13
Referências Bibliográficas

ANDERSEN BROTHERTON-RATCLIFF, **"Exact exotics"**, 1996.

BLACK F. & SCHOLES M. "**The Pricing of Options & Corporate Liabilities**", The Journal of Political Economy. 1973

BOYLE BROADIE GLASSERMAN, **"Monte carlo methods for security pricing"**. Journal of Economic Dynamics and Control, 1997.

COX ROSS RUBINSTEIN, **"Option Pricing: A Simplified Approach."** Journal of Financial Economics, 1979.

DOUADY, **"Monte-Carlo Path Weighing"**. Working paper (CIBC and Ecole Normale Superieure of Cachan), 1999.

DUPIRE B, **"Monte-Carlo Methodologies and Applications for Pricing and Risk Management"**,1998.

FISHMAN, **"Monte Carlo: Concepts, Algorithms, and Applications"**, 1996.

GLASSERMAN, **"Monte Carlo Methods in Financial Engineering"**, 2004.

GRANT D, G. VORA, D. WEEKS. "**Simulation and the early-exercise option problem. Journal of Financial Engineering"**, 1996.

HEYNEN R., & KAT H. "**Lookback options with Discrete and Partial Monitoring of the Underlying Price**", Applied Mathematical Finance, Vol 2, p273-283. 1995

HIGHAM DESMOND J., **"An Introduction to Financial Option Valuation"**, 2004.

HULL JOHN C., **"Options, Futures and Other Derivatives"**, 2000.

KEMNA A. G. Z. & VORST A. C. F., "**A Pricing Method for Options Based on Average Asset Values**", Journal of Banking and Finance, 14, p113-129, 1990

LONGSTAFF SCHWARTZ: **"Valuing American Options by Simulation: A Simple Least-Squares Approach"**, Review of Financial Studies vol. 14, 2001

REINER E. & RUBINSTEIN M., **"Unscrambling the Binary Code"**. 1991

RUBINSTEIN M., "**Options for the Undecided"**, 1991

ZHOU AGLEHOLE DYBVIG , **"Going to extremes: Correcting simulation in exotic option valuation"**. Financial Analysts Journal, 1997.

PUC-Rio - Certificação Digital Nº 0611773/CA

Apêndice 1

Calculo Estocástico

Para um titulo S que segue um movimento geométrico browniano:

$$dS = \mu.S.dt + \sigma.S.dZ \qquad \textbf{(1)}$$

μ: taxa ajustada ao risco do titulo

σ: volatilidade do titulo

dt: incremento de tempo

dZ: incremento de Wiener (dZ segue N(0,dt))

e trocando μ por r a taxa livre de risco na equação (1),

$$dS = r.S.dt + \sigma.S.dZ \qquad \textbf{(2)}$$

assim podemos escrever:

$$d(\ln(S)) = \frac{dS}{S} = r.dt + \sigma.dZ \qquad \textbf{(3)}$$

Para um derivativo $F(S,t) = \ln(S)$ seguindo um processo d'Itô temos:

$$dF(S,t) = \frac{\partial F}{\partial S} dS + \frac{\partial F}{\partial t} dt + \frac{1}{2} \frac{\partial^2 F}{\partial S^2} dS^2 \qquad \textbf{(4)}$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Analise do incremento de Wiener dZ :

Os termos dt^i com $i > 1$ são considerados nulos, considerando as expressões truncadas na ordem 1 somente.

$$dt^i = 0 \ \textbf{para} \ i > 1 \qquad \textbf{(5)}$$

Calcular $(dS)^2$:

$$(dS)^2 = (r.S.dt + \sigma.S.dZ)^2 = r^2.S^2.dt^2 + 2.r.\sigma.S^2.dt.dZ + \sigma^2.S^2.dZ^2 \qquad \textbf{(6)}$$

Distribuição de dZ :

dZ segue por definição uma distribuição normal, media 0 e variância dt : seja $N(0, dt)$.

Distribuição de dZ^2:

Sabendo a distribuição acima, podemos escrever:

$$Var(dZ) = E\left[dZ^2\right] - \left(E\left[dZ\right]\right)^2 = E\left[dZ^2\right] - 0 = dt \qquad \textbf{(7)}$$

Então

$$E\left[dZ^2\right] = dt \quad \textbf{(8)}$$

$$Var(dZ^2) = E\left[(dZ^2)^2\right] - \left(E\left[dZ^2\right]\right)^2 = E\left[dZ^4\right] - \left(E\left[dZ^2\right]\right)^2 = 3.dt^2 - dt^2 = 2.dt^2$$

$$Var(dZ^2) = 0 \ \textbf{(9)}$$

Então com (8) e (9), dZ^2 segue a distribuição $N(dt, O)$, seja é determinístico.

$$dZ^2 = dt \qquad \textbf{(10)}$$

Finalmente usando (5) e (10), (6) da:

$$(dS)^2 = \sigma^2.S^2.dt \qquad \textbf{(11)}$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Calculo das derivadas de $F(S,t) = \ln(S)$**:**

$$\frac{\partial F(S,t)}{\partial S} = \frac{\partial(\ln S)}{\partial S} = \frac{dS}{S}$$

$$\frac{\partial^2 F(S,t)}{\partial S^2} = \frac{\partial^2(\ln S)}{\partial S^2} = \frac{\partial}{\partial S}\left(\frac{dS}{S}\right) = -\frac{1}{2}\frac{dS^2}{S^2}$$

$$\frac{\partial F(S,t)}{\partial t} = \frac{\partial(\ln S)}{\partial t} = 0$$

Substituindo as expressões, temos finalmente a formula (4) que define o movimento do derivativo F, e sendo o movimento geométrico browniano do ativo:

$$dF(S,t) = \left(r - \frac{\sigma^2}{2}\right).dt + \sigma.dZ \qquad \textbf{(12)}$$

Terminando o calculo onde "eps" esta uma variável aleatória tirada da distribuição N(0,1):

$$\Delta F(S,t) = \Delta \ln S = \left(r - \frac{\sigma^2}{2}\right)\Delta t + \sigma.eps.\sqrt{\Delta t}$$

$$\Delta \ln S = \ln S_t - \ln S_0 = \ln \frac{S_t}{S_0} = \left(r - \frac{\sigma^2}{2}\right)\Delta t + \sigma.eps.\sqrt{\Delta t}$$

$$\frac{S_t}{S_0} = \exp\left((r - \frac{\sigma^2}{2})\Delta t + \sigma.eps\sqrt{\Delta t}\right)$$

$$\boxed{S_t = S_0 e^{\left(r - \frac{\sigma^2}{2}\right)\Delta t + \sigma.eps.\sqrt{\Delta t}}} \qquad \textbf{(13)}$$

Assim podemos simular o preço S_t para um ativo qualquer, de preço inicio S_0, volatilidade σ e para um tempo futuro distante de Δt .

PUC-Rio - Certificação Digital Nº 0611773/CA

Dividindo o tempo total da simulação T em vários intervalos de tempo "*dt* " todos iguais, cada $S_{n.dt}$ pode ser calculado usando o termo precedente da seguinte maneira:

$$S_{n.dt} = S_{(n-1).dt} e^{\left(r - \frac{\sigma^2}{2}\right)dt + \sigma.eps.\sqrt{dt}} \qquad (14)$$

iniciando com S_0, o preço da ação no instante inicial.

A fórmula com "*dividend yield*" esta quase igual, trocando o r acima para "$r - y$" onde y representa a taxa de dividendos continua da ação:

$$S_{n.dt} = S_{(n-1).dt} e^{\left(r - y - \frac{\sigma^2}{2}\right)dt + \sigma.eps.\sqrt{dt}} \qquad (15)$$

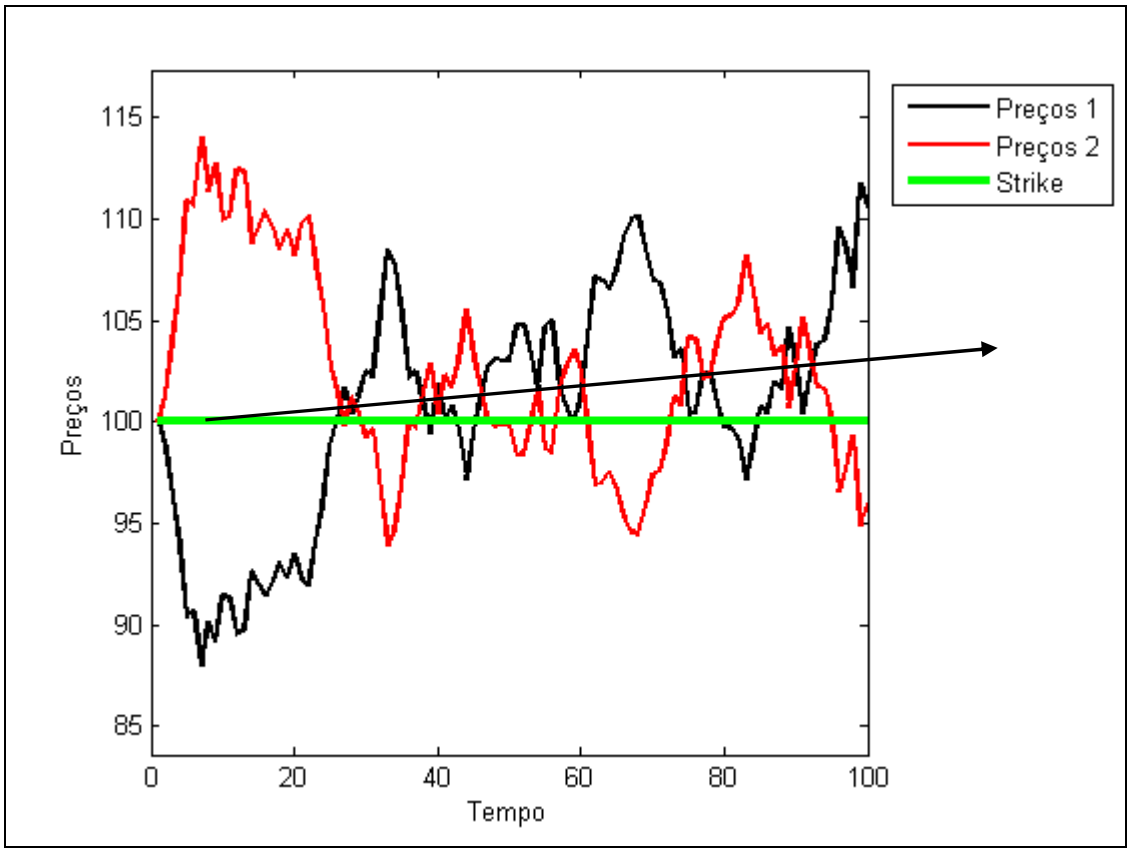
Exemplo 1:

Preços do ativo no inicio	S=100
Strike da opção	k=100
Tempo a expiração	T=1
Taxa livre de risco	r=5%
Volatilidade do ativo	v=20%
Dividend Yield	y=0

PUC-Rio - Certificação Digital Nº 0611773/CA

A simulação dos preços foi feita sobre 100 divisões iguais do tempo de um ano da opção.

O "preços 1" representa o caminho normal, e "preços 2" o caminho "antitético" correspondente.



Podemos ver que ao longo do tempo, a tendência normal do preço esta de aumentar. A simulação dos preços com esses parâmetros tem um "*drift*" como visto no modelo que explica a variação da variável $dF(S,t)$ em função de dt e de dZ. O "*drift*" esta a tendência que ocorre no tempo, seja o parâmetro fator de dt:

$$dF(S,t) = \left(r - y - \frac{\sigma^2}{2}\right).dt + \sigma.dZ \qquad Drift = \left(r - y - \frac{\sigma^2}{2}\right)$$

Nesse exemplo: $$Drift = \left(r - y - \frac{\sigma^2}{2}\right) = \left(0.05 - \frac{0.2^2}{2}\right) = 0.03 = 3\%$$

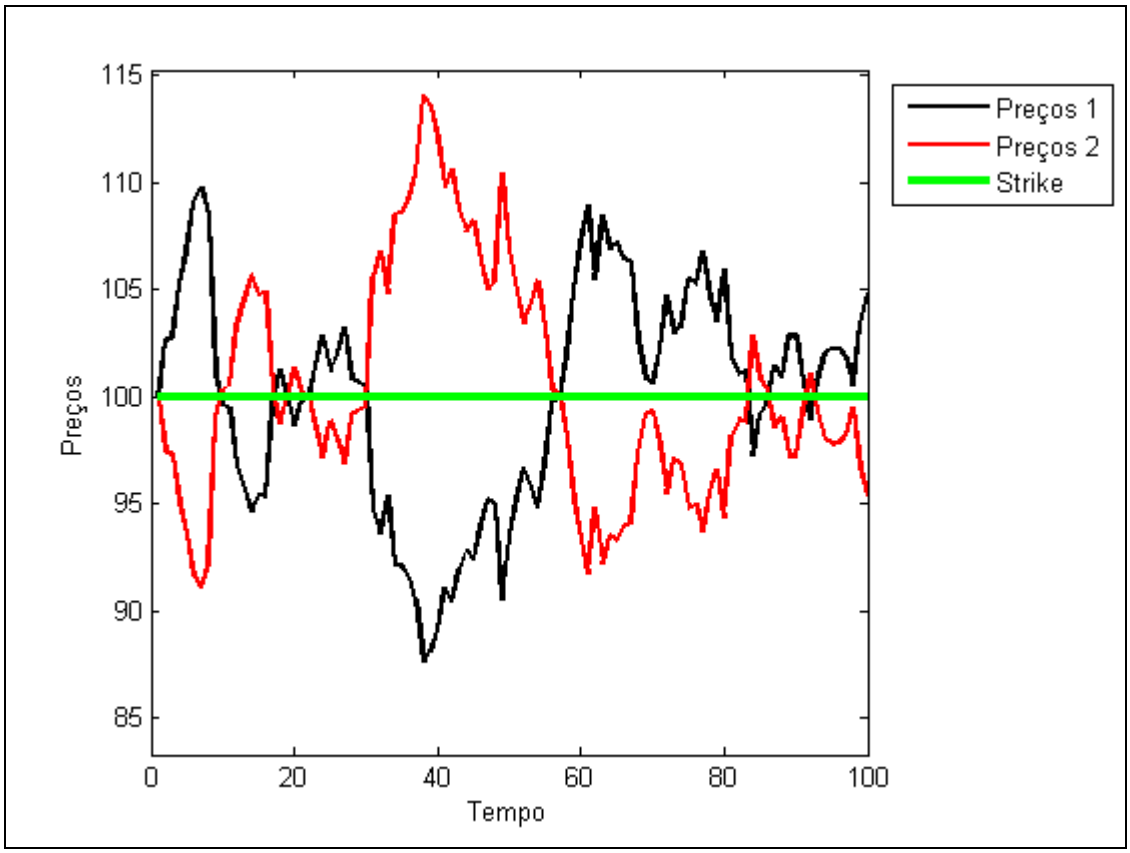
PUC-Rio - Certificação Digital Nº 0611773/CA

Exemplo 2: Sem drift

Usando parâmetros ajustado tais que $Drift = \left(r - y + \frac{\sigma^2}{2} \right) = 0$, seja $r = \frac{\sigma^2}{2}$ com $y = 0$, a média dos dois caminhos antitéticos será exatamente igual a zero.

