



Escola de Artes, Ciências e Humanidades  
Universidade de São Paulo

# MCMC e Amostrador de Gibbs

SIN5008 - Estatística Computacional

Alex Ceccon e Leonardo C. Santos

São Paulo / 2025



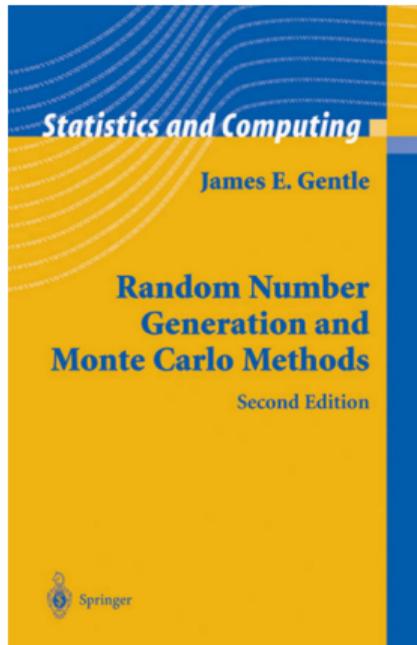
# Agenda

- ① MC - Markov Chain
- ② MC - Monte Carlo Method
- ③ MCMC - Markov Chain Monte Carlo
- ④ Algoritmos
  - Metropolis
  - Amostrador de Gibbs
  - Hamiltonian Monte Carlo (HMC)
- ⑤ Curiosidades



# Monte Carlo Methods

Referência Principal



Gentle, J. E. (2003).  
*Random Number Generation and Monte Carlo Methods.*

2nd ed. Springer.

Chapter 7 - Monte Carlo Methods



# Linha Histórica

MCMC — marcos principais

Ano	Marco	Nota
1913	Markov	Cadeias com dependência apenas do estado atual
1949	Monte Carlo	Primeiras aplicações computacionais (Ulam, von Neumann)
1953	Metropolis	Primeiro algoritmo MCMC prático (Metropolis et al.)
1970	Hastings	Generaliza a proposta não simétrica (Metropolis–Hastings)
1984	Gibbs Sampling	Atualizações por condicionais completas (Geman & Geman)
1987	HMC	Dinâmica Hamiltoniana para propostas eficientes (Duane et al.)
2011	NUTS	HMC sem tuning manual do número de passos (Hoffman & Gelman)
2012+	Stan	Sistema moderno que populariza HMC/NUTS (modelagem Bayesiana)

# Cadeia de Markov



# Cadeia de Markov

Processos Estocásticos e Dependência de Estado

Uma **Cadeia de Markov** é um processo estocástico em que a **probabilidade do próximo estado depende apenas do estado atual**, e não dos anteriores. Essa característica é chamada de *propriedade de Markov*.

- O sistema é composto por **estados** (situação atual) e **transições**
- Cada transição possui uma **probabilidade associada**
- Representação comum: **matriz de transição**
- Aplicações: previsão do tempo, reconhecimento de fala, comportamento de usuários, etc



# Cadeia de Markov

Processos Estocásticos e Dependência de Estado

Estado atual →	Próximo: Feliz	Próximo: Neutro	Próximo: Triste
<b>Feliz</b>	0.6	0.3	0.1
<b>Neutro</b>	0.4	0.4	0.2
<b>Triste</b>	0.2	0.3	0.5

- Estados: *Feliz, Neutro, Triste*
- Exemplo: se a pessoa está *Feliz*, há 60% de chance de permanecer assim no próximo dia

$$P(X_{n+1} = x \mid X_n = y, X_{n-1}, \dots) = P(X_{n+1} = x \mid X_n = y)$$



# Cadeia de Markov

## Resultados Teóricos

Conceito	Descrição
<b>Estado</b>	Situação possível do sistema (ex.: Sol, Chuva). Pode ser transitório, recorrente ou absorvente.
<b>Transição</b>	Mudança entre estados com probabilidade $p_{ij} = P(X_{n+1} = j   X_n = i)$ .
<b>Matriz <math>P</math></b>	kernel de transição - contém todas as probabilidades $p_{ij}$ ; cada linha soma 1.
<b>Distribuição estacionária</b>	Vetor $\pi$ tal que $\pi P = \pi$ . Representa o equilíbrio do sistema.
<b>Cadeia ergódica</b>	Cadeia que converge para $\pi$ independentemente do estado inicial.

*Propriedade de Markov: o futuro depende apenas do presente, não do passado.*



# Cadeia de Markov

## Resultados Teóricos

---

Conceito	Descrição
Estado	<p><b>Transiente:</b> o sistema pode sair dele e talvez nunca mais retornar;</p> <p><b>Recorrente:</b> o sistema sempre retorna a esse estado com o tempo;</p> <p><b>Absorvente:</b> uma vez alcançado, o sistema permanece nele para sempre.</p>

---



# Cadeia de Markov

## Resultados Teóricos

---

Conceito	Descrição
<b>Cadeia Ergódica</b>	<p><b>Irredutível:</b> é possível ir de qualquer estado a qualquer outro (direta ou indiretamente)</p> <p><b>Aperiódica:</b> não há ciclos fixos (o retorno a um estado não ocorre em múltiplos fixos de tempo)</p> <p><b>Positivamente recorrente:</b> o tempo esperado para retornar a um estado é finito</p>

---



# Cadeia de Markov

## Propriedades

- Sequência de variáveis aleatórias  $(X_0, X_1, X_2, \dots)$  que satisfaz a **propriedade de Markov**:

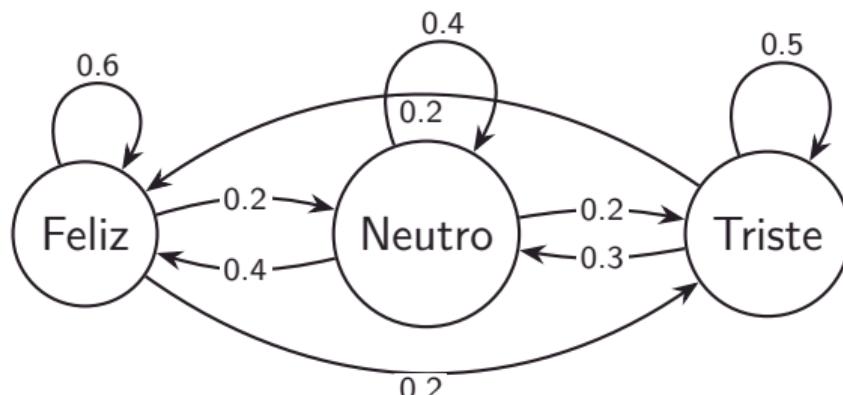
$$P(X_{t+1} | X_t, X_{t-1}, \dots) = P(X_{t+1} | X_t)$$

- **Ergodicidade:** a cadeia eventualmente visita todos os estados possíveis
- **Balanço detalhado:**  $\pi(x)P(y|x) = \pi(y)P(x|y)$  garante que  $\pi$  é estacionária
- **Convergência:** sob certas condições, a cadeia converge para uma única distribuição estacionária  $\pi$
- **Irredutibilidade:** qualquer estado pode ser alcançado a partir de outro
- Em MCMC, o kernel de transição  $P(y|x)$  é projetado para que  $\pi$  seja a **distribuição alvo**

# Cadeia de Markov — Feliz, Neutro e Triste



Diagrama e matriz de transição



Matriz de transição  $P$ :

