Monte Carlo - Uma variável aleatória

Renato Martins Assunção

DCC - UFMG

28 de agosto de 2020

Simulação Monte Carlo

- *Simular:* Fazer aparecer como real uma coisa que não o é; fingir.
- *Simulação:* a imitação do comportamento ou das características de um sistema estocástico utilizando um gerador de números aleatórios num computador: simulação Monte Carlo.
- Estes números possuem uma distribuição de probabilidade de interesse.
- Pode ser a distribuição normal (gausssiana), de Poisson, de Pareto (power law) ou outra.
- Os números aleatórios gerados servem para estudar propriedades complexas de algoritmos ou aspectos do problema que não podem ser deduzidos analiticamente (por fórmulas).

Tudo começa com uma uniforme

- Existe uma base para gerar números aleatórios.
- Praticamente todos os métodos conhecidos geram uma variável aleatória U com distribuição uniforme no intervalo $(0, 1)$.
- Isto é, U é um número escolhido ao acaso em $(0, 1)$ com densidade uniforme.
- A probabilidade de selecionar X num intervalo (a, b) é o seu comprimento: $b - a$.
- A seguir, eles transformam U de forma a obter uma variável com a distribuição de interesse.
- Assim, todas as variáveis são obtidas a partir da distribuição $\mathcal{U}(0, 1)$.

Aleatórios mesmo?

- De fato, os núemros aleatórios gerados no computador não são realmente aleatórios mas sim determinísticos.
- Muito trabalho de pesquisa já foi feito para criar bons geradores de números aleatórios.
- São procedimentos que geram uma seqüência de valores $U_1, U_2, \ldots$.
- Para todos os efeitos práticos, eles podem ser considerados i.i.d. com distribuição uniforme em $(0, 1)$.
- Além disso, por causa da representação finita nos computadores, não conseguimos de fato gerar números reais com precisão inifinita.

O que veremos agora...

- Não veremos em detalhes os geradores de números com distribuição uniforme no intervalo $(0, 1)$.
- Este é um assunto bastante técnico e de pouco uso na prática da análise de dados.
- Vamos dar apenas um ligeira idéia de como eles funcionam.
- Vamos ver um dos algoritmos mais simples existentes.

Divisão inteira

- Eles dependem da operação de divisão inteira. Resto da divisão de um inteiro por outro.
- Por exemplo, a divisão inteira de 18 por 7 é 2 com um resto de 4.
- O resto deve ser um dos inteiros: $0, 1, 2, \ldots, 6$.
- $18 = 2 * 7 + 4$.
- A expressão é única (com resto entre 0 e 6).
- Mais um exemplo: $20 = 2 * 7 + 6$

Divisão inteira

- Dado um inteiro $p > 0$, um inteiro n pode ser escrito de forma única como $n = kp + r$.
- k é um inteiro e $r = 0, \ldots, p - 1$.
- O resto é o valor r que vai variar de 0 a $p - 1$.
- Por exemplo,
  - $21 = 7 \times 3 + 0$ e a divisão inteira de 21 por 7 deixa resto 0.
  - $22 = 7 \times 3 + 1$ e o resto é 1.
  - $27 = 7 \times 3 + 6$ e o resto é 6.
  - $4 = 7 \times 0 + 4$ e o resto é 4.
  - Finalmente, $0 = 7 \times 0 + 0$ e o resto é 0.
- Notação: $n \equiv r \pmod{p}$.

Gerador congruencial misto

- Valor inicial inteiro positivo x_0 arbitrário, chamado de *semente* (*seed*).
- Recursivamente, calcule $x_1, x_2, \ldots$ por meio da fórmula:

$$ax_{i-1} + b \equiv x_i \pmod{p}$$

onde a, b, e p são inteiros positivos.

- x_i é um dos inteiros $0, 1, \ldots, p-1$.
- A sequência

$$u_1 = x_1/p, u_2 = x_2/p, \ldots$$

é uma aproximação para uma sequência de valores de variáveis *independentes* e com distribuição uniforme em $(0, 1)$.

- A qualidade desta aproximação: testes estatísticos incapazes de detectar padrões (não aleatórios) nas sequências geradas.

Exemplo

- Gerador dado por

$$32749x_{i-1} + 3 \equiv x_i \pmod{32777}$$

- Iniciando-se com semente $x_0 = 100$, obtenha $32749 \times 100 + 3 = 3274903$.
- A seguir, o resto da divisão inteira por $p = 32777$: temos $3274903 = 99 \times 32777 + 29980$
- Assim, $x_1 = 29980$
- Primeiro número aleatório entre 0 e 1 é

$$u_1 = x_1/p = 29980/32777 = 0.9146658$$

.

Exemplo

- O segundo valor x_2 é obtido de forma análoga.
- Temos

$$32749 \times 29980 + 3 = 981815023 = 29954 \times 32777 + 12765$$

- Assim, $x_2 = 12765$
- Portanto, $u_2 = 12765/32777 = 0.3894499$.
- E assim por diante: $u_1 = 0.91466577$, $u_2 = 0.38944992$, $u_3 = 0.09549379$, $u_4 = 0.32626537$, $u_5 = 0.86466120$, $u_6 = 0.78957806$, $u_7 = 0.89190591, \ldots$

Não são contínuos

- O gerador do exemplo gera 32777 restos x_i distintos: os inteiros $0, 1, \ldots, 32776$.
- Assim, apenas 32776 números u_i do intervalo $(0, 1)$ podem ser gerados por este procedimento:

$$0/32777, 1/32777, 2/32777, \ldots, 32776/32777$$

- Quanto maior o valor de p, maior o número de valores u_i distintos possíveis.

São pseudo-aleatórios

- $u_1, u_2, \ldots$ não são realmente aleatórios
- Resultam de uma função matemática aplicada de forma recursiva.
- Usando o mesmo gerador e a mesma semente x_0, vamos obter sempre os mesmos números.
- Além disso, a sequência de números pseudo-aleatórios rapidamente se repetir.
- Por exemplo, se $a = 3$, $b = 0$, $m = 30$ e $x_0 = 1$, teremos a sequência $\{3, 9, 27, 21, 3, 9, 27, 21, 3, 9, 27, 21, 3, 9, 27, 21, 3, \ldots\}$.

São pseudo-aleatórios

- Com probabilidade 1, depois de certo tempo, obtem-se um valor x_i igual a algum valor x_{i-k} já obtido anteriormente.
- A partir daí, teremos a sequência repetindo-se com $x_{i+j} = x_{i-k+j}$.
- O número de passos k até obter-se uma repetição numa sequência é chamado de período do gerador.
- Uma importante biblioteca de subrotinas científicas, a NAG, utiliza um gerador congruencial com $a = 13^{13}$, $b = 0$ e $p = 2^{59}$, que possui um período igual a $2^{57} \approx 1.44 \times 10^{17}$.
- Bons geradores tem períodos tão grandes que podem ser ignorados na prática.

A semente

- A semente x_0 costuma ser determinada pelo relógio interno do computador.
- Pode também ser pré-especificada pelo usuário.
- Isto garante que se repita a mesma sequência de números aleatórios.
- De qualquer forma, é um número arbitrário para iniciar o processo.

Gerador de uniforme em $(0,1)$

- Temos um gerador de números (pseudo)-aleatórios reais no intervalo $(0,1)$.
- Isto é, geramos $U \sim Unif(0,1)$.
- U escolhe um número real completamente ao acaso no intervalo $(0,1)$.
- Se (a,b) é um intervalo contido em $(0,1)$. Então

$$\mathbb{P}(U \in (a,b)) = (b-a) = \text{comprimento do intervalo}$$

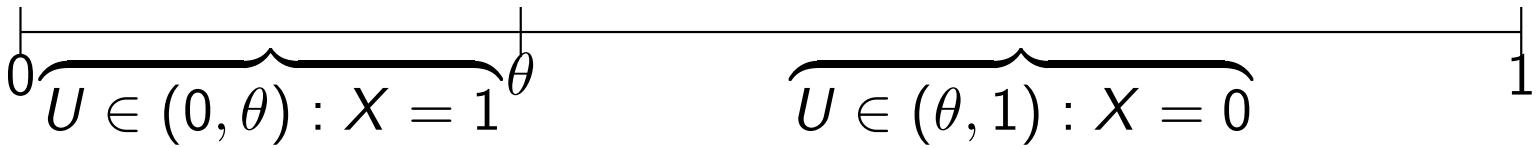
- O comando `runif(1)` em R gera um valor $U(0,1)$.
- `runif(n)` gera `n` valores $U(0,1)$ independentes.

Caso mais simples: Bernoulli

- Como gerar

$$X \sim Bernoulli(\theta): \quad \begin{cases} P(X=1)=\theta \\ P(X=0)=1-\theta \end{cases}$$

- Selecione U ao acaso no intervalo $(0,1)$.



Gerando uma Bernoulli(0.35)

- Suponha $\theta = p = 0.35$, por exemplo.
- Podemos usar:

```
p = 0.35
U = runif(1)
if(U <= p) X = 1
     else X = 0
```

- Mais simples em *R*: `X = runif(1) <= p`
- Gerando 215 valores i.i.d.: `X = runif(215) <= p`

Gerando uma Binomial(m, θ)

- Para gerar $X \sim$ Bin(m, p), basta repetir o algoritmo Bernoulli m vezes independentemente.
- Por exemplo, se $m = 100$ e $\theta = 0.35$, então:

```
m <- 100
p <- 0.35
X <- 0
for(i in 1:m) if(runif(1) < p) X <- X+1
```

- Em R, vetorizando fica muito mais simples:

```
X = sum(runif(m) <= p)
```

Gerando Binomial no R

- Na verdade, o R já possui um gerador de binomial.
- Help do *R*: `rbinom(n, size, prob)`: geramos n valores, cada um deles de uma Bin(`size`, `prob`).
- WARNING: No HELP do *R*, o argumento n refere-se a quantos valores binomiais Bin(`size`, `prob`) queremos gerar. Não confundir com a notação usual em que escrevemos Bin(n, θ) para umavariável binomial.
- Por exemplo, para gerar $n = 10$ valores indepedentes de uma Bin$(100, 0.17)$ (isto é, `size`=100 e `prob`= $\theta = 0.17$), digitamos:

```
> rbinom(10, 100, 0.17)
 [1] 14 20 20 14  8 14 12 13 17 14
```

Calculando $\mathbb{P}(X = k)$

A função `dbinom(x, size, prob)` calcula a $P(X = x)$ quando X é uma v.a. binomial Bin(size,prob).

Por exemplo, se $X \sim \text{Bin}(100, 0.17)$ então $\mathbb{P}(X = 13)$ é

```
> dbinom(13, 100, 0.17)
[1] 0.06419966
```

Podemos pedir vários valores de uma única vez:

```
> dbinom(13:17, 100, 0.17)
[1] 0.06419966 0.08171369 0.09595615 0.10441012 0.10566807
```

Gerando v.a. discreta arbitrária

Vamos ver um procedimento geral, que serve para qualquer distribuição discreta, mesmo para aquelas com infinitos valores, como a Poisson, Geométrica e Pareto.