Preços do ativo no inicio	S=100
Strike da opção	k=100
Tempo a expiração	T=1
Taxa livre de risco	r=2%
Volatilidade do ativo	v=20%
Dividend Yield	y=0

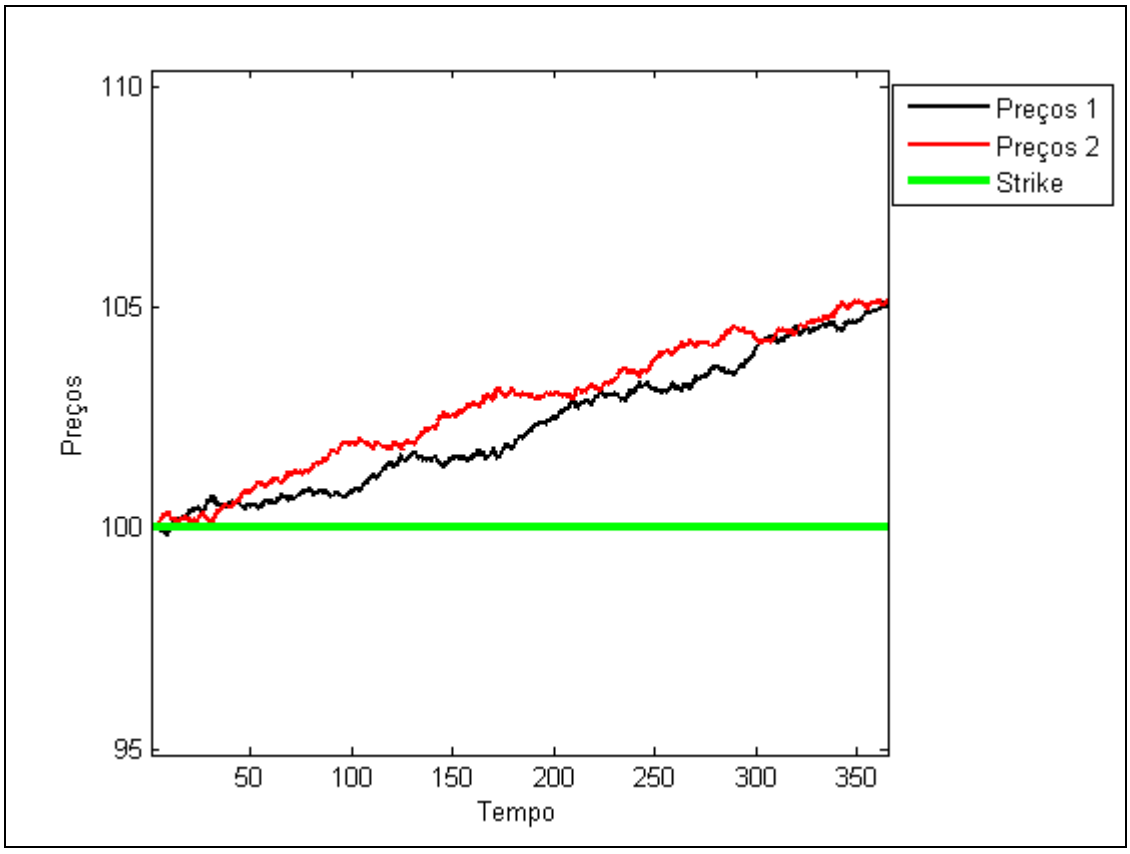
Assim, $r = 2\%$ e $\frac{\sigma^2}{2} = \frac{0.2^2}{2} = 2\%$. Segue o gráfico abaixo onde as duas curvas são perfeitamente simétricas para o valor inicial de 100:



PUC-Rio - Certificação Digital Nº 0611773/CA

Exemplo 3: Volatilidade baixa

Preços do ativo no inicio	S=100
Strike da opção	k=100
Tempo a expiração	T=1
Taxa livre de risco	r=5%
Volatilidade do ativo	v=0.1%
Dividend Yield	y=0



Com volatilidade fraca, os valores simulados ficam mais perto do valor esperado, seguindo o aumento clássica de *5%* da taxa livre de risco, para atingir *105* em 1 ano.

$$Drift = \left(r - y - \frac{\sigma^2}{2} \right) = 0.05 - \frac{0.01^2}{2} \approx 5\%$$

O valor esperado da simulação, com valor *100* no instante inicial, e um drift de *5%* é: $E[\Pr eços] = 100.(1+0.05) = 105$

PUC-Rio - Certificação Digital Nº 0611773/CA

Apêndice 2

Demonstração Black-Scholes

O objetivo é de provar a formula de Black-Scholes para precificação de opções, ou seja:

$$w = x.\underline{N}(d_1) - C.e^{-r(T-t)}.\underline{N}(d_2)$$

Com

$$d_1 = \frac{\ln(X/C) + (r + \frac{v^2}{2})(T-t)}{v.\sqrt{T-t}} \qquad \text{e} \qquad d_2 = \frac{\ln(X/C) + (r - \frac{v^2}{2})(T-t)}{v.\sqrt{T-t}}$$

Onde

w	valor da opção
x	valor do ativo objeto
C	preço de exercício da opção
t	data atual
T	data de exercício da opção
r	taxa livre de risco
v	volatilidade do ativo

PUC-Rio - Certificação Digital Nº 0611773/CA

Queremos achar uma formula para $w(x,t)$ em qualquer instante de tempo. Mas sabemos que no vencimento ($(T=t)$):

$$w(x,T) = \max(x_T - C, 0) \qquad \textbf{(1)}$$

Para encontramos $w(x,t)$, podemos trazer a expressão $w(x,T)$ a valor presente, de forma que sendo neutro ao risco:

$$w(x,t) = e^{-r(T-t)}.E\left[\max(x_T - C, 0)\right] \quad \textbf{(2)}$$

Considerando que a variação do preço das opções siga um movimento geométrico browniano MGB:

$$dx = r.x.dt + \sigma.x.dZ \qquad \textbf{(3)}$$

Onde o incremento de Wiener $dZ \sim N(0, dt)$

Seja $f = f(x)$ uma função apenas da variável estocástica x. Do lema de Itô, teremos:

$$df(x) = \frac{\partial f}{\partial x}dx + \frac{1}{2}\frac{\partial^2 f}{\partial^2 x}dx^2$$

Usando $f(x) = \ln(x)$, e aplicando o lema de Itô:

$$d\ln(x) = \frac{\partial \ln(x)}{\partial x}dx + \frac{1}{2}\frac{\partial^2 \ln(x)}{\partial^2 x}dx^2$$

$$d\ln(x) = \frac{1}{x}dx + \frac{1}{2}\left(-\frac{1}{x^2}\right)dx^2$$

$$d\ln(x) = \frac{1}{x}dx - \frac{dx^2}{2x^2} \quad \textbf{(6)}$$

Do MGB, na equação **(4)** temos:

$$(dx)^2 = (r.x.dt + \sigma.x.dZ)^2 = r^2.x^2.dt^2 + 2.r.\sigma.x^2.dt.dZ + \sigma^2.x^2.dZ^2$$

e finalmente:

$$(dx)^2 = \sigma^2.x^2.dt \qquad \textbf{(7)} \qquad \text{(ver anexo 1 p2 para detalho)}$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Substituindo **(4)** e **(7)** em **(6)**:

$$d\ln(x) = \frac{1}{x}(r.x.dt + v.x.dz) - \frac{1}{2x^2}v^2.x^2.dt$$

$$d\ln(x) = (r - \frac{v^2}{2}).dt + v.dz \qquad \textbf{(9)}$$

Integrando a expressão acima **(9)** de t a T:

$\int_t^T d\ln(x) = \int_t^T (r - \frac{v^2}{2}).dt + \int_t^T v.dz_x$ (a segunda parte é uma integral estocástica que sabemos resolver)

$$\ln(x_T) - \ln(x_t) = (r - \frac{v^2}{2})(T-t) + v.(z_T - z_t) \quad \textbf{(10)} \qquad (\text{onde } (z_T - z_t) \sim N(0, T-t))$$

$$\ln(x_T) = \ln(x_t) + (r - \frac{v^2}{2})(T-t) + v.(z_T - z_t) \quad \textbf{(11)}$$

Seja y_T a função definida para: $y_T = (r - \frac{v^2}{2})(T-t) + v.(z_T - z_t)$ **(12)**

Como $(z_T - z_t)$ é a única parcela estocástica com distribuição $N(0, T-t)$, y_T também terá distribuição normal com parâmetros:

Media:

$$E[y_T] = (r - \frac{v^2}{2})(T-t) + E[v.(z_T - z_t)]$$

$E[y_T] = (r - \frac{v^2}{2})(T-t)$ devido á distribuição da variável estocástica z.

Variância:

$$Var[y_T] = Var[v.(z_T - z_t)]$$

$$Var[y_T] = v^2.Var(z_T - z_t)$$

$$Var[y_T] = v^2.(T-t)$$

y_T tem a distribuição $N((r - \frac{v^2}{2})(T-t), v^2.(T-t))$ **(13)**

PUC-Rio - Certificação Digital Nº 0611773/CA

Ainda sobre y_T , da expressão **(11)**:

$$\ln(\frac{x_T}{x_t}) = y_T \qquad\qquad x_T = x_t.e^{y_T} \qquad \textbf{(14)}$$

Pela teoria das probabilidades temos para quaisquer funções g e h:

$$g(y_T) = h(x_T).\left|\frac{dx_T}{dx_t}\right|$$

Mas temos $\frac{dx_T}{dx_t} = x_T.e^{y_T}$ que é sempre positivo, então

$$g(y_T).dy_T = h(x_T).dx_T \qquad \textbf{(15)}$$

Voltando na definição de $w(x,t)$ **(2)** que queremos calcular:

$$w(x,t) = e^{-r(T-t)}.E\left[\max(x_T - C, 0)\right]$$

Da definição de valor esperado temos:

$$w(x,t) = e^{-r(T-t)}.\int_{-\infty}^{+\infty} \max(x_T - C, 0).f(x_T).d(x_T)$$

Considerando que a integral é nula para $x_T < C$ pela definição da função

$$\max(x_T - C, 0): \qquad w(x,t) = e^{-r(T-t)}.\int_{C}^{+\infty} (x_T - C).f(x_T).d(x_T)$$

E usando **(15)**:

$$w(x,t) = e^{-r(T-t)}.\int_{u}^{+\infty} (x_t.e^{y_T} - C).g(y_T).d(y_T)$$

Onde u é o novo limite inferior para a integral em $d(y_T)$ de forma que:

$u = \ln(C/x_t)$.

Finalmente

$$w(x,t) = e^{-r(T-t)}.\int_{\ln(C/x_t)}^{+\infty} (x_t.e^{y_T} - C).g(y_T).d(y_T)$$

$$w(x,t) = x_t.I_1 - C.e^{-r(T-t)}.I_2 = x_t.e^{-r(T-t)}.\int_{\ln(C/x_t)}^{+\infty} e^{y_T}.g(y_T).d(y_T) - C.e^{-r(T-t)}.\int_{\ln(C/x_t)}^{+\infty} g(y_T).d(y_T)$$

(16)

PUC-Rio - Certificação Digital Nº 0611773/CA

Calcular I_1:

Como y_T tem distribuição normal com parâmetros dados na expressão **(13)**:

$$I_1 = e^{-r(T-t)}.\int_{\ln(C/x_t)}^{+\infty} e^{y_T}.\frac{1}{\sqrt{2\pi.v^2.(T-t)}}.\exp\left[-\frac{1}{2}\left(\frac{y_T-(r-\frac{v^2}{2}).(T-t)}{v.\sqrt{(T-t)}}\right)^2\right].dy_T$$

$$I_1 = \int_{\ln(C/x_t)}^{+\infty} e^{y_T}.\frac{1}{\sqrt{2\pi.v^2.(T-t)}}.\exp\left[-r.(T-t)+y_T-\frac{1}{2}\left(\frac{y_T-(r-\frac{v^2}{2}).(T-t)}{v.\sqrt{(T-t)}}\right)^2\right].dy_T$$

(17)

Abrindo a expressão do expoente de e na integração, teremos:

$$A = -r.(T-t)+y_T-\frac{1}{2}\left(\frac{y_T-(r-\frac{v^2}{2}).(T-t)}{v.\sqrt{(T-t)}}\right)^2$$

$$A = -\frac{1}{2.v^2.(T-t)}\left[\left(y_T-(r-\frac{v^2}{2}).(T-t)\right)^2+(-2).\left(y_T-r.(T-t)\right)v^2.(T-t)\right]$$

$$A = -\frac{1}{2}\left[\frac{y_T-(r+\frac{v^2}{2}).(T-t)}{v.\sqrt{(T-t)}}\right]^2 \qquad \textbf{(18)}$$

Observação:
usando uma distribuição normal com média μ_x e variância $\sigma_x{}^2$, temos:

$$I = \frac{1}{\sqrt{2\pi.\sigma_x{}^2}}\int_b^{+\infty}\exp\left[-\frac{1}{2}\left(\frac{x-\mu_x}{\sigma_x}\right)^2\right].dx$$

fazendo $Z = \frac{x-\mu_x}{\sigma_x}$, $dx = \sigma_x.dZ$. Para $x = b$, temos: $Z = \frac{b-\mu_x}{\sigma_x}$

PUC-Rio - Certificação Digital Nº 0611773/CA

$$I = \frac{1}{\sqrt{2\pi}} \int_{\frac{b-\mu_x}{\sigma_x}}^{+\infty} \exp\left[-\frac{1}{2}(Z)^2\right].dx \qquad \text{e} \qquad I = \int_{\frac{b-\mu_x}{\sigma_x}}^{+\infty} N(Z).dZ = \int_{-\infty}^{\frac{\mu_x - b}{\sigma_x}} N(Z).dZ$$

Onde $N(Z)$ é a função de densidade de probabilidade da $N(0,1)$, função simétrica.

De forma análoga em I_1 temos com esses parâmetros:

$$l = \ln(C/x) \qquad \mu_x = (r + \frac{v^2}{2}).(T-t) \qquad \sigma_x = \sqrt{v^2.(T-t)}$$

$$I_1 = \underline{N}\left(\frac{\ln(x/c) + (r + \frac{v^2}{2}).(T-t)}{\sqrt{v^2.(T-t)}}\right) \quad \textbf{(19)}$$

onde $\underline{N}$ é a função cumulativa da densidade de probabilidade normal N.

$$I_1 = \underline{N}(d_1) \qquad \text{onde} \qquad d_1 = \frac{\ln(x/c) + (r + \frac{v^2}{2}).(T-t)}{\sqrt{v^2.(T-t)}}$$

Calcular I_2:

$$I_2 = \int_{\ln(C/x_t)}^{+\infty} e^{y_T}.\frac{1}{\sqrt{2\pi.v^2.(T-t)}}.\exp\left[-\frac{1}{2}\left(\frac{y_T - (r - \frac{v^2}{2}).(T-t)}{v.\sqrt{(T-t)}}\right)^2\right].dy_T$$

Da mesma forma e usando os parâmetros seguintes:

$$l = \ln(C/x) \qquad \mu_x = (r - \frac{v^2}{2}).(T-t) \qquad \sigma_x = \sqrt{v^2.(T-t)}$$

$$I_1 = N\left(\frac{\ln(x/c) + (r - \frac{v^2}{2}).(T-t)}{\sqrt{v^2.(T-t)}}\right) \quad \textbf{(20)}$$

PUC-Rio - Certificação Digital Nº 0611773/CA

$$I_2 = \underline{N}\left(d_2\right) \qquad \text{onde} \qquad d_2 = \frac{\ln(x/c) + (r - \frac{v^2}{2}).(T-t)}{\sqrt{v^2.(T-t)}}$$

Usando **(16)**, **(19)** e **(20)**:

A fórmula de Black-Scholes

$$w(x,t) = x_t.\underline{N}\left(d_1\right) - C.e^{-r(T-t)}.\underline{N}\left(d_2\right)$$

Onde

$$d_1 = \frac{\ln(x/c) + (r + \frac{v^2}{2}).(T-t)}{\sqrt{v^2.(T-t)}} \qquad d_2 = \frac{\ln(x/c) + (r - \frac{v^2}{2}).(T-t)}{\sqrt{v^2.(T-t)}}$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Apêndice 3