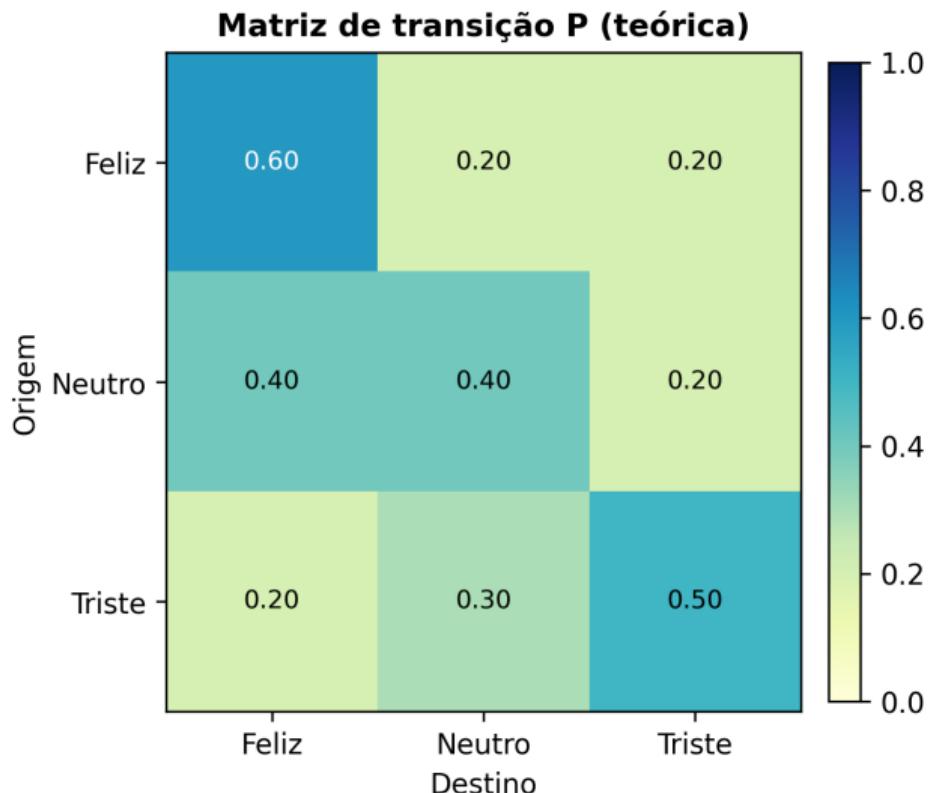
$$P = \begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.4 & 0.4 & 0.2 \\ 0.2 & 0.3 & 0.5 \end{bmatrix}$$

Linhas: estado atual    Colunas: próximo estado



# Cadeia de Markov

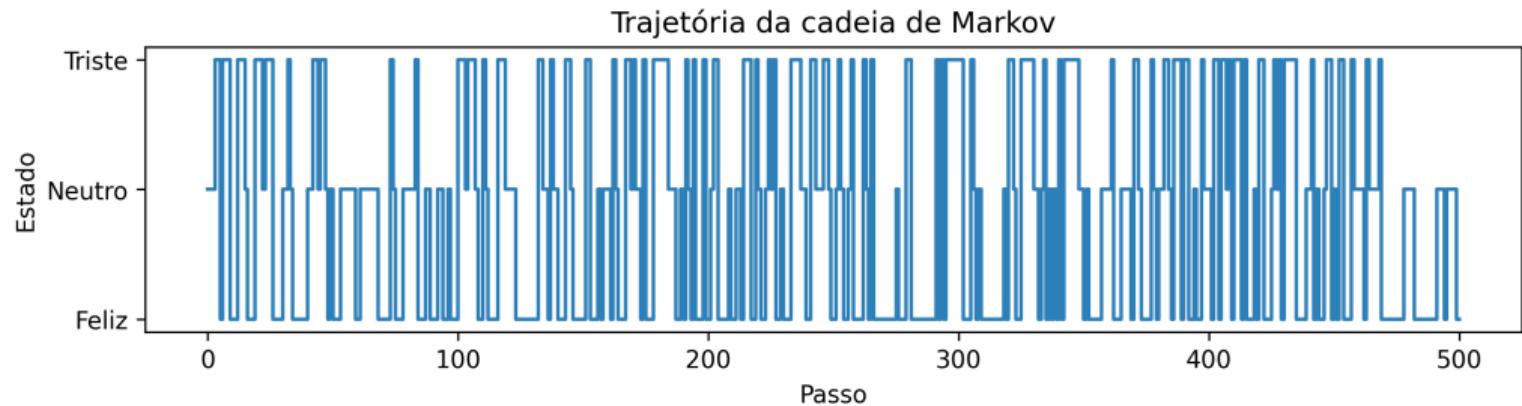
Modelo probabilístico de transição entre estados





# Cadeia de Markov

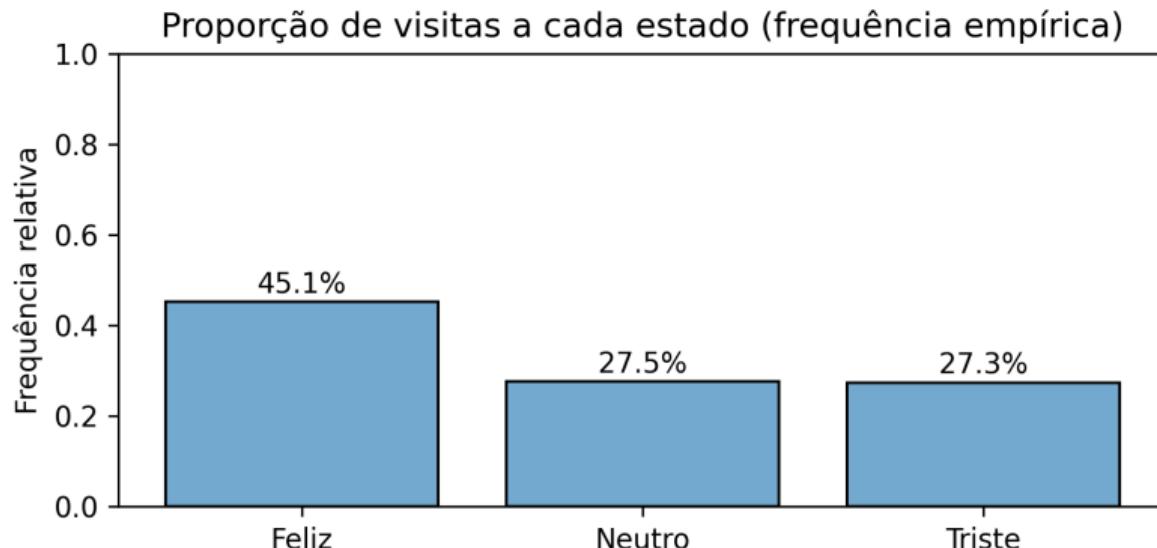
Modelo probabilístico de transição entre estados