Distribuição de X é dada por:

x_i	$P(x = x_i) = p_i$
x_1	p_1
x_2	p_2
x_3	p_3
$\vdots$	$\vdots$
Total	$\sum_i p_i = 1$

Tabela: Distribuição da v.a. discreta X com valores possíveis $x_1, x_2, \ldots$

Gerando v.a. discreta arbitrária

- Acumulamos as probabilidades obtendo $F(x_k) = P(X \leq x_k) = \sum_{i=1}^{k} p_i$.
- Por exemplo,

$$\begin{aligned} F(x_1) &= p_1 \\ F(x_2) &= p_1 + p_2 \\ F(x_3) &= p_1 + p_2 + p_3 \quad \text{Etc.} \end{aligned}$$

- Se $0 < U < F(x_1) = p_1$ faça $X = x_1$
- Se $p_1 \leq U < p_1 + p_2$ faça $X = x_2$
- Se $p_1 + p_2 \leq U < p_1 + p_2 + p_3$ faça $X = x_3$
- Etc.

Gerando v.a. discreta arbitrária

- Em resumo, faça $X = g(U)$:

$$X = g(U) = \begin{cases} x_0, & \text{se } U < p_0 \\ x_1, & \text{se } p_0 \leq U < p_0 + p_1 \\ x_2, & \text{se } p_0 + p1 \leq U < p_0 + p_1 + p_2 \\ \dots & \dots \\ x_i, & \text{se } \sum_{k=1}^{i-1} p_k \leq U < \sum_{k=0}^{i} p_k \\ \dots & \dots \end{cases}$$

Exemplo

- Gerar X com a seguinte distribuição de probabilidade discreta:

$$X = \begin{cases} -1, & \text{com probabilidade } p_0 = 0.25 \\ 2, & \text{com probabilidade } p_1 = 0.35 \\ 7, & \text{com probabilidade } p_2 = 0.17 \\ 12, & \text{com probabilidade } p_3 = 0.23 \end{cases}$$

- Gere $U \sim U(0,1)$ e faça

$$g(U) = X = \begin{cases} -1, & \text{se } U < 0.25 \\ 2, & \text{se } 0.25 \leq U < 0.60 \\ 7, & \text{se } 0.60 \leq U < 0.77 \\ 12, & \text{se } 0.77 \leq U < 1.00 \end{cases}$$

- Por exemplo, se $U = 0.4897$ então $X = 2$ pois $0.25 \leq 0.4897 < 0.60$.

Exemplo - Poisson

- Para o caso de $X \sim$ Poisson(1.61) teríamos:

$$X = g(U) = \begin{cases} 0 & \text{se } U < 0.1998876 \\ 1 & \text{se } 0.1998876 \leq U < 0.5217067 \\ 2 & \text{se } 0.5217067 \leq U < 0.7807710 \\ \ldots & \ldots \\ i & \text{se } 0.1998876 \sum_{k=1}^{i-1} (1.61)^i/i! \leq U < 0.1998876 \sum_{k=0}^{i} (1.61)^i/i! \\ \ldots & \ldots \end{cases}$$

- Algoritmo? É impossível listar os infinitos possíveis valores de X e só então verificar onde o valor de X caiu.

Algoritmo Poisson

- Trabalhar sequencialmente.
- Verifique se U cai no primeiro intervalo.
- Se sim, pare e retorne $X = 0$.
- Se não, calcule o intervalo seguinte e verifique se U cai neste novo intervalo.
- Se sim, pare e retorne $X = 1$.
- E etc.

Casos especiais

- Para facilitar o cálculo podemos usar uma relação de recorrência entre as probabilidades sucessivas de uma Poisson com parâmetro λ:

$$p_{i+1} = \frac{\lambda}{i+1} p_i$$

- Com $\lambda = 1.61$:

```
lambda = 1.61
x = -1
i = 0; p = exp(-lambda); F = p
while(x == -1){
 if(runif(1) < F) x = i
  else{
    p = lambda*p/(i+1)
    F = F + p
    i = i+1
  }
 }
```

Transformada inversa

- X é v.a. contínua com distribuição acumulada $F_X(x)$.
- Por exemplo, se $X \sim \exp(3)$ então $F_X(x) = 1 - \exp(-3x)$ para $x \geq 0$.
- Gere uma variável uniforme $U \sim U(0, 1)$.
- A seguir, transforme usando $Y = F_X^{-1}(U)$.
- A v.a. Y possui a mesma distribuição que X.
- Isto é, a função distribuição acumulada de Y no ponto y é exatamente $F_X(y)$.

Exemplo

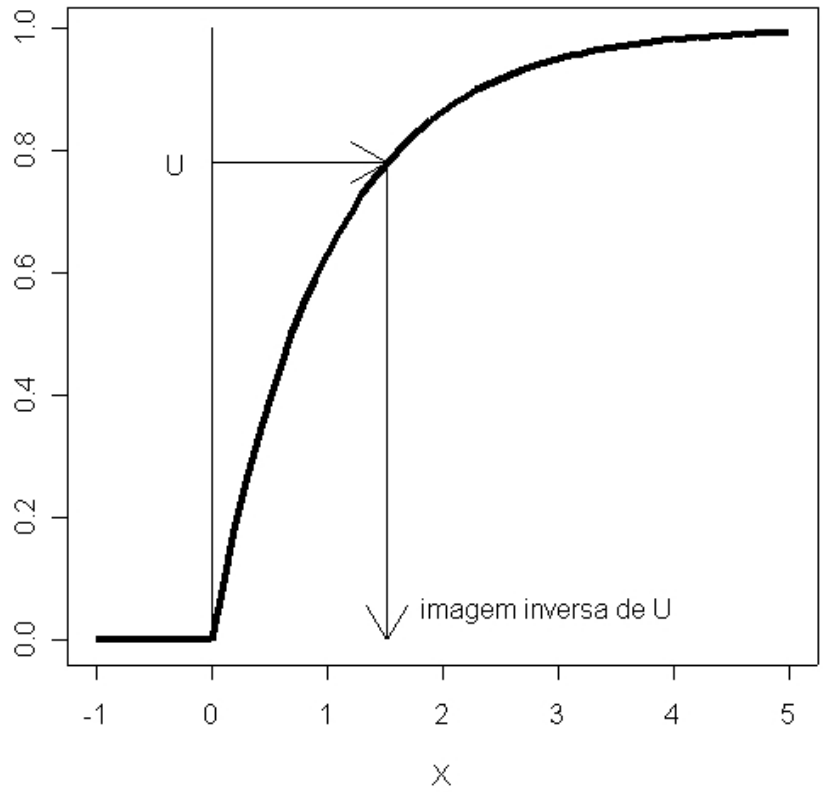
- Gerar $X \sim \exp(1)$. Então

$$F_X(x) = \begin{cases} 1 - \exp(-x), & \text{se } x > 0 \\ 0, & \text{caso contrário} \end{cases}$$

- Se $u = 1 - \exp(-x)$, então $x = -\log(1-u) = F_X^{-1}(u)$.
- Gere $U \sim U(0,1)$ e aplique $W = F^{-1}(U) = -\log(1-U)$.
- $W \sim \exp(1)$.

Intuição gráfica

- Gere $U \sim U(0,1)$ e coloque-o no eixo vertical.
- Obtenha a imagem inversa F_X^{-1}.



Exemplo da prova no caso particular da exp(1)

- Gerar $X \sim \exp(1)$:

$$F_X(x) = \begin{cases} 1 - \exp(-x), & \text{se } x > 0 \\ 0, & \text{caso contrário} \end{cases}$$

- Se $u = 1 - \exp(-x)$, então $x = -\log(1-u) = F_X^{-1}(u)$.
- $W = F^{-1}(U) = -\log(1-U)$.
- W possui distribuição exponencial 1.
- De fato, se $w > 0$, nós temos

$$\begin{aligned} \mathbb{P}(W \leq w) &= \mathbb{P}(-\log(1-U) \leq w) \\ &= \mathbb{P}(1-U \leq e^{-w}) \\ &= \mathbb{P}(U \leq 1 - e^{-w}) \\ &= 1 - e^{-w} \end{aligned}$$

Prova no caso geral

- $U \sim U(0,1)$
- Defina a v.a. $W = F_X^{-1}(U)$.
- Como uma função de distribuição acumulada é não decrescente, se $a \leq b$, então $F_X(a) \leq F_X(b)$.
- Além disso, $P(U \leq a) = a$ se $a \in [0,1]$.
- Assim,

$$\begin{aligned} F_W(w) &= \mathbb{P}(W \leq w) \\ &= \mathbb{P}(\mathbb{F}_X^{-1}(U) \leq w) \\ &= \mathbb{P}(\mathbb{F}_X(\mathbb{F}_X^{-1}(U)) \leq \mathbb{F}_X(w)) \\ &= \mathbb{P}(U \leq \mathbb{F}_X(w)) \\ &= \mathbb{F}_X(w) \end{aligned}$$

Observações

- Como U e $1-U$ possuem a mesma distribuição uniforme $U(0,1)$.
- Então $X = -\log(U)$ também é exponencial com parâmetro 1.
-
- Se $X \sim \exp(1)$ então $Y = X/\beta \sim \exp(\beta)$.
- Assim, pode-se gerar $Y \sim \exp(\beta)$ usando a transformação $Y = -1/\beta \ \log(U)$.

Seguro de vida: idade ao morrer

- Uma distribuição muito importante para o mercado de seguros é a distribuição de Gompertz.
- Ela modela muito bem o tempo de vida a partir dos 22 anos.
- A função de de distribuição acumulada $F(x)$ é

$$F(x) = 1 - \exp\left(-\frac{B}{\log(c)}(c^x - 1)\right)$$

onde $c > 1$ e $B > 0$.

- O parâmetro c usualmente possui um valor em torno de 1.09.
- Um valor típico para B é 1.02×10^{-4}.