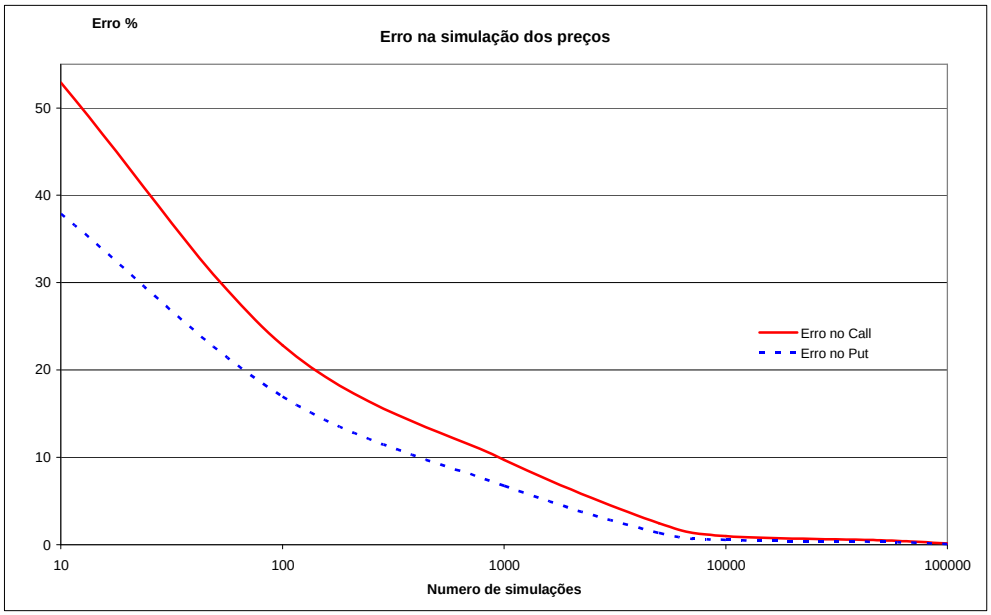
Gráficos de sensibilidade

Opção européia

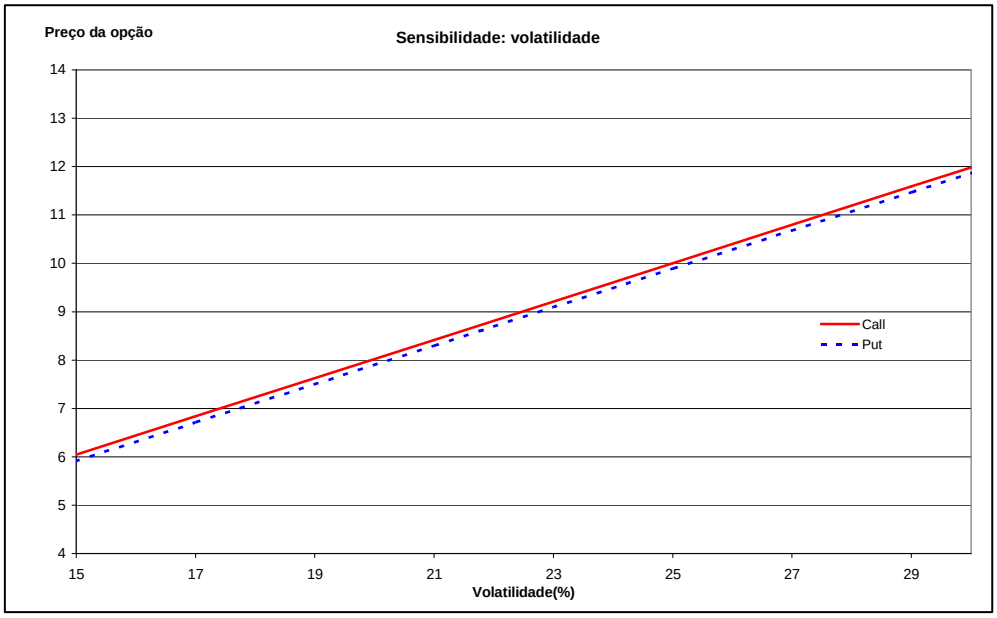
Sensibilidade em relação ao número de simulações

Evolução do erro com o aumento do numero de simulações, em comparação no valor teórico da formula de Black e Scholes.

PUC-Rio - Certificação Digital Nº 0611773/CA



Sensibilidade em relação a mudanças na volatilidade

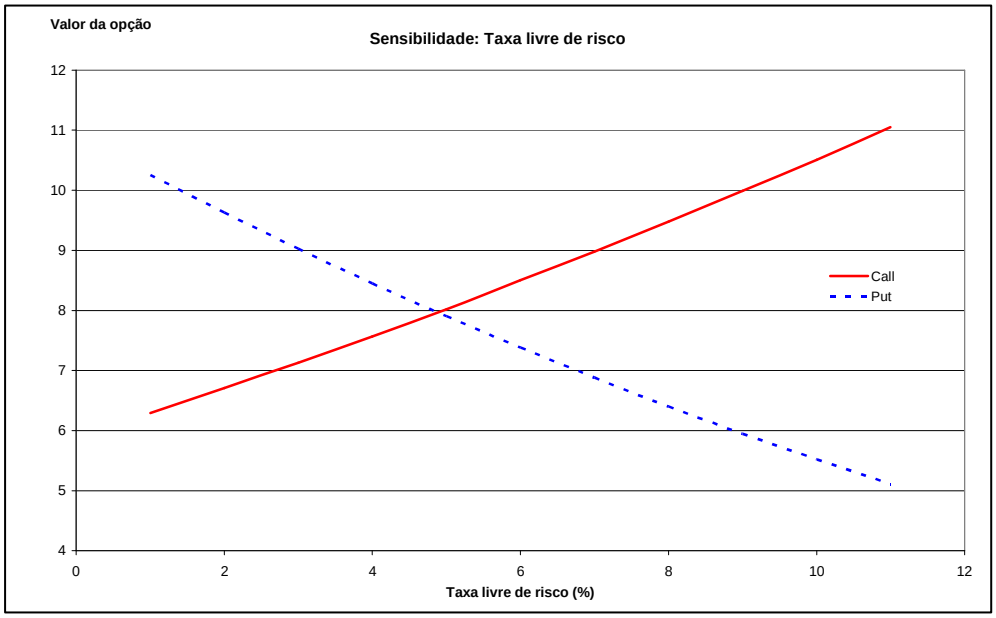


Ilustrando a linearidade com a volatilidade v, uma regressão linear mostra que:

$$\text{Pr}\,eçoCall = 0{,}3951\,.v + 0.107 \qquad \text{Pr}\,eçoPut = 0{,}3953\,.v - 0{,}0167$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação a mudanças na taxa livre de risco



Usando uma regressão quadrática para aproximar as curvas, obtemos:

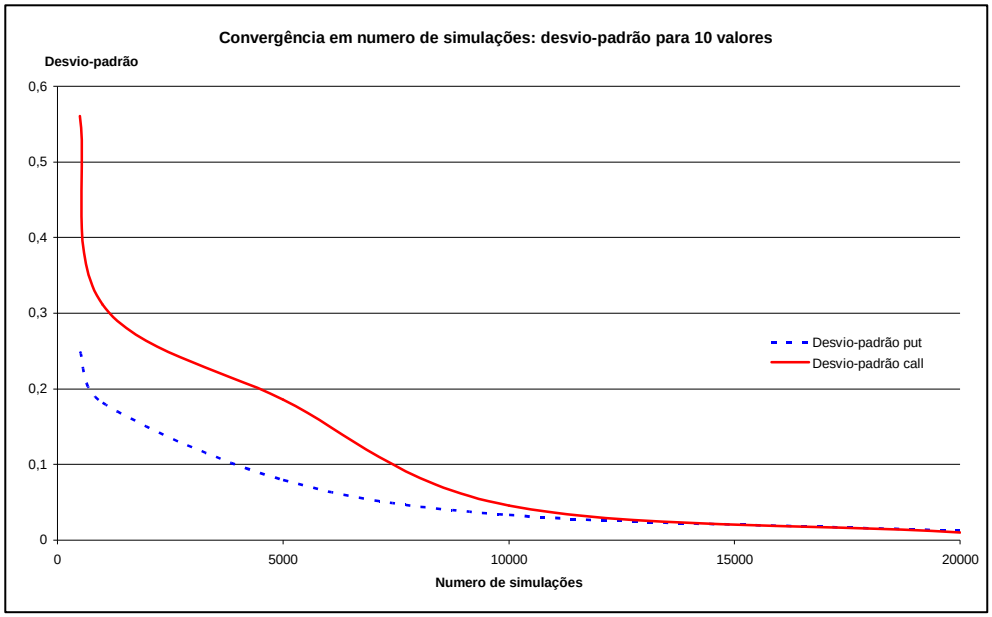
$$\text{PreçoCall} = 0{,}007.r^2 + 0{,}3921.r + 5{,}8922$$
$$R^2 = 1$$

$$\text{PreçoPut} = 0{,}0121.r^2 - 0{,}6591.r + 10{,}897$$
$$R^2 = 1$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção americana

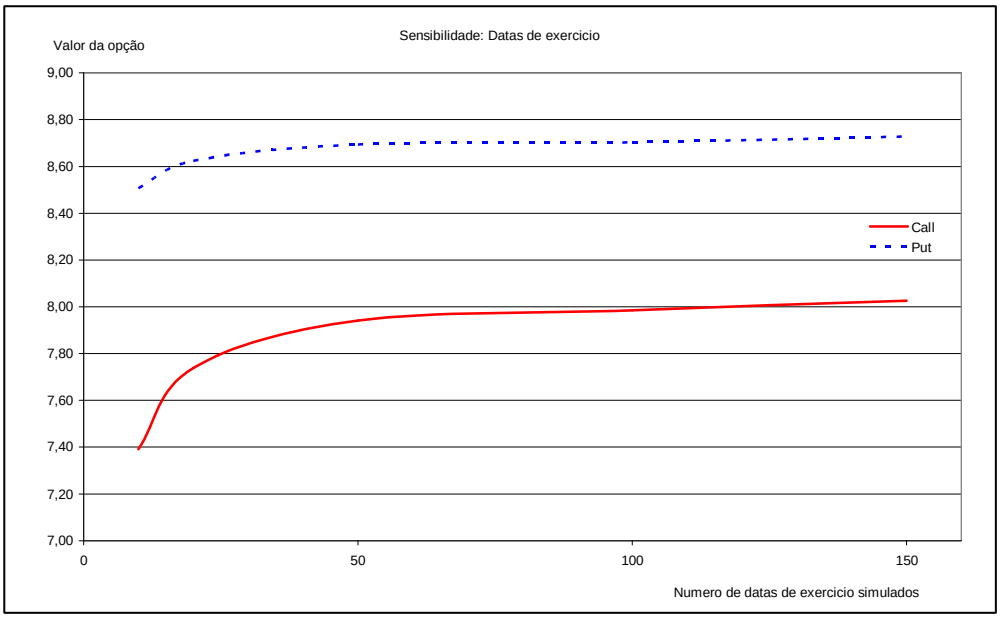
Sensibilidade em mudança no numero de simulações



A precisão no método de Monte-Carlo é obtida através o aumento dos eventos possíveis simulados.

PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em mudança no numero de datas de exercício simuladas

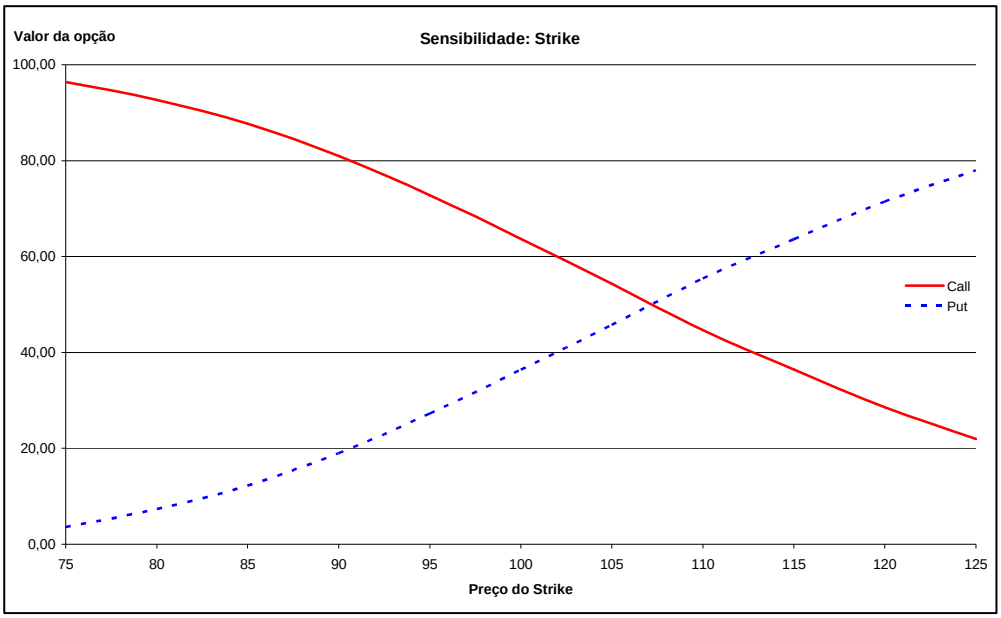


Com o aumento de datas de exercício simuladas, os preços aumentam junto com a flexibilidade de poder exercer em qualquer momento.

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção *asset-or-nothing*

Sensibilidade em relação à mudança no preço do strike



As curvas são simétricas, conseqüentemente a: $Call + Put = C$, com C uma constante. (aqui $C = 100$ nesse exemplo).

Os limites são:

$Strike \to 0$: $Call \to 100$ porque sempre vai ficar "*in-the-money*", sendo o preço do ativo maior que o strike

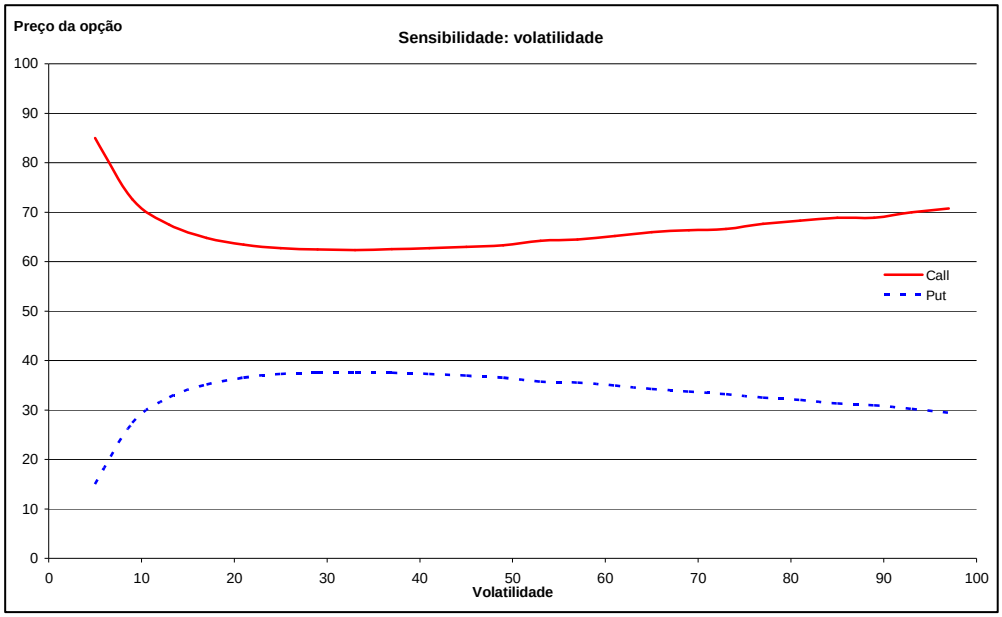
$Put \to 0$ porque sempre "*out-of-the-money*", pela mesma razão.

$Strike \to +\infty$: $Call \to 0$ O preço do ativo tem maior probabilidade de ficar menor que o strike.

$Put \to 100$ pela mesma razão.

