Hastings-Algorithmus

Um den Metropolis-Hastings-Algorithmus effizienter zu gestalten und vor allem auf multivariate Verteilungen anzupassen, wird die Zufallsvariable $X$ oft in einzelne Komponenten $X_1, \ldots, X_n$ unterschiedlicher Dimension aufgeteilt. Diese Methode ermöglicht komponentenweise Aktualisierungen, wobei eine Iteration des Komponentenweisen Metropolis-Hastings-Algorithmus aus $n$ Update-Schritten besteht.

Die folgende Gleichung beschreibt alle Komponenten von $X$ außer die $i$-te.

$$X_{.,-i} := \{X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n\}$$

$X(k,i)$ sei der Stand von $X(.,i)$ nach der $k$-ten Iteration. Für die Komponente $i$ der Iteration $k+1$ wird der Metropolis-Hastings-Algorithmus auf $X(k,i)$ angewandt.

Der Kandidat $X(k^*,i)$ wird mit der Vorschlags-Wahrscheinlichkeit $q_i(X(k^*.,i)\,|\,X(k,i),X(k,-i))$ erzeugt, wobei

$$X(k,-i) := \{X(k+1,1),\ldots,X(k+1,i-1),X(k,i+1),\ldots,X(k+1,n)\}$$

den Stand von $X_{.,-i}$ am Ende des $i-1$-ten Schrittes der Iteration $k+1$ bezeichnet.

Die Akzeptanz-Wahrscheinlichkeit

$$\alpha(X(.,\text{–}i),X(.,i),X(k^*,i) = \min\left(1,\frac{q_i(X(.,i)\,|\,X(k^*,i),X(.,\text{–}i)}{q_i(X(k^*,i)\,|\,X(.,-i),X(.,i))}\cdot\frac{\pi(X(k^*,i\,|\,X(.,\text{–}i)}{\pi(X(.,i)\,|\,X(.,\text{–}i)}\right)$$

$\pi(x(.,i)|X(.,-i))$ wird dabei als *Vollständige bedingte Verteilung* für ein $X$ unter einem $\pi$ bezeichnet

Wenn $X(k^*,i)$ akzeptiert wird, setzt man $X(k+1,i) = X(k^*,i)$, andernfalls bleibt $X(k+1,i) = X(k,i)$. Die restlichen Komponenten bleiben im $i$-ten Schritt unverändert.

Die Eindeutigkeit der invarianten Wahrscheinlichkeitsverteilung $\pi$ ist gewährleistet, da $\pi$ durch die Menge ihrer vollständig bedingten Verteilungen eindeutig bestimmt ist.

Wir beweisen folgendes Lemma, um zu zeigen, dass $\pi$ tatsächlich invariant ist.

**Lemma 2.2.** *Sei $\pi(X)$ eine stationäre Verteilung der resultierenden Markov-Kette $(X(k))$.*

*Beweis.* Wir müssen zeigen, dass für jedes $X_{k^*,1:n}$ gilt:

$$\sum_{X(.,1:n)} \pi(X(.,1:n))P(X(k^*,1:n)\,|\,X(.,1:n)) = \pi(X(k^*,1:n))$$

wobei $P$ die Übergangsmatrix ist, die durch den (systematischen) Gibbs-Sampler impliziert wird. Nun, dies lässt sich wie folgt umschreiben:

$$\begin{aligned}
&\sum_{X(.,1:n)} \pi(X(.,1:n))P(X(k^*,1:n)\,|\,X(.,1:n))\\
&= \sum_{X(.,1:n)} \pi(X(.,1:n))\pi_1(X(k^*,1)\,|\,X(k^*,2:n))\pi_2(X(k^*,2)\,|\,X(k^*,1),X(k^*,3:n))\ldots\pi_n(X(k^*,n)\,|\,X(k^*,2:n))\\
&= \sum_{X(.,2:n)} \pi(X(.,2:n))\pi_1(X(k^*,1)\,|\,X(k^*,2:n))\pi_2(X(k^*,2)\,|\,X(k^*,1),X(k^*,3:n))\ldots\pi_n(X(k^*,n)\,|\,X(k^*,2:n))\\
&= \sum_{X(.,2:n)} \pi(X(k^*,1),X(2:n))\pi_2(X(k^*,2)\,|\,X(k^*,1),X(k^*,3:n))\ldots\pi_n(X(k^*,n)\,|\,X(k^*,2:n))\\
&\vdots\\
&= \sum_{X(.,3:n} \pi(X(k^*,1:n-1),X(3:n))\pi_n(X(k^*,n)\,|\,X(k^*,2:n))\\
&= \pi(X(k^*,1:n))
\end{aligned}$$