# Cadeia de Markov

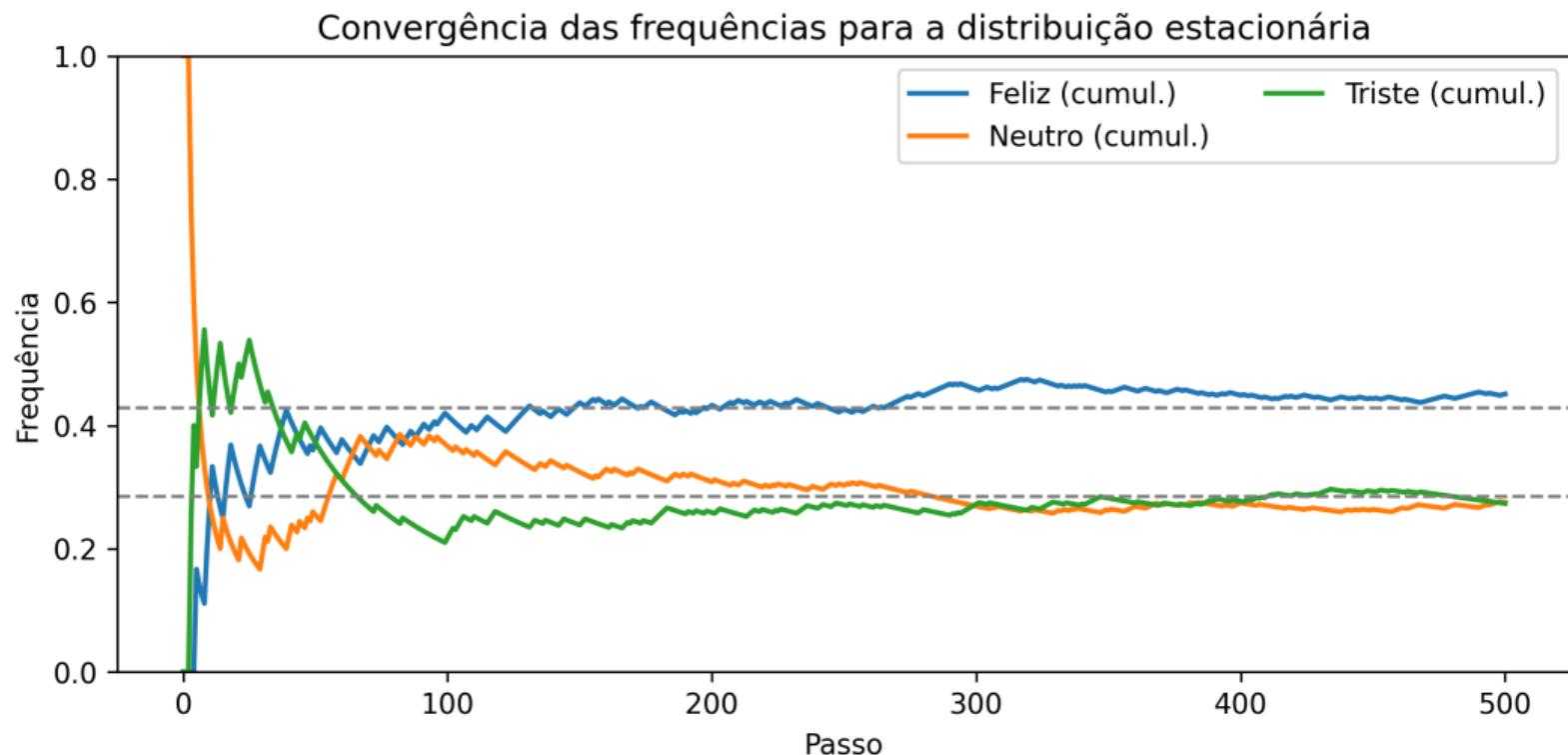
Modelo probabilístico de transição entre estados





# Cadeia de Markov

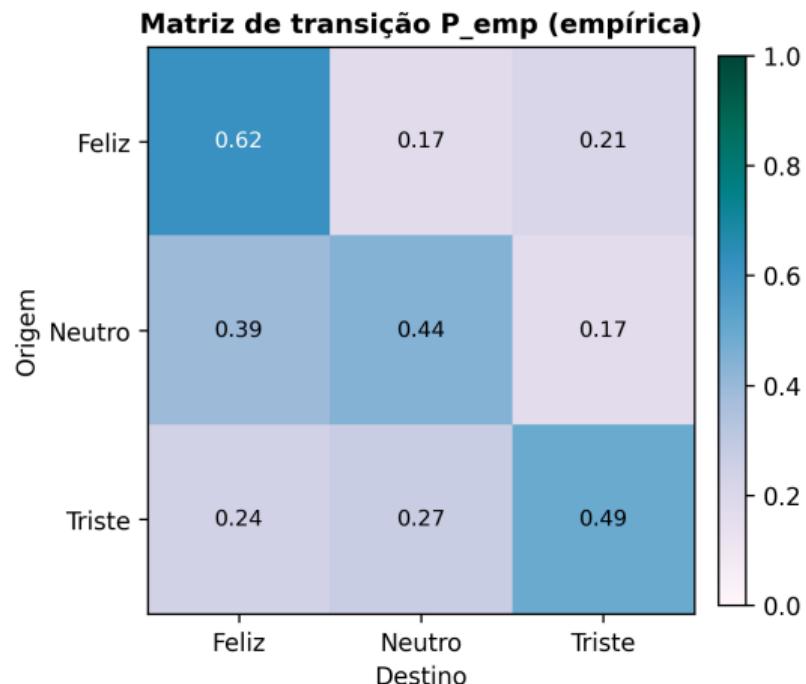
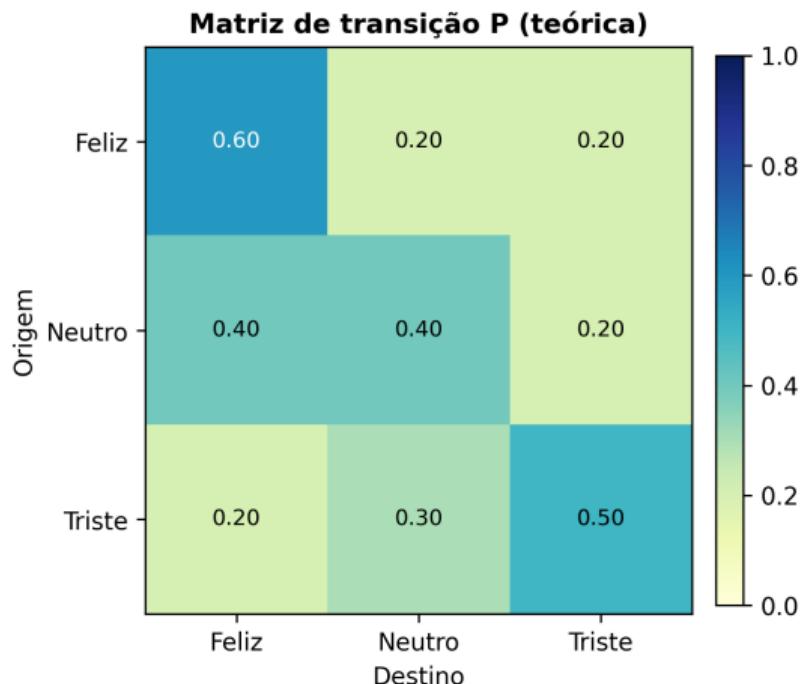
Modelo probabilístico de transição entre estados





# Cadeia de Markov

Modelo probabilístico de transição entre estados



# Cadeia de Markov

`z01_markov_chain.py`



# MC - Monte Carlo



# MC - Monte Carlo Method

## Overview

- Conjunto de métodos estatísticos baseados em **amostragem aleatória** para resolver problemas numéricos complexos
- Estima valores de **expectativas, integrais ou probabilidades** por meio de simulações repetidas
- Fundamenta-se na **Lei dos Grandes Números**: quanto maior o número de amostras, mais próxima a estimativa do valor real
- Aplicável a fenômenos que envolvem incerteza, variabilidade ou sistemas estocásticos



# MC - Monte Carlo Method

Quando usar o Método de Monte Carlo?

