

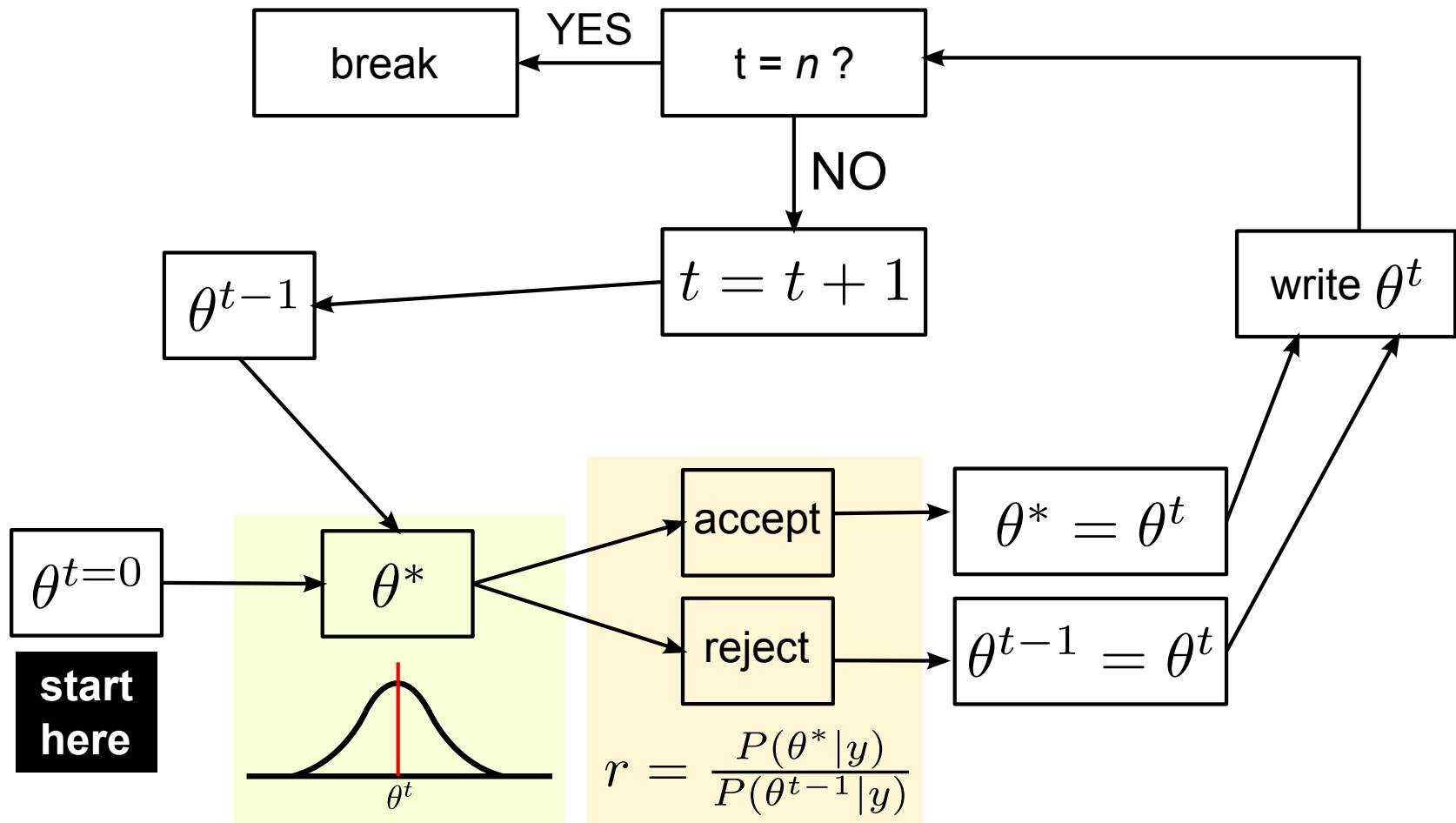
Caminhar é preciso

Um foco importante de pesquisa sobre o MCMC são as estratégias de proposta de novos valores para os parâmetros.