Somit ist $\pi(X)$ tatsächlich eine stationäre Verteilung der Markov-Kette. □

Der **Gibbs-Sampler** wurde 1984 von Geman und Geman eingeführt und ist eine spezielle Form des komponentenweisen Metropolis-Hastings-Algorithmus. Er beruht darauf, dass eine neue Position $X(k^*) \in S$ nur dann eine positive Wahrscheinlichkeit des Vorschlags $q(X(k^*)|X(k)$ besitzt, wenn die neue Position $X(k^*)$ und die ursprüngliche Position $X(k)$ nicht stark voneinander abweichen, indem sie sich zum Beispiel nur in einer Koordinate unterscheiden.

Es gibt verschiedene Varianten des Gibbs-Samplers, darunter:

- **Randomisiert:** Zufällige Auswahl einer (Block von) Koordinate(n) $i$ in $X(.,i)$ und Änderung entsprechend $q_i$.
- **Zyklisch:** Durchlaufen jeder (Block von) Koordinate(n) $i$ in $X(.,i)$ und Änderung entsprechend $q_i$.

Diese Variationen erlauben eine flexible Anpassung an die Struktur des Problems und ermöglichen eine effiziente Exploration des Parameterraums.

# Kapitel 3

# MCMC Approximationen und Verbesserungen

## 3.1 Konvergenz von Approximationen mit MCMC

Um die Konvergenz des Metropolis-Hastings-Algorithmus nachweisen zu können, müssen zunächst die dafür notwendigen Voraussetzungen erfüllt sein. $\rho(x(k), x(k^*))$ wird so konstruiert, dass die Markovkette reversibel ist. Damit gilt der folgende Satz:

**Satz 3.1.** *Wird eine Markovkette mit Hilfe des Metropolis-Hastings-Algorithmus konstruiert, so ist $\pi$ deren invariante Wahrscheinlichkeitsverteilung.*

*Beweis.* Es seien $\pi$ eine Wahrscheinlichkeitsverteilung, $Q$ eine Vorschlagsverteilung, $\rho(x(k), x(k^*)) = \min\left(1, \frac{\pi(x(k^*))q(x(k)|x(k^*))}{\pi(x(k))q(x(k^*)|x(k))}\right)$, und es gelte
$p(x(k), x(k^*)) = q(x(k^*)|x(k))\rho(x(k), x(k^*))$ für $x(k) \neq x(l)$ und
$p(x(k), x(k)) = 1 - \sum_{x(k)\neq x(k^*)} p(x(k), p(x(k^*))$

Dabei ist $p(x(k), x(k^*))$ ein Eintrag in der Übergangsmatrix $P$. Für die Invarianz ist zu zeigen, dass

$$\begin{aligned}&\pi(x(k))p(x(k), x(k^*)) = \pi(x(k))q(x(k^*)|x(k))\rho(x(k), x(k^*))\\ &= \pi(x(k^*))q(x(k)|x(k^*))\rho(x(k^*), x(k)) = \pi(x(k^*))p(x(k^*), x(k))\end{aligned}$$

erfüllt ist, da aus der Reversibilität von $\pi$ auch die Invarianz folgt.

Im Fall $\pi(x(k))q(x(k^*)|x(k)) = 0$ ist diese Aussage trivial, da dann $\rho(x(k^*), x(k)) = 0$ gilt und somit auf beiden Seiten 0 resultiert.

Es sei also $\pi(x(k))q(x(k^*)|x(k)) > 0$. Dann sind drei Fälle zu unterscheiden:

**Fall 1:** Es gilt

$$\rho(x(k), x(k^*)) = \min\left(1, \frac{\pi(x(k^*))q(x(k)|x(k^*))}{\pi(x(k))q(x(k^*)|x(k))}\right) = \frac{\pi(x(k^*))q(x(k)|x(k^*))}{\pi(x(k))q(x(k^*)|x(k))}$$

.

