Fundação Getulio Vargas
Estatística Computacional
Prof. Eduardo Mendes
Aluno: Luiz Fernando Maia

Relatório da replicação do artigo "Slice Sampling" de Radford M. Neal

1 Introdução

O intuito deste relatório é apresentar métodos de slice sampling descritos em Neal (2003) e exibir uma replicação da demonstração presente nesse artigo.

A Seção 2 discute possíveis benefícios desses métodos em relação à outros métodos baseados em cadeias de Markov, a Seção 3 apresenta a ideia geral dos métodos de slice sampling enquanto que as Seções 4 e 5 exibem estratégias para utilizar estes métodos em distribuições univariadas e multivariadas, respectivamente. Duas extensões dos métodos de slice sampling que evitam um corpotamento de passeio aleatório são aprentadas nas Seções 6 e 7, a Seção 8 mostra a demonstração presente em Neal (2003) que expõe alguns dos benefícios do slice sampling além de uma replicação desta demonstração. Finalmente, na Seção 9 são apresentadas as conclusões.

2 Motivação

Métodos de cadeias de Markov como o amostrador de Gibbs (GEMAN; GEMAN, 1984) e o algoritmo de Metropolis-Hastings (HASTINGS, 1970) podem ser utilizados para amostrar de muitas das distribuições complexas encontradas na estatística. Aplicações destes métodos, em geral, envolvem amostrar valores de distribuições posterioris de parâmetros de diversos modelos.

No entanto, para implementação do Gibbs, é necessário saber gerar valores das distribuições condicionais completas e para o Metropolis é preciso encontrar uma proposta adequada para que a amostragem seja eficiente. Essas necessidades limitam o uso destes métodos e o desenvolvimento de softwares que automatizem a construção de cadeias de Markov a partir da especificação de modelos.

Em geral, esses amostradores são ineficientes pois eles propõem mudanças que não se adaptam bem às propriedades locais da função, o que faz com que estas mudanças devem ser feitas em pequenos passos e estes pequenos passos geralmente são passeios aleatórios que precisam de muitos mais passos para realizar um deslocamento que poderia ser feito mais facilmente se estes passos fossem feitos consistentemente em uma direção.

Os métodos da classe slicing sample podem ser aplicados em uma grande variedade de distribuições. Formulações simples do slice sampling univariado são uma alternativa para o amostrador de Gibbs pois evitam a necessidade de amostrar das distribuições condicionais e um grande benefício desses métodos é que eles podem adaptativamente mudar a escala das mudanças, o que os torna mais fáceis de serem ajustados do que os algoritmos de Metropolis e evita problemas que podem surgir quando a escala adequada das mudanças propostas varia na distribuição.

Além disso, para muitas distribuições, os métodos de slice sampling não vão amostrar de maneira mais eficiente do que o Gibbs ou um Metropolis com proposta adequada mas, em geral, vão demandar menos esforço para implementação e ajuste. No entanto, métodos mais complexos de slice sampling podem se adaptar às dependências entre as variáveis, no caso de uma distribuição multivariada, e permitem mudanças maiores do que seria possível com o Gibbs ou o Metropolis, o que faz com que eles sejam mais eficientes em algumas distribuições.

3 A ideia do slice sampling

Suponha que queremos gerar valores de uma distribuição de uma variável x, que tem suporte em algum subconjunto do $\mathbb{R}^n$ e densidade proporcional a $f(x)$. Para realizar isto, podemos amostrar uniformemente da região $(n+1)$-dimensional abaixo do gráfico de $f(x)$. Formalizamos esta ideia introduzindo y, uma variável auxiliar e definindo a distribuição conjunta de x e y como sendo uniforme na região $U = \{(x, y) : 0 < y < f(x)\}$. Desse modo,

$$p(x, y) = \begin{cases} 1/Z, \text{ se } 0 < y < f(x) \\ 0, \text{ caso contrário} \end{cases}$$

em que $Z = \int f(x)dx$, portanto,

$$p(x) = \int_0^{f(x)} (1/Z)dy = f(x)/Z$$

que é a distribuição de interesse.

Para amostrar de x, podemos amostrar da conjunta (x, y) e ignorar y. No entanto, gerar pontos independentes uniformemente de U pode não ser fácil, podemos então definir uma cadeia de Markov que convergirá para esta distribuição uniforme. Uma opção é o amostrador de Gibbs: amostramos alternadamente de $Y \,|\, X$ que é uniforme em $(0, f(x))$ e de $X \,|\, Y$ que é uniforme em $S = \{x : y < f(x)\}$, chamamos esta região S de slice. Porém, gerar um ponto de S pode ainda ser difícil, neste caso, podemos utilizar algum valor de x que deixe a distribuição uniforme em S invariante. As próximas seções apresentarão algumas estratégias para realizar essas amostragens.

4 Slice sampling univariado

O caso mais simples do slice sampling é quando apenas uma variável está sendo atualizada. Esse é o caso quando a distribuição de interesse é univariada ou quando cada variável de uma distribuição multivariada está sendo atualizada de cada vez.

Seja x a variável a ser atualizada, com densidade proporcional a $f(x)$, para atualizar o valor atual x_0 por um novo valor x_1, realizamos o seguinte procedimento:

(a) Amostramos y uniformemente de $(0, f(x_0))$, definindo o slice $S = \{x : y < f(x)\}$.

(b) Encontramos um intervalo, $I = (L, R)$, ao redor de x_0 que contenha todo, ou muito, do slice.

(c) Amostramos x_1 da parte do slice deste intervalo.

No primeiro passo, ao invés de computar $f(x)$, podemos utilizar $g(x) = log(f(x))$ para evitar problemas de underflow. Neste caso, utilizamos a variável auxiliar $z = log(y) = g(x_0) - e$ em que $e \sim Exp(1)$ e definimos $S = \{x : z < g(x)\}$. Os passos (b) e (c) podem ser implementados de várias maneiras desde que a a cadeia de Markov resultante deixe a distribuição definida por $f(x)$ invariante. As próximas subseções descreverão algumas estratégias para estes dois últimos passos.

4.1 Encontrado um intervalo apropriado

Nessa etapa, o objetivo é encontrar um intervalo $I = (L, R)$ que contém x_0 e em que será amostrado x_1, queremos este intervalo contenha tanto quanto for factível do slice para que

x_1 seja o mais diferente possível de x_0 mas também é de interesse que este intervalo não seja muito maior do que o slice pois isso irá afetar negativamente a eficiência do próximo passo.

Existem diversas estratégias para encontrar um intervalo:

1. Idealmente, podemos fazer $L = \inf(S)$ e $R = \sup(S)$, ou seja, I é o menor intervalo que contém S.
2. Se o suporte de x é limitado, podemos definir I como este suporte.
3. Dada uma estimativa, w, do tamanho de S, podemos escolher aleatoriamente um intervalo inicial de tamanho w contendo x_0 e, se necessário, expandi-lo em passos de tamanho w até que os extremos do intervalo estejam fora do slice (procedimento "stepping out").
4. Similarmente, podemos escolher aleatoriamente um intervalo de tamanho w e dobrar este intervalo até que os extremos estejam fora do slice (procedimento "doubling").