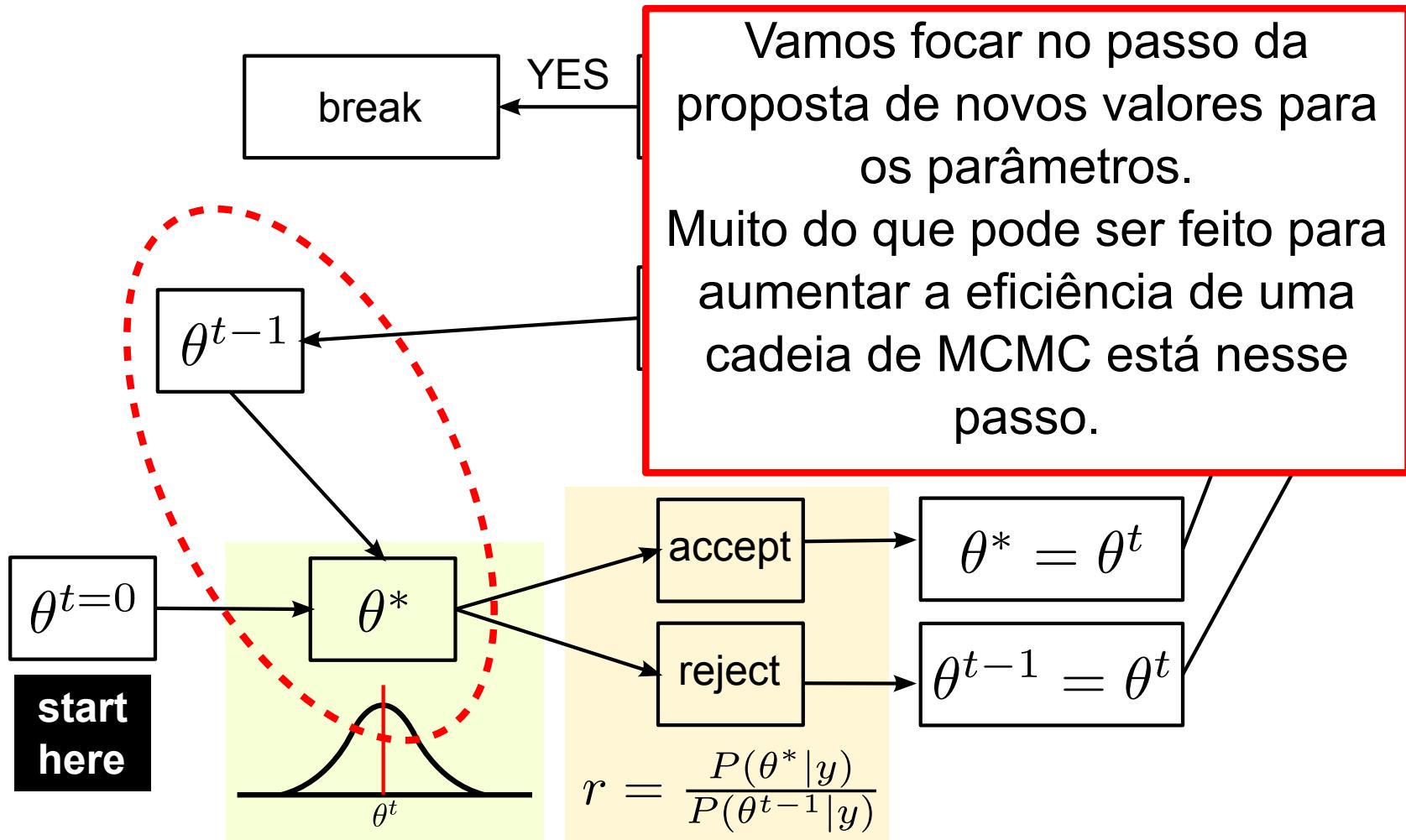
Quando propostas não são aceitas por muitas gerações nós estamos desperdiçando tempo. Isso pode fazer grande diferença quando a sua cadeia de MCMC já leva dias para rodar.



Caminhar é preciso

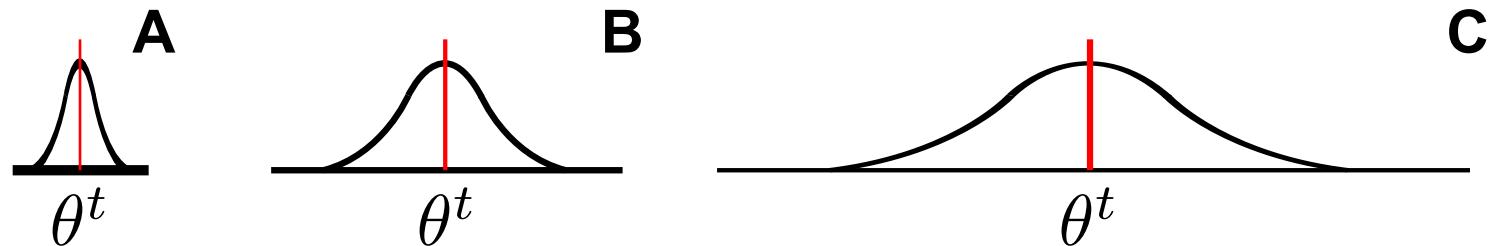


Caminhar é preciso



Caminhar é preciso

A largura de cada passo influencia a frequência do aceite das propostas (*acceptance ratio*):



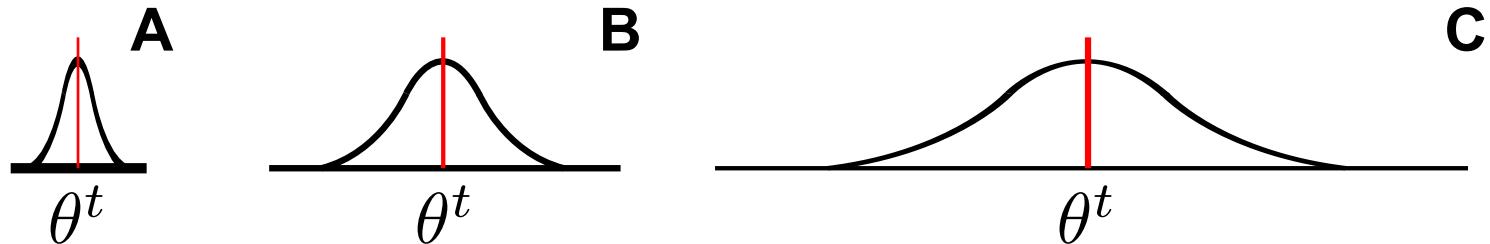
A -- Alta acceptance ratio, baixo mixing.