- Quando o problema é **difícil de ser resolvido analiticamente**
- Quando há **muitas variáveis**, incertezas ou relações complexas entre parâmetros
- Para **avaliar riscos, incertezas ou cenários** em modelos físicos, financeiros ou computacionais
- Quando se busca uma **estimativa aproximada** com margem de erro controlada



# MC - Monte Carlo Method

Estimativa de  $\pi$  por Monte Carlo

- Considere um **quadrado unitário** e um **quadrante de círculo** de raio  $r = 1$ .
- A área do quadrado é  $A_q = 1^2 = 1$ .
- A área do quadrante é  $A_c = \frac{\pi r^2}{4} = \frac{\pi}{4}$ .
- Gerando  $N$  pontos aleatórios  $(x_i, y_i)$  uniformemente em  $[0, 1] \times [0, 1]$ :

$$x_i^2 + y_i^2 \leq 1 \Rightarrow \text{ponto dentro do círculo.}$$

- A proporção de pontos dentro do círculo aproxima a razão das áreas:

$$\frac{n_{\text{dentro}}}{N} \approx \frac{\pi}{4}$$

- Logo, estima-se:

$$\boxed{\pi \approx 4 \times \frac{n_{\text{dentro}}}{N}}$$

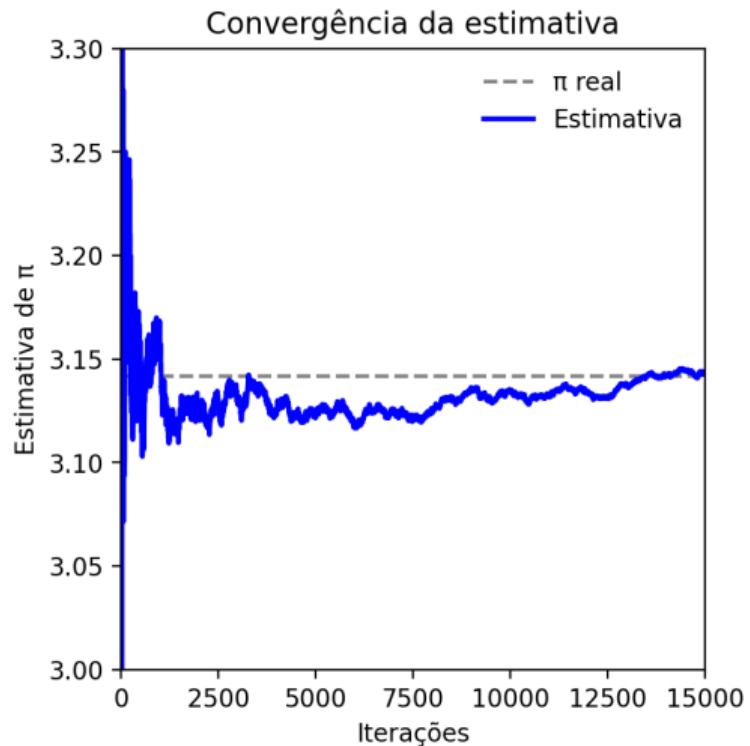
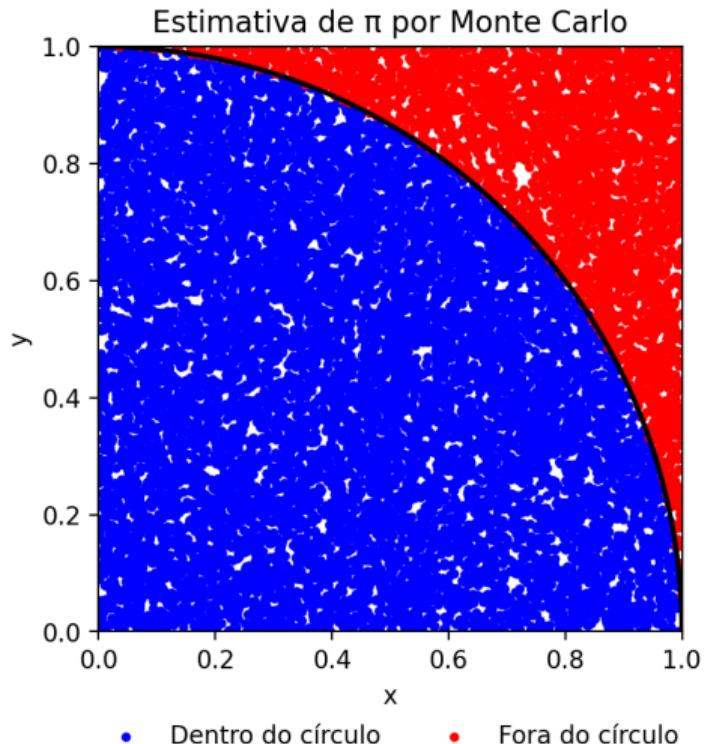
# Monte Carlo Method

`z02_monte_carlo_pi.py`



# MC - Monte Carlo Method

Estimativa de  $\pi$  por Monte Carlo





# MC - Monte Carlo Method

Estimativa de  $\pi$  por Monte Carlo

- O método baseia-se na **Lei dos Grandes Números**: quanto maior o número de amostras  $N$ , mais próxima a média amostral fica do valor esperado
- A precisão cresce proporcionalmente a  $1/\sqrt{N}$
- É um exemplo clássico de uso de **simulação estocástica** para resolver problemas numéricos



# MC - Monte Carlo Method

Lei dos Grandes Números e comportamento do erro

- Cada simulação de Monte Carlo estima um valor médio a partir de  $N$  amostras aleatórias:

$$\hat{p} = \frac{1}{N} \sum_{i=1}^N X_i, \quad \text{com } X_i \in \{0, 1\}$$

- Pela **Lei dos Grandes Números**,  $\hat{p}$  converge para o valor esperado  $p$  conforme  $N$  cresce
- Pelo **Teorema do Limite Central**, a variância da média amostral é:

$$\text{Var}(\hat{p}) = \frac{p(1-p)}{N}$$

- Portanto, o erro padrão da estimativa é:

$$\sigma_{\hat{p}} = \sqrt{\frac{p(1-p)}{N}} \Rightarrow \boxed{\text{Erro} \propto \frac{1}{\sqrt{N}}}$$

A precisão cresce lentamente: aumentar  $N$  em  $4\times$  reduz o erro apenas pela metade!

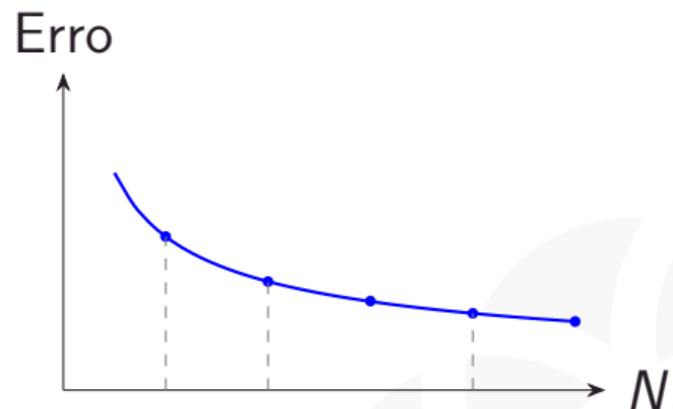


# MC - Monte Carlo Method

Relação entre número de simulações e erro esperado

Amostras ( $N$ )	Erro relativo	Precisão
100	~ 0.10	baixa
1000	~ 0.03	moderada
10000	~ 0.01	boa
100000	~ 0.003	alta

Para dobrar a precisão, é necessário quadruplicar o número de simulações.



Grandes aumentos em  $N$  geram apenas pequenas reduções no erro.

# MCMC - Markov Chain Monte Carlo





# Markov Chain Monte Carlo (MCMC)

Ideia geral

- Combina dois conceitos fundamentais:
  - **Cadeia de Markov:** cada amostra depende apenas da anterior
  - **Monte Carlo:** uso de amostragem aleatória para resolver problemas numéricos
- Permite amostrar de distribuições complexas quando o cálculo direto é inviável
- A cadeia é construída de forma que sua **distribuição estacionária** seja a distribuição alvo
- Amplamente utilizado em **inferência Bayesiana**, onde as distribuições a posteriori são intratáveis



# Inferência Bayesiana

Distribuição a posteriori

- O objetivo da inferência bayesiana é obter a **distribuição a posteriori** dos parâmetros  $\theta$  com dados observados  $\mathcal{D}$ :

$$p(\theta | \mathcal{D}) = \frac{p(\mathcal{D} | \theta) p(\theta)}{p(\mathcal{D})}$$

- Onde:
  - $p(\theta)$ : distribuição a **priori**
  - $p(\mathcal{D} | \theta)$ : **verossimilhança** dos dados
  - $p(\mathcal{D})$ : termo de normalização (ou evidência)
- Essa distribuição resume o que se sabe sobre  $\theta$  após observar os dados.