Para cada uma dessas estratégias, também precisamos encontrar o conjunto A de sucessores aceitáveis de x_0 que é dado por:

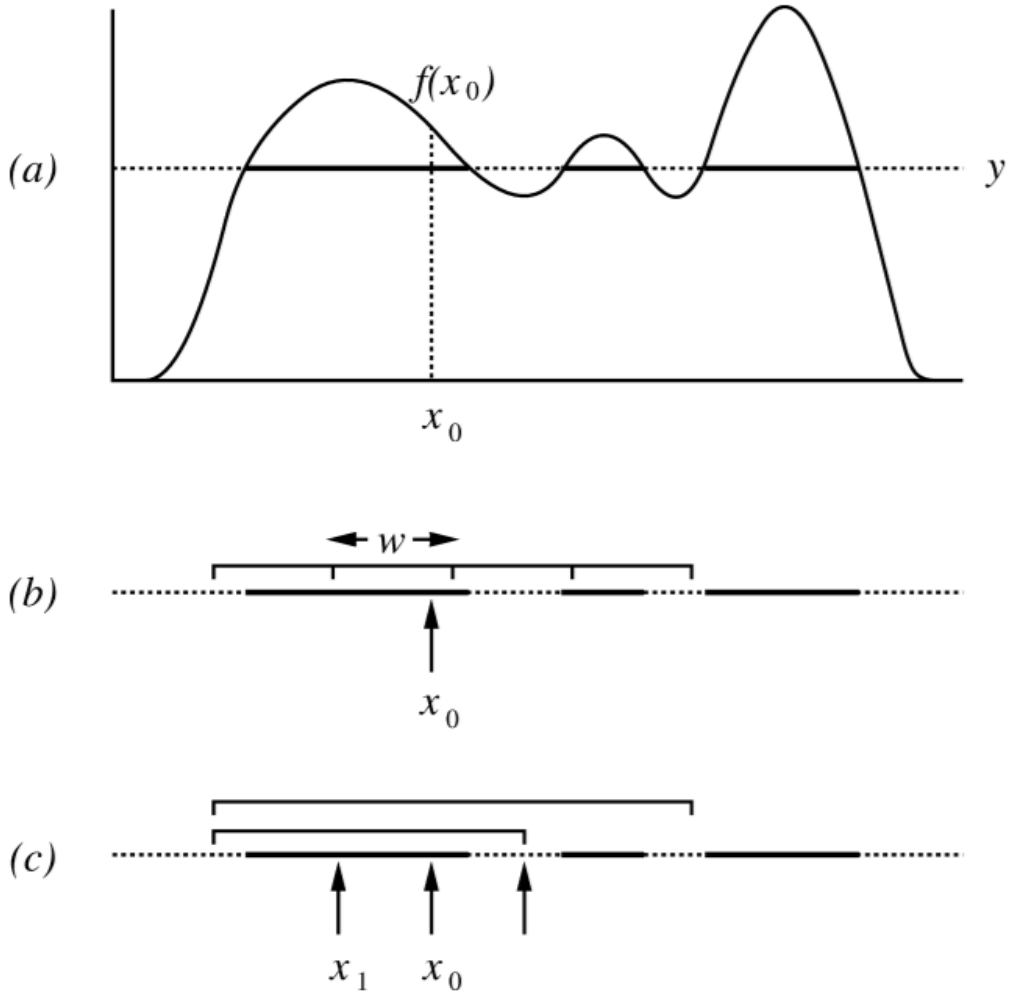
$$A = \{x : x \in S \cap I \text{ e } P(\text{Selecionar } I \mid \text{No estado } x) = P(\text{Selecionar } I \mid \text{No estado } x_0).$$

Ou seja, A é o conjunto de estados dos quais seria tão provável escolher I quanto é escolher I do estado atual. Para (1) e (2), temos que $A = S$ visto que todo slice está no intervalo, para (3), $A = S \cap I$ e para (4) é necessário efetuar um teste para verificar se um estado está em A.

A estratégia (1) é interessante quando todas as soluções de $f(x) = y$ podem ser encontradas analiticamente ou por métodos numéricos. No entanto, isto geralmente não acontece, em alguns casos, até o número de intervalos disjuntos que compõem S é difícil de se determinar. A estratégia (2) pode ser empregada utilizando transformações no suporte de x mas se o slice é usualmente muito menor do que o suporte, o algoritmo será ineficiente.

Por sua vez, o procedimento "stepping out", que é exemplificado e descrito na Figura 1, é apropriado para qualquer distribuição desde que tenhamos uma estimativa razoavél de w. Finalmente, a Figura 2 apresenta o procedimento "doubling" que pode expandir o intervalo mais rapidamento do que o "stepping out", logo, pode ser mais eficiente quando w é pequeno.

Figura 1: Atualização do slice sampling univariado utilizando os procedimentos "stepping out" e "shrinkage"



Um novo ponto, x_1, é selecionado a partir de x_0 em três passos. (a) O slice, indicado em negrito, é definido amostrando-se uniformemente de $(0, f(x_0))$. (b) Um intervalo de largura w é posicionado aleatoriamente ao redor de x_0 e expandido em passos de tamanho w até que ambos extremos estejam fora do slice. (c) Um novo ponto, x_1, é encontrado amostrando-se uniformemente desse intervalo até que um ponto dentro do slice é encontrado. Pontos amostrados que estejam fora do slice são utilizados para encolher o intervalo. Fonte: Neal (2003).

4.2 Amostrando da parte do slice dentro do intervalo

Dado um intervalo $I = (L, R)$ que contenha x_0, o último passo do slice sampling univariado é amostrar o novo ponto, x_1 da parte do slice deste intervalo. Dois métodos podem ser utilizados para isso:

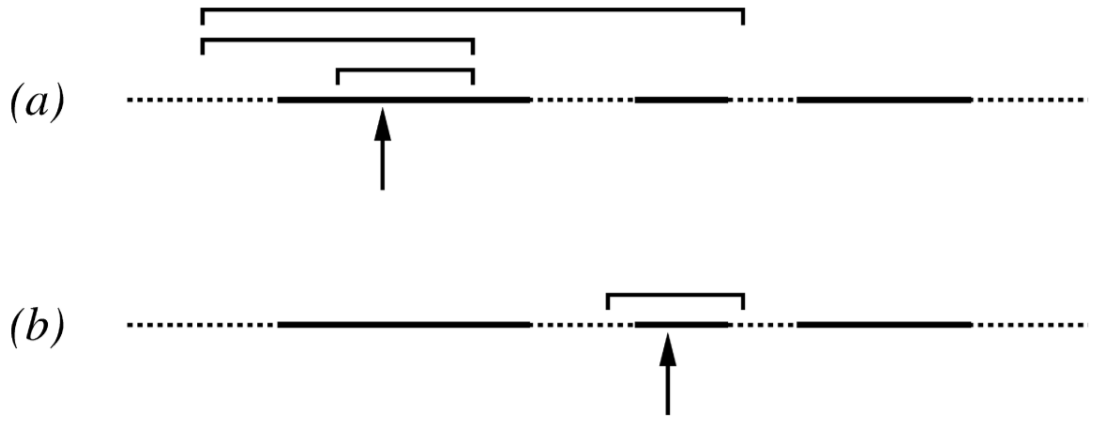
(i) Amostrar uniformemente de I até que seja selecionado um ponto de A.