B -- Média acceptance ratio, bom mixing.

C -- Baixa acceptance ratio, baixo mixing.

Caminhar é preciso

A largura de cada passo influencia a frequência do aceite das propostas (*acceptance ratio*):



O cálculo da acceptance ratio é simples:

$$\frac{\text{número de passo aceitos}}{\text{número de passo rejeitados}}$$

O número mágico

Valor ótimo do acceptance ratio: **0.234**

"This means that, at least under their [Roberts et al., 1997] (strong) assumptions, the optimal acceptance probability is 23.4%, which leads to the fastest limiting speed regardless of the target density."

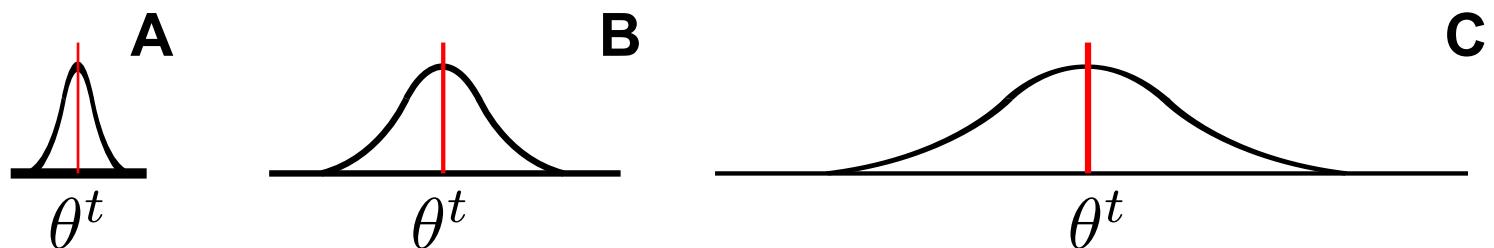
Rosenthal J. S. (2014)

Roberts, G. O., Gelman, A., and Gilks, W. R. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. *The Annals of Applied Probability*, 7:110–120.

Caminhar é preciso

Queremos ajustar as distribuições de proposta de forma que o acceptance ratio seja próximo de 0.234.

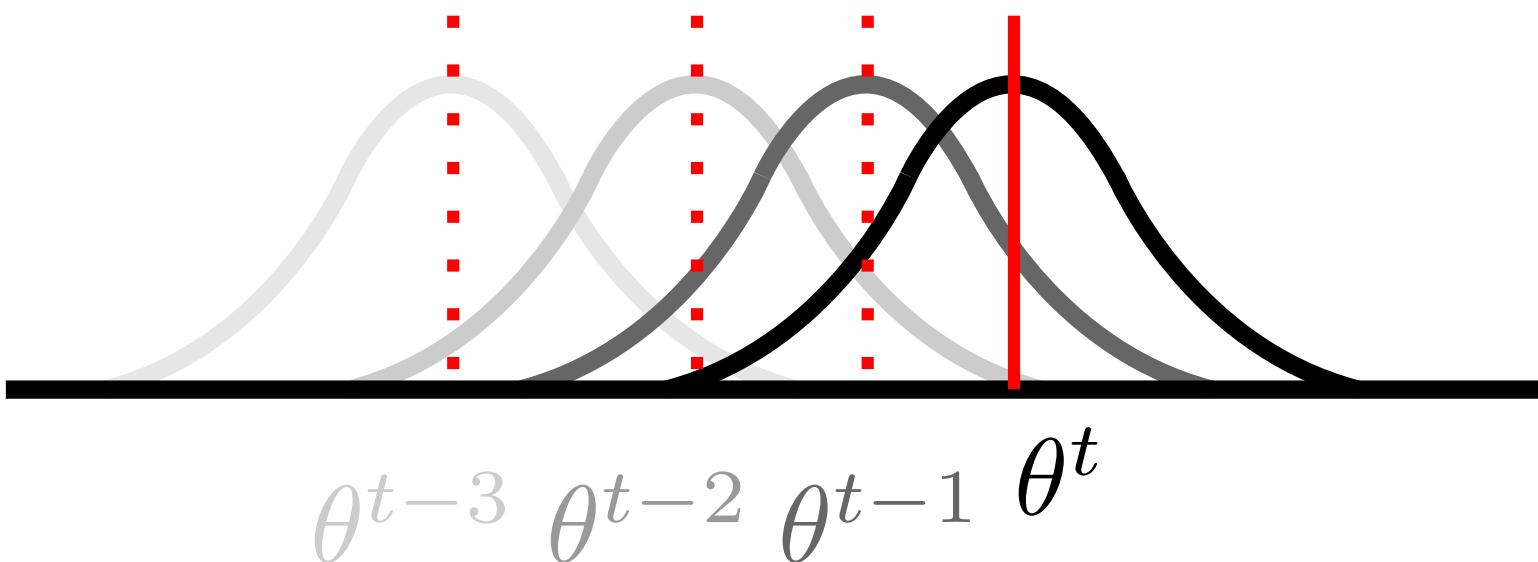
Infelizmente esse ajuste é empírico. Para chegar em uma boa distribuição de proposta é necessário fazer muitas simulações.



Caminhar é preciso

Sliding window

As propostas abaixo são chamas de sliding window, pois são centradas no valor atual θ^t . Ou seja, a cada passo elas são centradas novamente.