# O Problema da Normalização

Integral intratável

- O termo de normalização  $p(\mathcal{D})$  é definido como:

$$p(\mathcal{D}) = \int p(\mathcal{D} | \theta) p(\theta) d\theta$$

- Essa integral é frequentemente **intratável**, especialmente em modelos com muitos parâmetros ou hierarquias complexas
- Sem o valor exato de  $p(\mathcal{D})$ , não é possível calcular diretamente a distribuição a posteriori normalizada



# Solução com MCMC

Amostragem sem calcular o termo de normalização

- Os métodos MCMC permitem gerar amostras da distribuição **a posteriori** sem calcular o termo  $p(\mathcal{D})$
- Para isso, utilizam apenas o **núcleo da distribuição**:

$$p(\theta | \mathcal{D}) \propto p(\mathcal{D} | \theta) p(\theta)$$

- Construindo uma cadeia de Markov cuja **distribuição estacionária** é exatamente  $p(\theta | \mathcal{D})$
- Assim, as amostras geradas convergem para a distribuição de interesse, permitindo inferências bayesianas de forma aproximada



# MCMC - Markov Chain Monte Carlo

Amostragem indireta

- O MCMC gera uma sequência que forma uma **cadeia de Markov**
- Essa cadeia é construída de modo que sua distribuição estacionária seja a distribuição **a posteriori** desejada
- Imagine uma pessoa caminhando aleatoriamente sobre uma montanha: com o tempo, a frequência de visitas a cada ponto revela o formato do relevo
- Do mesmo modo, o MCMC “explora” a distribuição e amostra regiões com maior probabilidade



# MCMC - Markov Chain Monte Carlo

Algoritmos e aplicações

## Principais algoritmos

- **Metropolis-Hastings:** propõe novos valores e decide aceitá-los conforme uma probabilidade de aceitação
- **Gibbs Sampling:** atualiza cada parâmetro condicionalmente aos demais
- **Hamiltonian Monte Carlo (HMC):** utiliza gradientes para explorar o espaço de forma mais eficiente

## Aplicações

- Inferência Bayesiana em modelos estatísticos complexos
- Simulações em Física e Química (sistemas de partículas, energia, etc.)
- Aprendizado de Máquina (redes Bayesianas, modelos latentes, estimadores Bayesianos)

# Metropolis



# Metropolis

## Etapas do algoritmo

- ① **Inicialização:** Escolha um ponto inicial  $\theta_0$  no espaço de parâmetros
- ② **Proposição:** Gere uma proposta  $\theta^*$  a partir de uma distribuição simétrica

$$\theta^* \sim q(\cdot | \theta_{t-1})$$

normalmente  $q$  é uma distribuição Normal centrada em  $\theta_{t-1}$

- ③ **Cálculo da razão de aceitação:**

$$r = \frac{p(\theta^*)}{p(\theta_{t-1})}$$

- ④ **Regra de aceitação:** Aceite  $\theta^*$  com probabilidade  $\alpha = \min(1, r)$  caso contrário, mantenha  $\theta_t = \theta_{t-1}$
- ⑤ **Repetição:** Repita o processo gerando a sequência  $(\theta_1, \theta_2, \dots)$



# Metropolis

## Propriedades principais

- **Distribuição estacionária:** Após várias iterações, a cadeia converge para a distribuição alvo  $p(\theta)$
- **Proposta simétrica:** Quando  $q(\theta^* | \theta) = q(\theta | \theta^*)$ , a razão de aceitação simplifica para:

$$r = \frac{p(\theta^*)}{p(\theta_{t-1})}$$

- **Aceitação/rejeição:** As propostas para regiões de alta probabilidade são geralmente aceitas propostas para regiões de baixa probabilidade são aceitas com menor chance
- **Equilíbrio detalhado:** Garante que a distribuição alvo seja estacionária e que o processo satisfaça

$$p(\theta_i)P(\theta_i \rightarrow \theta_j) = p(\theta_j)P(\theta_j \rightarrow \theta_i)$$

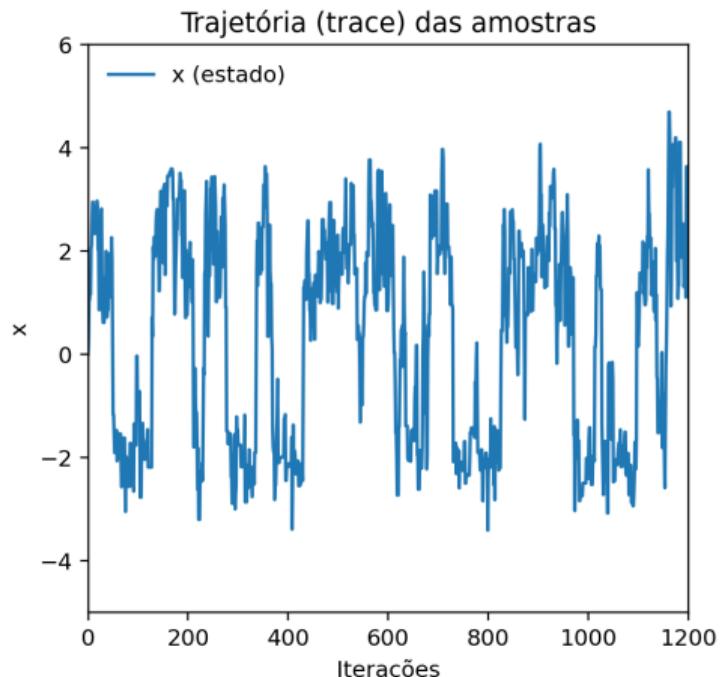
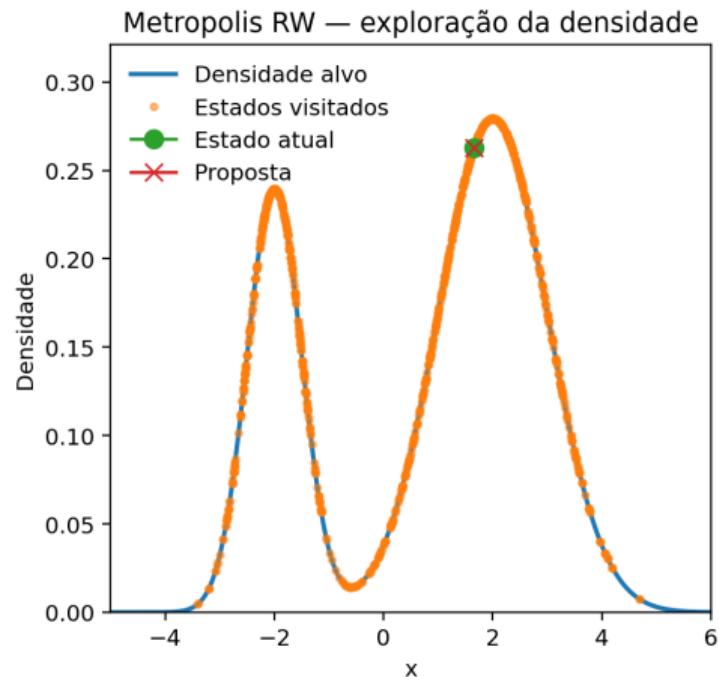
- **Resultado:** A sequência de amostras  $(\theta_t)$  representa a distribuição  $p(\theta)$ , mesmo sem conhecê-la completamente