(ii) Amostrar uniformemente de um intervalo que inicialmente é igual a I, e que é encolhido cada vez que um ponto selecionado não é de A, até que um ponto de A é encontrado, este é o procedimento "shrinkage".

Em geral, o método (ii) é mais adequado pois garante que o número de pontos amostrados não será muito grande. Além disso, um valor muito alto de w, a estimativa do tamanho do slice, fará com que o número esperado de pontos amostrados seja grande enquanto que um valor baixo fará com que x_1 não seja muito distante de x_0, o que aumentaria a correlação da cadeia.

Como dito anteriormente, para testar se o ponto amostrado no procedimento "doubling" é aceito, precisamos realizar um teste com o objetivo de que a cadeia resultante seja reversível: começamos o "doubling" de trás para frente e verificamos se nenhum dos intervalos tem extremidades fora do slice, o que terminaria o procedimento prematuramente. Por exemplo, na Figura 2, se o ponto indicado em (b) for gerado pelo intervalo de (a), este ponto não seria aceito e faríamos $x_1 = x_0$.

Figura 2: O procedimento "doubling"



Em (a), o intervalo inicial é dobrado duas vezes até que ambos os extremos estejam fora do slice. Em (b), onde o estado inicial é diferente, e os extremos já estão fora do slice, nada é feito. Fonte: Neal (2003).

5 Slice sampling multivariado

Ao invés de utilizar técnicas univariadas para cada componente do vetor $x = (x_1, ..., x_n)$, podemos generalizar estes métodos. Uma possível generalização é substituir o intervalo $I = (L, R)$ por um hiper-retângulo $H = \{x : L_i < x_i < R_i \text{ para todos } i = 1, ..., n\}$.

Desse modo, o procedimento para encontrar o novo valor, $x_1 = (x_{1,1}, ..., x_{1,n})$, a partir de $x_0 = (x_{0,1}, ..., x_{0,n})$ é:

(a) Amostrar y uniformemente de $(0, f(x_0))$, definindo o slice $S = \{x : y < f(x)\}$.

(b) Encontrar um hiper-retângulo $H = (L_1, R_1) \times ... \times (L_n, R_n)$, ao redor de x_0, que contenha, preferencialmente, boa parte do slice.

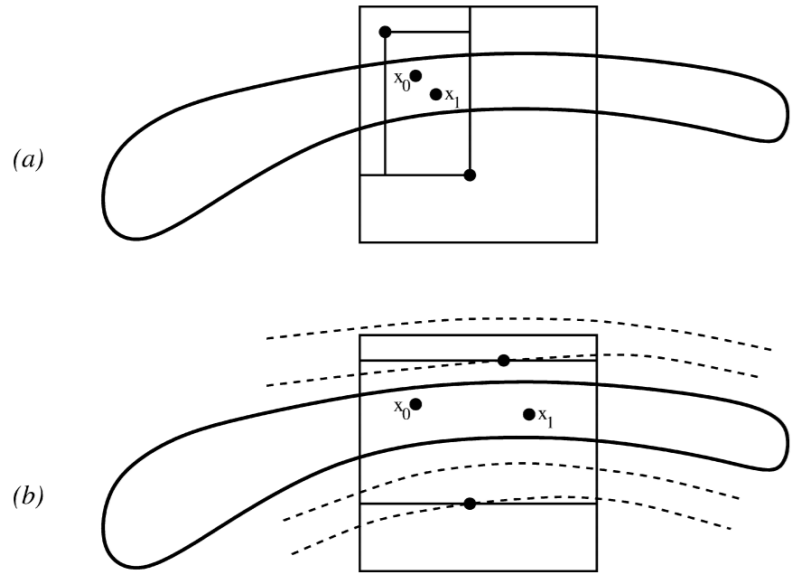
(c) Amostrar x_1 da parte do slice deste hiper-retângulo.

O procedimento "stepping out" não é bem generalizado pois um hiper-retângulo n-dimensional possui 2^n vértices, e testar se todos estes vértices estão fora do slice é muito custoso. Por sua vez, o procedimento "doubling" possui uma generalização melhor e podemos usar como critério de parada se um ponto uniformemente amostrado do hiper-retângulo atual está fora do slice. Outro esquema possível é utilizar um hiper-retângulo posicionado aleatoriamente sem nenhuma expansão, neste caso, é crucial que as estimativas de w não sejam muito menores do que a realidade.

Essa técnica de hiper-retângulos, em geral, é menos eficiente do que aplicar o slice sampling univariado em uma variável por vez porque encolher todas as dimensões do hiper-retângulo até que um ponto dentro do slice seja encontrado pode não ser eficiente pois a função densidade pode não variar muito em algumas dessas dimensões.

Uma solução é realizar o encolhimento em apenas uma dimensão, baseado no gradiente de $\log f(x)$ no último ponto. Para isso, apenas o eixo correspondente a variável x_i é encolhido em que i maximiza o produto $(R_i - L_i)|G_i|$ onde G é o gradiente de $\log f(x)$ no último ponto escolhido. Esse produto nos dá uma estimativa do quanto $\log f(x)$ varia no eixo i e o eixo que possuir a maior variação é o mais indicado para se fazer o encolhimento. Finalmente, a Figura 3 ilustra esses conceitos.

Figura 3: Slice sampling multivariado com hiper-retângulos



O quadrado é hiper-retângulo original e a curva representa o slice. Em (a), o hiper-retângulo é encolhido em todas as direções quando um ponto fora do slice é amostrado até que um ponto dentro do slice seja encontrado. Em (b), o hiper-retângulo é encolhido apenass no eixo determinado pelo gradiente da função e as dimensões atuais do hiper-retângulo. As linhas tracejadas indicam a direção do gradiente. Fonte: Neal (2003).

6 Overrelaxed slice sampling

Quando variáveis são atualizadas sem levar em conta suas dependências, as mudanças devem ser pequenas e muitas iterações são necessárias para mover de uma parte da distribuição para outra. É possível melhorar a eficiência do amostrador evitando o comportamento de passeio aleatório de amostradores mais simples, como o de Gibbs. Uma maneira de alcançar isso é utilizando "atualizações exageradas" ou "overrelaxed updates".

Os métodos de "overrelaxation" atualizam uma variável de cada vez como o Gibbs. Ao invés de amostrar um valor da distribuição condicional independente do valor atual, o novo valor é escolhido do outro lado da moda do valor atual e em geral, espera-se que a distribuição alvo seja unimodal.

Uma das maneiras de fazer "overrelaxation" com slice sampling é:

1. Aplicamos o procedimento "stepping out" para encontrar um intervalo ao redor de x_0.

2. Se o intervalo inicial do "stepping out" já possui os extremos fora do slice, encolhemos seus extremos pela metade até que a mediana do intervalo esteja no slice.

