Introdução à Simulação de Monte Carlo com Python

1 Geração de números aleatórios.

1.1 O método da inversão.

A técnica no método da inversão consiste em simular a variável aleatória X com função de distribuição de probabilidade $f(x)$ mediante ao uso da variável aleatória uniforme $U(0,1)$, e a inversão de $F(x)$. Considere uma função densidade de probabilidade, $f(x)$, normalizada sobre o intervalo (a,b). Seja a correspondente função de distribuição de probabilidade e a função $F(u)$ dada pela seguinte integral:

$$F(u) = \int_a^u f(x)dx \tag{1}$$

Onde u é um valor a ser determinado.
Sabemos que:

$$\int_a^b f(x)dx = 1 \tag{2}$$

Se gerarmos um número aleatório U uniforme entre $[0:1]$ e definirmos a igualdade:

$$U = F(u) = \int_a^u f(x)dx \tag{3}$$

Como U é equiprovável no intervalo $[0:1]$ e como $U = F(u)$ onde $u = F^{-1}(U)$, u é amostrado equiprovavelmente, mas à medida que u varia, $F(u)$ varia de acordo com o comportamento de $f(x)$, veja a figura abaixo:

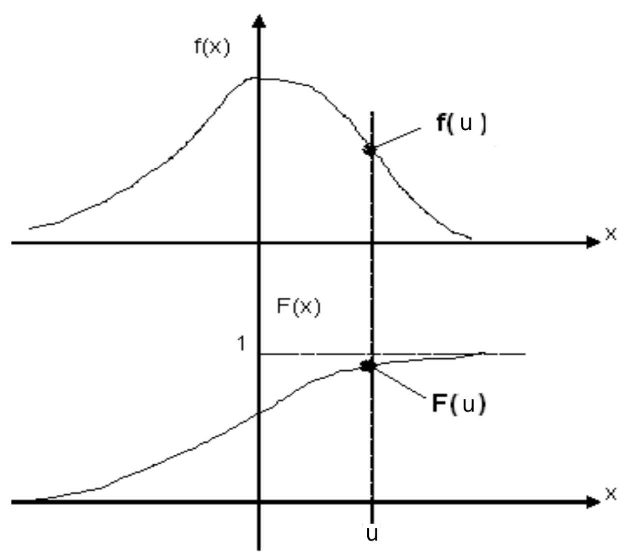

Figura 1:

Para calcular a inversa numericamente use uma adaptação do método de newton, como mostra a seguir. Use $u_0 = U$.

$$u_{i+1} = u_i - \frac{F(u_i) - U}{F'(u_i)} \tag{4}$$

Para o calculo de $F'(u_i)$, fizemos o uso da aproximação da definição de derivada:

$$F'(u_i) = \lim_{h \to 0} \frac{F(u_i + h) - F(u_i)}{h} \tag{5}$$

$$\lim_{h \to 0} \frac{F(u_i + h) - F(u_i)}{h} \approx \frac{F(u_i + h) - F(u_i)}{h} \tag{6}$$

Para $h \approx 0$.

1.2 Algoritmo

Dada a $f(x)$.

Passo 1 : Obtenha $F(x)$ a partir de $f(x)$;
Passo 2 : Gere $U(0, 1)$.
Passo 3 : Faça $F^{-1}(U)$ e obtenha u.
Passo 4 : Repita os Passos anteriores 2 e 3, N vezes para gerar uma amostra de tamanho N.

2 Projetos

Projeto - 1: Gere uma amostra aleatória de tamanho n da distribuição $exp(\theta)$. Para isso adapte o método de Newton para calcular a inversa numericamente. Para testar se seu algoritmo funciona, gere uma amostra muito grande $n = 100000$ e compare os 4 primeiros momentos amostrais com os populacionais. Ou seja:

$$E(X) = \frac{1}{\theta} \quad \approx \quad \frac{1}{n}\sum_{i=1}^{n} X_i \tag{7}$$

$$E(X^2) = \frac{2}{\theta^2} \quad \approx \quad \frac{1}{n}\sum_{i=1}^{n} X_i^2 \tag{8}$$

$$E(X^3) = \frac{6}{\theta^3} \quad \approx \quad \frac{1}{n}\sum_{i=1}^{n} X_i^3 \tag{9}$$

$$E(X^4) = \frac{24}{\theta^4} \quad \approx \quad \frac{1}{n}\sum_{i=1}^{n} X_i^4 \tag{10}$$

(11)

Note que você terá que fixar um valor de θ, por exemplo $\theta = 0.6$.

DICA: Use os códigos `mmc_exp.py` e `mmc_exp.py`.
Projeto - 2: Realize uma simulação de Monte Carlo para testar o estimador de (α, β) da distribuição *Birnbaum-Saunders* dado pelo sistema abaixo. Para gerar uma amosta $(X_1, ..., X_i, ...X_n)$, $X \sim BS(\alpha, \beta)$ com dessa distribuição use a seguinte equação:

$$X_i = \beta\left[\alpha Z_i/2 + \sqrt{(\alpha Z_i/2)^2 + 1}\right]^2 \tag{12}$$

Em que $Z_i \sim N(0, 1)$.

$$\begin{cases} \frac{1}{n}\sum_{i=0}^{n} X_i - \widehat{\beta}(1 + \frac{\widehat{\alpha}^2}{2}) & = \quad 0 \\ \frac{1}{n}\sum_{i=0}^{n} X_i^2 - \widehat{\beta}^2(1 + 2\widehat{\alpha}^2 + \frac{3\widehat{\alpha}^4}{2}) & = \quad 0 \end{cases} \tag{13}$$

DICA: Use o código `mmc_gamma.py` como base.