Transformada inversa de Gomperz

- Invertendo:

$$F^{-1}(u) = \log\left(1 - \log(c)\log(1-u)/B\right)/\log(c)$$

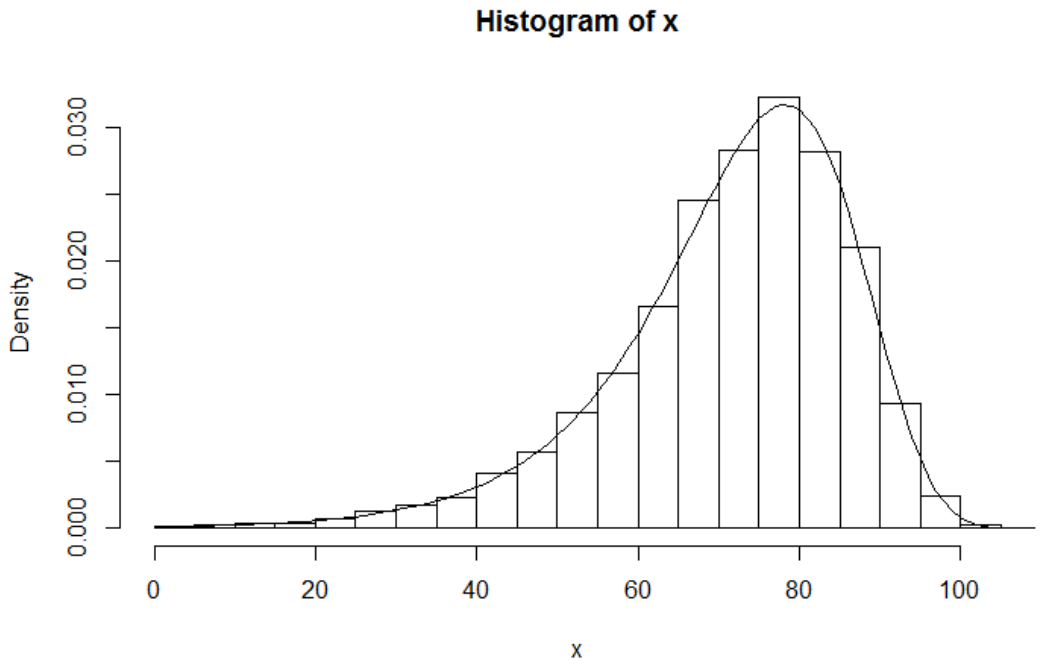
Assim um código em R para obter a amostra é o seguinte:

```
 # Amostra de 10 mil valores iid de Gompertz
 ## fixa as constantes
ce <- 1.09; B <- 0.000102; k <- B/log(ce)
u <- runif(10000)  ## gera valores iid U(0,1)
 ## Gompertz por metodo da transformada inversa
x <- 1/log(ce) * (log( 1- log(1-u)/k))
```

10 mil vidas Gomperz

- Fazendo um histograma dos 10 mil valores gerados e acrescentando a densidade Gomperz:

```
hist(x, prob=T)
eixox <- seq(0,120,by=1)
dens <- B * ceˆeixox * exp(-k * (ceˆeixox - 1))
lines(x,y)
```



Pareto ou power-law em seguros

- Perdas monetárias associadas com uma apólice
- Um parâmetro: $x_0 > 0$, é o valor mais baixo que uma perda pode ter.
- x_0 é um valor de franquia ou um valor *stop-loss*.
- Seguradora só toma conhecimento de sinistros com valores acima de x_0.
- Cobre toda a perda acima do valor x_0.
- Pareto é costuma se ajustar bem a este tipo de dados.
- O 2o. parâmetros, $\alpha > 0$, controla o peso da cauda superior da distribuição em relação aos valores próximos de x_0.
- Quanto menor α, maior a chance de observarmos valores extremos.

Densidade da Pareto

- Pareto com parâmetros (x_0, α) é dada por

$$f_X(x) = \begin{cases} 0, & \text{se } x \leq x_0 \\ \frac{\alpha}{x_0}\left(\frac{x_0}{x}\right)^{\alpha+1}, & \text{se } x > x_0 \end{cases}$$

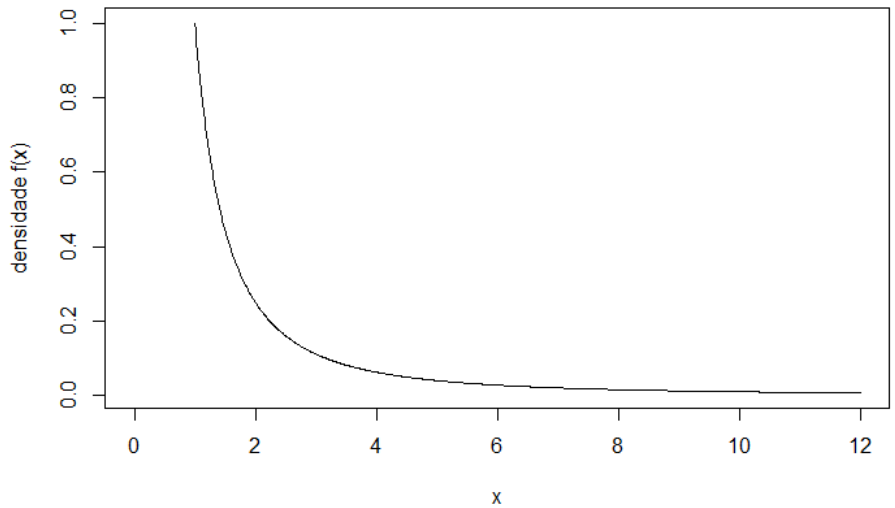


Figura: Densidade da Pareto com $x_0 = 1$ e $\alpha = 1$

α típicos

- Quais os valores típicos de α na prática de seguros e resseguros?
- A Swiss Re, a maior companhia européia de resseguros, fez um estudo.
- Nos casos de perdas associadas com incêndios, $\alpha \in (1, 2.5)$.
- Esta faixa pode ser mais detalhada: para incêndios em instalações industriais de maior porte, temos $\alpha \approx 1.2$.
- Para incêndios ocorrendo em pequenos negócios e serviços temos $\alpha \in (1.8, 2.5)$.
- No caso de perdas associadas com catástrofes naturais: $\alpha \approx 0.8$ para o caso de perdas decorrentes de terremotos; $\alpha \approx 1.3$ para furacões, tornados e vendavais.

Gerando Pareto ou power-law

- Temos $F_X(x) = 1 - (x_0/x)^\alpha$ se $x > x_0$
- Então

$$X \sim F_X^{-1}(U) = x_0/(1-U)^{-1/\alpha}$$

- Basta digitar `x0/(1-runif(1000))^(1/a)` no R.

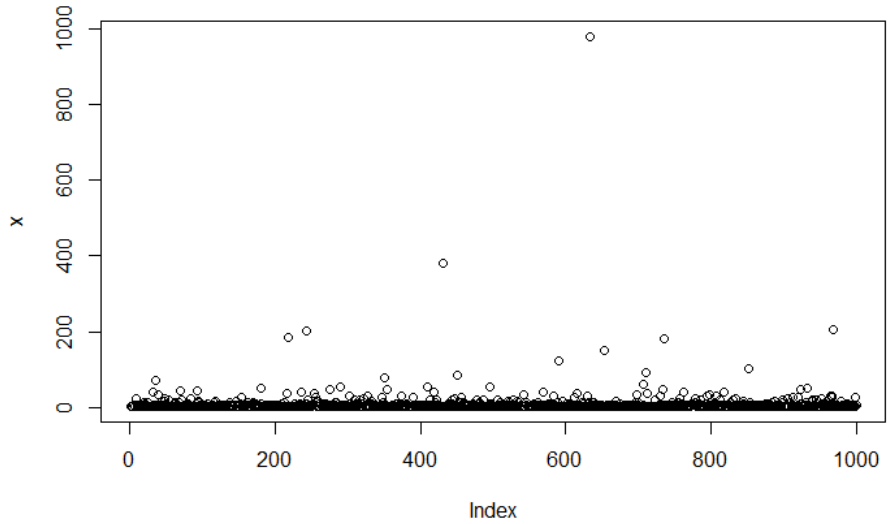


Figura: Amostra de 1000 valres i.i.d. de uma Pareto com $x_0 = 1$ e $\alpha = 1$

Amostras de Pareto

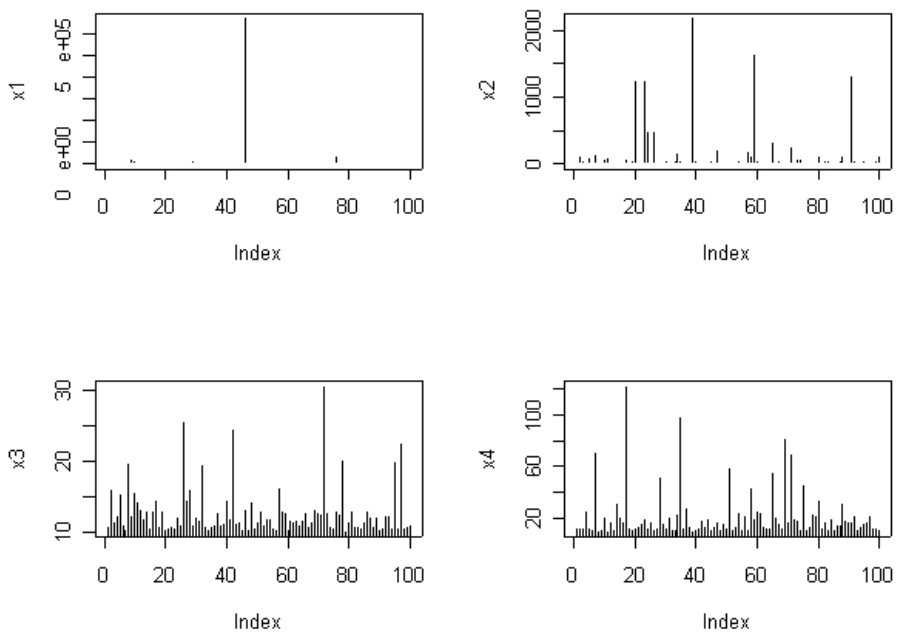


Figura: Amostras de 100 valores Pareto com (x_0, α) igual a $(1.3, 0.25)$ (canto superior esquerdo), $(1.3, 0.5)$ (canto superior direito), $(10, 5)$ (canto inferior esquerdo) e $(10, 2)$ (canto inferior direito).

Gerando gaussianas $N(0,1)$

- Numa gaussiana, $F(x)$ não possui uma fórmula analítica.
- O uso da técnica de transformação $F_X^{-1}(U)$ de variáveis uniformes não pode ser usado.
- Box e Muller propuseram um algoritmo muito simples para gerar gaussianas.
- Pode-se mostrar matematicamente que:
  - se $\theta \sim U(0, 2\pi)$ e $V \sim \exp(1/2)$, duas v.a.'s independentes
  - então $X = \sqrt{V}\cos(\theta) \sim N(0,1)$.
- Como você sabe gerar uniformes e exponenciais...código em R para gerar n valores independentes de uma $N(0,1)$.
- Em R:

```
minharnorm = function(n) sqrt(rexp(n, 0.5)) * cos(runif(n, 0, 2 * pi))
```

Amostra de gaussiana $N(0,1)$

```
set.seed(123)
minharnorm = function(n){ sqrt(rexp(n, 0.5)) * cos(runif(n, 0, 2 * pi))}
hist(minharnorm(1000), prob=T)
plot(dnorm, -3,3, add=T)
```

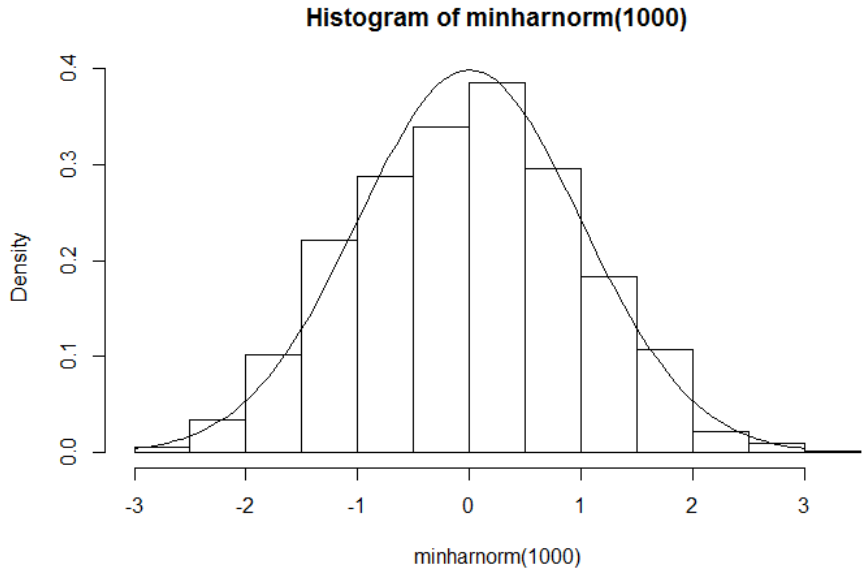


Figura: Histograma padronizado de 1000 valores $N(0,1)$ gerados com `minhanorm` com a densidade $f(x)$ sobreposta.

