

## Algorithm for an American put option

Input:  $x_{min}, x_{max}, M, N, K, T$  and the parameters of the model

$$\delta\tau = \frac{\sigma^2 T}{2N}, \quad \delta x = \frac{x_{max} - x_{min}}{M}$$

Calculate  $\tau_\nu, \nu = 0, 1, \dots, N$ , and  $x_i, i = 0, 1, \dots, M$

For  $i = 0, 1, \dots, M$

$$w_{i,0} = e^{-rT}(K - e^{x_i})^+$$

For  $\nu = 0, 1, \dots, N - 1$

$$w_{0,\nu+1} = Ke^{-r(T - \frac{2\tau_{\nu+1}}{\sigma^2})}$$

$$w_{M,\nu+1} = 0$$

For  $i = 1, 2, \dots, M - 1$

$$u_1 = \lambda w_{i+1,\nu} + (1 - 2\lambda)w_{i,\nu} + \lambda w_{i-1,\nu}$$

$$u_2 = e^{-r(T - \frac{2\tau_{\nu+1}}{\sigma^2})} \left( K - e^{x_i - (\frac{2r}{\sigma^2} - 1)\tau_{\nu+1}} \right)^+$$

$$w_{i,\nu+1} = \max(u_1, u_2)$$

Output:  $w_{i,\nu}$  for  $i = 0, 1, \dots, M, \quad \nu = 0, 1, \dots, N$

Computational Finance - p. 1

Figure 1: Algorithm - American BS

```
# Runge-Kutta4 coefficients
k1 = self.dt * R
k2 = self.dt * (R + 0.5 * k1)
k3 = self.dt * (R + 0.5 * k2)
k4 = self.dt * (R + k3)

# Derivated function in time domain
self.U_st[s_i, t_i] = self.U_st[s_i, t_i - 1] + (1.0 / 6.0) * (k1 + 2.0 * k2 + 2.0 * k3 + k4)
```

Figure 2: European Option computation

```

# Runge-Kutta4 coefficients
k1 = self.dt * R
k2 = self.dt * (R + 0.5 * k1)
k3 = self.dt * (R + 0.5 * k2)
k4 = self.dt * (R + k3)

# Derivated function in time domain
o = Option(s, self.k, 1, self.r, self.sigma)
actual_price = o.euro_vanilla_call()
calculated_price = self.U_st[s_i, t_i - 1] + (1.0 / 6.0) * (k1 + 2.0 * k2 + 2.0 * k3 + k4)

if actual_price >= calculated_price:
    self.U_st[s_i, t_i] = actual_price
else:
    self.U_st[s_i, t_i] = calculated_price

```

Figure 3: American Option computation

Assumptions for numerical algorithm:  $S \in (0, K)$  with  $dS$  step,  $t \in (0, N)$  with  $dt$  step.

European Option algorithm:

```

1 For S in (0, k*dS):
2     1. Calculate V[k*dS, 0] -> Initial condition
3
4 For t in (1, n*dt):
5     For S in (0, (k-1)*dS):
6         2. Calculate V1 price at [k*dS, n*dt] using numerical
7             methods (Runge-Kutta)
8
9         3. Option price = max(V, 0)
10
11    4. Set boundary conditions for V[0, n*dt] and for V[K, n*dt]

```

American Option algorithm:

```

1 For S in (0, k*dS):
2     1. Calculate V[k*dS, 0] -> Initial condition
3
4 For t in (1, n*dt):
5     For S in (0, (k-1)*dS):
6         2. Calculate V1 price at [k*dS, n*dt] using numerical
7             methods (Runge-Kutta)
8
9         3. Calculate V2 price using Black-Scholes formula
10            (at S = k*dS, and expiry time T = dt)
11
12        4. Choose higher price: V = max(V1, V2)
13
14        5. Option price = max(V, 0)
15
16    6. Set boundary conditions for V[0, n*dt] and for V[K, n*dt]