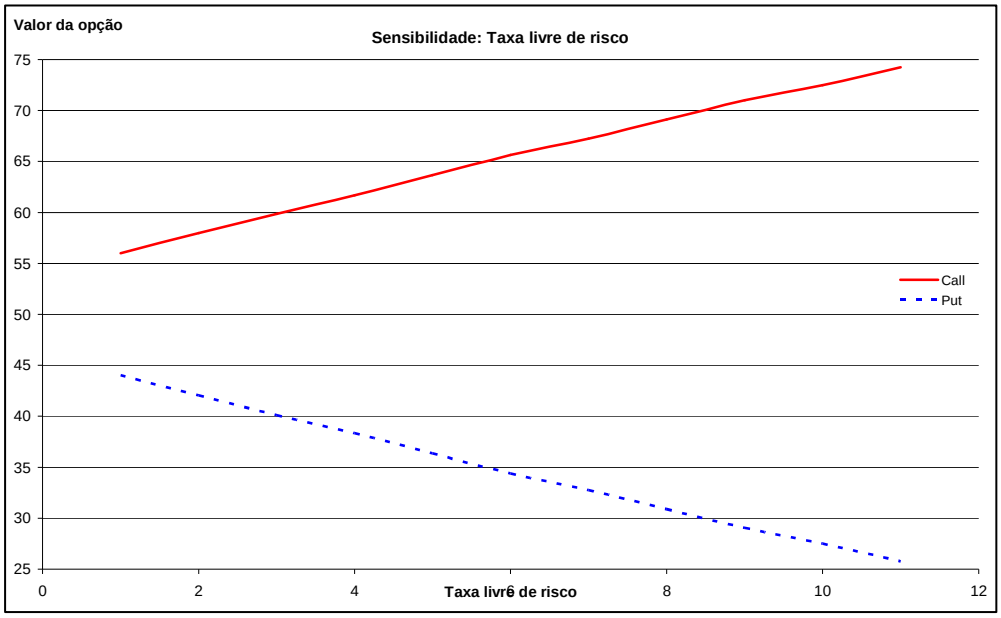
PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação à mudança na volatilidade



PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação à mudança na taxa livre de risco



Usando uma regressão linear, obtemos as equações seguintes:

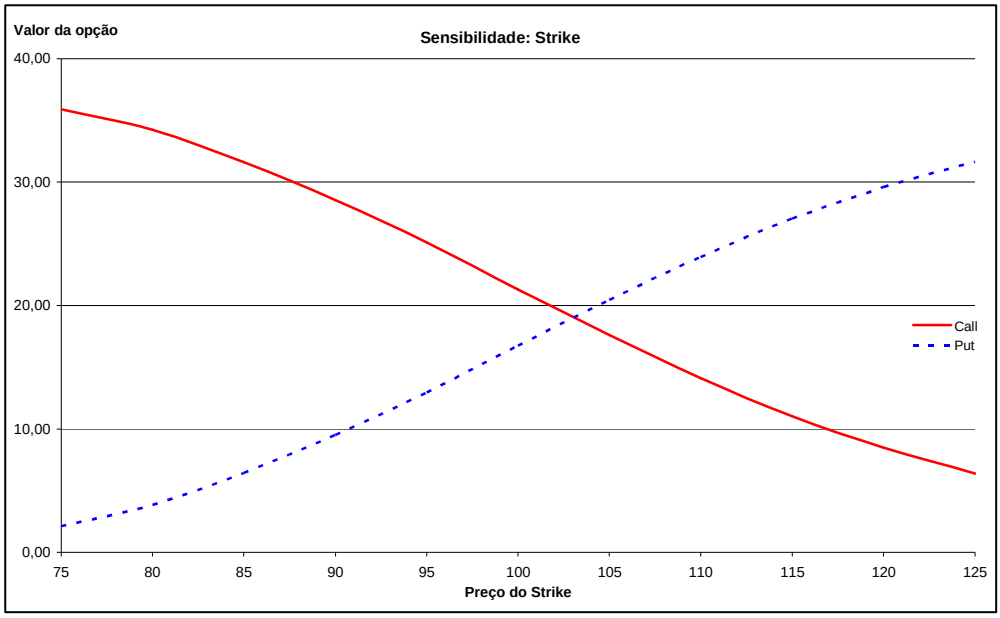
$$\text{PreçoCall} = 1{,}8287.r + 54{,}372 \qquad R^2 = 0{,}9993$$

$$\text{PreçoPut} = -1{,}8286.r + 45{,}624 \qquad R^2 = 0{,}9992$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção *cash-or-nothing*

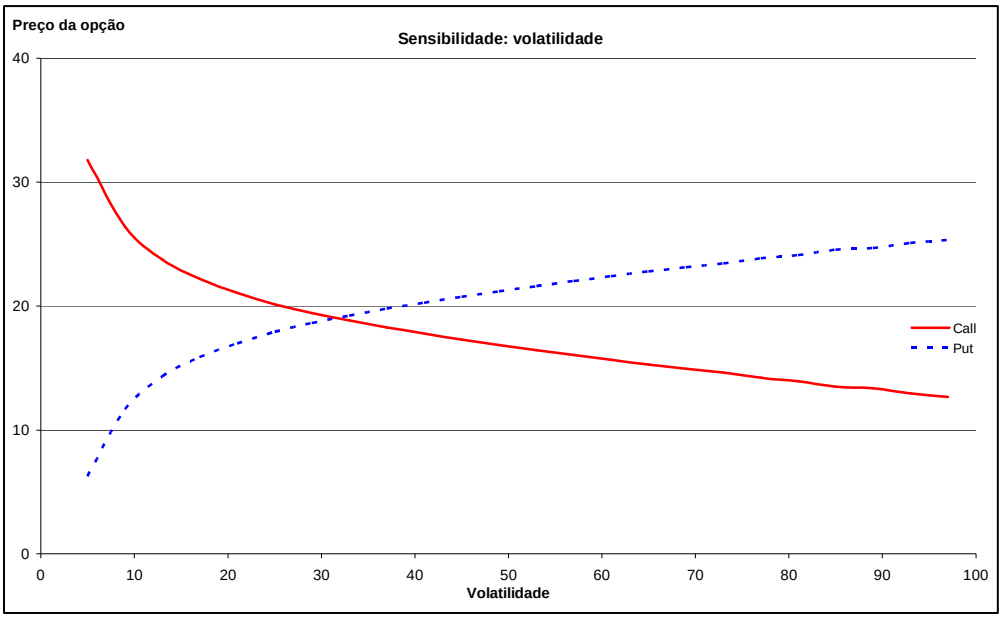
Sensibilidade em relação à mudança no preço do strike



Da mesma forma que para a opção *asset-or-nothing*, as curvas são simétricas e as mesmas conclusões podem ser tiradas quando as limites das curvas em 0 e em $+\infty$

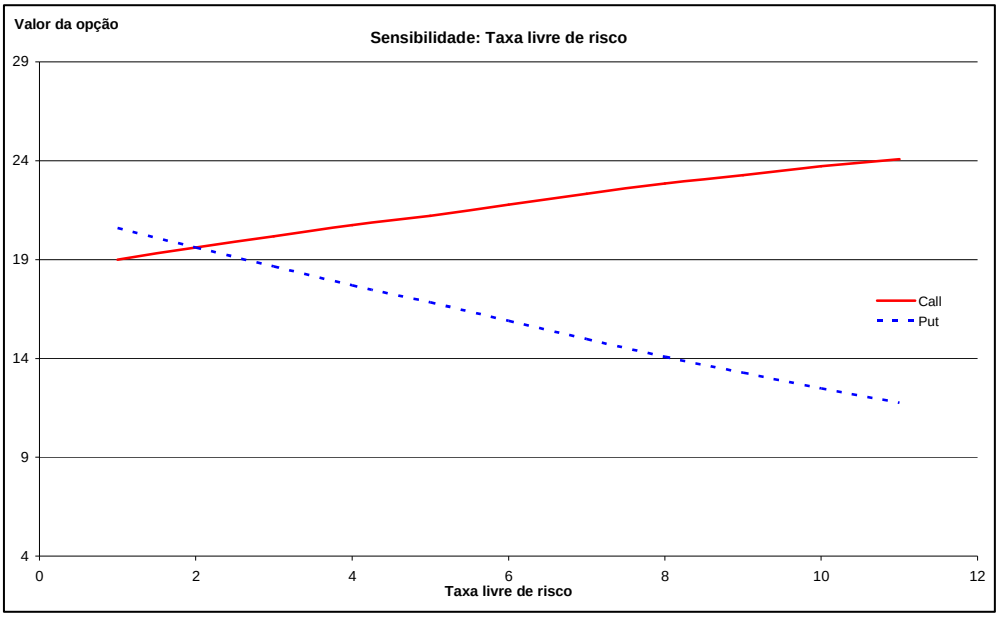
PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação à mudança na volatilidade



PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação à mudança na taxa livre de risco



Usando uma regressão linear, obtemos as equações seguintes:

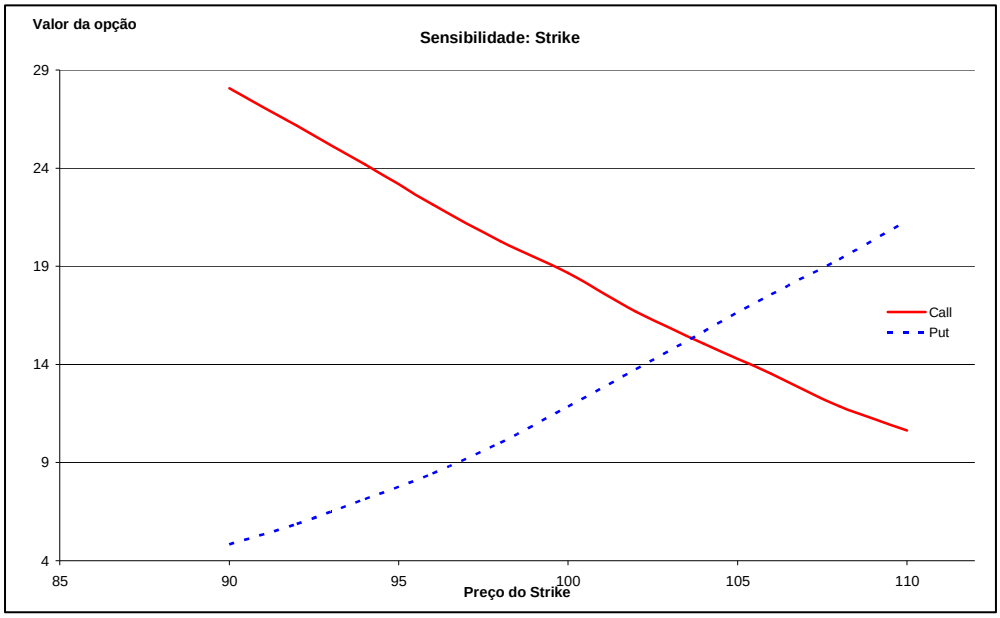
$$\text{PreçoCall} = 0{,}5137.r + 18{,}621 \qquad R^2 = 0{,}9969$$

$$\text{PreçoPut} = -0{,}8905.r + 21{,}329 \qquad R^2 = 0{,}9986$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção *Lookback fixed strike*

Sensibilidade em relação à mudança no preço do strike



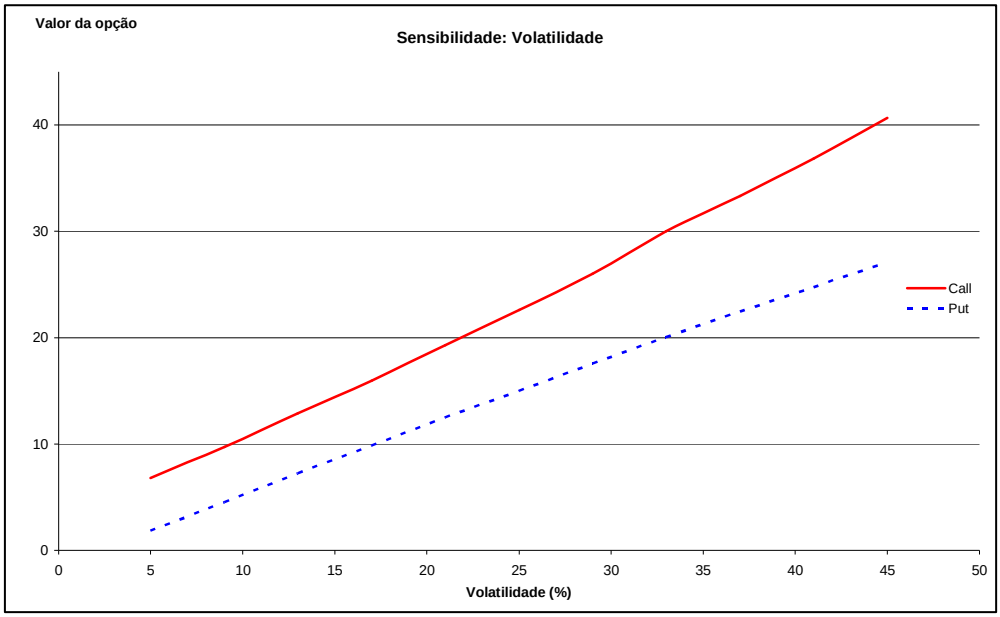
Usando uma regressão quadrática, obtemos as equações seguintes:

$$\text{PreçoCall} = 0{,}0083.k^{2} - 2{,}5413.k + 189{,}66$$
$$R^{2} = 0{,}9998$$

$$\text{PreçoPut} = 0{,}0122.k^{2} - 1{,}5959.k + 49{,}449$$
$$R^{2} = 0{,}9996$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação à mudança na volatilidade



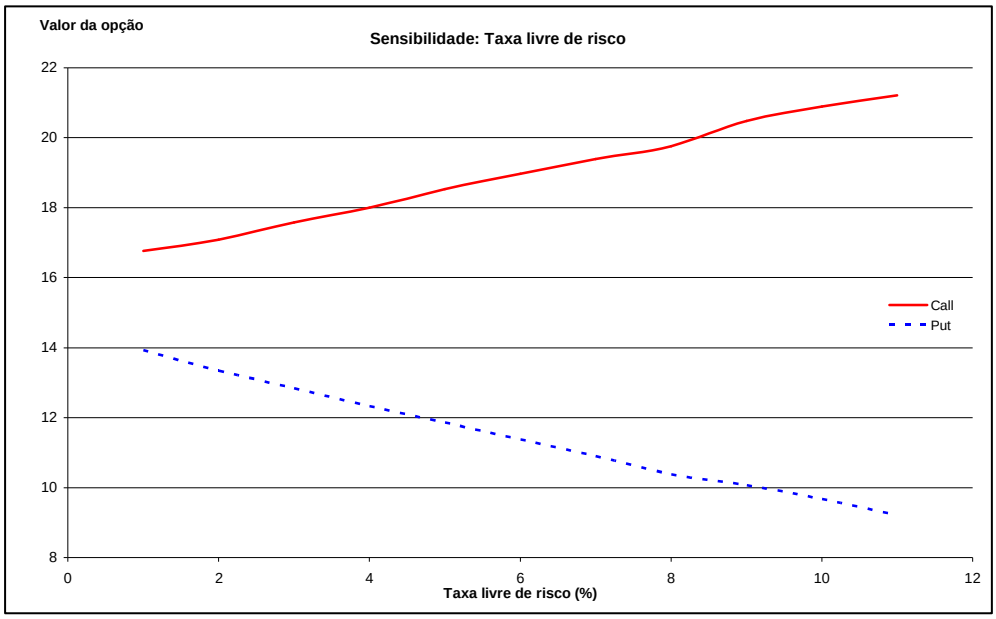
Usando uma regressão linear, obtemos as equações seguintes:

PreçoCall= 0,8493.v+1,8699 $R^2 = 0,9987$

PreçoPut= 0,6325.v-1,0124 $R^2 = 0,9993$

PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação à mudança na taxa livre de risco



Usando uma regressão linear, obtemos as equações seguintes:

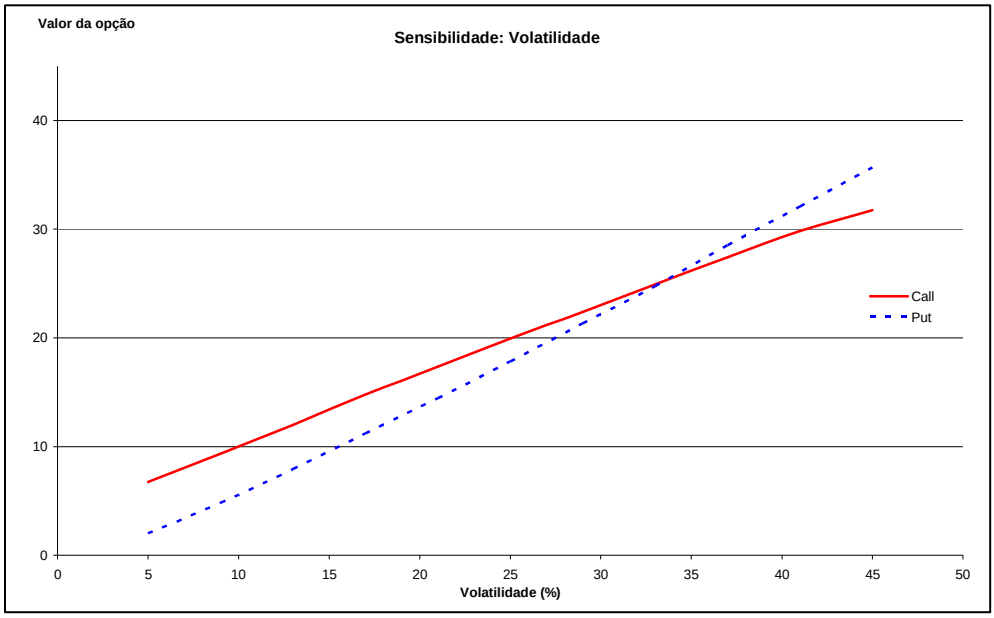
$$\text{PreçoCall} = 0{,}4591.r + 16{,}215 \qquad R^2 = 0{,}9975$$

$$\text{PreçoPut} = -0{,}4675.r + 14{,}253 \qquad R^2 = 0{,}9969$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção *Lookback floating strike*