Das bedeutet $\pi(x(k^*))q(x(k)|x(k^*)) < \pi(x(k))q(x(k^*)|x(k))$ und somit gilt
$\rho(x(k^*), x(k)) = \min\left(1, \frac{\pi(x(k))q(x(k^*)|x(k))}{\pi(x(k^*))q(x(k)|x(k^*))}\right) = 1.$

Es ergibt sich:

$$\begin{aligned}&\pi(x(k))p(x(k), x(k^*)) = \pi(x(k))q(x(k^*)|x(k))\rho(x(k), x(k^*))\\ &= \pi(x(k))q(x(k^*)|x(k))\frac{\pi(x(k^*))q(x(k)|x(k^*))}{\pi(x(k))q(x(k^*)|x(k))}\\ &= \pi(x(k^*))q(x(k)|x(k^*))\\ &= \pi(x(k^*))q(x(k)|x(k^*))\rho(x(k^*), x(k))\\ &= \pi(x(k^*))p(x(k^*), x(k))\end{aligned}$$

**Fall 2:** Es gilt

$$\rho(x(k), x(k^*)) = \min\left(1, \frac{\pi(x(k^*))q(x(k)|x(k^*))}{\pi(x(k))q(x(k^*)|x(k))}\right) = 1$$

.

Das heißt, es ist entweder $\pi(x(k^*))q(x(k)|x(k^*)) = \pi(x(k))q(x(k^*)|x(k))$ (siehe Fall 3) oder es gilt $\pi(x(k^*))q(x(k)|x(k^*)) > \pi(x(k))q(x(k^*)|x(k))$ und somit
$\rho(x(k^*), x(k)) = \min\left(1, \frac{\pi(x(k))q(x(k^*)|x(k))}{\pi(x(k^*))q(x(k)|x(k^*))}\right) = \frac{\pi(x(k))q(x(k^*)|x(k))}{\pi(x(k^*))q(x(k)|x(k^*))}$
In diesem Fall erhält man:

$$\begin{aligned}
&\pi(x(k))p(x(k), x(k^*)) = \pi(x(k))q(x(k^*)|x(k))\rho(x(k), x(k^*)) \\
&= \pi(x(k))q(x(k^*)|x(k)) \cdot 1 \\
&= \pi(x(k))q(x(k^*)|x(k)) \cdot \frac{\pi(x(k^*))q(x(k)|x(k^*))}{\pi(x(k^*))q(x(k)|x(k^*))} \\
&= \pi(x(k^*))q(x(k)|x(k^*))\rho(x(k^*), x(k)) \\
&= \pi(x(k^*))p(x(k^*), x(k))
\end{aligned}$$

**Fall 3:** Es gilt $\pi(x(k^*))q(x(k)|x(k^*)) = \pi(x(k))q(x(k^*)|x(k))$. Hieraus folgt $\rho(x(k), x(k^*)) = \rho(x(k^*), x(k)) = 1$ und somit trivialerweise:

$$\begin{aligned}
&\pi(x(k))p(x(k), x(k^*)) = \pi(x(k))q(x(k^*)|x(k))\rho(x(k), x(k^*)) \\
&= \pi(x(k))q(x(k^*)|x(k)) \cdot 1 \\
&\pi(x(k^*))p(x(k^*), x(k)) \\
&= \pi(x(k^*))q(x(k)|x(k^*))\rho(x(k^*), x(k)) \\
&= \pi(x(k^*))q(x(k)|x(k^*)) \cdot 1
\end{aligned}$$

□

Die Wahl von Q beeinflusst weiterhin, ob die konsturierte Markovkette aperiodisch ung irreduzibel ist. $Q$ hat aufgrund der Definition von
$p(x(k), x(l)) = q(x(l)|x(k))\rho(x(k), x(l))$ für $x(k) \neq x(l)$ und
$p(x(k), x(k)) = 1 - \sum_{x(k)\neq x(l)} p(x(k), p(x(l))$
einen direkten Einfluss auf die Irreduzibilität von $P$. Es ist also $q(x(l)|x(k)) > 0$ $\forall x(k), x(l)$ hinreichend, aber nicht notwendig. Nur in der Original-Version des Algorithmus, dem *Metropolis-Algorithmus* ist die Akzeptanzwahrschienlichkeit $\rho$ unabhängig von Q. Nur wenn alle Bedingungen erfüllt sind, kann der *Fundamentalsatz für ergodische Markovketten* angewandt werden, um eine Konvergenzaussage zu treffen.