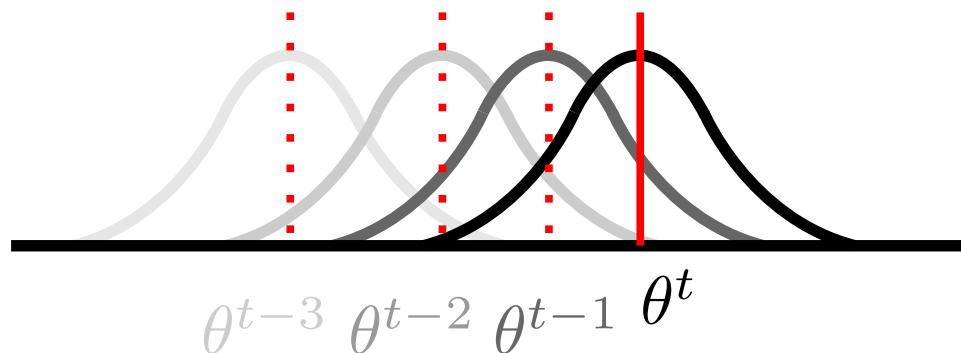


Caminhar é preciso

Sliding window

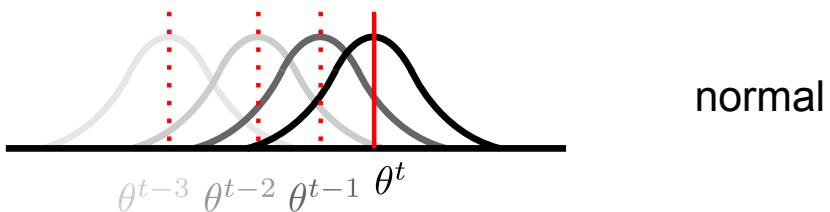
As propostas abaixo são chamas de sliding window, pois são centradas no valor atual θ^t . Ou seja, a cada passo elas são centradas novamente.

Importante lembrar que o próximo passo na cadeia somente depende do passo imediatamente anterior.

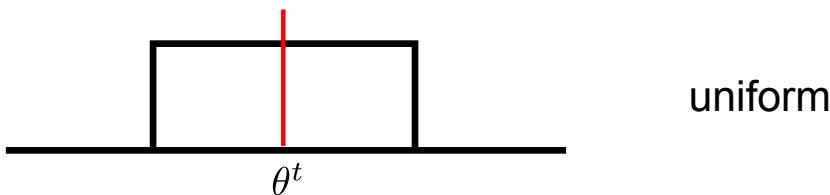


Caminhar é preciso

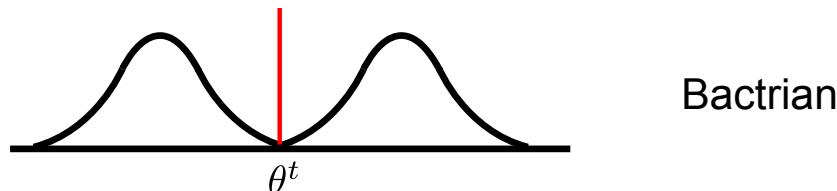
O esquema de proposta de sliding window pode ter diferentes distribuições.



normal



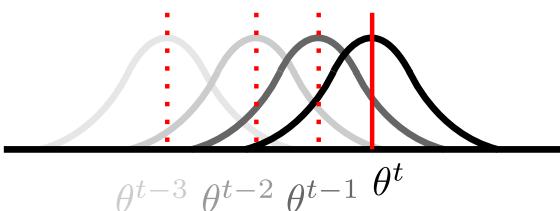
uniform



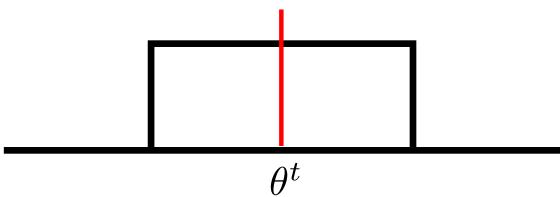
Bactrian

Caminhar é preciso

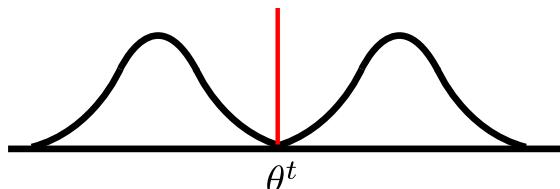
O esquema de proposta de sliding window pode ter diferentes distribuições.



normal



uniform



Bactrian

Estudo recente na PNAS aponta que a distribuição 'Bactrian' é a mais eficiente quando comparada com a uniforme e normal (todas otimizadas).

Estudo também sugere que a eficiência de diferentes 'proposal kernels' não é equivalente, contrário do que se assumia.

Yang and Rodríguez, PNAS, 2013

Caminhar é preciso

"We develop a Bactrian kernel, which uses a mixture of two Gaussian distributions and which has the overall shape of a Bactrian camel, to reduce the positive autocorrelation (p 1 in Eq. 5)."