```

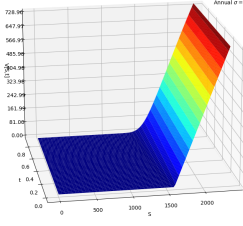
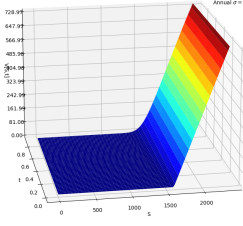
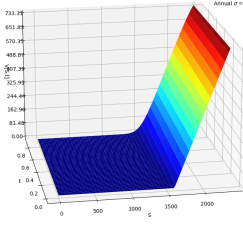
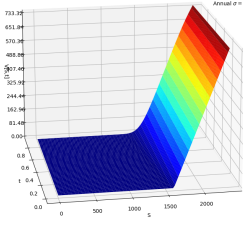
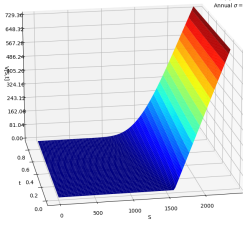
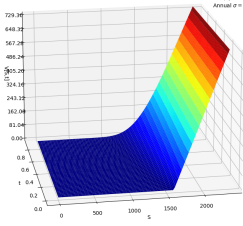
European	American
<p>Method of lines: Linear Black-Scholes equation. Strike price: <math>K=1600</math>. Initial condition: <math>V(t, S) = \max(S-K, 0)</math>. Boundary condition: <math>V(t, 0) = 0</math>. Boundary condition: <math>V(t, S) = S - K</math>. <math>S \in [0, 2500]</math> and <math>t \in [0, 0.9]</math>. Annual <math>r=0.2</math>, <math>\sigma=0.3</math>.</p>  <p>1.324</p>	<p>Method of lines: Linear Black-Scholes equation. Strike price: <math>K=1600</math>. Initial condition: <math>V(t, S) = \max(S-K, 0)</math>. Boundary condition: <math>V(t, 0) = 0</math>. Boundary condition: <math>V(t, S) = S - K</math>. <math>S \in [0, 2500]</math> and <math>t \in [0, 0.9]</math>. Annual <math>r=0.2</math>, <math>\sigma=0.3</math>.</p>  <p>1.325</p>
<p>Method of lines: Linear Black-Scholes equation. Strike price: <math>K=1600</math>. Initial condition: <math>V(t, S) = \max(S-K, 0)</math>. Boundary condition: <math>V(t, 0) = 0</math>. Boundary condition: <math>V(t, S) = S - K</math>. <math>S \in [0, 2500]</math> and <math>t \in [0, 0.9]</math>. Annual <math>r=0.2</math>, <math>\sigma=0.3</math>.</p>  <p>1.671</p>	<p>Method of lines: Linear Black-Scholes equation. Strike price: <math>K=1600</math>. Initial condition: <math>V(t, S) = \max(S-K, 0)</math>. Boundary condition: <math>V(t, 0) = 0</math>. Boundary condition: <math>V(t, S) = S - K</math>. <math>S \in [0, 2500]</math> and <math>t \in [0, 0.9]</math>. Annual <math>r=0.2</math>, <math>\sigma=0.3</math>.</p>  <p>1.676</p>
<p>Method of lines: Linear Black-Scholes equation. Strike price: <math>K=1600</math>. Initial condition: <math>V(t, S) = \max(S-K, 0)</math>. Boundary condition: <math>V(t, 0) = 0</math>. Boundary condition: <math>V(t, S) = S - K</math>. <math>S \in [0, 2500]</math> and <math>t \in [0, 0.9]</math>. Annual <math>r=0.25</math>, <math>\sigma=0.3</math>.</p>  <p>3.395</p>	<p>Method of lines: Linear Black-Scholes equation. Strike price: <math>K=1600</math>. Initial condition: <math>V(t, S) = \max(S-K, 0)</math>. Boundary condition: <math>V(t, 0) = 0</math>. Boundary condition: <math>V(t, S) = S - K</math>. <math>S \in [0, 2500]</math> and <math>t \in [0, 0.9]</math>. Annual <math>r=0.25</math>, <math>\sigma=0.3</math>.</p>  <p>3.400</p>

Table 1: Option pricing for European and American style using linear Black-Scholes model.  $S_0 = 1500$  and  $K = 1600$