Die Aperiodizitätsbedingung für den Metropolis-Hastings-Algorithmus lautet wie folgt:

$$\mathbb{P}\left(\pi(x(k))q(x(k^*)|x(k)) \leq \pi(x(k)q(X(k)|x(k^*))\right) < 1$$

Hierbei steht $\pi$ für die Zielverteilung, $x(k)$ ist der aktuelle Zustand der Kette, $x(k^*)$ ist ein Vorschlag aus der Vorschlagsverteilung $q(x(K^*)|x(k))$, und $P$ ist die Wahrscheinlichkeit.

Das bedeutet, dass die Wahrscheinlichkeit, dass der Übergang $x(k) \to x(k^*)$ genauso wahrscheinlich oder wahrscheinlicher ist als $x(k^*) \to x(k)$, kleiner als eins sein muss.

## 3.2 Erweiterungen des Algorithmus

Der Metropolis-Hastings (MH) Algorithmus, obwohl für alle Dichten anwendbar, kann oft eine langsame Konvergenz aufweisen. Dies liegt teilweise am sogenannten "random walk behavior"des Algorithmus. Hierbei werden neue Samples $x(k^*)$ willkürlich vorgeschlagen, ohne die spezielle Struktur der Zielverteilung $\pi$ zu berücksichtigen.

Um diesem Problem entgegenzuwirken und die intrinsische Struktur von $\pi$ besser auszunutzen, wurde der Metropolis Adjusted Langevin Algorithmus (MALA) entwickelt.

Der MALA-Algorithmus integriert Informationen über den Gradienten der Verteilung $\pi$, repräsentiert durch den Gradienten des Logarithmus der Dichte, in die Generierung neuer Vorschläge. Dies ermöglicht eine gezieltere Exploration des Zustandsraums und kann zu einer effizienteren Konvergenz des Algorithmus führen.

Die Akzeptanzwahrscheinlichkeit im MH-Algorithmus berücksichtigt zwar bereits die Zielverteilung $\pi$, aber der MALA-Algorithmus geht einen Schritt weiter, indem er auch Informationen über die Lokalstruktur von $\pi$ nutzt. Dieser Ansatz kann insbesondere in hochdimensionalen Zustandsräumen zu einer verbesserten Performance führen.

### 3.2.1 Metropolis Adjusted Langevin Algorithmus (MALA)

Der Metropolis Adjusted Langevin Algorithmus (MALA) verwendet ein spezielles Proposal für den Metropolis-Hastings (MH) Algorithmus, um die Konvergenz zu verbessern. Der Vorschlag $x(k^*)$ wird wie folgt definiert:

$$x(k^*) = x(k) + \varepsilon \nabla \log(\pi(x(k))) + \sqrt{2\varepsilon}\, \xi_k$$

Hierbei ist $\varepsilon > 0$ die Schrittweite, oft mit $\varepsilon \in (0, 0.5)$ als Hyperparameter festgelegt, und $\xi_k$ ist standardnormalverteilt. Der Vorschlag setzt sich aus drei additiven Teilen zusammen: $x(k)$, $\varepsilon \nabla \log(\pi(x(k)))$ und $\sqrt{2\varepsilon}\, \xi_k$. Diese Teile bewirken, dass der neue Punkt in der Nähe des vorherigen liegt, einen Schritt in die Richtung mit der höchsten Annahmewahrscheinlichkeit macht und einen zufälligen Schritt in den Zustandsraum hinzufügt.

Es ergibt Sinn, den Vorschlag so zu wählen, da MALA den Gradienten $\nabla$ von $\pi$ verwendet, um die Markov Chain in die Richtung von Punkten zu lenken, die die höchste Annahmewahrscheinlichkeit haben. Dies führt dazu, dass die Markov Chain nicht mehr einem zufälligen Spaziergang ähnelt.

Die Vorteile von MALA liegen darin, dass er den Gradienten von $\pi$ verwendet, um $\pi$ lokal zu beschreiben, eine gute Performance im Vergleich zum einfachen MH-Algorithmus bietet und ein einfacher Algorithmus mit einem soliden theoretischen Fundament ist. Ein Nachteil ist jedoch, dass der Hyperparameter $\varepsilon$ oft schwierig und teuer abzustimmen ist, und auch der Gradient $\nabla$ von $\log(\pi)$ benötigt wird. Zudem erkundet MALA den Zustandsraum intelligent, aber möglicherweise langsam, was zu hoher Autokorrelation zwischen den Samples führen kann. In solchen Fällen könnte der Hamiltonian Monte Carlo (HMC) eine bessere Alternative sein.