Sensibilidade em relação à mudança na volatilidade



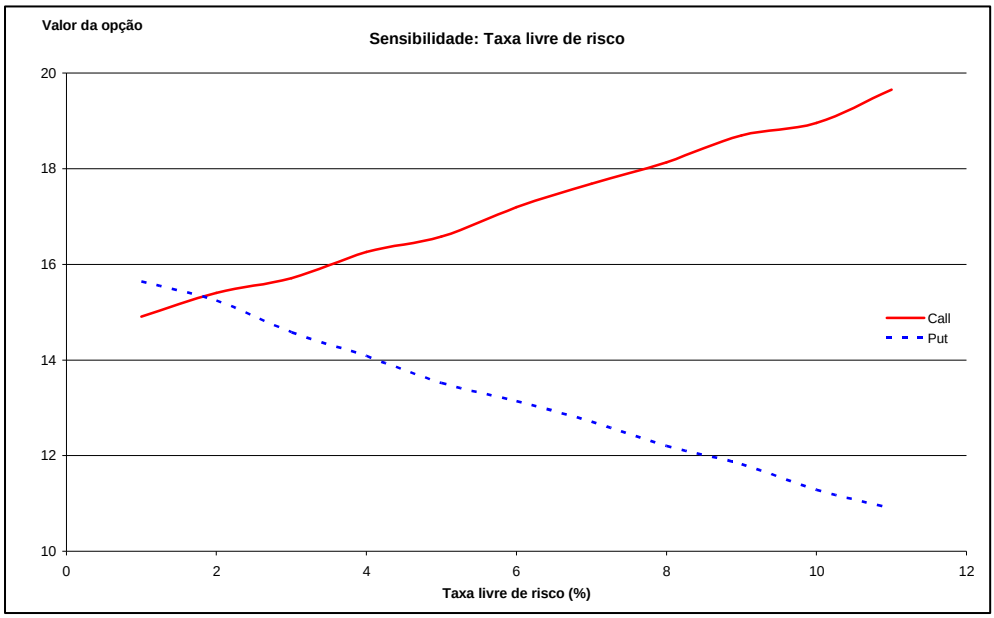
Usando uma regressão linear, obtemos as equações seguintes:

$$\text{PreçoCall} = 0{,}633.v + 3{,}8557 \qquad R^2 = 0{,}999$$

$$\text{PreçoPut} = -0{,}8482.v - 2{,}9737 \qquad R^2 = 0{,}9989$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação à mudança na taxa livre de risco



Usando uma regressão linear, obtemos as equações seguintes:

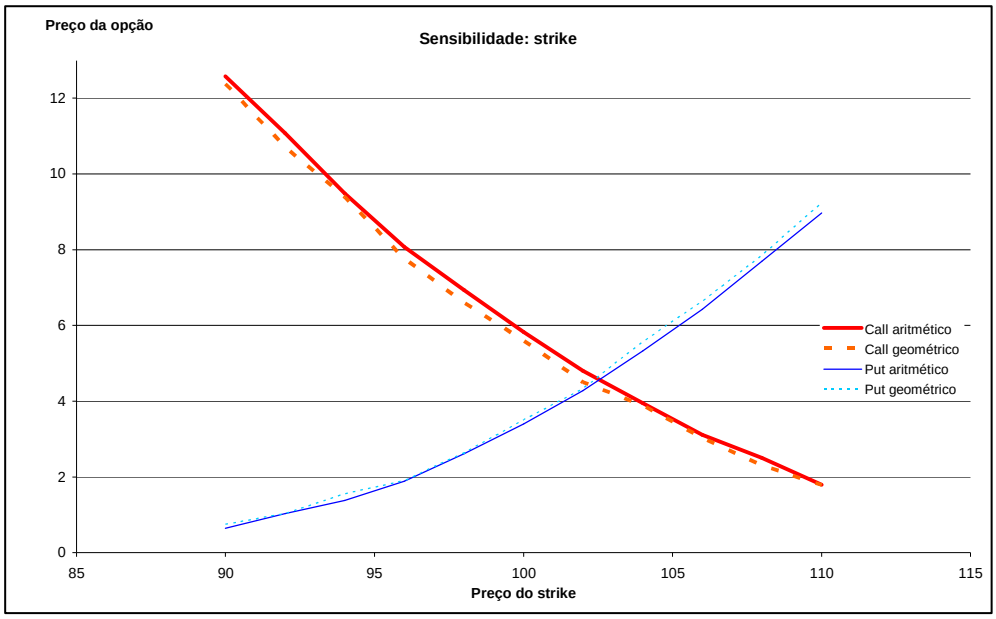
$$\text{PreçoCall} = 0{,}4704.r + 14{,}374 \qquad R^2 = 0{,}9971$$

$$\text{PreçoPut} = -0{,}4761.r + 16{,}049 \qquad R^2 = 0{,}9974$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção Asiática

Sensibilidade em relação à mudança no preço do strike



Usando uma regressão quadrática, obtemos as equações seguintes:

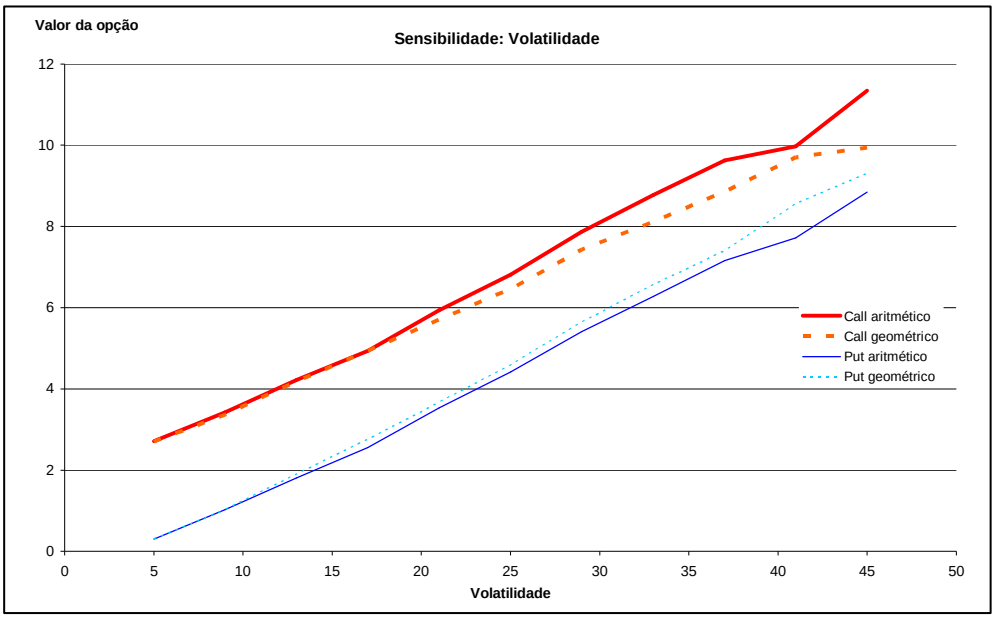
$$PreçoCall = 0{,}0141.k^2 - 1{,}5959.k + 49{,}449$$
$$R^2 = 0{,}9998$$

$$PreçoPut = 0{,}0144.k^2 - 2{,}4689.k + 105{,}93$$
$$R^2 = 0{,}9998$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação à mudança na volatilidade

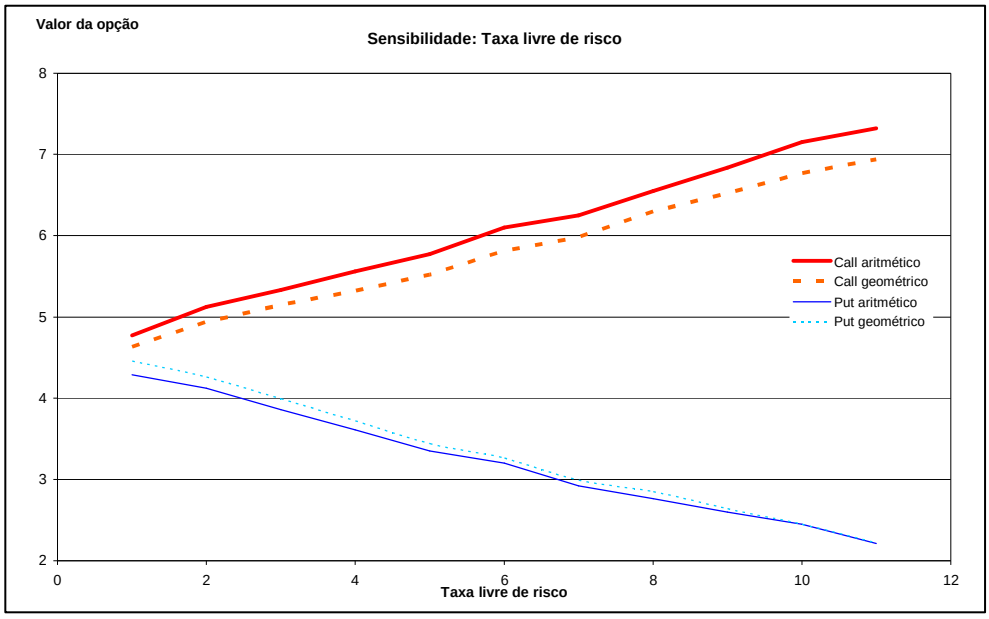
Usando uma regressão linear, obtemos as equações seguintes:



Aritmético	Geométrico
$PreçoCall = 0{,}2163.v + 1{,}4683$	$PreçoCall = 0{,}1901.v + 1{,}7335$
$R^2 = 0{,}9963$	$R^2 = 0{,}9967$
$PreçoPut = 0{,}2156.v - 0{,}9331$	$PreçoPut = -0{,}23.v - 1{,}0466$
$R^2 = 0{,}9983$	$R^2 = 0{,}9987$

PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação à mudança na taxa livre de risco



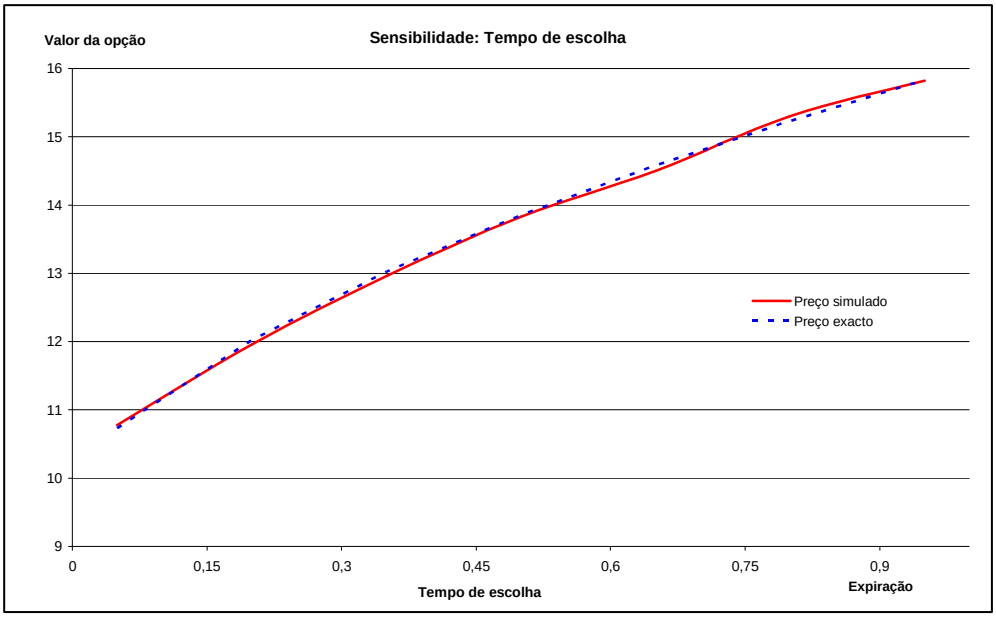
Usando uma regressão linear, obtemos as equações seguintes:

Aritmético	Geométrico
$Preço Call = 0,2533.r + 4,5495$	$Preço Call = 0,2309.r + 4,4218$
$R^2 = 0,9972$	$R^2 = 0,9974$
$Preço Put = -0,209.r + 4,4695$	$Preço Put = -0,2244.r + 4,6444$
$R^2 = 0,9943$	$R^2 = 0,9952$

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção *Chooser*

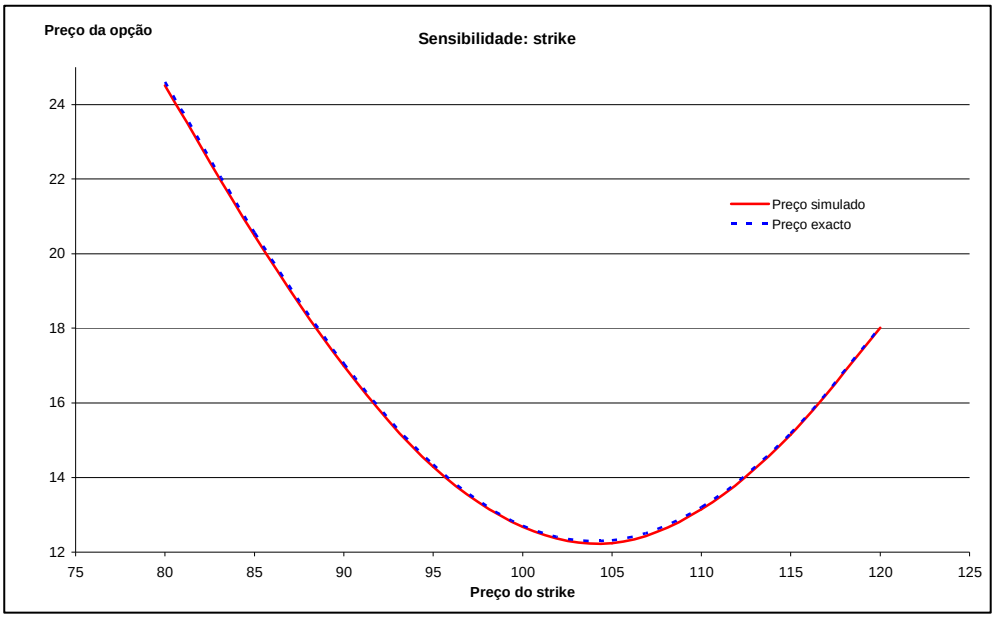
Sensibilidade em relação à mudança no tempo de escolha



O preço da opção aumenta com o tempo de escolha, a incerteza sobre o valor final do ativo sendo reduzida.

PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação à mudança no preço do strike

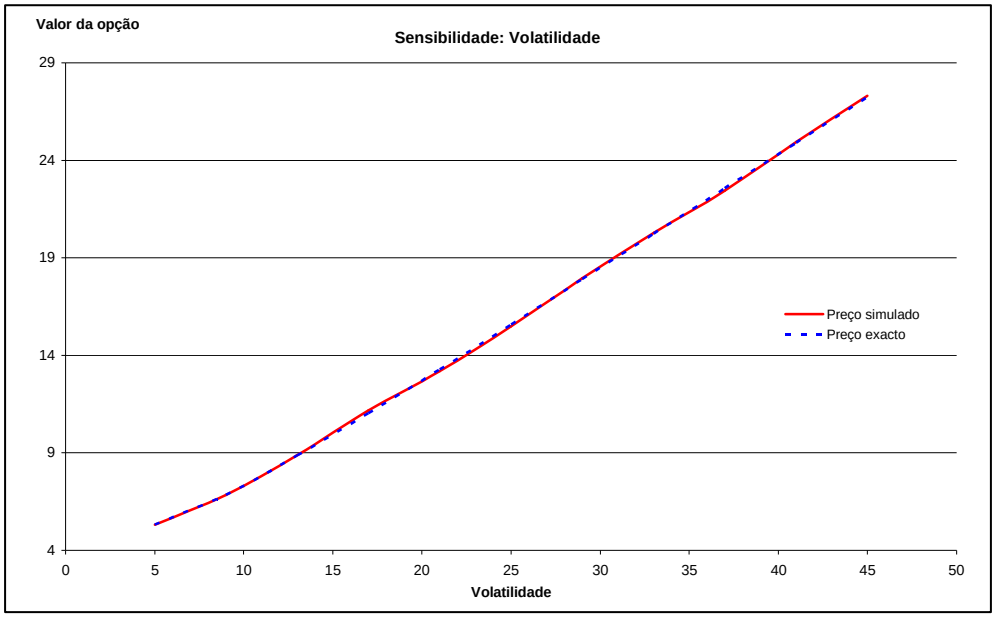


O mínimo do valor do preço da opção depende do *drift* da simulação, então dos parâmetros do ativo.

O preço da opção aumenta com strike baixo, e com strike alto, a opção *chooser* sendo um call e um put no mesmo tempo até o tempo de escolher.

PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação à mudança na volatilidade



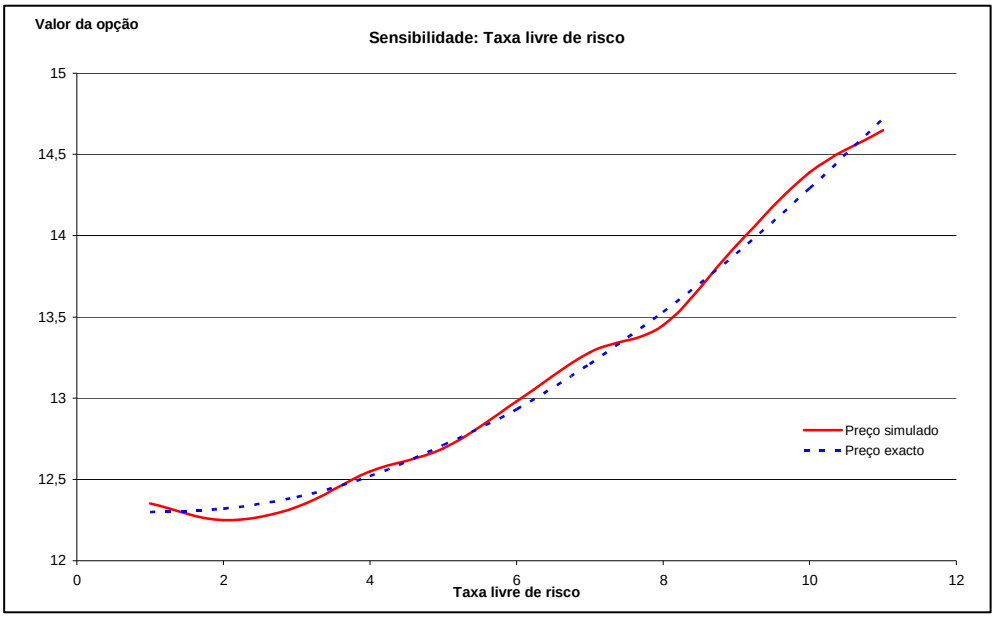
Uma relação linear existe para valores maiores que *10%*:

$$\text{PreçoChooser} = 0{,}5588.v + 11{,}8189$$

$$R^2 = 0{,}9984$$

PUC-Rio - Certificação Digital Nº 0611773/CA

Sensibilidade em relação à mudança na taxa livre de risco



Para o preço teórico existe uma relação quadrática:

$$\text{PreçoChoos er} = 0{,}0231.r^{2} - 0{,}0315.r + 12.292$$

$$R^{2} = 0{,}9998$$

O preço simulado converge para todos os valores da taxa livre de risco.

PUC-Rio - Certificação Digital Nº 0611773/CA

Apêndice 4

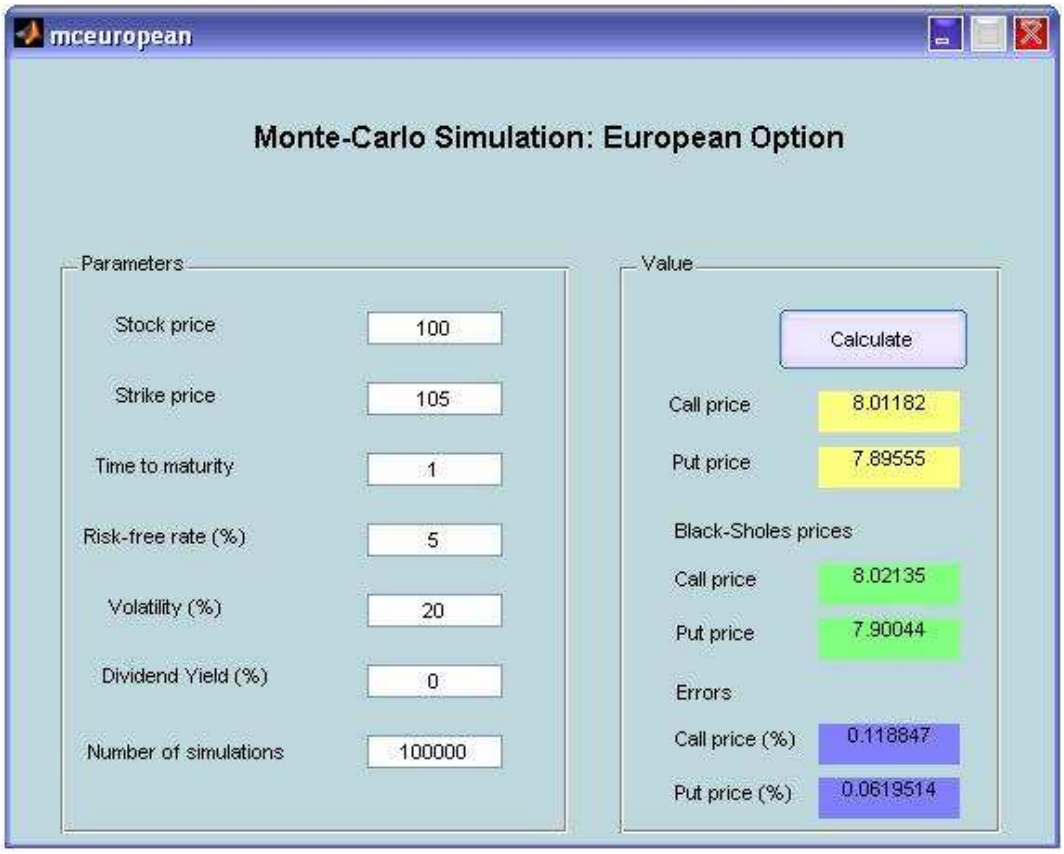
Apresentação dos programas

Para iniciar um calculo de opção, abrir "*Matlab*" e iniciar o programa: "*maingui.m*" (RUN).
A janela seguinte aparece:

Selecionar o tipo de opção desejado, e apertar "*Go to calculate*" para entrar no programa particular da opção.
Para cada tipo de opção, a janela que aparece é a seguinte:

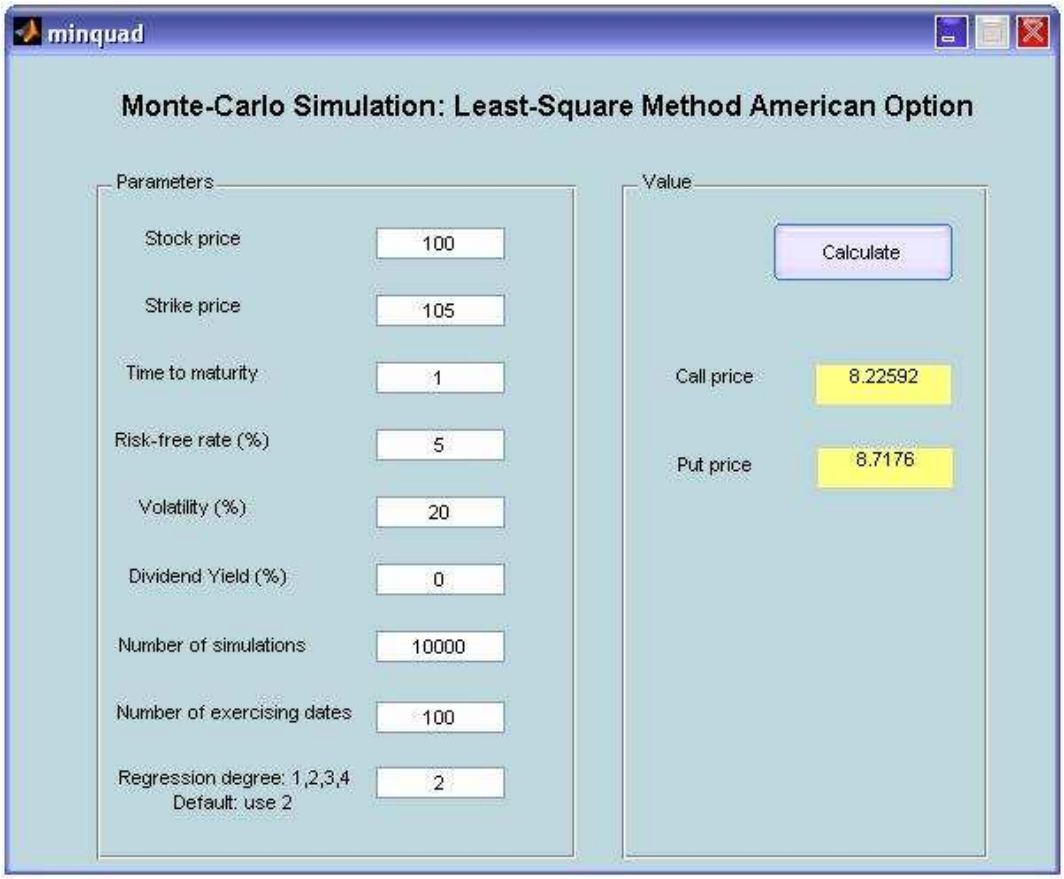
PUC-Rio - Certificação Digital Nº 0611773/CA

European option



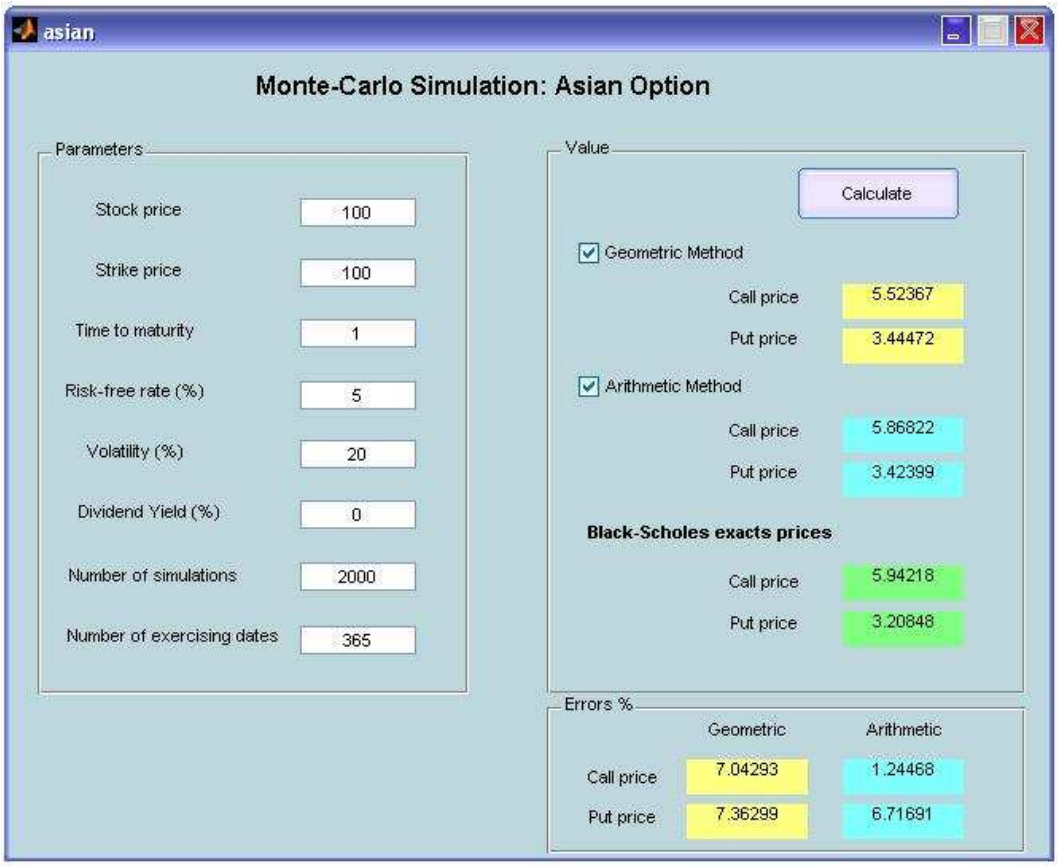
PUC-Rio - Certificação Digital Nº 0611773/CA

American option Least-square method



PUC-Rio - Certificação Digital Nº 0611773/CA

Asian option



PUC-Rio - Certificação Digital Nº 0611773/CA

Asset-or-nothing option

PUC-Rio - Certificação Digital Nº 0611773/CA

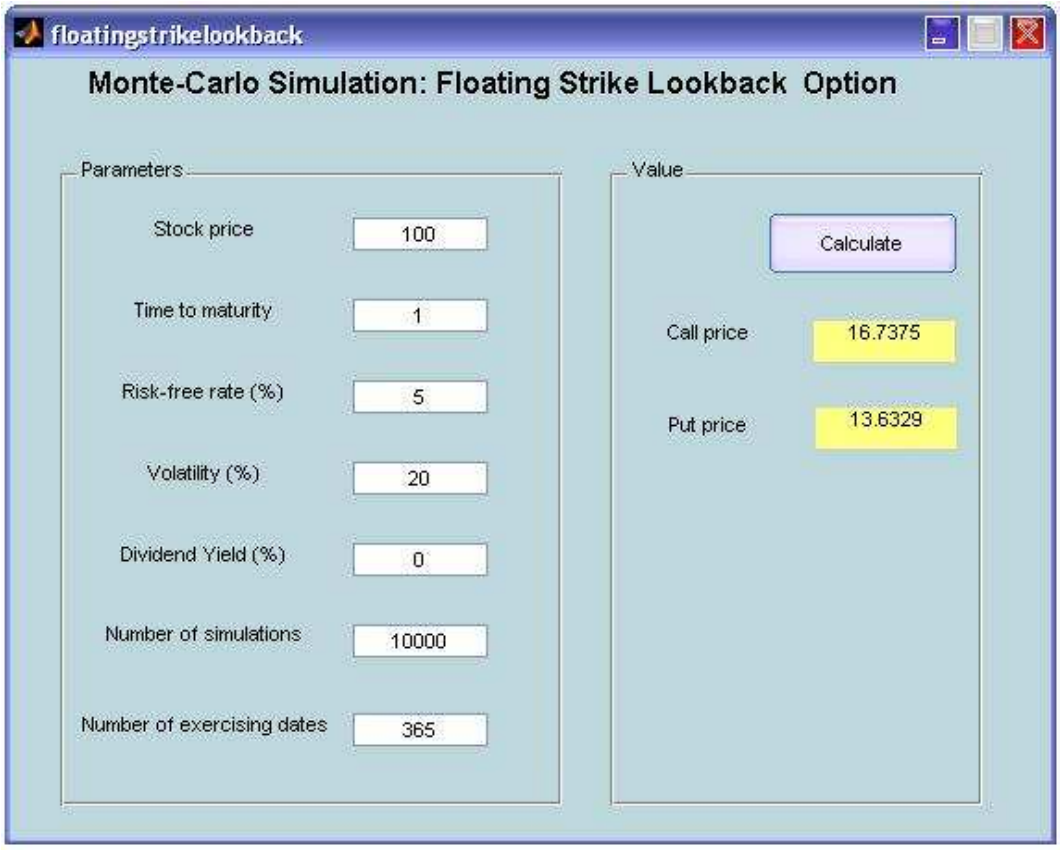
Cash-or-nothing option

PUC-Rio - Certificação Digital Nº 0611773/CA

Lookback option: fixed strike

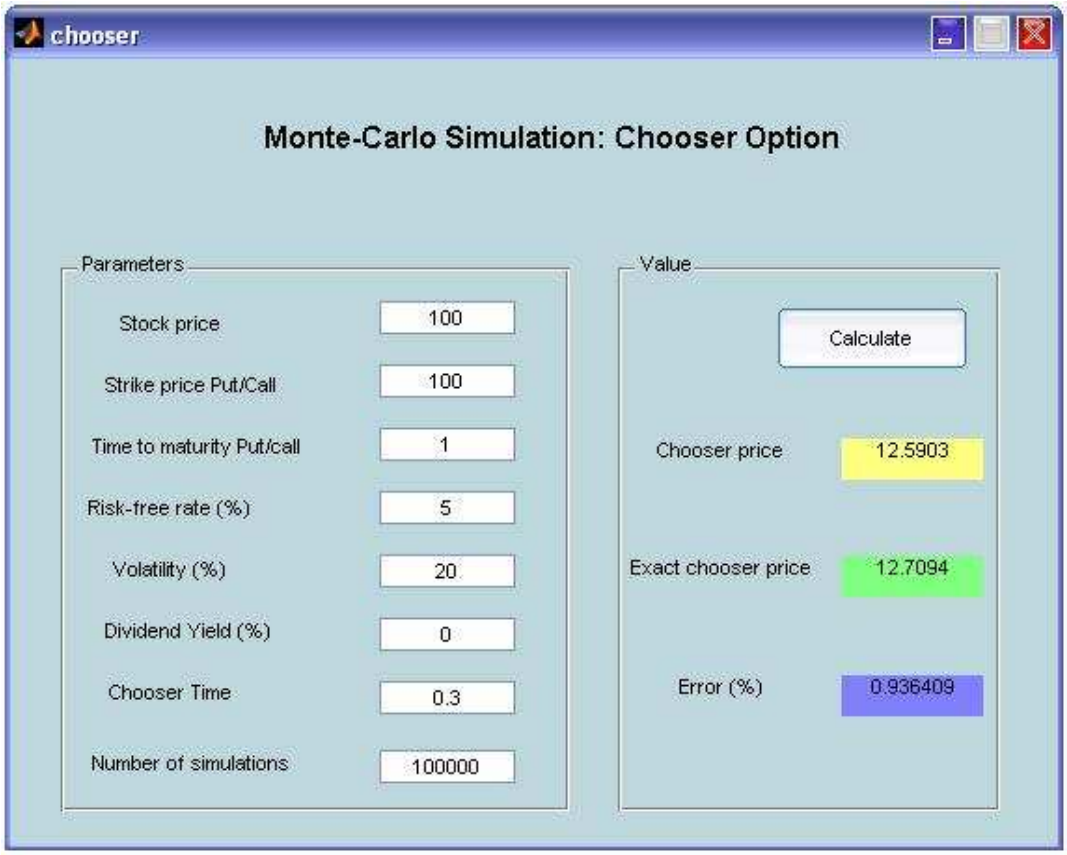
PUC-Rio - Certificação Digital Nº 0611773/CA