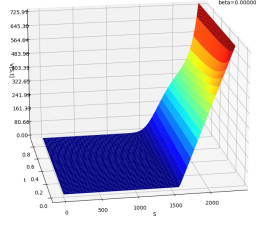
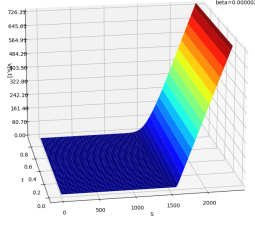
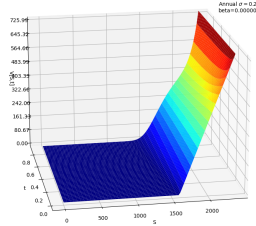
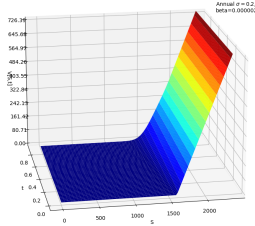
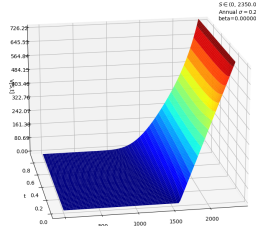
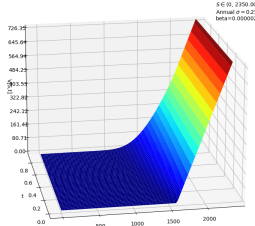
European	American
<p>Method of lines, non-linear Black-Scholes equation. Strike price: <math>K=1600.0</math> Initial condition: <math>V(t, S) = \max(S-K, 0)</math> Boundary condition: <math>V(S=0, t) = 0</math> Boundary condition: <math>V(t, S=2000) = S-K</math> <math>\sigma \in [0, 0.25]</math>, <math>r = 0.05</math> and <math>\rho \in [0, 0.9]</math> Annual <math>\rho = 0.25</math>, <math>r = 0.05</math> beta=0.000002</p>  <p>1.392</p>	<p>Method of lines, non-linear Black-Scholes equation. Strike price: <math>K=1600.0</math> Initial condition: <math>V(t, S) = \max(S-K, 0)</math> Boundary condition: <math>V(S=0, t) = 0</math> Boundary condition: <math>V(t, S=2000) = S-K</math> <math>\sigma \in [0, 0.25]</math>, <math>r = 0.05</math> and <math>\rho \in [0, 0.9]</math> Annual <math>\rho = 0.25</math>, <math>r = 0.05</math> beta=0.000002</p>  <p>1.392</p>
<p>Method of lines, non-linear Black-Scholes equation. Strike price: <math>K=1600.0</math> Initial condition: <math>V(t, S) = \max(S-K, 0)</math> Boundary condition: <math>V(S=0, t) = 0</math> Boundary condition: <math>V(t, S=2000) = S-K</math> <math>\sigma \in [0, 0.25]</math>, <math>r = 0.05</math> and <math>\rho \in [0, 0.9]</math> Annual <math>\rho = 0.25</math>, <math>r = 0.05</math> beta=0.000002</p>  <p>1.907</p>	<p>Method of lines, non-linear Black-Scholes equation. Strike price: <math>K=1600.0</math> Initial condition: <math>V(t, S) = \max(S-K, 0)</math> Boundary condition: <math>V(S=0, t) = 0</math> Boundary condition: <math>V(t, S=2000) = S-K</math> <math>\sigma \in [0, 0.25]</math>, <math>r = 0.05</math> and <math>\rho \in [0, 0.9]</math> Annual <math>\rho = 0.25</math>, <math>r = 0.05</math> beta=0.000002</p>  <p>1.912</p>
<p>Method of lines, non-linear Black-Scholes equation. Strike price: <math>K=1600.0</math> Initial condition: <math>V(t, S) = \max(S-K, 0)</math> Boundary condition: <math>V(S=0, t) = 0</math> Boundary condition: <math>V(t, S=2000) = S-K</math> <math>\sigma \in [0, 0.25]</math>, <math>r = 0.05</math> and <math>\rho \in [0, 0.9]</math> Annual <math>\rho = 0.25</math>, <math>r = 0.05</math> beta=0.000002</p>  <p>3.570</p>	<p>Method of lines, non-linear Black-Scholes equation. Strike price: <math>K=1600.0</math> Initial condition: <math>V(t, S) = \max(S-K, 0)</math> Boundary condition: <math>V(S=0, t) = 0</math> Boundary condition: <math>V(t, S=2000) = S-K</math> <math>\sigma \in [0, 0.25]</math>, <math>r = 0.05</math> and <math>\rho \in [0, 0.9]</math> Annual <math>\rho = 0.25</math>, <math>r = 0.05</math> beta=0.000002</p>  <p>3.575</p>

Table 2: Option pricing for European and American style using non-linear Black-Scholes model.  $S_0 = 1500$  and  $K = 1600$