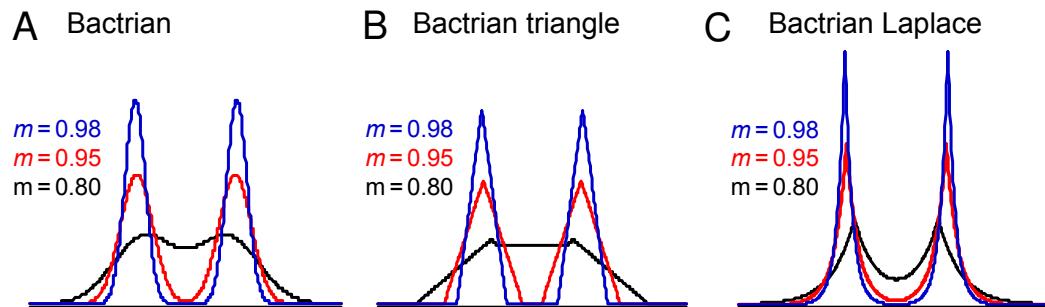
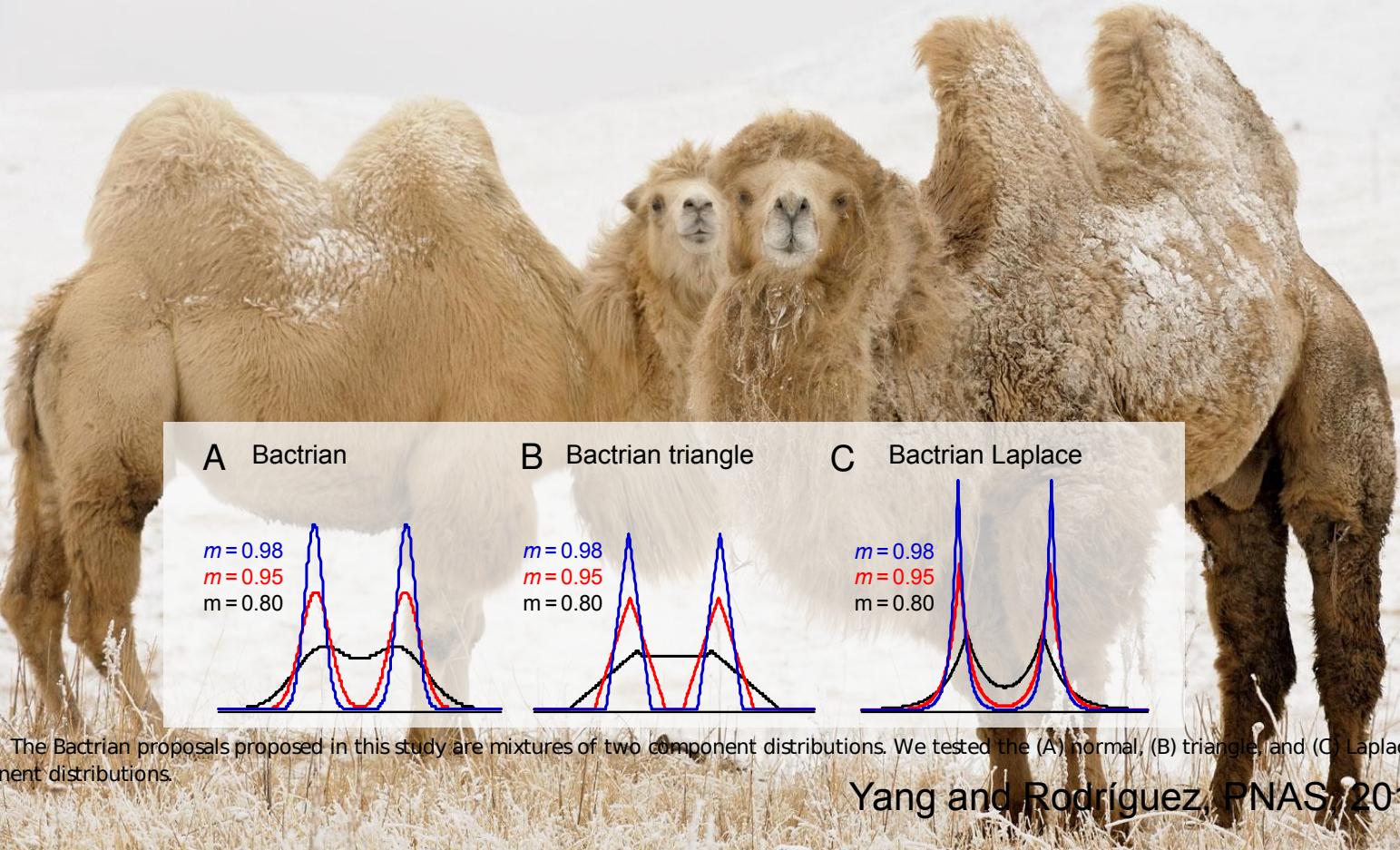


Fig. 1. The Bactrian proposals proposed in this study are mixtures of two component distributions. We tested the (A) normal, (B) triangle, and (C) Laplace component distributions.