Gerando gaussianas $N(\mu, \sigma^2)$

- Como gerar uma gaussiana $N(\mu, \sigma)$: centrada em μ e com dispersão σ em torno de μ.
- Propriedade de probabilidade: se $Z \sim N(0, 1)$ então $X = \mu + \sigma Z \sim N(\mu, \sigma^2)$
- Sabemos gerar $Z \sim N(0, 1)$.
- Se quiser $X \sim N(10, 4)$ (digamos) basta gerar Z e em seguida tomar $X = 10 + \sqrt{4}Z$.
- Em R:

```
minharnorm = function(n) sqrt(rexp(n, 0.5)) * cos(runif(n, 0, 2 * pi))

x = 10 + sqrt4 * minhanorm(100)
```

- É claro que R já possui gerador de gaussianas: `rnorm(100, mean=0, sd=1)`

Estimando integrais por Monte Carlo

- Queremos calcular
$$\theta = \int_0^1 g(x)\,dx$$
- Podemos ver a integral θ como a esperança de uma v.a.: se $U \sim U(0,1)$ então $\theta = E[g(U)]$.
- Se $U_1, U_2, \ldots, U_n$ são i.i.d. $U(0,1)$ então as v.a.'s $Y_1 = g(U_1), Y_2 = g(U_2), \ldots, Y_n = g(U_n)$ também são i.i.d. com esperança θ.
- Pela Lei dos Grandes Números, se $n \to \infty$,
$$\frac{1}{n}\sum_{i=1}^{n} Y_i = \frac{1}{n}\sum_{i=1}^{n} g(U_i) \to E[g(U)] = \theta$$
- Assim, se n é grande, θ é aprox. a média aritmética dos valores simulados $g(u_i)$.

Exemplo

- Queremos

$$\theta = \int_0^1 x^2\, dx = \frac{1}{3}$$

- Uma amostra i.i.d. de 1000 variáveis aleatórias $U(0,1)$ é gerada:

$$u_1 = 0.4886415, u_2 = 0.1605763, u_3 = 0.8683941, \ldots, u_{1000} = 0.3357509$$

- Calculamos então

$$\begin{aligned} \hat{\theta} &= \left(u_1^2 + u_2^2 + \ldots + u_{1000}^2\right)/1000 \\ &= \left((0.4886415)^2 + (0.1605763)^2 + \ldots + (0.3357509)^2\right)/1000 \\ &= 0.33406 \approx \theta \end{aligned}$$

- Nova geração, com semente diferente, vai produzir $\hat{\theta}$ ligeiramente diferente.
- Outros 1000 valores da uniforme produzem $\hat{\theta} = 0.3246794$.
- Aumentando tamanho da amostra variação diminui: escolha do tamanho da amostra precisa de desigualdades em probabilidade (logo mais).

Integrais e probabilidades gaussianas

- Se $X \sim N(0,1)$ então

$$\mathbb{P}(X \in (0,1)) = \int_0^1 \frac{\exp(-x^2/2)}{\sqrt{2\pi}}\, dx = \theta$$

- Não existe fórmula para esta integral, deve ser obtida numericamente.
- Usando as funções nativas em R:
  `pnorm(1) - pnorm(0)` que retorna $0.8413447 - 0.5 = 0.3413447$
- Gere 1000 valores i.i.d. de uma $U(0,1)$ e calcule
  $(y_1 + y_2 + \ldots + y_{1000})/1000$ onde $y_i = (2\pi)^{-0.5} \exp(-u_i^2/2)$.
- Por exemplo, se $u_i = 0.4886$ então
  $y_i = (2\pi)^{-0.5} \exp(-0.4886^2/2) = 0.3541$.
- Em R: `mean((2*pi)^(-0.5) * exp(-runif(1000)^2/2))`
- Quatro simulações sucessivas (e independentes) com 1000 valores:
  $0.3425249, 0.3413119, 0.3432939$ e 0.3400479.
- Comparando com $\theta = 0.3413447$, os erros de estimação são pequenos.

Limites genéricos

- Nem sempre a integral terá os limites 0 e 1.

$$\theta = \int_a^b g(x)\,dx$$

- Fazer mudança de variável linear: tome $x = a + (b-a)y$ e $dx = (b-a)dy$.
- Então

$$\theta = \int_a^b g(x)\,dx = \int_0^1 g\left(a + (b-a)y\right)(b-a)\,dy = \int_0^1 h(y)\,dy$$

onde $h(y) = (b-a)g(a + (b-a)y)$.

- Usamos $U(0,1)$ mesmo quando a integral é num intervalo $(a,b) \neq (0,1)$

Exemplo

- Calcule o valor aproximado de

$$\theta = \int_3^9 \log(2 + |\sin(x)|) e^{-x/20}\, dx$$

- Uma amostra i.i.d. de 1000 $U(0,1)$ é gerada e calcula-se

$$\bar{w} = \frac{1}{n}(w_1 + \ldots + w_{1000})$$

onde

$$w_i = h(u_i) = 6 \log(2 + |\sin(3 + 6u_i)) \exp\left(-\frac{3 + 6u_i}{20}\right)$$

- Três simulações deram: 4.285739, 4.327516, 4.310637.
- Neste exemplo, não sabemos o verdadeiro valor θ da integral mas as simulações dão aproximadamente o mesmo valor.
- Isto é um sinal de que, ao usar qualquer um deles como estimativa, a integral deve estar sendo estimada com pequeno erro.

Método de aceitação-rejeição

- Queremos gerar amostra de densidade $f(x)$.
- Não conseguimos obter $F(x)$ analiticamente.
- O método da transformada inversa não pode ser usado.
- Uma alternativa: *método de aceitação-rejeição*
- Idéia básica: gerar de *outra distribuição* que seja fácil.
- A seguir, retemos alguns dos valores gerados e descartamos os demais.
- Isto é feito de tal maneira que a amostra que resta tem exatamente a densidade $f(x)$.

Essência da ideia

- Sabemos gerar com facilidade da densidade $g(x)$ (linha tracejada).
- Amostra de $g(x)$ produz o histograma abaixo.
- Mas queremos amostra de $f(x)$.
- Eliminamos de forma seletiva alguns valores gerados.

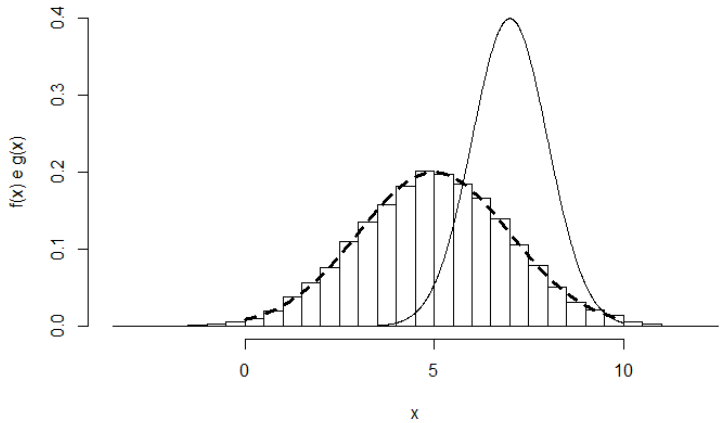


Figura: Linha contínua: densidade $f(x)$ de onde queremos amostrar. Linha tracejada: densidade $g(x)$ de onde sabemos amostrar. histograma de amostra de 20000 elementos de $f(x)$.

Essência da ideia

- Se o processo seletivo for feito de maneira adequada,
- terminamos com uma amostra que, no fim dos dois processos (geração e aceitação-rejeição), é gerada de $f(x)$.

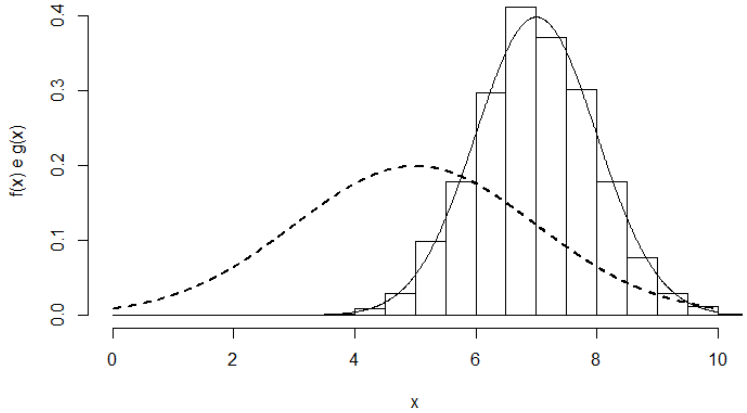


Figura: Linha contínua: densidade $f(x)$ de onde queremos amostrar. Linha tracejada: densidade $g(x)$ de onde sabemos amostrar. histograma de amostra de 20000 elementos de $f(x)$. Histograma dos 3696 elementos da amostra anterior que restaram após rejeitar seletivamente 16304 dos elementos gerados.

Compatibilizando os suportes

- Fixe uma densidade-alvo $f(x)$.
- Quais $g(x)$ podemos escolher?
- Suporte de $g(x)$ deve ser maior que aquele de $f(x)$.
- Isto é, se $f(x)$ pode gerar um valor x então $g(x)$ também deveria ser capaz de gerar este x.
- Ou seja, se $f(x) > 0$ então $g(x) > 0$.
- $g(x)$ pode gerar valores impossíveis sob $f(x)$
- Mas não podemos permitir que valores possíveis sob $f(x)$ sejam impossíveis sob $g(x)$.
- Isto é bem razoável: se inicialmente, usando $g(x)$, gerarmos valores impossíveis sob $f(x)$, podemos rejeitá-los no segundo passo do algoritmo.
- Mas se nunca gerarmos valores de certas regiões possíveis sob $f(x)$, nossa amostra final não será uma amostra de $f(x)$.