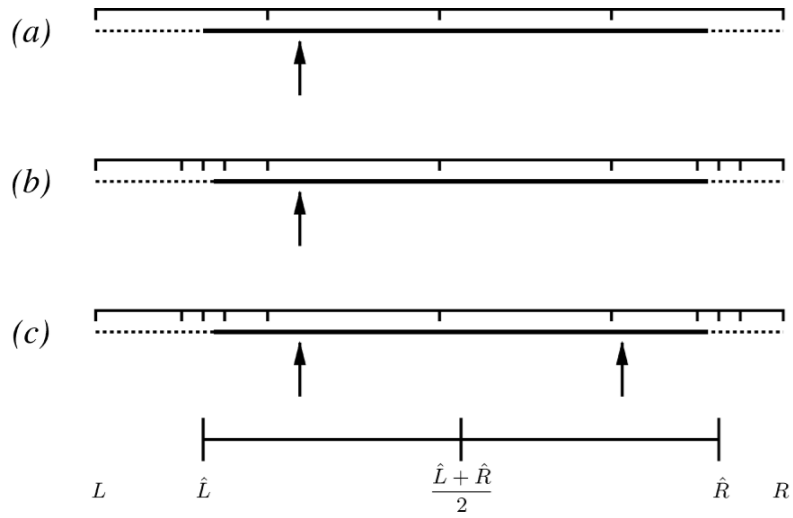
3. Localizamos os extremos do intervalo de forma mais precisa utilizando várias bissecções.

4. Aproximamos o slice pelo intervalo $(\hat{L}, \hat{R})$ formado pelos extremos do intervalo após as bissecções.

5. Fazemos $x_1 = \frac{\hat{L}+\hat{R}}{2} - (x_0 - \frac{\hat{L}+\hat{R}}{2}) = \hat{L} + \hat{R} - x_0$, ou seja, x_1 tem a mesma distância da mediana de $(\hat{L}, \hat{R})$ do que x_0, só que está do lado oposto.

Assim como no procedimento "doubling", em alguns casos, é necessário rejeitar o canditado a x_1 e fazer $x_1 = x_0$. Rejeitamos x_1 em dois casos. Se ele está fora do intervalo que foi encontrado antes das bisecções, pois o intervalo encontrado a partir de x_1 seria diferente, porém isso não pode acontecer em distribuições unimodais. Ou se x_1 está fora do slice, o que acontece até para uma distribuição unimodal quando os extremos do intervalo não foram localizados de forma exata, no entanto, podemos reduzir a frequência deste caso aumentando o número de bissecções. Por fim, a Figura 4 exemplifica o processo de "overrelaxation".

Figura 4: Overrelaxation utilizando stepping out e bissecção



Em (a), um intervalo $I = (L, R)$, com ambos os extremos fora do slice é encontrado empregando o procedimento stepping out. Em (b), os extremos do slice são localizados de forma mais precisa utilizando bissecções. Em (c), um ponto candidato a x_1 é encontrado do outro lado da mediana da aproximação dos extremos do slice. Nesse caso, esse ponto é aceito pois ele está no slice e no intervalo anterior à bissecção. Fonte: Neal (2003).

7 Slice sampling reflexivo

Métodos de slice sampling multivariados também podem ser desenvolvidos com o intuito de evitar o comportamento de passeio aleatório. Uma forma são os métodos que "refletem" nas fronteiras do slice, esses métodos podem ser vistos como uma especialização para distribuições uniformes das dinâmicas do Hamiltonian Monte Carlo (DUANE et al., 1987).

Novamente, suponhamos que temos uma distribuição em $\mathbb{R}^n$ proporcional a $f(x)$ e que conseguimos calcular $f(x)$ e seu gradiente. Em cada iteração, vamos amostrar y de uma $Unif(0, f(x))$ e definir o slice n-dimensional $S = \{x : y < f(x)\}$. Além disso, introduziremos n variáveis auxiliares de "momento" em um vetor p que indicarão a velocidade e direção do movimento.

No início de cada iteração, geramos um valor de p, independente de x, de alguma distribuição rotacionalmente simétrica, geralmente Gaussiana com média 0 e covariância identidade. Se o x' resultante está fora do slice, devemos trazê-lo de volta utilizando algum tipo de reflexão.

Idealmente, gostaríamos de refletir no ponto exato em que o movimento na direção p nos tira do slice, modificando p até que este movimento toque em outra fronteira do slice. Quando atingimos a fronteira do slice no ponto em que o gradiente de $f(x)$ é h, a reflexão ideal altera p da seguinte forma:

$$p' = p - 2h\frac{p \cdot h}{|h|^2}.$$

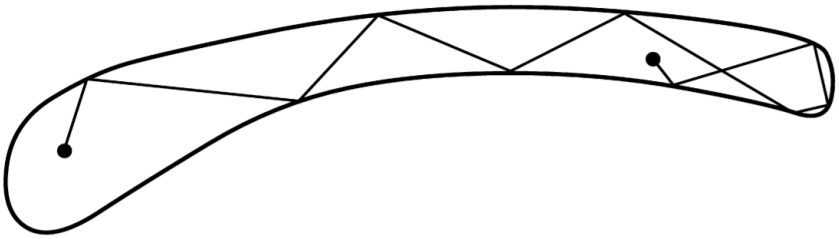
No entanto, este tipo de reflexão, que é apresentado na Figura 5, é de difícil implementação pois é necessário o cálculo exato de onde a trajetória atual intersecta a fronteira do slice. Podemos utilizar duas estratégias de aproximação: reflexão "interior" ou reflexão "exterior". Em ambas estratégias, será necessário rejeitar alguns valores para que elas sejam válidas.

Quando caminhar de x para $x' = x+wp$ nos tira do slice, podemos tentar a reflexão do último ponto de dentro, x, ao invés do ponto exato em que a trajetória intersecta com a fronteira, utilizando o gradiente de $f(x)$ neste ponto. Para este método ser válido, precisamos checar se a trajetória reversa também refletiria neste ponto, verificando se um passo na direção oposta nos tiraria do slice. Essa estratégia é exibida na Figura 6.

Outra alternativa é refletir do ponto exterior, x', utilizando o gradiente neste ponto. Depois de realizar um número pré-determinado de passos, aceitamos a trajetória se o ponto final está no slice. Para este método ser válido, devemos refletir sempre que o ponto atual

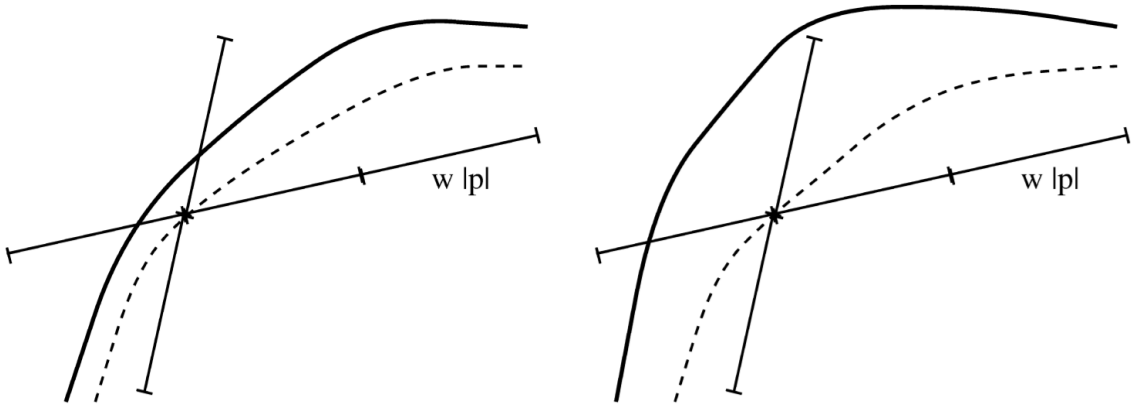
estiver fora do slice, até mesmo se a reflexão não traga o ponto de volta para o slice. A Figura 7 descreve esse tipo de reflexão.