# Metropolis

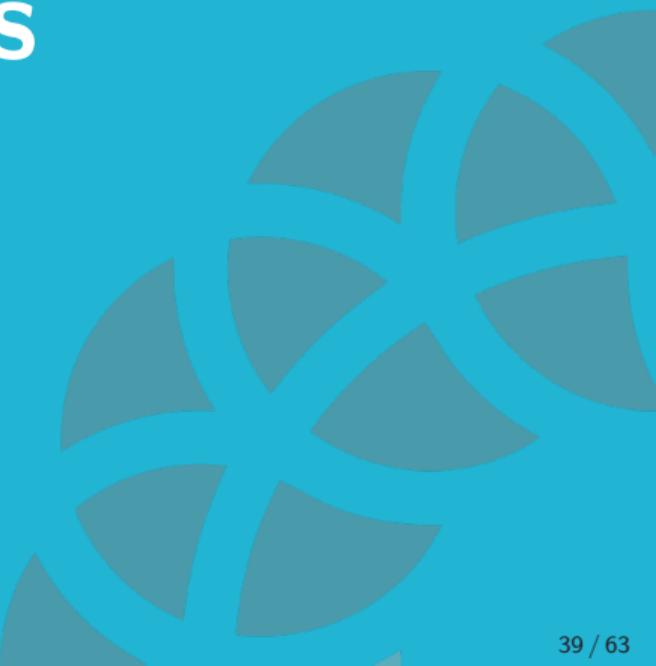
## Random Walk

**Metropolis RW — acc≈0.661 | ESS≈37 |  $\sigma_{\text{proposta}}=1.0$**



# Metropolis

z03\_Metropolis\_RW.py



# Amostrador de Gibbs



# Amostrador de Gibbs

Teoria e aplicações

- O amostrador de Gibbs é um caso particular do algoritmo de Metropolis-Hastings, utilizado quando é possível amostrar diretamente das distribuições condicionais completas
- Considerando uma variável multidimensional  $\mathbf{x} = (x_1, x_2, \dots, x_d)$ , o Gibbs gera amostras sequencialmente de cada componente:

$$x_i^{(t+1)} \sim p(x_i | x_{-i}^{(t)}),$$

onde  $x_{-i}$  representa todos os componentes exceto  $x_i$



# Amostrador de Gibbs

## Teoria e aplicações

- O processo alterna entre as condicionais, atualizando uma variável por vez, até que a cadeia atinja o equilíbrio
- O amostrador converge para a distribuição conjunta  $p(\mathbf{x})$ , mesmo que esta seja difícil de amostrar diretamente
- O desempenho do Gibbs depende da correlação entre as variáveis:
  - Correlações altas reduzem a eficiência (amostras mais dependentes)
  - Correlações baixas favorecem a mistura da cadeia
- É amplamente utilizado em inferência Bayesiana, modelos hierárquicos e análise de redes probabilísticas



# Amostrador de Gibbs

## Teoria e aplicações

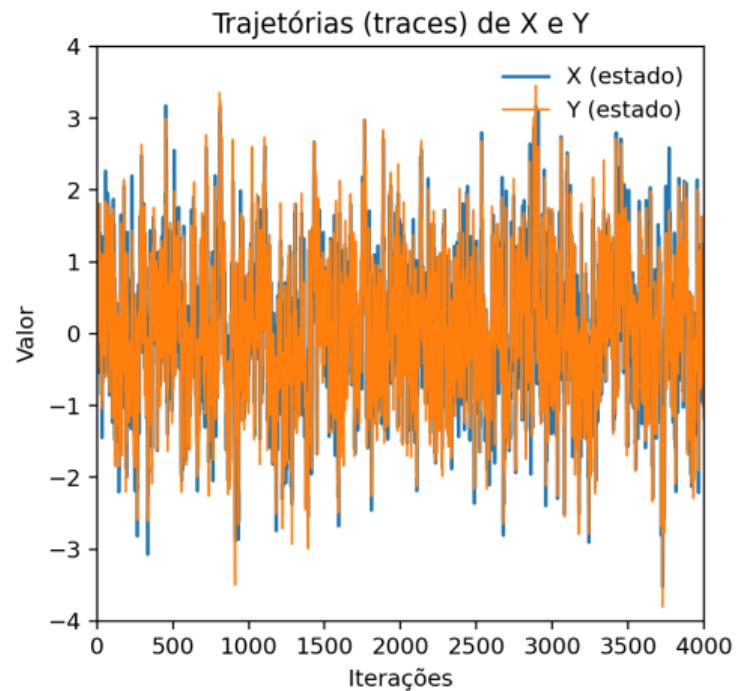
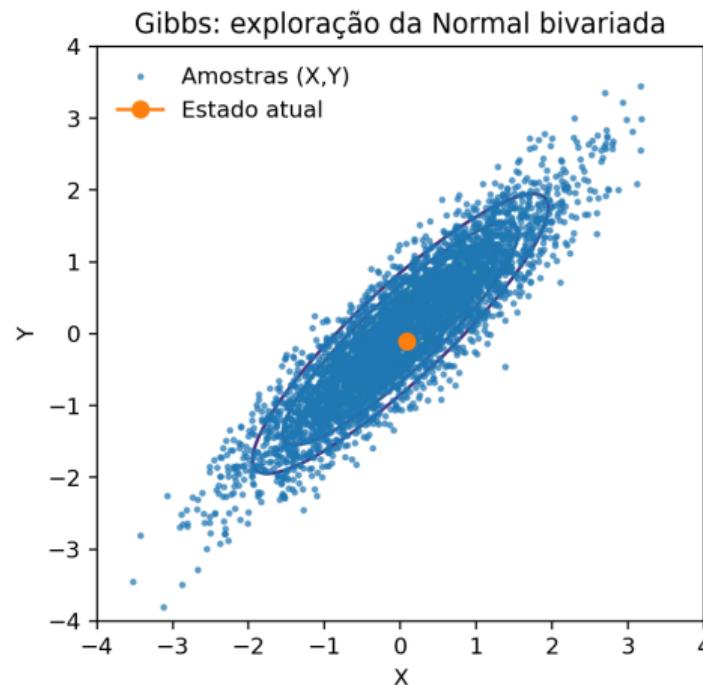
- O amostrador de Gibbs é um caso particular do MCMC em que as **amostras são geradas de forma sequencial**, a partir das **distribuições condicionais** de cada variável
- A ideia central é que, mesmo sem conhecer a distribuição conjunta completa, podemos amostrar de suas condicionais conhecidas:
  - Dada uma variável  $Y$ , amostramos  $X \sim p(X | Y)$
  - Em seguida, atualizamos  $Y \sim p(Y | X)$
  - Esse processo alternado se repete até que a cadeia atinja o equilíbrio
- Após um período de aquecimento (*burn-in*), as amostras  $(X_t, Y_t)$  passam a seguir a distribuição conjunta alvo  $p(X, Y)$
- O Gibbs é especialmente útil quando as condicionais têm **formas conhecidas** (ex.: Normais, Gamas, Bernoullis) e são **fáceis de amostrar**