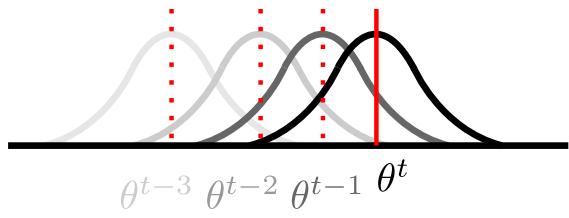
Yang and Rodríguez, PNAS, 2013

Caminhar é preciso

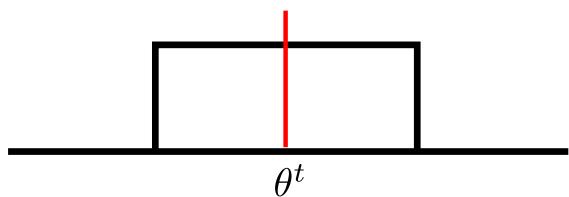


Caminhar é preciso

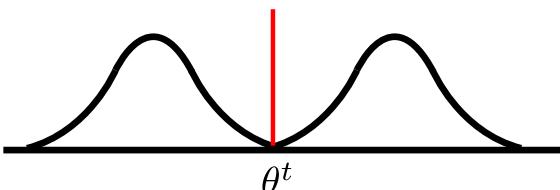
Distribuições de propostas são sempre simétricas?



O algoritmo de Metropolis assume que a transição entre os estados da cadeia de Markov do MCMC seja simétrica.



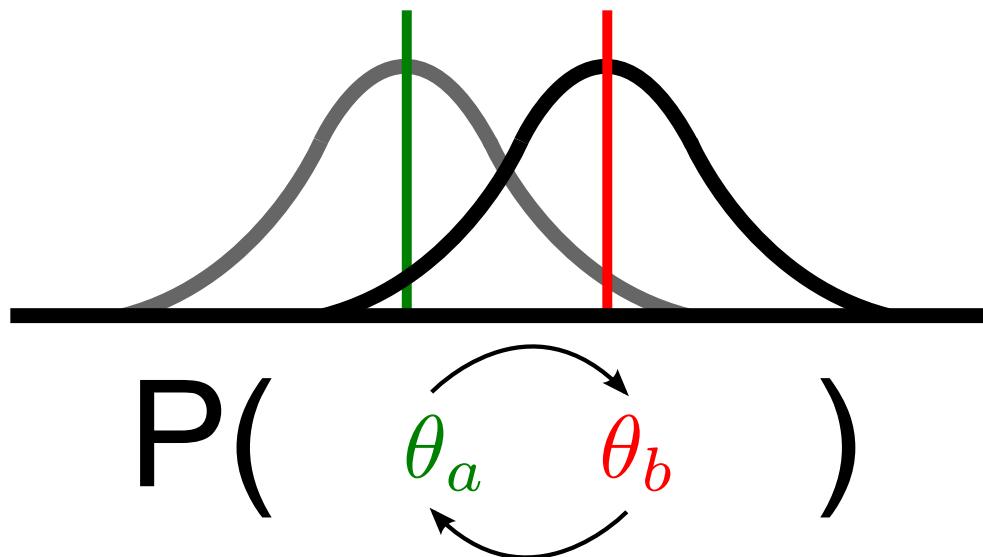
Ou seja, que a chance de pular de A para B seja igual a chance de fazer o reverso.



$$J_t(\theta_a | \theta_b) = J_t(\theta_b | \theta_a)$$

Caminhar é preciso

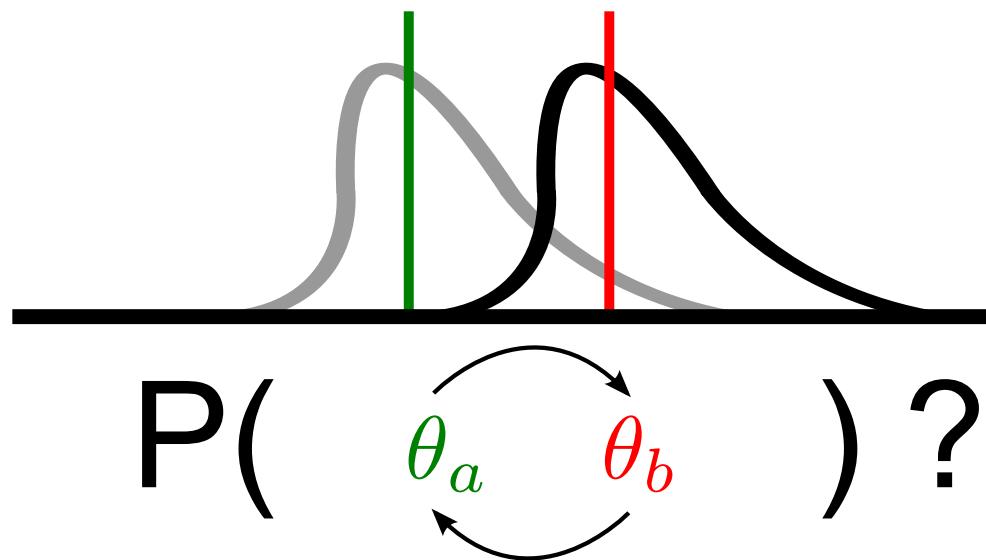
Simetria da distribuição



$$J_t(\theta_a | \theta_b) = J_t(\theta_b | \theta_a)$$

Caminhar é preciso

O que acontece quando a distribuição de proposta é assimétrica?