Lookback option: floating strike



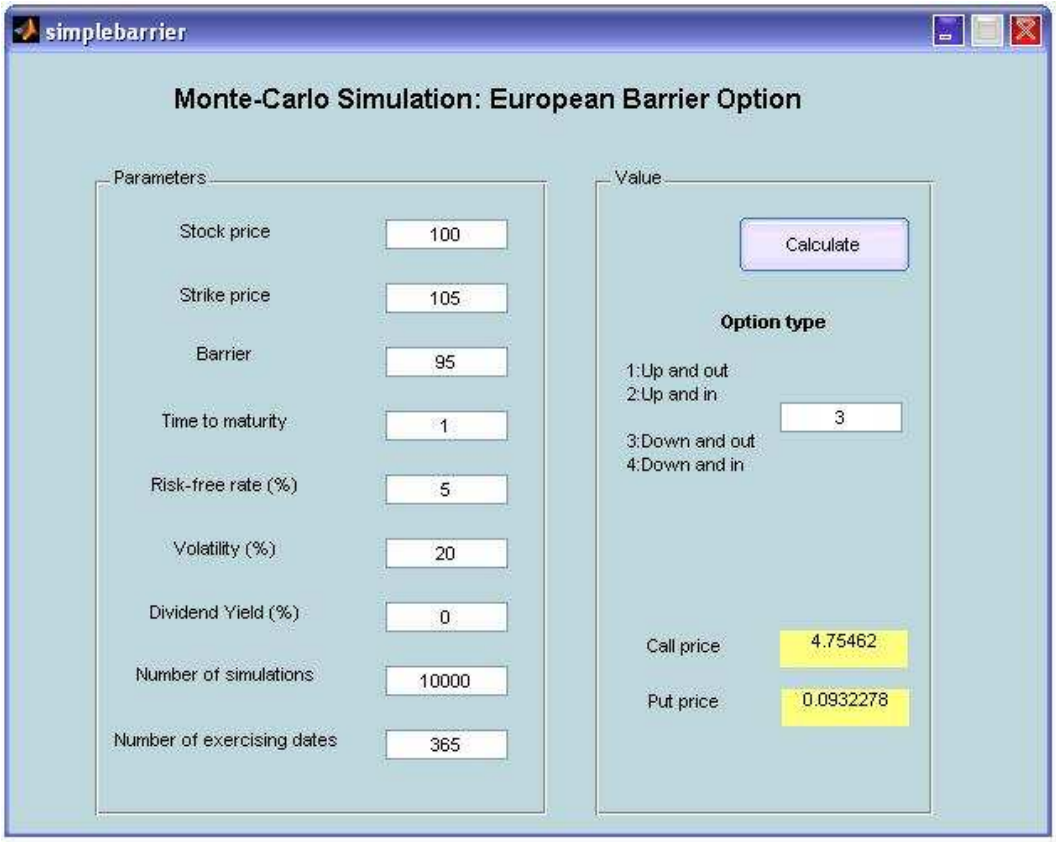
PUC-Rio - Certificação Digital Nº 0611773/CA

Chooser option



PUC-Rio - Certificação Digital Nº 0611773/CA

Barrier option



PUC-Rio - Certificação Digital Nº 0611773/CA

Apêndice 5

Método de Monte-Carlo

Código "Matlab" dos programas

PUC-Rio - Certificação Digital Nº 0611773/CA

Variáveis padrões dos programas:

s: Preço inicial
k: Strike
T: Tempo
r: Taxa livre de risco
v: Volatilidade
y: Dividend yield

NumSim: Numero de simulações
DivTemp: Numero de datas de exercício

grau: Grau da regressão

premium: Prêmio

b: Barrier
type: Tipo de opção *barrier*

Opção européia

```
function [call,put] =
mceuropeanoption(price,strike,time,riskfreerate,volatility,dividendyield,NumSim)

x=price;
k=strike;
r=riskfreerate;
v=volatility;
n=NumSim;
t=time;
y=dividendyield;

 %Initializacao e calculo das variaveis intermediarios
F =(r-y-v^2/2)*t;
FF=v*sqrt(t);
call_vector    =0;
call_vector_inv=0;
 put_vector    =0;
 put_vector_inv=0;

for i=1:n/2
   %Optimizar por "antithetic"
   eps    =random('Normal',0,1);
   eps_inv=-eps;

    %Calculo do valor da opcao para cada simulacao
   call_vector     =max( x*exp(F+FF*eps)-k,0)     +call_vector;
   call_vector_inv =max( x*exp(F+FF*eps_inv)-k,0) +call_vector_inv;
    put_vector     =max(-x*exp(F+FF*eps)+k,0)     +put_vector;
    put_vector_inv =max(-x*exp(F+FF*eps_inv)+k,0) +put_vector_inv;
end

%Calculo da media do valor do payoff
c=(1/n)*(call_vector + call_vector_inv);
p=(1/n)*( put_vector +  put_vector_inv);

%Valor da opcao
call=exp(-r*t)*c;
put=exp(-r*t)*p;
```

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção americana: Método dos mínimos quadrados de Monte-Carlo

```
function [ put_value,call_value ] = minquadoption(s,k,T,r,v,y,NumSim,DivTemp,grau)

%Divisão de tempo usada pela simulação, (tempo total dividido pelo numero de datas de
exercicios)
dt = T/DivTemp;
%Simulaçao dos preços
prices=zeros(NumSim,DivTemp);
%Inicializaçao da primeira coluna da simulaçao dos preços
for i=1:NumSim
   prices(i,1)=s;
end
%Simulaçao dos precos do ativo basico
for j=2:DivTemp
   for i=1:NumSim/2
      eps=normrnd(0,1);
      prices(i,j)        =prices(i        ,j-1)*exp( (r-y-v^2/2)*dt + v*eps*sqrt(dt)  );
      prices(i+NumSim/2,j)=prices(i+NumSim/2,j-1)*exp( (r-y-v^2/2)*dt - v*eps*sqrt(dt)
);
   end
end

%Calculo do Put
%Criaçao da matriz de calculo do preço da opção no exercicio imediato em cada tempo.
option = max( k-prices , 0);
%Inicialisação do valor de continuação ao tempo final,seja o preço da opção.
continue_value=option(:,DivTemp);
%Metodo do MQ para calculo da opção
for j=DivTemp-1:-1:2
   temp=regression( exp(-r*dt)*continue_value , prices(:,j) , option(:,j) , grau);
   %Condicao de exercicio da opcao (exercicio imediatio > valor esperado).
   for i=1:NumSim
      if (option(i,j) > temp(i))
         continue_value(i)=option(i,j);
      else
         continue_value(i)=exp(-r*dt)*continue_value(i);
      end
   end
end
put_value=sum(continue_value)/NumSim;

%Calculo do Call
%Criaçao da matriz de calculo do preço da opção no exercicio imediato em cada tempo.
option = max( prices-k , 0);
%Inicialisação do valor de continuação ao tempo final,seja o preço da opção.
continue_value=option(:,DivTemp);
%Metodo do MQ para calculo da opção
for j=DivTemp-1:-1:2
   temp=regression( exp(-r*dt)*continue_value , prices(:,j) , option(:,j) , grau);
   %Condicao de exercicio da opcao (exercicio imediatio > valor esperado).
```

PUC-Rio - Certificação Digital Nº 0611773/CA

```
   for i=1:NumSim
      if (option(i,j) > temp(i))
         continue_value(i)=option(i,j);
      else
         continue_value(i)=exp(-r*dt)*continue_value(i);
      end
   end
end
call_value=sum(continue_value)/NumSim;
```

Função de regressão

```
function [vector]=regression(A,B,C,grau)
%Função de calculo dos parametros da regreção e do valor de continuaçao da opçao.

%Vetor de valor de continuaçao
Y=[];
Y=A;
%Vetor de valor dos precos
X=[];
X=B;
%Vetor de valor da opcao corespondante
Z=[];
Z=C;
%Vetor marcando os caminhos "ITM"
W=[];
n=length(Y);
YY=[];
XX=[];

%Seleçao dos preços "in-the-money" e do vetor dos caminhos "in-the-money"
%A linha tem o valor "1" se "ITM", "0" se "out-of-the-money"
i=1;
for j=1:n
   if Z(j)==0
      W(j,1)=0;
      else
         YY(i,1)=Y(j);
         XX(i,1)=X(j);
         W(j,1) =1;
         i=i+1;
   end
end

%Calculo da matriz dos valores de (XX^0,XX,XX^2,XX^3...) para o calculo dos
%coeficientes da regressao, so com os caminhos 'in-the-money".
%Dependendo do grau desejado:
if (grau==1)                        %regressao linear
   matrixx=[ones(size(XX)) XX];
   matrix=[ones(size(X)) X];
elseif (grau==2)                    %regressao quadratica
   matrixx=[ones(size(XX)) XX XX.*XX];
   matrix=[ones(size(X)) X X.*X];
```

PUC-Rio - Certificação Digital Nº 0611773/CA

```
elseif (grau==3)                    %regressao grau 3
   matrixx=[ones(size(XX)) XX XX.*XX XX.*XX.*XX];
   matrix=[ones(size(X)) X X.*X X.*X.*X];
elseif (grau==4)                    %regressao grau 4
   matrixx=[ones(size(XX)) XX XX.*XX XX.*XX.*XX XX.*XX.*XX.*XX];
   matrix=[ones(size(X)) X X.*X X.*X.*X X.*X.*X.*X];
elseif (grau==5)                    %regressao grau 5
   matrixx=[ones(size(XX)) XX XX.*XX XX.*XX.*XX XX.*XX.*XX.*XX
XX.*XX.*XX.*XX.*XX];
   matrix=[ones(size(X)) X X.*X X.*X.*X X.*X.*X.*X X.*X.*X.*X.*X];
end
%Calculo dos coeficientes da regressão
coef = matrixx\YY;
%Calculo dos precos esperados
vector = (matrix*coef).*W;
```

Opção asset-or-nothing

```
function [call,put] = assetornothingoption(s,k,T,r,v,y,NumSim)

%Inicializaçao das variaveis usadas
eps      =0;
price    =0;
price_inv=0;
valuecall=0;
valueput =0;
call     =0;
put      =0;
%Simulação dos preços do ativo basico e calculo do valor do payoff em cada
tempo
for i=1:NumSim/2
   eps      =normrnd(0,1);
   price    =s*exp( (r-y-v^2/2)*T + v*eps*sqrt(T)  );
   price_inv=s*exp( (r-y-v^2/2)*T - v*eps*sqrt(T)  );
     %Calculo do payoff do call
   if(price>k)
   valuecall =valuecall + price;
   end
   if(price_inv>k)
   valuecall =valuecall + price_inv;
   end
     %Calculo do payoff do put
   if(k>price)
   valueput =valueput + price;
   end
   if(k>price_inv)
   valueput =valueput + price_inv;
   end
end
call=exp(-r*T)*(valuecall/NumSim);
put=exp(-r*T)* (valueput /NumSim);
```

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção cash-or-nothing

```
function [call,put] = cashornothingoption(s,k,T,r,v,y,premium,NumSim)

%Inicializaçao das variaveis usadas
eps      =0;
price    =0;
price_inv=0;
valuecall=0;
valueput =0;
call     =0;
put      =0;

%Simulação dos preços do ativo basico e calculo do valor do payoff em cada
%tempo
for i=1:NumSim/2
   eps      =normrnd(0,1);
   price    =s*exp( (r-y-v^2/2)*T + v*eps*sqrt(T)  );
   price_inv=s*exp( (r-y-v^2/2)*T - v*eps*sqrt(T)  );

   %Calculo do payoff do call
   if(price>k)
   valuecall =valuecall + premium;
   end
   if(price_inv>k)
   valuecall =valuecall + premium;
   end

   %Calculo do payoff do put
   if(k>price)
   valueput =valueput + premium;
   end
   if(k>price_inv)
   valueput =valueput + premium;
   end
end

call=exp(-r*T)*(valuecall/NumSim);
put=exp(-r*T)* (valueput /NumSim);
```

```
%Função de calculo exacto:
function [call,put] =
exactcashornothingoption(s,k,T,r,v,y,premiu
m)

%Reiner e Rubinstein 1991 soluçao
d=(log(s/k)+(r-y-v^2/2)*T)/(v*sqrt(T));
call=exp(-r*T)*normcdf(d) *premium;
put =exp(-r*T)*normcdf(-d)*premium;
```

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção lookback: *fixed strike*

```
function [call,put] = fixedstrikelookbackoption(s,k,T,r,v,y,NumSim,DivTemp)

%Divisão de tempo usada pela simulação, (tempo total dividido pelo numero de
datas de exercicios)
dt = T/DivTemp;
%Simulaçao dos preços
prices=zeros(NumSim,DivTemp);
%Inicializaçao da primeira coluna da simulaçao dos preços
for i=1:NumSim
   prices(i,1)=s;
end
%Simulaçao dos preços do ativo basico
for j=2:DivTemp
   for i=1:NumSim/2
      eps=normrnd(0,1);
      prices(i,j)        =prices(i        ,j-1)*exp( (r-y-v^2/2)*dt + v*eps*sqrt(dt)  );
      prices(i+NumSim/2,j)=prices(i+NumSim/2,j-1)*exp( (r-y-v^2/2)*dt -
v*eps*sqrt(dt)  );
   end
end

%Pesquisando o minimo e o maximo para cada caminho
for i=1:NumSim
   callpayoff(i,1)=max( max(prices(i,:))-k ,0);
   putpayoff (i,1)=max( k-min(prices(i,:)) ,0);
end

call=exp(-r*T)*(sum(callpayoff)/NumSim);
put =exp(-r*T)*(sum(putpayoff) /NumSim);
```

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção lookback: *floating strike*

```
function [call,put] = floatingstrikelookbackoption(s,T,r,v,y,NumSim,DivTemp)

%Divisão de tempo usada pela simulação, (tempo total dividido pelo numero de
datas de exercicios)
dt = T/DivTemp;
%Simulaçao dos preços
prices=zeros(NumSim,DivTemp);
%Inicializaçao da primeira coluna da simulaçao dos preços
for i=1:NumSim
   prices(i,1)=s;
end
%Simulaçao dos preços do ativo basico
for j=2:DivTemp
   for i=1:NumSim/2
      eps=normrnd(0,1);
      prices(i,j)        =prices(i        ,j-1)*exp( (r-y-v^2/2)*dt + v*eps*sqrt(dt)  );
      prices(i+NumSim/2,j)=prices(i+NumSim/2,j-1)*exp( (r-y-v^2/2)*dt -
v*eps*sqrt(dt)  );
   end
end


%Pesquisando o minimo e o maximo para cada caminho
for i=1:NumSim
   callpayoff(i,1)=max( prices(i,DivTemp)-min(prices(i,:)) ,0);
   putpayoff (i,1)=max( max(prices(i,:))-prices(i,DivTemp) ,0);
end

call=exp(-r*T)*(sum(callpayoff)/NumSim);
put =exp(-r*T)*(sum(putpayoff) /NumSim);
```