# Amostrador de Gibbs

Teoria e aplicações

Gibbs — Normal bivariada |  $\rho=0.9$  | ESS\_X≈485 | ESS\_Y≈487





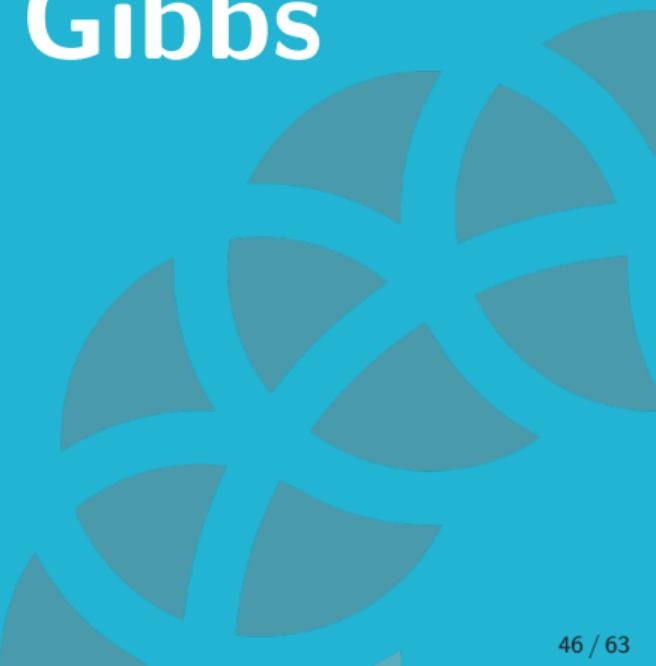
# Amostrador de Gibbs

## Teoria e aplicações

- **Painel esquerdo:** contornos da densidade bivariada e pontos gerados
  - Cada ponto  $(x_t, y_t)$  é obtido alternando amostras condicionais
  - A trajetória forma uma cadeia que se desloca gradualmente pela distribuição
  - O padrão diagonal reflete a correlação entre  $X$  e  $Y$  na Normal bivariada
- **Painel direito:** *traces* das variáveis  $X_t$  e  $Y_t$  ao longo das iterações
  - As flutuações mostram dependência entre as amostras, típica de MCMC
  - A boa sobreposição e ausência de tendência indicam convergência
- **ESS (Effective Sample Size):** mede o tamanho efetivo de amostras independentes; diminui conforme aumenta a correlação entre variáveis
- **Correlação ( $\rho$ ):** valores altos geram forte dependência entre  $X$  e  $Y$ , reduzindo a eficiência da cadeia

# Amostrador de Gibbs

`z04_gibbs.py`



# Hamiltonian Monte Carlo (HMC)





# Hamiltonian Monte Carlo (HMC)

Teoria e aplicações

- O método Hamiltonian Monte Carlo (HMC) é uma extensão do Metropolis-Hastings que utiliza informações de gradiente para explorar o espaço de parâmetros de forma mais eficiente
- Ele introduz variáveis auxiliares de momento  $\mathbf{p}$  e define uma **energia Hamiltoniana**:

$$H(\mathbf{q}, \mathbf{p}) = U(\mathbf{q}) + K(\mathbf{p}),$$

onde  $U(\mathbf{q}) = -\log \pi(\mathbf{q})$  representa a energia potencial e  $K(\mathbf{p})$  a energia cinética



# Hamiltonian Monte Carlo (HMC)

Teoria e aplicações

- A dinâmica Hamiltoniana é simulada via o integrador *leapfrog*:

$$\begin{cases} \mathbf{p}_{t+\frac{\varepsilon}{2}} = \mathbf{p}_t + \frac{\varepsilon}{2} \nabla_{\mathbf{q}} \log \pi(\mathbf{q}_t) \\ \mathbf{q}_{t+\varepsilon} = \mathbf{q}_t + \varepsilon \mathbf{p}_{t+\frac{\varepsilon}{2}} \\ \mathbf{p}_{t+\varepsilon} = \mathbf{p}_{t+\frac{\varepsilon}{2}} + \frac{\varepsilon}{2} \nabla_{\mathbf{q}} \log \pi(\mathbf{q}_{t+\varepsilon}) \end{cases}$$

- Após  $L$  passos, a nova proposta  $(\mathbf{q}', \mathbf{p}')$  é aceita com probabilidade:

$$\alpha = \min(1, e^{-[H(\mathbf{q}', \mathbf{p}') - H(\mathbf{q}, \mathbf{p})]})$$

- O HMC move-se ao longo das curvas de nível da densidade, produzindo saltos longos e eficientes sem reduzir a taxa de aceitação
- É altamente eficiente em espaços contínuos e de alta dimensão, sendo a base de algoritmos modernos como o No-U-Turn Sampler (NUTS)



# Hamiltonian Monte Carlo (HMC)

## Teoria e aplicações

- O HMC é um método de amostragem que combina conceitos da **mecânica Hamiltoniana** com o **algoritmo de Metropolis**
- A ideia central é explorar o espaço de parâmetros utilizando a **dinâmica de um sistema físico**:
  - Cada posição  $q = (x, y)$  representa um estado do sistema
  - Um momento artificial  $p$  é sorteado de uma distribuição Normal
  - O par  $(q, p)$  evolui segundo as equações de Hamilton por  $L$  passos de tamanho  $\varepsilon$  (método *leapfrog*)
  - Ao final, a nova posição proposta  $q'$  é aceita com probabilidade  $\min(1, e^{-\Delta H})$ , onde  $\Delta H$  é a variação da energia total



# Hamiltonian Monte Carlo (HMC)

Teoria e aplicações

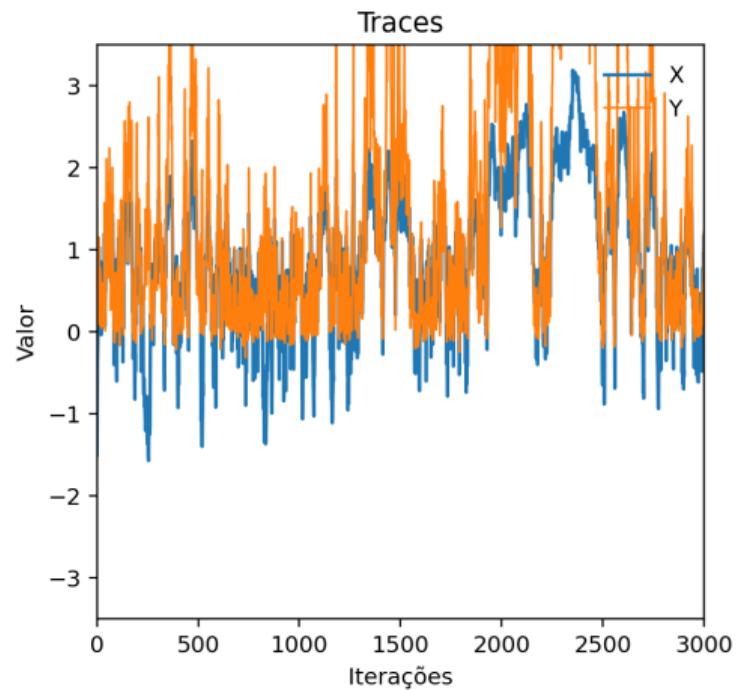
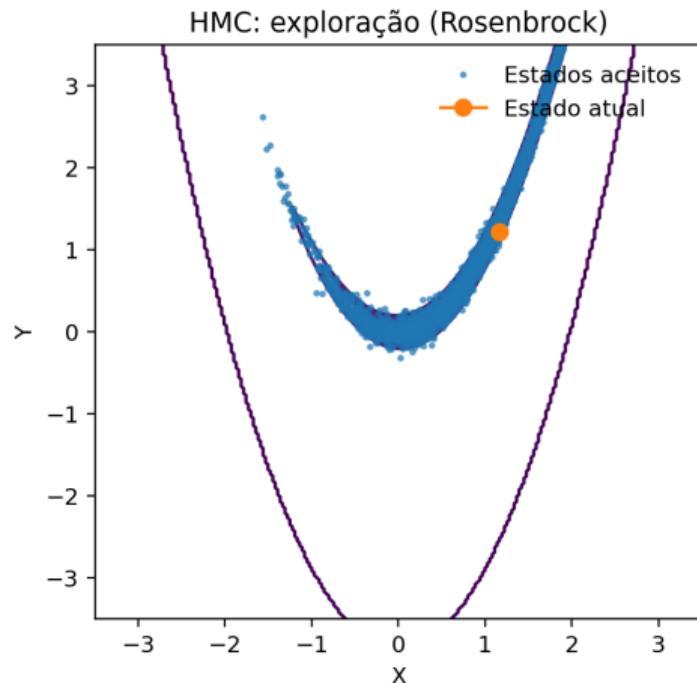
- O uso de gradientes da densidade permite movimentos longos e coerentes, evitando o comportamento aleatório do Metropolis tradicional
- O resultado é uma **exploração mais eficiente**, especialmente em distribuições curvas ou correlacionadas, como o vale de Rosenbrock