Ache M tal que $f(x) \leq Mg(x)$

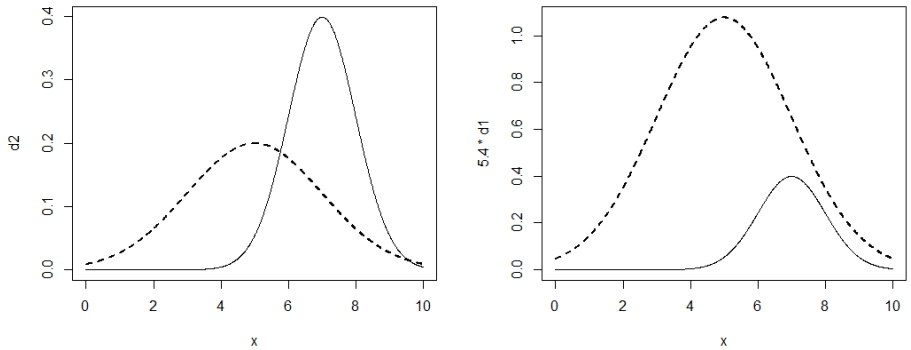
- Precisamos achar uma constante $M > 1$ tal que

$$f(x) \leq Mg(x)$$

  para todo x.
- Isto é, multiplicamos a densidade $g(x)$ de onde sabemos amostrar por uma constante $M > 1$ implicando em elevá-la.
- Por exemplo, se $M = 2$, comparamos o valor de $f(x)$ com $2g(x)$, duas vezes a altura da densidade g no ponto x.
- Devemos ter sempre $f(x) \leq Mg(x)$.

Exemplo

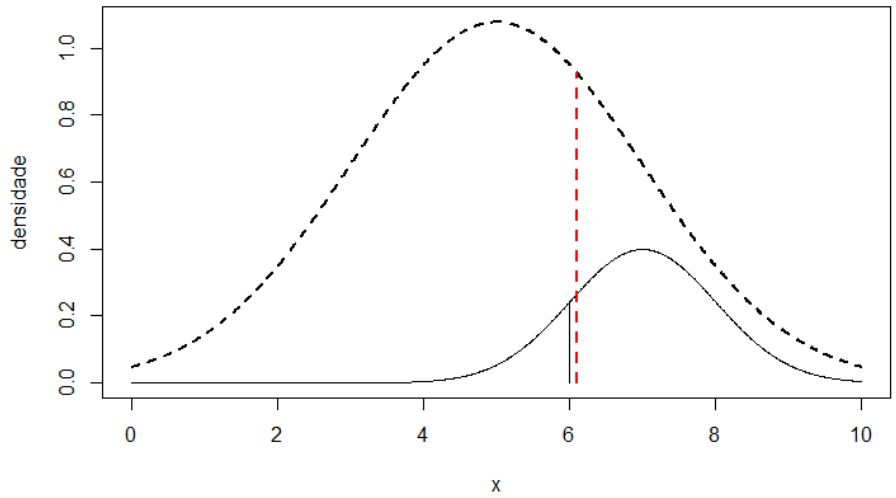
- Linha contínua é a densidade $f(x)$ de onde queremos amostrar
- Linha tracejada: densidade $g(x)$ de onde sabemos amostrar.
- Direita: gráfico de $f(x)$ e de $5.4 * g(x)$.
- Temos $f(x) \leq 5.4g(x)$ para todo x



Razão $r(x)$

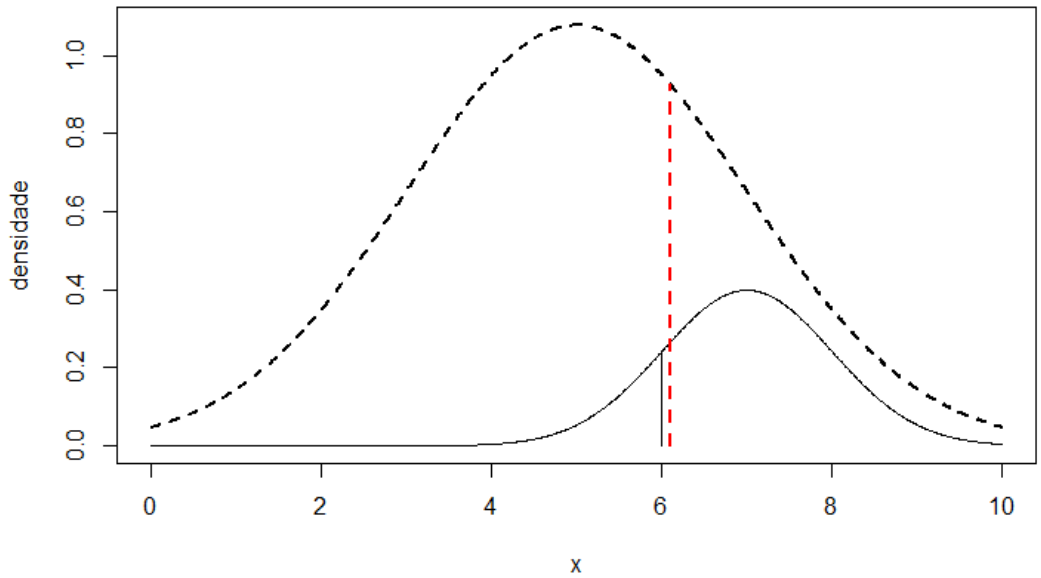
- Temos $f(x)$ e $Mg(x)$.
- No ponto $x = 6.0$ temos a altura $f(x)$ (contínua) e a a altura $5.4g(x)$ (tracejada).
- Para todo x, definimos a razão entre estas alturas

$$r(x) = \frac{f(x)}{Mg(x)} < 1 \quad \text{para todo } x.$$



$$r(x) = \frac{f(x)}{Mg(x)} < 1$$

- Sejam $x_1, x_2, \ldots$ o elementos da amostra de $g(x)$. Quais reter?
- Calcule $r(x_1), r(x_2), \ldots$
- Se $r(x_i) \approx 0$, vamos tipicamente rejeitar x_i
- Se $r(x_i) \approx 1$, vamos tipicamente reter x_i.



$r(x) = \frac{f(x)}{Mg(x)}$ é a probabilidade de retenção

- Para cada elemento x_i gerado por $g(x)$, jogamos uma moeda com probabilidade de cara igual a $r(x_i)$.
- Se sair cara, retemos x_i como um elemento vindo de $f(x)$.
- Se sair coroa, eliminamos x_i da amostra final.
- Se começarmos com n elementos retirados de $g(x)$, o tamanho final da amostra é aleatório e geralmente menor que n

Algoritmo

Y é um valor inicialmente gerado a partir de $g(x)$ e X é um dos valores finalmente aceitos no final do processo.

Algorithm 1 Método da Rejeição.

1: $I \leftarrow$ `True`
2: **while** I **do**
3: Gere $Y \sim g(y)$
4: Gere $U \sim \mathcal{U}(0,1)$
5: **if** $U \leq r(Y) = f(Y)/Mg(Y)$ **then**
6: $X \leftarrow Y$
7: $I =$ `False`
8: **end if**
9: **end while**

Exemplo

- Queremos gerar $X \sim Gamma(3,3)$ com densidade:

$$f(x) = \begin{cases} 0 & , \text{ se } x \leq 0 \\ \frac{27}{2} x^2 e^{-3x} & , \text{ se } x \geq 0 \end{cases} \quad (1)$$

- Sabemos gerar $W \sim exp(1)$ pois basta tomar $W = -\log(1-U)$ onde $U \sim \mathcal{U}(0,1)$.
- A densidade de W é:

$$g(x) = \begin{cases} 0, & \text{se } x < 0 \\ e^{-x}, & \text{se } x \geq 0 \end{cases} \quad (2)$$

- O suporte das duas distribuições é o mesmo, o semi-eixo real positivo.

Exemplo

- Então:

$$0 \leq \frac{f(x)}{g(x)} = \frac{\frac{27}{2}x^2e^{-3x}}{e^{-x}} = \frac{27}{2}x^2e^{-2x} \tag{3}$$

- Derivando e igualando a zero temos ponto de máximo $x_0 = 1$.
- Como $\frac{f(1)}{g(1)} = \frac{27}{2}1^2e^{-2} = 1.827 < 2$, temos $f(x) < 2g(x)$ para todo x.

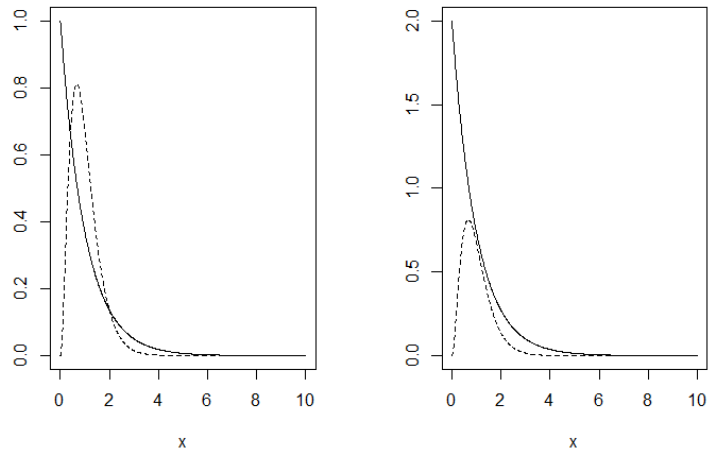


Figura: Esquerda: Densidade-alvo $f(x)$ (linha tracejada) e densidade $g(x)$ de onde sabemos gerar (linha contínua). Direita: Densidade $f(x)$ e a função $2g(x)$.