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção asiática: método geométrico

```
function [call,put] =
geometricasianoptionmontecarlo(s,k,T,r,v,y,NumSim,DivTemp)

%Divisão de tempo usada pela simulação, (tempo total dividido pelo numero de
datas de exercicios)
dt = T/DivTemp;
%Simulaçao dos preços
prices=zeros(NumSim,DivTemp);
%Inicializaçao da primeira coluna da simulaçao dos preços
for i=1:NumSim
   prices(i,1)=s;
end
%Simulaçao dos preços do ativo basico
for j=2:DivTemp
   for i=1:NumSim/2
      eps=normrnd(0,1);
      prices(i,j)        =prices(i        ,j-1)*exp( (r-y-v^2/2)*dt + v*eps*sqrt(dt)  );
      prices(i+NumSim/2,j)=prices(i+NumSim/2,j-1)*exp( (r-y-v^2/2)*dt -
v*eps*sqrt(dt)  );
   end
end
%Inicializaçao do vetor coluna recebendo o produto dos preços
for i=1:NumSim
   payoff(i,1)=1;
end
%Calculo do producto dos preços e da raiz "^(1/DivTemp)"
for j=1:DivTemp
   payoff(:,1) = payoff(:,1).*(prices(:,j).^(1/DivTemp));
end
%Calculo do payoff final de cada simulaçao
callpayoff(:,1)= max(payoff(:,1)-k,0);
putpayoff(:,1) = max(k-payoff(:,1),0);

call=sum(callpayoff(:,1))/NumSim;
call=call*exp(-r*T);

put=sum(putpayoff(:,1))/NumSim;
put=put*exp(-r*T);
```

```
%Função de calculo exacto:
function [ call,put ] =
geometricasianoption(s,k,T,r,v,y)
%Modificação dos parametros: volatilidade e
dividend yield
va = v/sqrt(3);
ya = (r-y-v^2/6)/2;
[call,put]=blsprice(s,k,r,T,va,ya);
```

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção asiática: método aritmético

```
function [call,put] =
arithmeticasianoptionmontecarlo(s,k,T,r,v,y,NumSim,DivTemp)

%Divisão de tempo usada pela simulação, (tempo total dividido pelo numero de
datas de exercicios)
dt = T/DivTemp;
%Simulaçao dos preços
prices=zeros(NumSim,DivTemp);
%Inicializaçao da primeira coluna da simulaçao dos preços
for i=1:NumSim
   prices(i,1)=s;
end
%Simulaçao dos preços do ativo basico
for j=2:DivTemp
   for i=1:NumSim/2
      eps=normrnd(0,1);
      prices(i,j)       =prices(i       ,j-1)*exp( (r-y-v^2/2)*dt + v*eps*sqrt(dt)  );
      prices(i+NumSim/2,j)=prices(i+NumSim/2,j-1)*exp( (r-y-v^2/2)*dt -
v*eps*sqrt(dt)  );
   end
end

%Inicializaçao do vetor coluna recebendo a soma dos preços
for i=1:NumSim
   payoff(i,1)=0;
end

%Calculo da soma dos preços
for j=1:DivTemp
   payoff(:,1) = payoff(:,1)+prices(:,j);
end

%Calculo da media do payoff em cada simulaçao
payoff(:,1)=payoff(:,1)./DivTemp;

%Calculo do payoff final de cada simulaçao
callpayoff(:,1)= max(payoff(:,1)-k,0);
putpayoff(:,1) = max(k-payoff(:,1),0);


call=sum(callpayoff(:,1))/NumSim;
call=call*exp(-r*T);

put=sum(putpayoff(:,1))/NumSim;
put=put*exp(-r*T);
```

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção chooser

```
function [chooservalue ] = chooseroption(s,k,T,t,r,v,y,NumSim)

price=0;
price_inv=0;
valuecall=0;
valueput =0;
numbercall=0;
numberput=0;
unpriceablepaths=0;

%Simulaçao dos precos do ativo basico e calculo do valor do payoff em cada
tempo
for i=1:NumSim/2
      eps =normrnd(0,1);
      epss=normrnd(0,1);

      %Simulaçao do preço dependendo do call ou do put escolhido no tempo t
      price           =s    *exp( (r-y-v^2/2)*t     + v*eps*sqrt(t)  );
      pricee          =price*exp( (r-y-v^2/2)*(T-t) + v*epss*sqrt(T-t)  );
      [blscall,blsput]  =blsprice(price,k,r,T-t,v,y);

      if (blscall>blsput)
      valuecall       =valuecall    + max(pricee-k,0);
      numbercall      =numbercall+1;
      elseif (blsput>blscall)
      valueput        =valueput     + max(k-pricee,0);
      numberput       =numberput+1;
      else
      unpriceablepaths  =unpriceablepaths+1;
      end

      price_inv       =s       *exp( (r-y-v^2/2)*t     - v*eps*sqrt(t)  );
      price_invv      =price_inv*exp( (r-y-v^2/2)*(T-t) - v*epss*sqrt(T-t)  );
      [blscalltemp,blsputtemp]  =blsprice(price_inv,k,r,T-t,v,y);

      if (blscalltemp>blsputtemp)
      valuecall       =valuecall  + max(price_invv-k,0);
      numbercall      =numbercall+1;
      elseif (blsputtemp>blscalltemp)
      valueput        =valueput + max(k-price_invv,0);
      numberput       =numberput+1;
      else
      unpriceablepaths=unpriceablepaths+1;
      end
end

average=(valuecall+valueput)/(numbercall+numberput);
chooservalue=average*exp(-r*T);
```

PUC-Rio - Certificação Digital Nº 0611773/CA

Opção *barrier*

```
function [call,put] = simplebarrieroption(s,k,b,T,r,v,y,NumSim,DivTemp,type)

%valor do "type":
% 1: up and out
% 2: up and in
% 3: down and out
% 4: down and in

%Inicialisaçao das variaveis
valuecall   =0;
valueput    =0;
optionactive=0;

%Divisão de tempo usada pela simulação, (tempo total dividido pelo numero de
datas de exercicios)
dt = T/DivTemp;
%Simulaçao dos preços
prices=zeros(NumSim,DivTemp);
%Inicializaçao da primeira coluna da simulaçao dos preços
for i=1:NumSim
   prices(i,1)=s;
end
%Simulaçao dos preços do ativo basico
for j=2:DivTemp
   for i=1:NumSim/2
      eps=normrnd(0,1);
      prices(i,j)        =prices(i        ,j-1)*exp( (r-y-v^2/2)*dt + v*eps*sqrt(dt)  );
      prices(i+NumSim/2,j)=prices(i+NumSim/2,j-1)*exp( (r-y-v^2/2)*dt -
v*eps*sqrt(dt)  );
   end
end



%type=1: up and out
if (type==1)
   for i=1:NumSim
      optionactive=b-max(prices(i,:));
      %Opçao esta ativa se optionactive positivo
      if (optionactive>0)
         valuecall=max(prices(i,DivTemp)-k,0)+valuecall;
         valueput =max(k-prices(i,DivTemp),0)+valueput ;
      end
   end
   call=exp(-r*T)*valuecall/NumSim;
```

PUC-Rio - Certificação Digital Nº 0611773/CA

```
   put =exp(-r*T)*valueput /NumSim;
end

%type=2: up and in
if (type==2)
   for i=1:NumSim
      optionactive=max(prices(i,:))-b;
      %Opçao esta ativa se optionactive positivo
      if (optionactive>0)
         valuecall=max(prices(i,DivTemp)-k,0)+valuecall;
         valueput =max(k-prices(i,DivTemp),0)+valueput ;
      end
   end
   call=exp(-r*T)*valuecall/NumSim;
   put =exp(-r*T)*valueput /NumSim;
end


%type=3: down and out
if (type==3)
   for i=1:NumSim
      optionactive=min(prices(i,:))-b;
      %Opçao esta ativa se optionactive positivo
      if (optionactive>0)
         valuecall=max(prices(i,DivTemp)-k,0)+valuecall;
         valueput =max(k-prices(i,DivTemp),0)+valueput ;
      end
   end
   call=exp(-r*T)*valuecall/NumSim;
   put =exp(-r*T)*valueput /NumSim;
end


%type=4: down and in
if (type==4)
   for i=1:NumSim
      optionactive=b-min(prices(i,:));
      %Opçao esta ativa se optionactive positivo
      if (optionactive>0)
         valuecall=max(prices(i,DivTemp)-k,0)+valuecall;
         valueput =max(k-prices(i,DivTemp),0)+valueput ;
      end
   end
   call=exp(-r*T)*valuecall/NumSim;
   put =exp(-r*T)*valueput /NumSim;
end
```

PUC-Rio - Certificação Digital Nº 0611773/CA

Apêndice 6

Opção americana sem dividendos
Modelo de Coxx-Ross-Rubinstein

Nesse modelo, r a taxa livre de risco é considerada constante.

Seja S_n o valor do ativo no instante n:

Assim existe duas constantes a e b tais que: $-1 < a < b$ e que $\frac{S_{n+1}}{S_n}$ só pode pegar os valores "$(1+a)$" ou "$(1+b)$" no instante $(n+1)$, com probabilidades "p" e "$(p-1)$" respectivamente:

PUC-Rio - Certificação Digital Nº 0611773/CA

Instante n	Instante (n+1)	
S_n	$S_{n+1} = S_n.(1+a)$	Valor S_U com probabilidade p
	$S_{n+1} = S_n.(1+b)$	Valor S_D com probabilidade $(p-1)$

Usando a definição da opção de venda seguinte:

Preços do ativo no inicio	S=100
Strike da opção	k=100
Tempo a expiração (ano)	T=1
Taxa livre de risco	R=5%
Volatilidade do ativo	v=20%
Dividend Yield	y=0

Para avaliar as opções americanas nesse contexto e usando o método binomial devemos escolher os parâmetros assim:

- N Numero de simulações
- $r = \frac{R.T}{N}$ A taxa livre de risco no período.
- $(1+a) = (1+r).e^{-\sigma.\sqrt{\frac{T}{N}}}$ Definição do parâmetro a
- $(1+b) = (1+r).e^{\sigma.\sqrt{\frac{T}{N}}}$ Definição do parâmetro b
- $p = \frac{b-r}{b-a}$ Definição da probabilidade neutra ao risco

Foi provado que com esses parâmetros o modelo binomial converge, por N grande, no valor da opção americana sem dividendos.

Obtemos um algoritmo rápido e determinístico para avaliar uma opção americana sem dividendos e onde o único parâmetro importante é "N" que deve ser alto para ter precisão no calculo. O tempo computacional para $N = 5000$ está de 1 minuto aproximadamente.

Os resultados da convergência são:

Numero de simulações	**Valor da opção de venda**
10	*6.1842*
50	*6.0917*
100	*6.0945*
500	*6.0915*
1000	*6.0898*
5000	*6.0904*

A precisão do calculo esta rapidamente boa, porque com $N = 1000$ o erro é só de 0,01% em comparação ao valor teórico, obtido com N=5000.

PUC-Rio - Certificação Digital Nº 0611773/CA

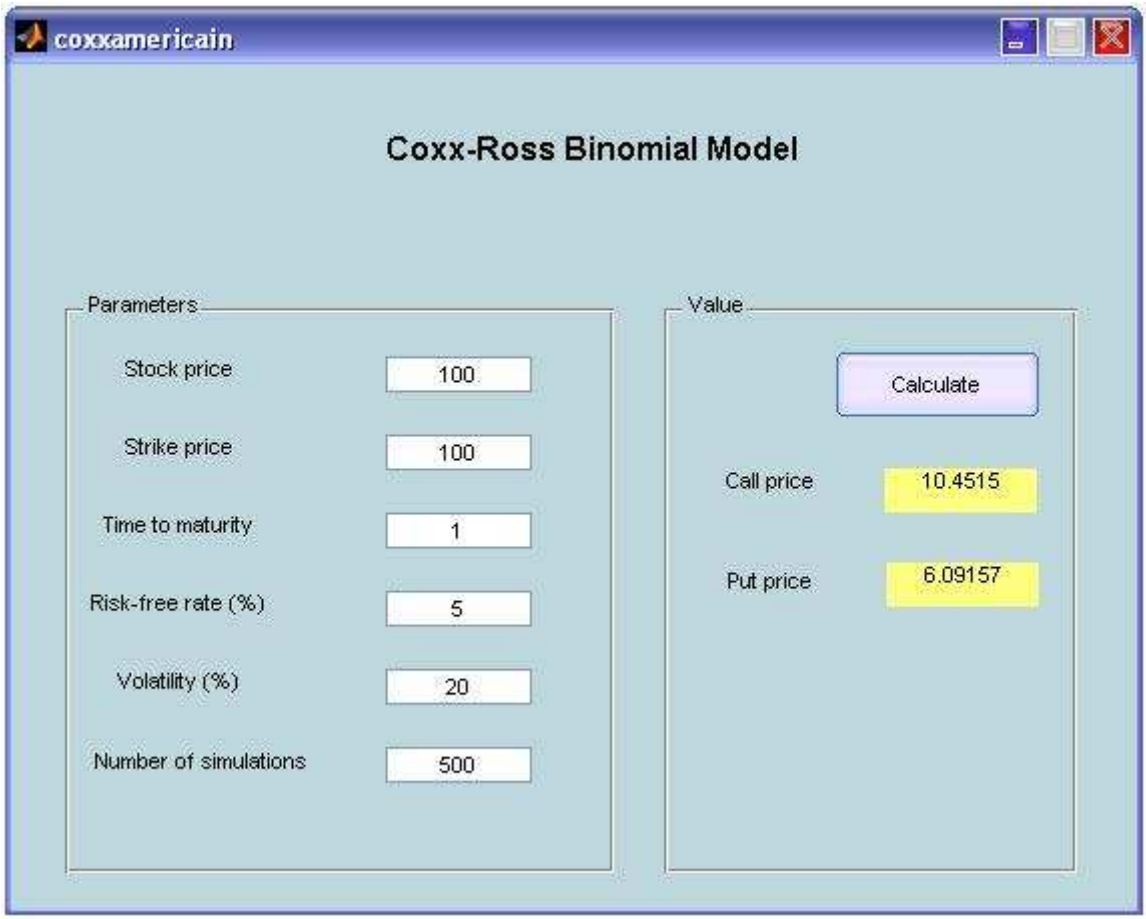
O preço da opção européia de venda com os mesmos parâmetros é 5.5735, valor menor que a opção americana.

O preço da opção de compra americana sem dividendos é o mesmo que a opção européia, porque foi provado que o exercício antecipado nunca é ótimo. Esse preço teórico é dado pela formula de Black-Scholes. (ver anexo 2).

Os resultados da convergência são:

Numero de simulações	**Valor do Call**	**Valor Black-Scholes**	**Erro (%)**
10	*10.6085*	*10.4506*	*1.51*
50	*10.4423*		*0.08*
100	*10.4586*		*0.07*
500	*10.4515*		*0.009*
1000	*10.4493*		*0.008*
5000	*10.4506*		*0*

PUC-Rio - Certificação Digital Nº 0611773/CA



Código Matlab do programa:

```
function [call,put] = coxxamericainoption(price,strike,time,riskfreerate,volatility,NumSim)

%Definição das constantes no calculo
s=price; k=strike; t=time; R=riskfreerate; v=volatility; n=NumSim;
 %Taxa ajustada no período
r=R*t/n;

A=(1+r)*exp(-v*sqrt(t/n));
B=(1+r)*exp(v*sqrt(t/n));

p=((B-1)-r) / ((B-1)-(A-1));      %Probabilidade neutra ao risco de queda
q=1-p;                            %Probabilidade neutra ao risco de subida
m=zeros(n+1,n+1);

%Calculo da opção call
   for j=1:n+1
      for i=1:j
         m(i,j)=  max ( s*(B)^(i-1)*(A)^(j-i)-k , 0 );
      end
```

PUC-Rio - Certificação Digital Nº 0611773/CA

```
    end

    for j=n+1:-1:2
        for i=j:-1:2
            m(i-1,j-1)= max ( (q*m(i,j)+p*m(i-1,j))/(1+r) , m(i-1,j-1) );
        end
    end
call = m(1,1);

%Calculo da opçao put
    for j=1:n+1
        for i=1:j
            m(i,j)=  max ( k-s*(B)^(i-1)*(A)^(j-i) , 0 );
        end
    end

    for j=n+1:-1:2
        for i=j:-1:2
            m(i-1,j-1)= max ( (q*m(i,j)+p*m(i-1,j))/(1+r) , m(i-1,j-1) );
        end
    end
put = m(1,1);
```

PUC-Rio - Certificação Digital Nº 0611773/CA