### 3.2.2 Hamiltonian Monte Carlo (HMC)

Der Hamiltonian Monte Carlo (HMC) verbessert das Metropolis-Hastings (MH) Sampling, indem er mehrere Schritte auf einmal macht, um einen Vorschlag zu generieren. Dies reduziert die Korrelation zwischen den einzelnen Zuständen und erhöht gleichzeitig die Annahmewahrscheinlichkeit. Die Idee besteht darin, mehrere Schritte gleichzeitig durchzuführen, um ein Proposal zu generieren, und dabei die Hamilton'schen Gleichungen zu verwenden.

Für jedes Sample $x(k)$ führt HMC eine normalverteilte Variable $p_i$ ein. Die Hamilton'schen Gleichungen werden gelöst, um einen Proposal-Punkt $x(k^*)$ zu generieren:

$$\frac{dx(k)}{dt} = \frac{dH}{dp_i}, \quad \frac{dp_i}{dt} = -\frac{dH}{dx(k)}$$

wobei $H(x, p) = -\log(\pi(x)) + 0.5 p^T p$. Die Hamilton'schen Gleichungen werden numerisch mit dem Leapfrog Integrator gelöst.

Die Lösung der Hamilton'schen Gleichungen bewirkt, dass die Energie im System konstant bleibt, auch wenn wir uns weit bewegen. Das HMC ermöglicht schnelles Navigieren in Regionen mit hoher Wahrscheinlichkeit, wodurch Annahmewahrscheinlichkeiten von nahezu 1 erreicht werden.

HMC hat zwei Hyperparameter: die Schrittweite $\varepsilon$ und die Anzahl der Schritte $L$. Die Wahl von $\varepsilon$ ist das Ergebnis des Leapfrog Integrators und relativ einfach. Die Anzahl der Schritte $L$ ist jedoch problematisch, da es kein optimales $L$ gibt. Manche Samples würden

gerne länger erkundet werden, während andere bereits ausreichend erkundet wurden. Das No U-Turn Sampler ist eine Lösung für dieses Problem und verbessert die Leistung des Algorithmus erheblich.

### 3.2.3 No U-Turn Sampler (NUTS)

Der No U-Turn Sampler (NUTS) ist eine Erweiterung des Hamiltonian Monte Carlo (HMC), die die Anzahl der Schritte $L$ adaptiv entscheidet. Im Standard-HMC ist die Anzahl der Schritte für jedes Sample $x(k)$ festgelegt, was zu einem ineffizienten Algorithmus führen kann. NUTS löst dieses Problem, indem es die Anzahl der Schritte adaptiv für jedes Sample wählt.

Typischerweise liegt die Anzahl der Schritte in einem Bereich von 40 bis 100. Der Algorithmus funktioniert, indem er solange weitere Schritte macht, bis das neue Proposal näher am Ausgangspunkt ist als der vorherige Punkt. Dies wird als Ü-Turn"bezeichnet und erfolgt, wenn die euklidische Distanz zwischen den Punkten einen bestimmten Schwellenwert erreicht.

Der NUTS-Algorithmus ermöglicht eine adaptive Anpassung der Anzahl der Schritte für jedes Sample, was zu einer effizienteren Exploration des Zustandsraums führt. Nachdem die Schritte durchgeführt wurden, wird aus allen generierten Punkten uniform ein Punkt als Sample $x(k^*)$ ausgewählt.

# Kapitel 4

# Zusammenfassung

Markov Chain Monte Carlo (MCMC) ist eine Methode zur Schätzung von Verteilungen und Erwartungswerten von Zufallsvariablen. Die grundlegende Idee besteht darin, eine Markov-Kette zu erzeugen, deren stationäre Verteilung mit der gewünschten Zielverteilung übereinstimmt. Dies ermöglicht die Approximation von komplexen Wahrscheinlichkeitsverteilungen.

Die Monte Carlo Simulation kann bei hochdimensionalen ineffizient sein. Traditionelle Methoden können Schwierigkeiten bei der Exploration solcher Räume haben, was zu langen Konvergenzzeiten führt. Hier setzt MCMC an, um eine bessere Exploration zu ermöglichen.

MCMC kann sinnvoller sein, da es eine gezielte Exploration des Zustandsraums ermöglicht. Anstatt zufällige Samples zu generieren, folgt MCMC einer Markov-Ketten-Struktur, um relevante Bereiche des Raums zu entdecken. Dies ist besonders nützlich in hochdimensionalen Räumen.