Script R

```
set.seed(123); M = 2; nsim = 10000
x = rexp(nsim, 1)
razao = dgamma(x, 3, 3)/(M * dexp(x, 1))
aceita = rbinom(10000, 1, razao)
amostra = x[aceita == 1]
par(mfrow=c(2,1))
xx = seq(0, 4, by=0.1); yy = dgamma(xx, 3, 3)
hist(x, prob=T, breaks=50, xlim=c(0, 8),
             main="f(x) e amostra de g(x)")
lines(xx, yy)
hist(amostra, breaks=20, prob=T, xlim=c(0,8),
             main="f(x) e amostra de f(x)")
lines(xx, yy)
```

Resultado

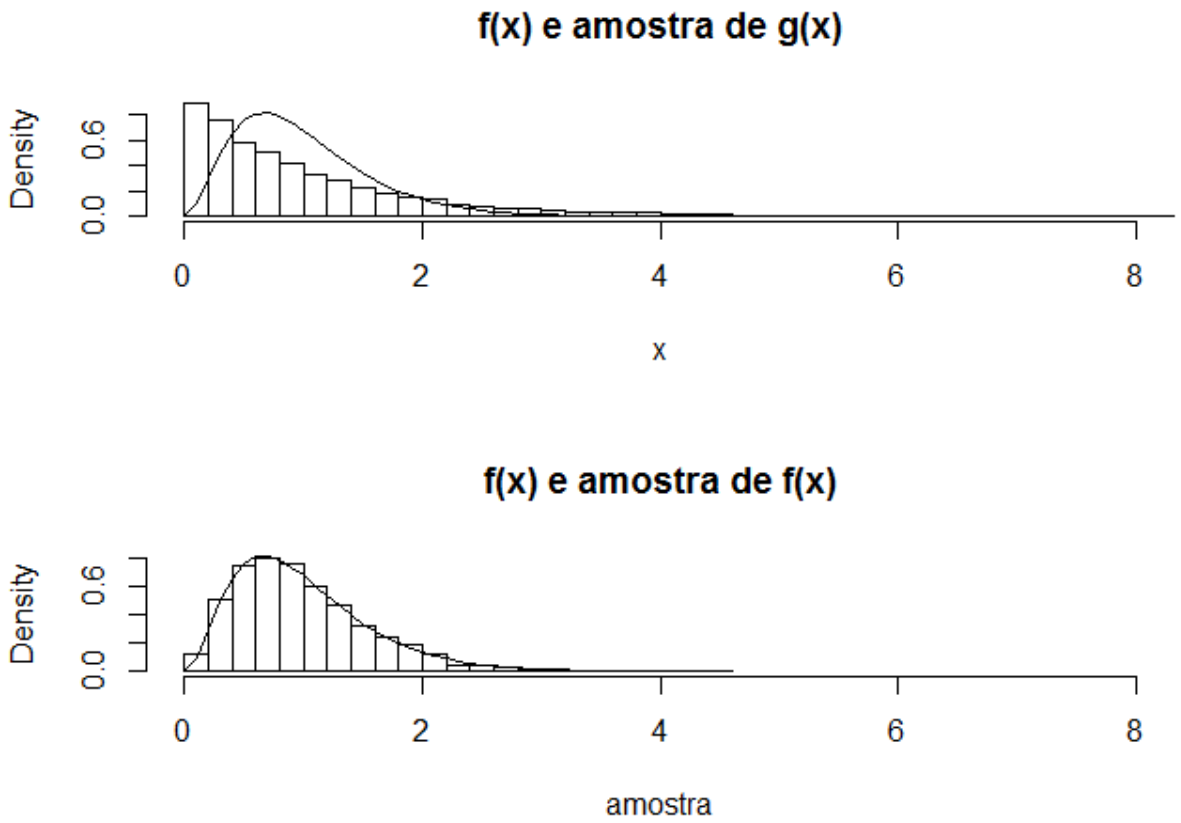


Figura: Amostra de 10 mil valores de uma $g(x) = \exp(1)$; rejeitando aprox 5000 valores terminamos com amostra de $f(x) = \text{Gama}(3, 3)$.

Script R mais simples

```
set.seed(123)
M = 2; nsim = 10000
x = rexp(nsim, 1)
amostra = x[ runif(nsim)<dgamma(x,3,3)/(M*dexp(x, 1)) ]
```

Pseudo-code

1: $l \leftarrow true$
2: **while** l **do**
3: Selecione $U \sim \mathcal{U}(0,1)$
4: Selecione $U^* \sim \mathcal{U}(0,1)$
5: Calcule $\omega = -\log(1-U)$
6: **if** $U^* \leq \frac{f(\omega)}{2g(\omega)} = (27/4)\omega^2 \exp(-2\omega)$ **then**
7: $x \leftarrow \omega$
8: $l = False$
9: **end if**
10: **end while**

Os dois teoremas

Theorem

(Aceitação-Rejeição gera valores de $f(x)$) A variável aleatória X gerada pelo método de aceitação-rejeição possui densidade $f(x)$.

Prova: Leitura opcional, documento disponível no moodle

Theorem

(Impacto de M) O número de iterações necessários até que um valor seja aceito possui distribuição geométrica com valor esperado M.

Prova: Leitura opcional, documento disponível no moodle

Impacto de M

- Método funciona com qualquer M tal que $f(x) \leq Mg(x)$.
- M_1 é muito maior que M_2, ambos satisfazendo a condição.
- Se rodarmos o método em paralelo com os dois valores de M, aquele com o maior valor rejeitaria mais frequentemente que o método com o M menor.
- Pelo teorema, devemos selecionar, em média, M valores até que aceitemos um deles.
- Quanto menor M, menos rejeição.
- Não é difícil provar que M deve ser maior ou igual a 1.

Impacto de M

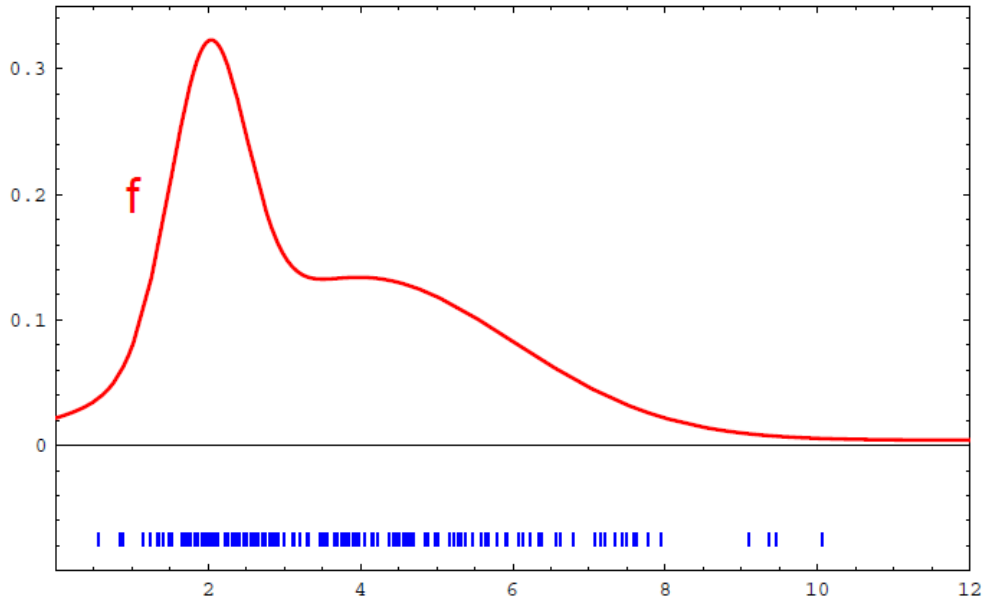
- O máximo de eficiência é obtido quando $M = 1$.
- Mas neste caso, como a área total debaixo de $f(x)$ e $g(x)$ é igual a 1, devemos ter $f(x) = g(x)$.
- Isto é, a densidade de onde geramos é idêntica à densidade-alvo $f(x)$ e todos os valores são aceitos.
- Se selecionarmos $g(x)$ muito diferente de $f(x)$, especialmente se tivermos $g(x) \approx 0$ numa região em que $f(x)$ não é desprezível, é possível que tenhamos de usar um valor de M muito grande para satisfazer $f(x) \leq Mg(x)$ para todo x.
- Esta será uma situação em que o método de aceitação-rejeição será pouco eficiente pois muitas amostras devem ser propostas (em média, M) para que uma delas seja eventualmente aceita).

Amostragem por importância

- Método muito importante para a geração de simultânea de várias variáveis aleatórias relacionadas entre si (correlacionadas): sabemos gerar facilmente de normal multivariada mas não de outras distribuições multivariadas.
- No método de aceitação-rejeição:
  - selecionamos de uma densidade $g(x)$ de onde sabemos amostrar
  - retemos alguns elementos e rejeitamos outros
  - os elementos retidos possuem a densidade desejada $f(x)$
- Na amostragem por importância, selecionamos de $g(x)$ mas retemos tudo, não rejeitamos nada.
- Mas ao usar a amostra, damos um peso diferente e apropriado a cada elemento amostrado.
- No final, isto corrige a distorção de não termos uma amostra de $f(x)$.

Densidade-alvo: o que queremos

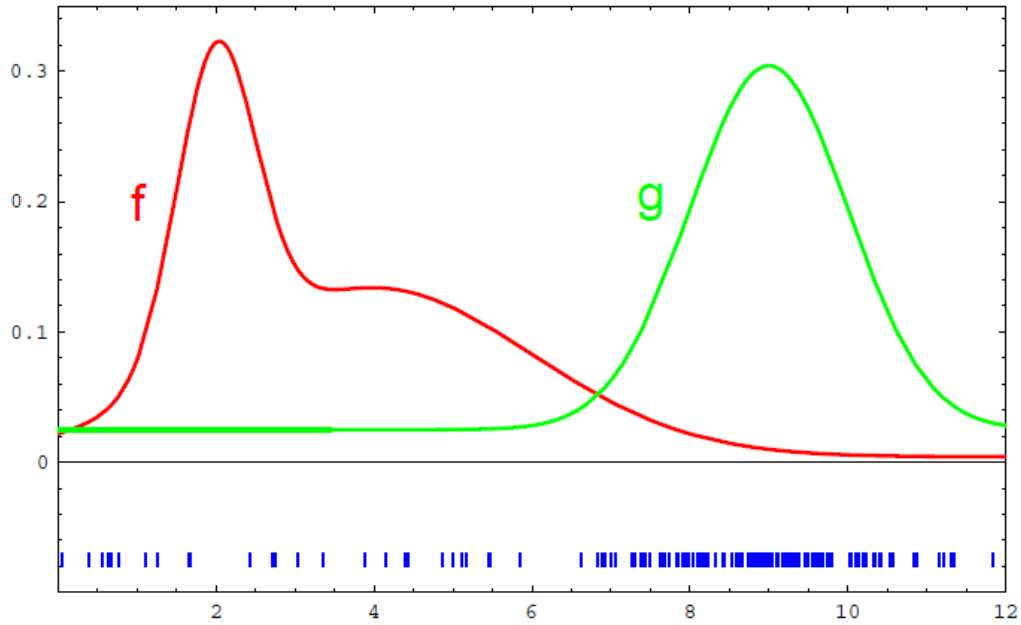
- $f(x)$, a densidade da distribuição-alvo, de onde queremos amostrar.