Figura 5: Caminhando em um slice bidimensional utilizando reflexão exata



Fonte: Neal (2003).

Figura 6: Reflexão de um ponto interior

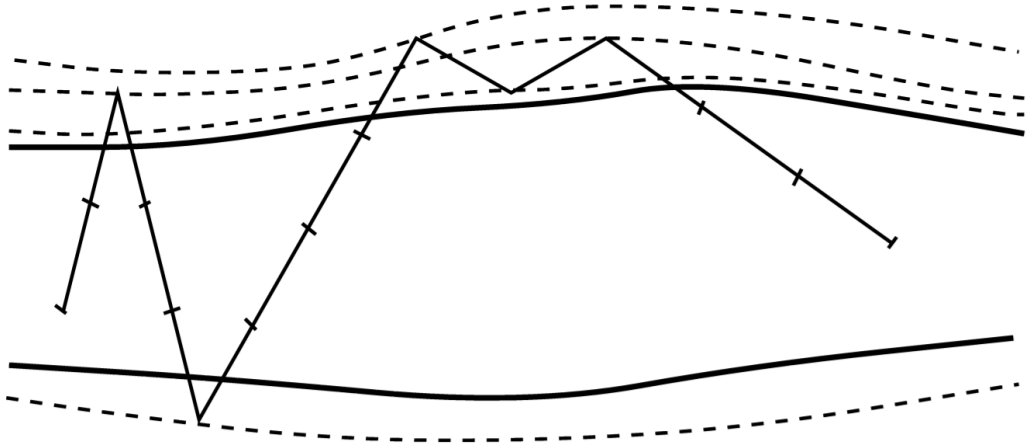


As trajetórias vão em passos de tamanho $w|p|$, começando do canto superior direito até que um ponto fora do slice seja encontrado onde a reflexão interior é feita com base no gradiente de $f(x)$. Em (a), a reflexão é um sucesso; em (b), o ponto encontrado precisa ser rejeitado pois a trajetória reversa não refletiria no mesmo ponto. Fonte: Neal (2003).

8 Demonstração e replicação

Para demonstrar os benefícios da natureza adaptativa do slice sampling, Neal (2003) mostra como ele pode evitar um cenário em que a resposta errada é obtida sem indicações de que algo está errado. Suponha que queremos amostar de uma distribuição composta de 10 variáveis, v e x_1 até x_9, v tem distribuição marginal normal com média 0 e desvio-padrão 3, condicionais a v, x_1 até x_9 são independentes e têm distribuição normal com média 0 e variância e^v.

Figura 7: Reflexão de pontos exteriores



Começando da esquerda, duas reflexões nos trazem de volta para o slice no próximo passo. O passo após a terceira reflexão ainda está fora do slice, então outras reflexões têm que ser feitas. Neste caso, a trajetória eventualmente retorna ao slice, portanto, o ponto final será aceito. Fonte: Neal (2003).

Também vamos supor que não sabemos essa resposta, caso contrário, poderíamos gerar valores de v e depois das distribuições condicionais. O autor utiliza quatro estratégias para amostrar valores desta distribuição e as compara com a distribuição verdadeira. Essas quatro simulações foram replicadas utilizando o software R (R Core Team, 2019) e os códigos podem ser encontrados nesse repositório[1].

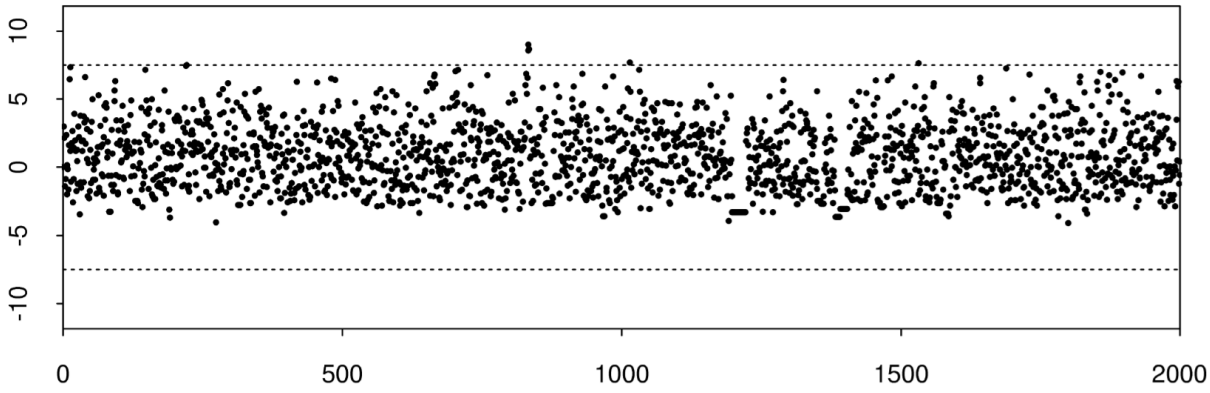
Os valores de v da primeira simulação são apresentada nas Figuras 8 e 9, foram feitas 2000 iterações em que cada iteração consistiu de 10000 atualizações de Metropolis-Hastings multivariado. A proposta utilizada foi uma Gaussiana esférica centrada no valor atual com desvio-padrão 1 para todas as variáveis, os valores iniciais foram $v = 0$ e $x_i = 1$ para todo i.

Podemos ver que a amostragem não foi muito bem feita, sabemos que a distribuição marginal de v é normal, esperaríamos que a distribuição fosse simétrica e que aproximadamente 4,8% dos pontos fossem menores do que -5, o que não acontece. A fonte do problema é na baixa probabilidade de aceitação quando v é pequeno; por exemplo, quando v é -4, o desvio padrão de x_i condicionado a v é 0,135 e a chance da proposta produzir valores para todos os x_i se torna muito pequena.