Für den Fundamentalsatz für ergodische Markovketten sind Irreduzibilität, Aperiodizität und positive Rekurrenz, sowie die Existenz einer invarianten Verteilung erforderlich. Irreduzibilität stellt sicher, dass jeder Zustand von jedem anderen Zustand erreicht werden kann. Aperiodizität sorgt dafür, dass die Kette nicht in regelmäßigen Intervallen zu einem Zustand zurückkehrt. Die Detailed Balance Bedingung stellt sicher, dass die Markov-Kette im Gleichgewicht ist.

Der Fundamentalsatz besagt, dass jede irreduzible, aperiodische, positiv rekurrente Markov-Kette eindeutig eine stationäre Verteilung hat. Dies bedeutet, dass die Kette gegen eine stationäre Verteilung konvergiert.

Der Metropolis Hastings Algorithmus ist eine MCMC-Technik, die auf dem Metropolis Algorithmus basiert. Er verwendet eine Vorschlagsverteilung, um neue Samples zu generieren, und verwendet eine Akzeptanzwahrscheinlichkeit, um zu entscheiden, ob ein neues Sample akzeptiert wird. Der Algorithmus gewährleistet, dass die stationäre Verteilung erreicht wird.

Der Metropolis Hastings Algorithmus erfordert das Anpassen von Parametern wie der Schrittweite, der Burn-in-Phase und der Autokorrelation. Die Schrittweite beeinflusst die Exploration des Raums, die Burn-in-Phase hilft, sich von einem schlechten Startpunkt zu erholen, und die Autokorrelation zeigt die Abhängigkeit zwischen aufeinanderfolgenden Samples an. Die Konvergenz von MCMC ist entscheidend für zuverlässige Schätzungen. Es ist wichtig sicherzustellen, dass ausreichend viele unkorrelierte Samples generiert werden.

Erweiterungen wie der Metropolis Adjusted Langevin Algorithmus (MALA) und der No U-Turn Sampler (NUTS) wurden entwickelt, um spezifische Probleme in MCMC zu adressieren. MALA nutzt den Gradienten der Verteilung, um intelligentere Schritte zu generieren, während NUTS die Anzahl der Schritte adaptiv bestimmt, um effizientere Explorationen zu ermöglichen.

# Kapitel 5

# Quellenangabe

# Literaturverzeichnis


[1] https://de.wikipedia.org/wiki/metropolis-algorithmus.

[2] https://en.wikipedia.org/wiki/importance$_s ampling : : text = importance$

[3] https://en.wikipedia.org/wiki/markov$_c hain_m onte_c arlo$.

[4] https://itp.tugraz.at/mml/montecarlo/mcintro.pdf.

[5] https://link.springer.com/article/10.3758/s13423-016-1015-8.

[6] https://machinelearningmastery.com/markov-chain-monte-carlo-for-probability/.

[7] https://tu-dresden.de/mn/psychologie/ifap/methpsy/ressourcen/dateien/forschung/openscience/materialien/hoefl
bayesianische-statistik.pdf?lang=de.

[8] https://twiecki.io/blog/2015/11/10/mcmc-sampling/.

[9] https://www.hzdr.de/db/cms?poid=45798.

[10] Christina Gunkel. *Die Markov Ketten Monte Carlo Methode zum Testen stochastisch Unabhängigkeit*. 2008.

[11] Masanori Hanada and So Matsuura. *MCMC from Scratch: A Practical Introduction to Markov Chain Monte Carlo*. Springer Nature, Singapore Pte Ltd., 2022.

[12] Noemi Kurt. *Stochastik für Informatiker: Eine Einführung in einheitlich strukturierte Lerneinheiten*. Springer Nature, 2020.

[13] Handsruedi Künsch. *Stochastische Simulation: Skript zur Vorlesung SS06, ETH Zürich*. 2006.

[14] Edward Teller Nicholas Metropolis. *Equation of State Calculations by Fast Computing Machines*. 1953.

[15] Prof. Gesine Reinert. *Markov Chain Monte Carlo and Applied Bayesian Statistics*. 2005.

[16] Prof. Dr. Martin Slowik. *Skript zur Vorlesung "Markovketten*
. 2023.
[11] [12] [10] [14] [15] [16] [13] [7] [9] [1] [4] [3] [6] [8] [2] [5]