- O "tapete" de pontos embaixo representa uma amostra de $f(x)$
- Todos os pontos com pesos iguais. Não sabemos obter esta amostra.
- OBS: Esta figura e as duas seguintes vêm do livro *Probabilistic Robotics*

Amostre de $g(x)$ ao invés de $f(x)$

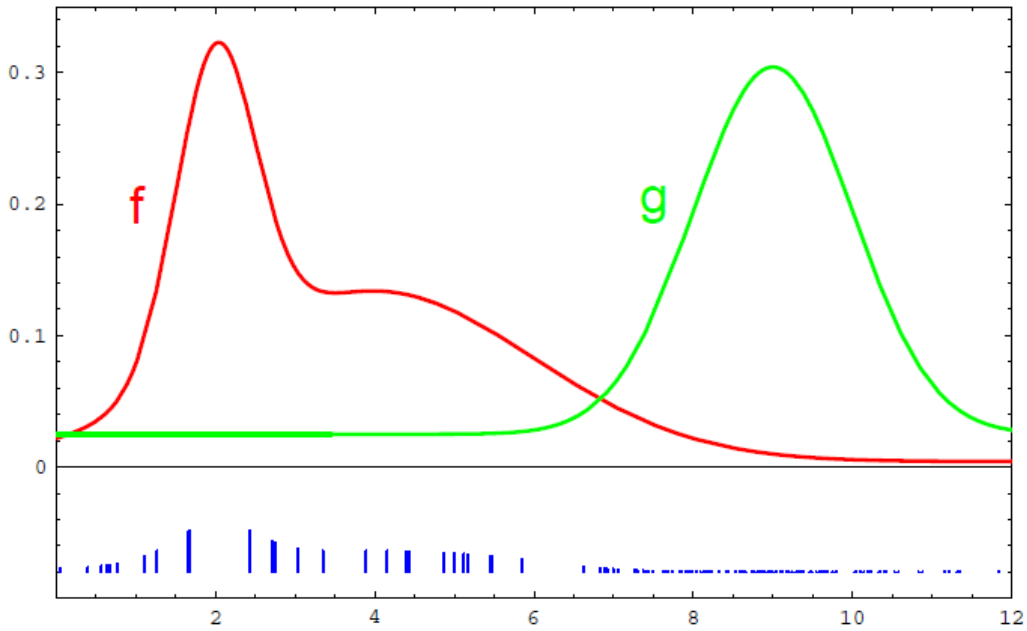
- Amostramos de $g(x)$ em vez de amostrar de $f(x)$.



- Terminamos com a amostra mostrada no "tapete", todos os pontos tem pesos iguais.
- Vamos agora dar pesos diferentes a estes elementos amostrados para que pareçam ter vindo de $f(x)$.
- Intuitivamente, como fazer? Quem recebe mais peso? E menos peso?

Pesos: mais ou menos importância

- Atribuímos pesos $w(x) = f(x)/g(x)$ aos elementos da amostra de $g(x)$.



- Esta amostra PONDERADA pode ser usada para fazer inferência sobre a distribuição $f(x)$
- Como fazer isto exatamente?

O que você quer saber sobre $f(x)$?

- Queremos uma amostra Monte Carlo para estimar (conhecer aproximadamente) alguns aspectos de uma v.a. X com distribuição-alvo $f(x)$.
- Por exemplo, podemos querer saber o seguinte:
  - $\mathbb{E}(X)$ sem precisar fazer a integral (pode ser muito difícil)
  - $\mathbb{V}(X) = \mathbb{E}(X^2) - (\mathbb{E}(X))^2$, a variância de X.
  - $\mathbb{P}(X > 2)$, a chance de observar X maior que 2, um valor-limite importante na aplicação.
  - $\mathbb{P}(e^{-|X|} > |X|)$, um cálculo probabilístico (uma integral).
  - $\mathbb{P}(X \in A)$, onde A é um conjunto complicado.

O truque: escreva como esperança

- Cada uma das quantidades de interesse pode ser escrita como o valor esperado de uma v.a. que é uma função $h(X)$ da v.a. X.
- Seja $\theta_1 = \mathbb{E}(X)$: Tome $h(X) = X$ e então $\theta_1 = \mathbb{E}(h(X))$.
- $\theta_2 = \mathbb{V}(X) = \mathbb{E}(X^2) - (\mathbb{E}(X))^2$: Se $h(X) = X^2$ então $\theta_2 = \mathbb{E}(h(X)) - \theta_1^2$, se tivermos uma estimativa de θ_1

O truque: escreva como esperança

- $\theta_3 = \mathbb{P}(X > 2) = E(h(X))$ onde $h(X) = I_{[X>2]}$, a função indicadora do evento $X > 2$.
- $\theta_4 = \mathbb{P}(e^{-|X|} > |X|) = \mathbb{E}_f(h(X))$ onde $h(X) = I_{[X \in A]}$ onde $A = \{x \text{ tais que } e^{-|x|} > |x|\}$.
- $\theta_5 = \mathbb{P}(X \in A) = \mathbb{E}(I_A)$, onde A é um conjunto complicado.

Estimando esperanças

- Pela idéia frequentista, $\mathbb{E}(X)$ é bem aproximada pela média aritmética de uma grande amostra de valores de X:

$$\mathbb{E}(X) \approx \frac{1}{n} \sum_{i=1}^{n} X_i$$

- Pelo mesmo raciocínio, se quisermos estimar o valor esperado $\mathbb{E}(h(X))$ de uma transformação $h(X)$ de X podemos usar a média aritmética dos $h(X_i)$:

$$\mathbb{E}(h(X)) \approx \frac{1}{n} \sum_{i=1}^{n} h(X_i)$$

- Por exemplo, se $h(X) = X^2$ temos

$$\mathbb{E}(X^2) \approx \frac{1}{n} \sum_{i=1}^{n} X_i^2$$

Estimando esperanças

- Simples: se quiser conhecer o valor esperado de X^2, tome uma amostra de X, aplique a função quadrática a cada valor e tome a sua média aritmética.
- Para estimar o valor esperado de qualquer função $h(X)$, transforme cada valor de uma grande amostra de $X \sim f(x)$ e tome sua média aritmética.
- Problema: não conseguimos gerar $X \sim f(x)$ desejada.
- Sabemos gerar de OUTRA distribuição $g(x)$.
- Aceitação-rejeição joga fora seletivamente vários elementos da amostra de modo a terminar com uma maostra de $f(x)$: é como dar pesos iguais a 0 ou 1 a cada valor.
- Amostragem por importância pondera TODOS os valores amostrados de $g(x)$ com pesos mais flexíveis.

Esperança sob QUAL densidade, g ou f?

- Queremos o valor esperado de $h(X)$ onde $X \sim f(x)$ com suporte $\mathcal{S}$.
- Isto é, queremos $\theta = \mathbb{E}_f(h(X))$.
- SUB-INDICE f para indicar a distribuição de X. A partir de agora,X pode ter densidade $g(x)$ ou $f(x)$ e queremos distinguir isto na notação.
- Sabemos gerar apenas de $g(x)$, com suporte maior ou igual a $\mathcal{S}$.
- Vamos mostrar que $\theta = \mathbb{E}_f(h(X))$ pode ser visto como a esperança de OUTRA funąão $h^*(X)$ quando X tem densidade g.
- Isto é, vamos mostrar que

$$\theta = \mathbb{E}_f(h(X)) = \theta = \mathbb{E}_g(h^*(X))$$

Por que o algoritmo funciona

- O truque mais barato da matemática: multiplique e divida por um memso valor...

$$\begin{aligned}\theta = \mathbb{E}_f(h(X)) &= \int_{\mathbb{R}} h(x)\ f(x)\ \mathrm{d}x \\ &= \int_{\mathbb{R}} \left(h(x)\ \frac{f(x)}{g(x)} \right)\ g(x)\mathrm{d}x \\ &= \int_{\mathbb{R}} h^*(x)\ g(x)\mathrm{d}x\end{aligned}$$

onde $h^*(x) = h(x)f(x)/g(x) = h(x)w(x)$ é uma nova função.

- Assim, podemos reconhecer a última expressão como uma nova esperança: o valor esperado de $h^*(X)$ quando X segue a densidade $g(x)$!!

Por que o algoritmo funciona

- Isto é,

$$\theta = \mathbb{E}_f(h(X)) = \mathbb{E}_g\left(h^*(X)\right) = \mathbb{E}_g\left(h(X)\frac{f(X)}{g(X)}\right) = \mathbb{E}_g\left(h(X)w(X)\right)$$

- Note que, na última esperança, a v.a. X possui distribuição $g(x)$ e não mais $f(x)$!!
- Tudo se resume a multiplicar e dividir por um mesmo valor dentro da integral e reconhecer que a nova integral é uma esperança de uma v.a. $h^*(X)$ onde X tem OUTRA distribuição $g(x)$.

O truque

- Repetindo

$$\theta = \mathbb{E}_f(h(X)) = \mathbb{E}_g\left(h(X)\frac{f(X)}{g(X)}\right) = \mathbb{E}_g\left(h(X)w(X)\right)$$

- A esperança de $h(X)$ com $X \sim f(x)$ é igual à esperança de $h^*(X) = h(X)w(X)$ onde $X \sim g(x)$.
- Como isto pode ser útil?
- Como sabemos amostrar de $X \sim g(x)$, a última esperança $\mathbb{E}_g\left(h(X)w(X)\right)$ pode ser estimada facilmente.

Exemplo

- Desejamos $\mathbb{E}_f(X)$ onde $X \sim \text{Gama}(3,3)$. Neste caso, $h(X) = X$.
- Geramos 200 valores de uma exp(1)
- Para cada um dos 200 valores $x_1, x_2, \ldots x_{200}$ calculamos os pesos

$$w(x_i) = \frac{f(x_i)}{g(x_i)} = \frac{\frac{27}{2} x_i^2 e^{-3x_i}}{e^{-x_i}} = \frac{27}{2} x_i^2 e^{-2x_i}$$

- Com estes pesos, estimamos

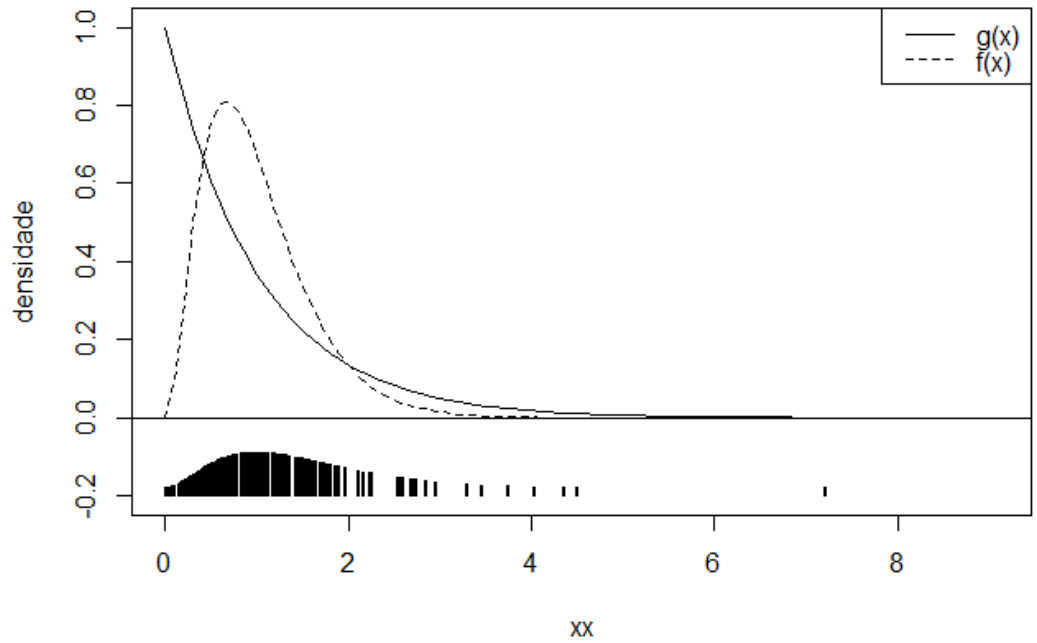
$$\mathbb{E}_f(X) = \mathbb{E}_g(h(X)\ w(X)) = \mathbb{E}_g(X\ w(X)) \approx \frac{1}{200} \sum_{i=1}^{200} x_i w(x_i)$$

- Com o script a seguir, obtive uma estimativa igual a 1.102366 quando o valor exato é igual a 1.