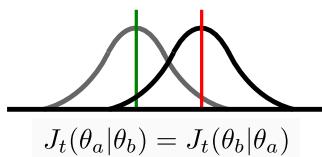
$$J_t(\theta_a | \theta_b) \neq J_t(\theta_b | \theta_a)$$

Caminhar é preciso

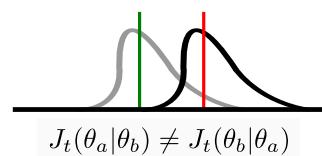
O que acontece quando a distribuição de proposta é assimétrica?

O pressuposto do *algoritmo de Metropolis* é violado.

Precisamos utilizar uma extensão do algoritmo que é chamada de *Metropolis-Hastings*.



algoritmo de *Metropolis*

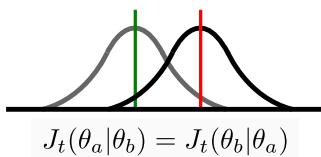


algoritmo de *Metropolis-Hastings*

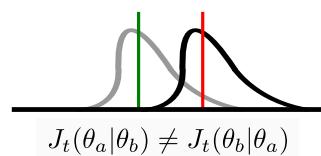
Caminhar é preciso

Para ser mais preciso o pressuposto de simetria da transição é associada à **cadeia de Markov**.

Precisamos disso para garantir a reversibilidade do processo.



algoritmo de *Metropolis*

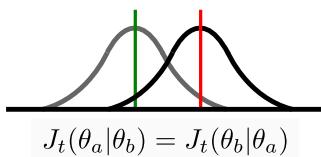


algoritmo de *Metropolis-Hastings*

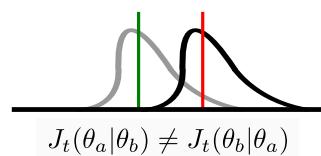
Caminhar é preciso

O algoritmo de Metropolis-Hastings adiciona um termo para "ajustar" a probabilidade de transição entre os estados.

Esse termo é chamado de "razão de Hastings" ou '*Hastings ratio*'.

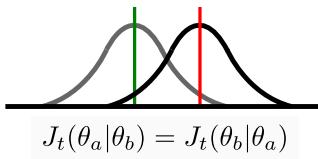


algoritmo de *Metropolis*



algoritmo de *Metropolis-Hastings*

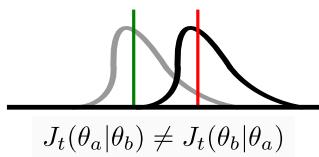
Caminhar é preciso



algoritmo de *Metropolis*

$$J_t(\theta_a|\theta_b) = J_t(\theta_b|\theta_a)$$

$$r = \frac{P(\theta^*|y)}{P(\theta^{t-1}|y)}$$

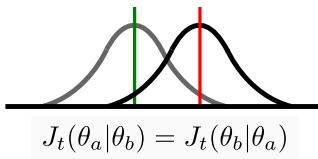


algoritmo de *Metropolis-Hastings*

$$J_t(\theta_a|\theta_b) \neq J_t(\theta_b|\theta_a)$$

$$r = \frac{P(\theta^*|y)}{P(\theta^{t-1}|y)} \frac{J_t(\theta^*|\theta^{t-1})}{J_t(\theta^{t-1}|\theta^*)}$$

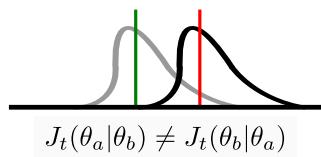
Caminhar é preciso



algoritmo de *Metropolis*

odds ratio

$$r = \frac{P(\theta^* | y)}{P(\theta^{t-1} | y)}$$

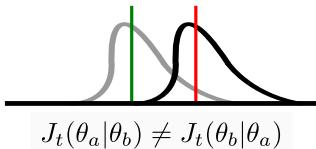


algoritmo de *Metropolis-Hastings*

odds ratio

$$r = \frac{P(\theta^* | y)}{P(\theta^{t-1} | y)} \cdot \frac{J_t(\theta^* | \theta^{t-1})}{J_t(\theta^{t-1} | \theta^*)} \cdot \textcolor{red}{\frac{Hastings}{ratio}}$$

Caminhar é preciso



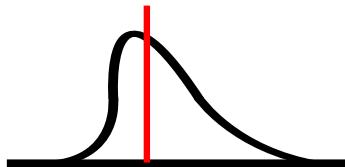
algoritmo de *Metropolis-Hastings*

odds ratio $r = \frac{P(\theta^*|y)}{P(\theta^{t-1}|y)} \cdot \frac{J_t(\theta^*|\theta^{t-1})}{J_t(\theta^{t-1}|\theta^*)}$

Hastings ratio

$$\frac{\text{probabilidade de } \theta^* \mid \text{distribuição centrada em } \theta^{t-1}}{\text{probabilidade de } \theta^{t-1} \mid \text{distribuição centrada em } \theta^*}$$

Caminhar é preciso



```
## Simulate proposal distribution.  
rlognorm(1000, mu=curr, sigma=1.5)
```

```
## Proposal:
```

```
## curr = current value.
```

```
## prop = proposed value.
```

```
prop <- rlognorm(1, mu=curr, sigma=1.5)
```

```
## Hastings ratio:
```

```
A <- dlognorm(prop, mu=curr, sigma=1.5)
```

```
B <- dlognorm(curr, mu=prop, sigma=1.5)
```

```
hast <- A / B
```

$$\frac{J_t(\theta^* | \theta^{t-1})}{J_t(\theta^{t-1} | \theta^*)}$$

Caminhar é preciso

Observação (1): O algoritmo de *Metropolis* é um caso especial de *Metropolis-Hastings* quando a transição é simétrica.

$$\begin{aligned} r &= \frac{P(\theta^* | y)}{P(\theta^{t-1} | y)} \frac{J_t(\theta^* | \theta^{t-1})}{J_t(\theta^{t-1} | \theta^*)} \\ &= \frac{P(\theta^* | y)}{P(\theta^{t-1} | y)} \frac{x}{x} \\ &= \frac{P(\theta^* | y)}{P(\theta^{t-1} | y)} \end{aligned}$$

Caminhar é preciso

Observação (2): Geralmente a função de likelihood e a razão de Hastings são derivadas algebricamente e não lembram as equações originais.