As Figuras 10 e 11 exibem o resultado da segunda estratégia em que cada uma das 2000 iterações consistiu de 1300 atualizações de Metropolis univariadas para cada variável

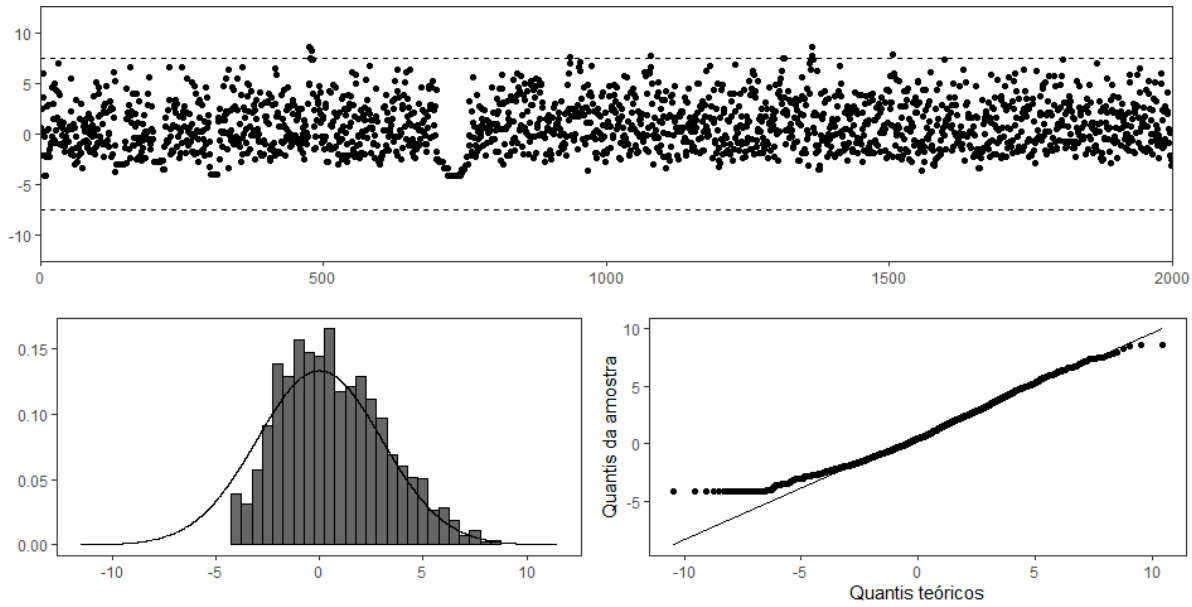
[1]https://github.com/luizfgnmaia/Slice-Sampling

Figura 8: Atualizações de Metropolis multivariado com desvio padrão 1 (original)



Fonte: Neal (2003).

Figura 9: Atualizações de Metropolis multivariado com desvio padrão 1 (replicação)



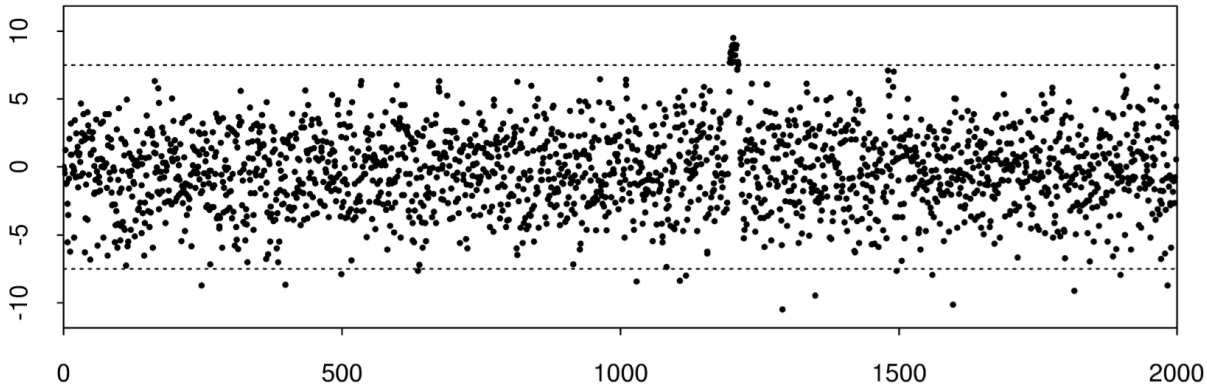
Fonte: Elaboração própria.

em sequência, foram utilizados os mesmos valores iniciais e a proposta foi uma normal padrão para cada componente.

Os resultados não estão tão ruins quanto na primeira simulação, valores baixos de v são bem amostrados desta vez, o problema agora são os valores altos. O problema agora é a exploração lenta feita em espaços pequenos; por exemplo, o desvio-padrão de x_i quando v é 7,5 é de 42,5 que é muito maior do que o passo proposto.

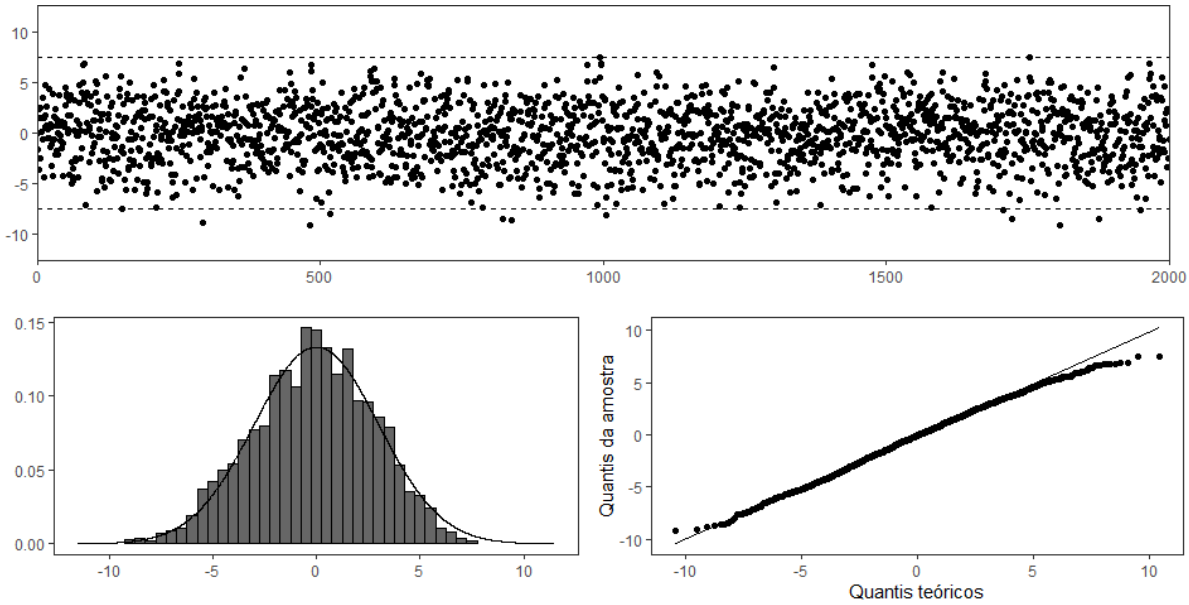
Com o intuito de evitar os problemas das duas primeiras simulações, a terceira

Figura 10: Atualizações de Metropolis univariado com desvio padrão 1 (original)



Fonte: Neal (2003).