Script R

```
set.seed(123)
nsim = 200
x = rexp(nsim, 1)
wx = dgamma(x, 3, 3)/dexp(x, 1)
theta1 = mean(x*wx)
par(mfrow=c(1,1))
xx = seq(0, 9.1, by=0.1)
fx = dgamma(xx, 3, 3)
gx = dexp(xx, 1)
plot(xx, gx, type="l", ylim=c(-0.2, 1), ylab="densidade")
lines(xx, fx, lty=2)
abline(h=0)
segments(x, -0.2, x, -0.18+wx/20, lwd=2)
legend("topright",lty=1:2,c("g(x)", "f(x)") )
```

Saída do script R



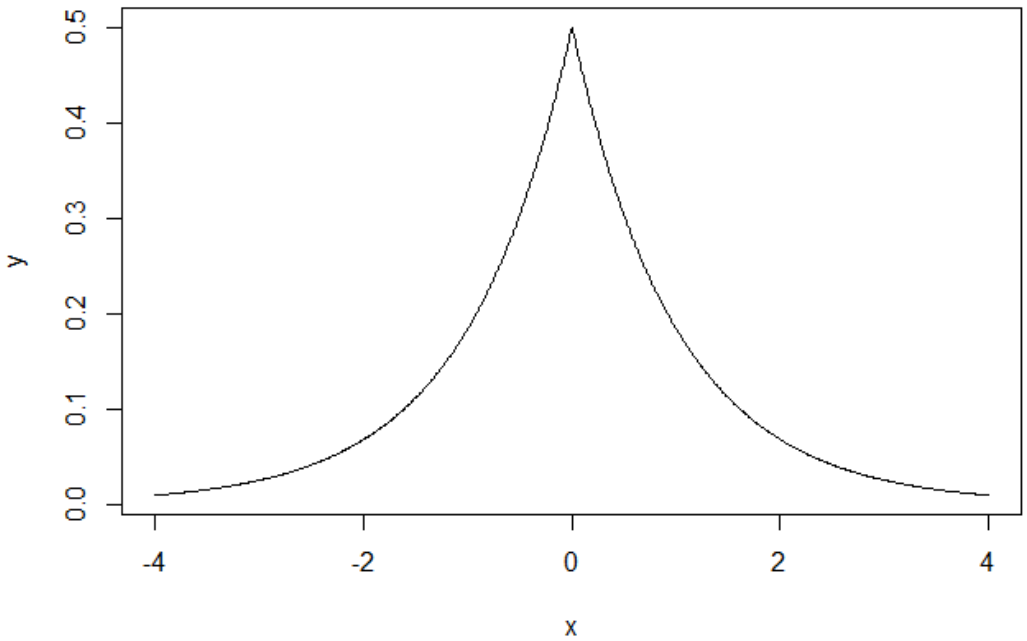
Exemplo

- Queremos gerar de uma $N(0,1)$ sem usar Box-Muller.
- Precisamos simular de uma distribuição com suporte na reta real.
- Sabemos gerar com facilidade uma v.a. Y com distribuição exp(1): $Y = -\log(U)$ onde $U \sim U(0,1)$.
- Problema: exp(1) possui suporte $(0, \infty)$ e normal possui suporte na reta inteira.
- Truque: selecionamos Y exp(1). A seguir, jogue uma moeda com probabilidade $1/2$: se cara, tome Y; se coroa, tome $-Y$.
- Esta distribuição é chamada de exponencial dupla ou distribuição de Laplace.
- Ver `http://en.wikipedia.org/wiki/Laplace_distribution`

Laplace ou exponencial dupla

- Densidade de Laplace padrão ($\mu = 0$ e $b = 1$):

$$g(x) = \frac{1}{2} e^{-|x|}$$



Exemplo

- Queremos calcular $\mathbb{E}_f(h(X))$ onde $X \sim f(x) = N(0,1)$
- Sabemos gerar de Laplace padrão.
- Queremos estimar:
  - $0 = \theta_1 = \mathbb{E}_f(X)$ onde $h(X) = X$
  - $1 = \theta_2 = \mathbb{V}_f(X) = \mathbb{E}_f(X^2) - (\mathbb{E}_f(X))^2$: Se $h(X) = X^2$ então $\theta_2 = \mathbb{E}_f(h(X)) - \theta_1^2$, se tivermos uma estimativa de θ_1
  - $0.02275 = \theta_3 = \mathbb{P}_f(X > 2) = \mathbb{E}_f(h(X))$ onde $h(X) = I_{[X>2]}$, a função indicadora do evento $X > 2$.
  - $\theta_4 = \mathbb{P}(e^{-|X|} > |X|) = E(h(X))$ onde $h(X) = I_A(X)$ onde $A = \{x \text{ tais que } e^{-|x|} > |x|\}$

Exemplo

- Simule $X_1, X_2, \ldots, X_B$ de uma Laplace.
- Calcule os pesos

$$w(x_i) = \frac{f(x_i)}{g(x_i)} = \frac{\frac{1}{\sqrt{2\pi}} e^{-x_i^2/2}}{\frac{1}{2} e^{-|x_i|}}$$

- Em seguida, estime

$$\theta_1 \approx \widehat{\theta_1} = \frac{1}{B} \sum_i x_i w_i$$

$$\theta_2 \approx \widehat{\theta_2} = \frac{1}{B} \sum_i x_i^2 w_i - \left(\widehat{\theta_1}\right)^2$$

$$\theta_3 \approx \widehat{\theta_3} = \frac{1}{B} \sum_i I_{[x_i > 2]} w_i = \text{média dos } w_i \text{ em que } x_i > 2$$

$$\theta_4 \approx \widehat{\theta_4} = \frac{1}{B} \sum_i I_{[e^{-|x_i|} > x_i]} w_i = \text{média dos } w_i \text{ em que } e^{-|x_i|} > x_i$$

Script R

```
B = 10000
x <- (2*(runif(B) > 0.5)-1) * rexp(B) # gerando de uma exponencial dupla
hist(x)
## estimativa via importance sampling
peso <- dnorm(x)/(exp(-abs(x))/2)
a1 <- mean(x * peso)
a2 <- mean( (x^2 *peso) ) - a1^2
a3 <- mean( (x > 2) * peso )
a4 <- mean( (exp(-abs(x)) > abs(x)) * peso )
c(a1, a2, a3, a4) # [1] -0.02186748  1.00444225  0.02259828  0.42595014
## Refazendo com B maior
B = 50000
x <- (2*(runif(B) > 0.5)-1) * rexp(B) # gerando de uma exponencial dupla
peso <- dnorm(x)/(exp(-abs(x))/2)
a1 <- mean(x * peso); a2 <- mean( (x^2 *peso) ) - a1^2;
a3 <- mean( (x > 2) * peso ); a4 <- mean( (exp(-abs(x)) > abs(x)) * peso )
c(a1, a2, a3, a4) # [1] 0.0004117619 0.9879179922 0.0222652043 0.4345303436
```

Reamostragem da amostragem por importância

- Sampling importance resampling (SIR)
- Para usar amostragem por importância, precisamos conhecer as densiaddes $f(x)$ e $g(x)$, incluindo as suas constantes de integração c_1 e c_2:

$$\begin{aligned} f(x) &= c_1 f_0(x) \\ g(x) &= c_2 g_0(x) \end{aligned}$$

- O algoritmo SIR dispensa o conhecimento de c_1 e c_2
- Isto nã é muito relevante nos casos de v.a. unidimensionais mas se quisermos gerar VETORES de v.a.'s correlacionadas, este problema aparece como uma grande dificuldade.

Algoritmo SIR

- Simule uma amostra $X_1, X_2, \ldots, X_B$ de $g(x)$
- Calcule os pesos $w_i = f_0(x_i)/g_0(x_i)$
- Normalize os pesos $w_i \leftarrow w_i/S$ onde $S = \sum_k w_k$
- REAMOSTRE os B dados da amostra original com reposição e com pesos w_i gerando $X_1^*, X_2^*, \ldots, X_n^*$
- Cada X_j^* assume um dos valores $X_1, X_2, \ldots, X_B$ com probab $w_1, w_2, \ldots, w_B$.

SIR

- Mostra-se que a distribuição de X_j^* tem densidade aproximadamente igual a $f(x)$.
- SIR começa gerando de $g(x)$ como importance sampling.
- Ao invés de reter todos os valores gerados atribuindo um peso...
- SIR REAMOSTRA os valores gerados com um peso.
- Voltando ao exemplo anterior, queremos estimar $\theta_1 = \mathbb{E}_f(X)$, $\theta_2 = \mathbb{V}_f(X)$, $\theta_3 = \mathbb{P}_f(X > 2)$, $\theta_4 = \mathbb{P}_f(e^{-|X|} > |X|)$. onde $X \sim f(x) = N(0, 1)$.
- Temos $f(x) = (2\pi)^{-1/2} e^{-x^2/2} \propto e^{-x^2/2}$
- Vamos SUPOR que não conhecemos a constante $(2\pi)^{-1/2}$
- Amostramos $X_1, \ldots X_B$ de $g(x)$, uma Laplace padrão.

SIR

- Reamostramos m elementos
- Reamostra $X_1^*, X_2^*, \ldots, X_m^*$ i.i.d com

$$X_j^* = \begin{cases} X_1 & \text{com probab } w_1 \\ \vdots & \\ X_B & \text{com probab } w_B \end{cases}$$

- No final, calculamos uma média aritmética simples de $h(X_j^*)$:

$$\widehat{\theta} = \frac{1}{m} \sum_{j=1}^{m} h(X_j^*)$$

- Ver script R

Script R

```
  B = 20000
## amostra de exponencial dupla (ou Laplace)
x <- (2*(runif(B) > 0.5)-1) * rexp(B)
## estimativa via SIR - sampling importance resampling
peso <- exp(-x^2/2)/(exp(-abs(x))/2)
peso <- peso/sum(peso)
xstar <- sample(x, 10000, replace=T, prob=peso)
a1 <- mean(xstar)
a2 <- mean(xstar^2) - a1^2
a3 <- mean(xstar > 2)
a4 <- mean(exp(-abs(xstar)) > abs(xstar))
c(a1, a2, a3, a4)
```

Escolha de $g(x)$

- Nos métodos de aceitação-rejeição, importance sampling e SIR geramos de $g(x)$ mas o objetivo é estimar quantidades associadas com $f(x)$.
- Como deve ser escolhida $g(x)$?
- Ela deve ter um suporte maior ou igual a $f(x)$.
- Além disso, ela deve ser o mais parecida possível com $f(x)$.
- Uma má escolha para $g(x)$ põe muita massa de probabilidade numa região (de onde amostramos frequentemente) e esta região tem baixa probabilidade sob $f(x)$.
- Pior: região onde $f(x)$ põe massa de probabilidade tem pouca chance de ser selecionada sob $g(x)$