# Hamiltonian Monte Carlo (HMC)

Teoria e aplicações

HMC — Rosenbrock | acc≈0.979 | ESS\_X≈15 | ESS\_Y≈13 | eps=0.02, L=25





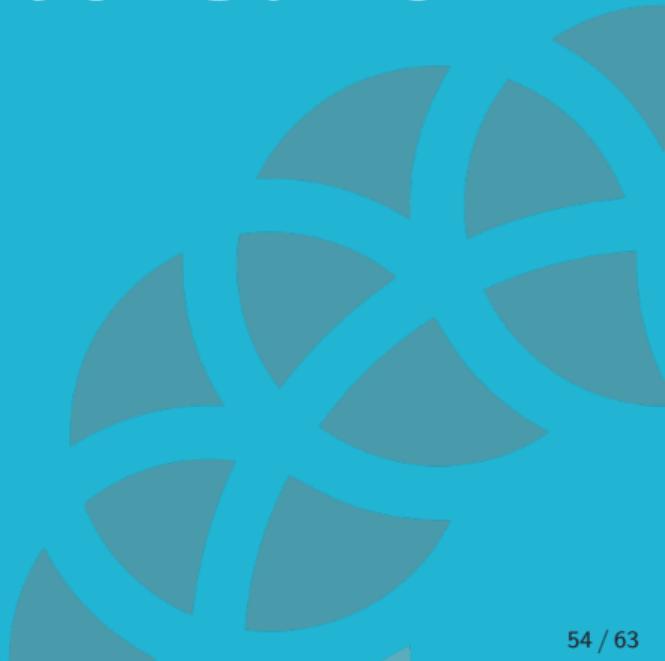
# Hamiltonian Monte Carlo (HMC)

Teoria e aplicações

- **Painel esquerdo:** contornos da densidade alvo e pontos amostrados
  - O HMC percorre o “vale” da distribuição de forma suave e contínua
  - Os pontos aceitos mostram uma boa cobertura da região de alta densidade
- **Painel direito:** trajetórias (*traces*) das variáveis  $x_t$  e  $y_t$ 
  - As flutuações são amplas e regulares, indicando boa mistura e independência entre amostras
- **Taxa de aceitação:** ideal entre 0.6 e 0.9 — valores altos indicam passos pequenos demais, e baixos sugerem integração instável
- **Variação de energia ( $|\Delta H|$ ):** deve ser pequena; controla a precisão do integrador *leapfrog*
- **ESS (Effective Sample Size):** mede a eficiência da cadeia. O HMC tende a produzir ESS muito maior que o Metropolis, pois as amostras são menos correlacionadas

# Hamiltonian Monte Carlo (HMC)

z05\_hmc.py





# Métodos MCMC

Características Principais — com Explicações

Metropolis	Gibbs Sampling	Hamiltonian Monte Carlo
<b>Ideia:</b> Gera uma proposta e decide aceitar ou rejeitar.	<b>Ideia:</b> Atualiza cada variável usando sua distribuição condicional.	<b>Ideia:</b> Usa gradientes para propor saltos mais eficientes no espaço.
<b>Vantagem:</b> Simples, flexível e fácil de implementar.	<b>Vantagem:</b> Sempre aceita amostras, garantindo estabilidade.	<b>Vantagem:</b> Excelente em alta dimensão, reduz autocorrelação.
<b>Limitação:</b> Pode ter alta rejeição e convergir lentamente.	<b>Limitação:</b> Requer condicionais conhecidas; pouco flexível.	<b>Limitação:</b> Complexo e computacionalmente custoso.



# Curiosidades

Aplicações que utilizam o Método Monte Carlo

- Wikipédia - Método Monte Carlo
- IBM - O que é a Simulação de Monte Carlo?
- Wikipédia - Método Monte Carlo e aplicações
- Medium - aplicações e códigos Python



# Obrigado!

Thanks! / ¡Gracias!



**Leonardo C. Santos**

[www.lattes.cnpq.br/5620610314140397](http://www.lattes.cnpq.br/5620610314140397)

[leonardo.cunha.santos@usp.br](mailto:leonardo.cunha.santos@usp.br)



**Alex Ceccon**

<http://www.lattes.cnpq.br/5978571904940435>

[alexceconi@gmail.com](mailto:alexceconi@gmail.com)



# Show me the Code!

Python & R



Todos os códigos apresentados  
estão no link abaixo!

**MCMC\_e\_Amostrador\_de\_Gibbs**



# Referências I

- Dellaportas, Petros and Gareth O. Roberts (2003). "An Introduction to MCMC". In: *Spatial Statistics and Computational Methods*. Ed. by Jesper Møller. Available in: SpringerLink. New York: Springer Science+Business Media, pp. 1–41. DOI: [10.1007/978-1-4613-0023-9\\_1](https://doi.org/10.1007/978-1-4613-0023-9_1).
- Gentle, James E. (2003). *Random Number Generation and Monte Carlo Methods*. 2nd ed. Statistics and Computing. New York: Springer. ISBN: 9780387001777.



# Referências II

- Hinojosa Luna, Adrian (2019). *Introdução aos Métodos de Monte Carlo Avançados*. Tech. rep. Notas de aula. Departamento de Estatística, Universidade Federal de Minas Gerais. URL: [https://www.est.ufmg.br/portal/wp-content/uploads/2023/01/RTE\\_01\\_2019.pdf](https://www.est.ufmg.br/portal/wp-content/uploads/2023/01/RTE_01_2019.pdf).
- Link, William A. and Mitchell J. Eaton (2012). "On Thinning of Chains in MCMC". In: *Methods in Ecology and Evolution* 3.1, pp. 112–115. DOI: 10.1111/j.2041-210X.2011.00131.x. URL: <https://doi.org/10.1111/j.2041-210X.2011.00131.x>.



# Referências III

- Luengo, David et al. (2020). “A Survey of Monte Carlo Methods for Parameter Estimation”. In: *EURASIP Journal on Advances in Signal Processing* 2020.25. Open Access. DOI: 10.1186/s13634-020-00675-6. URL: <https://doi.org/10.1186/s13634-020-00675-6>.
- *MCMC — Curso CE089, 08\_MCMC* (2025). [http://cursos.leg.ufpr.br/ce089/08\\_MCMC.html](http://cursos.leg.ufpr.br/ce089/08_MCMC.html). Acessado em: 13 12 2025.



# Referências IV

- Prado, Estevão Batista do (2016). “Comparação de estratégias de geração de propostas no algoritmo Metropolis-Hastings para um modelo Poisson log-linear”. Dissertação (Mestrado em Estatística). Belo Horizonte, MG, Brasil: Departamento de Estatística, Instituto de Ciências Exatas.
- Ravenzwaaij, Don van, Pete Cassey, and Scott D. Brown (2018). “A Simple Introduction to Markov Chain Monte–Carlo Sampling”. In: *Psychonomic Bulletin & Review* 25.1, pp. 143–154. DOI: 10.3758/s13423-016-1015-8. URL: <https://doi.org/10.3758/s13423-016-1015-8>.



# Referências V

- Robert, Christian P. and George Casella (2010). *Introducing Monte Carlo Methods with R*. Use R! New York, NY: Springer. ISBN: 978-1-4419-1575-7. DOI: 10.1007/978-1-4419-1576-4. URL: <https://doi.org/10.1007/978-1-4419-1576-4>.
- — (2011). “A Short History of Markov Chain Monte Carlo: Subjective Recollections from Incomplete Data”. In: *Statistical Science* 26.1, pp. 102–115. DOI: 10.1214/10-STS351. URL: <https://doi.org/10.1214/10-STS351>.