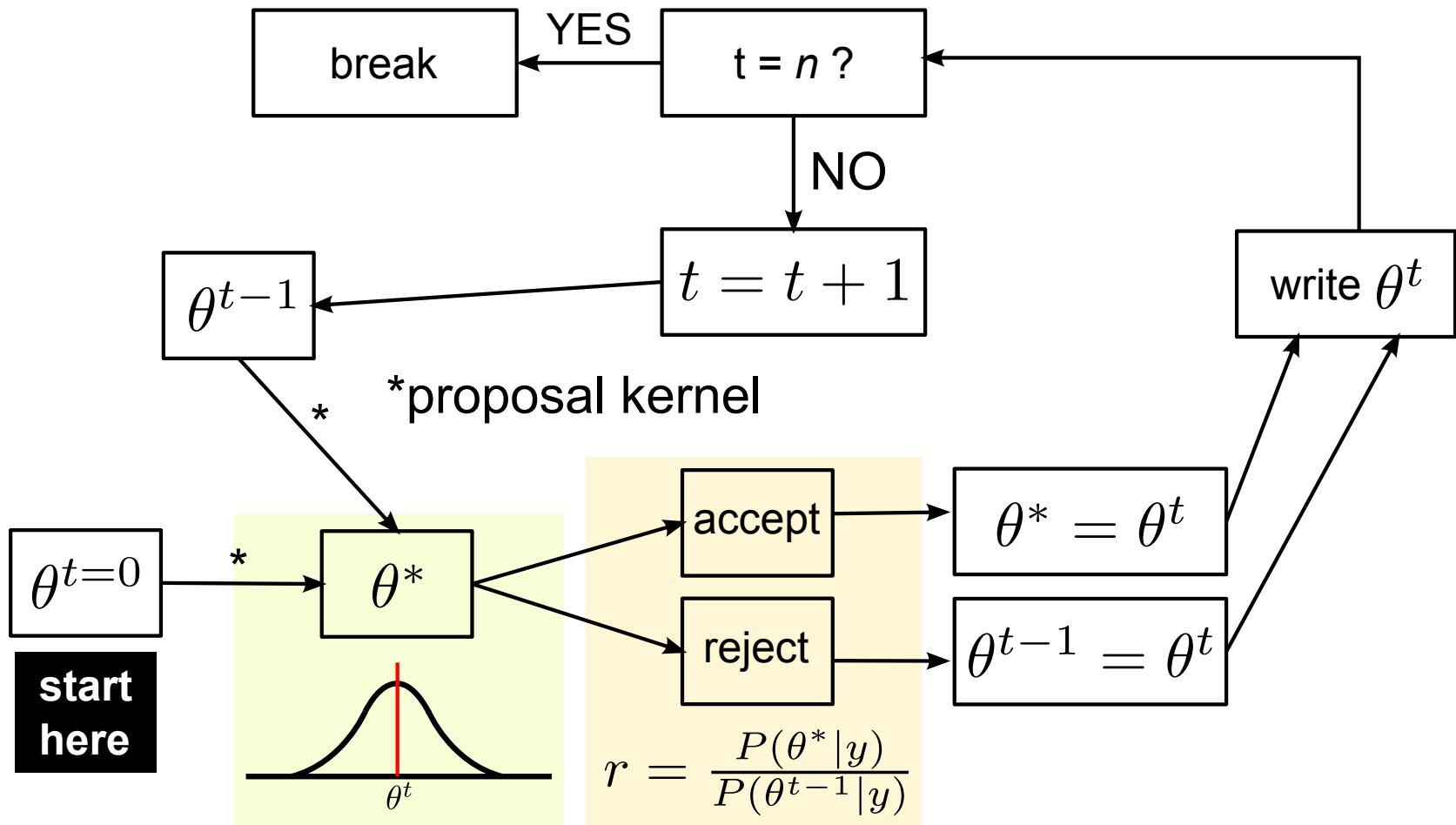
Mas isso não é necessário para a implementação.

$$r = \frac{P(\theta^* | y)}{P(\theta^{t-1} | y)} \frac{J_t(\theta^* | \theta^{t-1})}{J_t(\theta^{t-1} | \theta^*)}$$

Some magic . . .

$$r = 42 !?$$

Parabéns!!



A close-up, profile shot of a man's face, looking upwards and slightly to the right. He has dark hair, a mustache, and is wearing a light-colored shirt. The lighting is dramatic, with strong highlights on his forehead, nose, and cheek, while the rest of his face and the background are in deep shadow.

I know Markov chain Monte Carlo



Now show me.