Figura 11: Atualizações de Metropolis univariado com desvio padrão 1 (replicação)

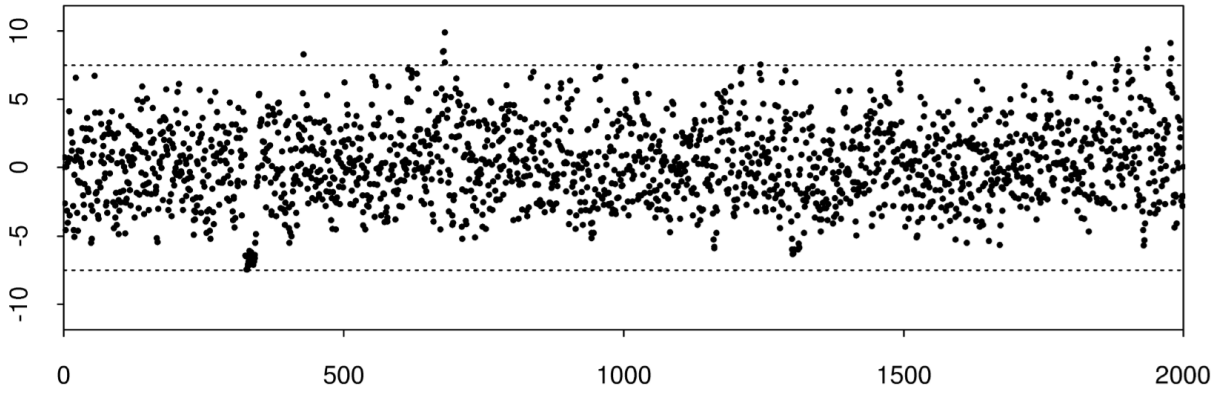


Fonte: Elaboração própria.

simulação, que é apresentada nas Figuras 12 e 13, foi feita com 2000 iterações onde cada iteração consistiu de 10000 atualizações de Metropolis multivariado, assim como na primeira. A diferença é a proposta, nesta estratégia, foi utilizada uma Gaussiana centrada no valor atual e com o $\log_{10}$ do desvio-padrão gerado de uma $Unif(-3, 3)$.

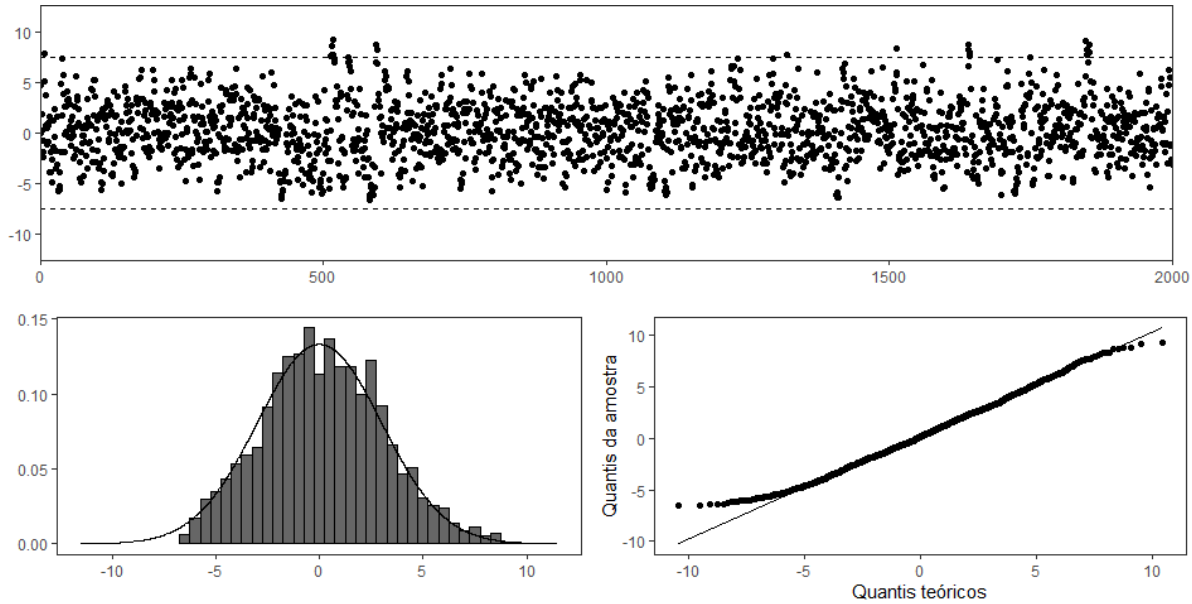
O resultado é um pouco melhor do que a primeira simulação mas o problema continua sendo a baixa amostragem de pequenos valores de v. Uma possível solução seria aumentar o intervalo em que o desvio-padrão da proposta é escolhido mas o custo computacional seria aumentado quando a escolha aleatória é inapropriada.

Figura 12: Atualizações de Metropolis multivariado com desvio padrão aleatório (original)



Fonte: Neal (2003).

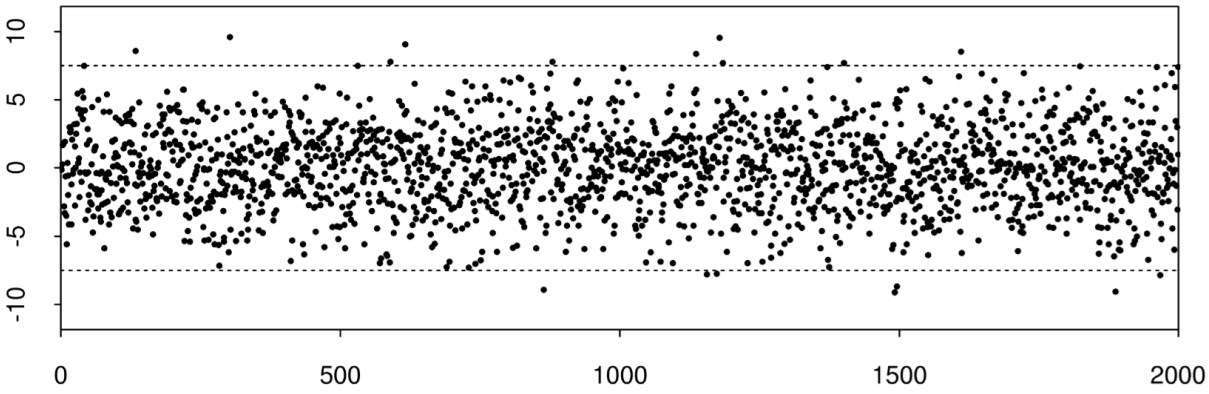
Figura 13: Atualizações de Metropolis multivariado com desvio padrão aleatório (replicação)



Fonte: Elaboração própria.

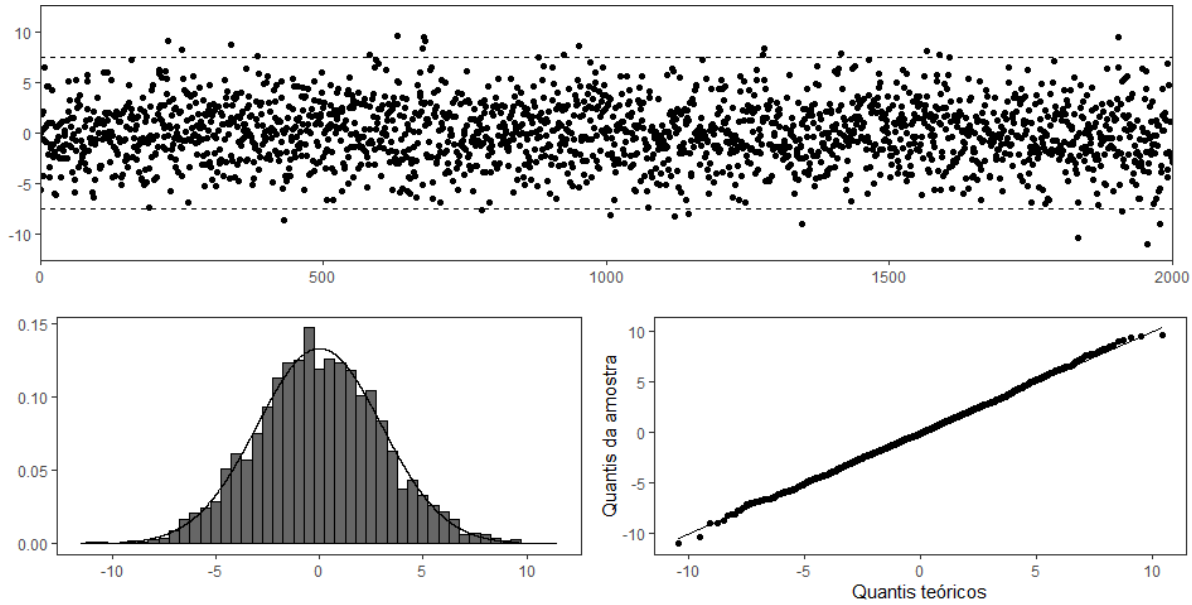
Por fim, o resultado de 2000 iterações em que cada iteração consistiu de 120 atualizações de slice sampling com os procedimentos "stepping out" e "shrinkage" para cada uma das variáveis é apresentado nas Figuras 14 e 15. Utilizou-se 1 como o comprimento inicial do intervalo do "stepping out". O resultado desta simulação é muito melhor do que as anteriores; ambas as caudas da distribuição foram amostradas na proporção correta. Este bom resultado é devido ao fato de que esse método consegue se adaptar às características locais da distribuição.

Figura 14: Slice sampling univariado com intervalo inicial 1 (original)



Fonte: Neal (2003).

Figura 15: Slice sampling univariado com intervalo inicial 1 (replicação)



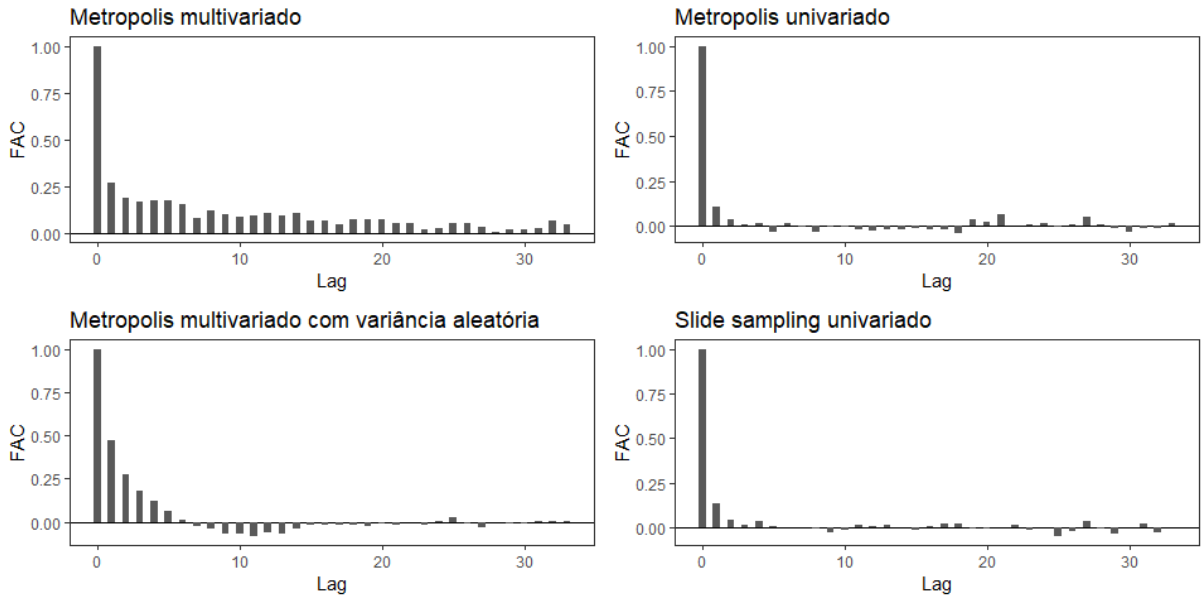
Fonte: Elaboração própria.

A Tabela 1 apresenta algumas estatísticas das cadeias de v e o tempo que cada simulação levou para ser executada nas replicações. Vemos que o slice sampling apresentou média e desvio-padrão muito próximos dos valores reais e que foi o método mais rápido, precisando de menos da metade do tempo para execução do que os outras estratégias de amostragem. Por sua vez, a Figura 16 exibe as funções de autocorrelação das cadeias de v em que podemos ver os que métodos multivariados possuem uma correlação maior nos primeiros lags do que os métodos univariados.

Tabela 1: Estatísticas das cadeias de v e tempo para execução

Simulação	Média	Desvio-padrão	Tempo para execução (minutos)
1	0,7107	2,5175	7,1656
2	-0,1599	2,906	6,5946
3	0,2317	2,9203	8,5322
4	-0,0086	3,0357	2,9553

Figura 16: Funções de autocorrelação das cadeias de v



Fonte: Elaboração própria.

9 Conclusões

Neste relatório, vimos que os métodos de slicing sample são alternativas para o algoritmo de Metropolis e o amostrador Gibbs que precisam de um ajuste menos elaborado e se adaptam melhor a distribuição alvo. Essa melhor adaptação ficou bastante evidenciada na demonstração.

No entanto, os métodos de Metropolis e slicing sample devem ter uma performance similiar se ambos forem bem ajustados. Finalmente, métodos como slicing sample reflexivo e overrelaxed slice sampling podem se adaptar as dependências entre as variáveis e possibilitar uma maior eficiência ao evitar o comportamento de passeio aleatório presente em outros algoritmos.

Referências

DUANE, S. et al. Hybrid monte carlo. *Physics letters B*, Elsevier, v. 195, n. 2, p. 216–222, 1987. Citado na página 10.

GEMAN, S.; GEMAN, D. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, IEEE, n. 6, p. 721–741, 1984. Citado na página 1.

HASTINGS, W. K. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, Biometrika Trust, v. 57, n. 1, p. 97–109, 1970. Citado na página 1.

NEAL, R. M. Slice sampling. *The annals of statistics*, Institute of Mathematical Statistics, v. 31, n. 3, p. 705–767, 2003. Citado 11 vezes nas páginas 1, 5, 6, 8, 9, 11, 12, 13, 14, 15 e 16.

R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria, 2019. Disponível em: ⟨https://www.R-project.org/⟩. Citado na página 12.