# Flex Option Price Calculator

YASHKIR CONSULTING
www.yashkir.com

## Contents



## 1 Flex Option specification

This option is based on rights of the buyer to receive arbitrary number of commodity units at the end of each time period $\Delta_k$ with maximal number of delivery units $N$. Delivery of $x_k$ units (partial excercise of the option at time $\tau_k$) is requested if market price of the commodity unit exceeds (for Call option) the strike price $K$. The contract may have special condition: if number of units received is below pre-agreed minimum $w$ then receiver is penalized at maturity $T$ by $[(w - x_k) \cdot f]$ where $f$ is the penalty fee per unit.

## 2 Methodology

We consider this option as a set of $k_m$ European-style options (with their values of $v_k$ for corresponding number of $x_k$). Since numbers $x_k$ are not known we derive the option price as maximum of the following function

$$F(x) = \sum_{k=1}^{k_m} v_k(x_k) - e^{-r \cdot T} \sum_{k=1}^{k_m} P_k(x_k) \tag{1}$$

where non-exercise penalty for time period $\tau_k$ is

$$P_k = f \cdot \max(0, w - x_k) \tag{2}$$

## 2.1 Price tree

The price of an underlying commodity unit (for the spot price of $S_0$, volatilitty of $\sigma$, and convenience yield of $q$) can be simulated using the trinomial tree algorithm. Time axis is presented with discreet time points $t_j = j \cdot dt$ ($dt$ is the time step and $T = n \cdot dt$ is the option maturity, $j = 0 \cdots n$). Simulated prices at time points $t_j$ and tree nodes $i$ are

$$S_{ji} = S_0 \cdot u^{i-j} \tag{3}$$
$$j = 0 \cdots n \tag{4}$$
$$i = 0 \cdots 2j \tag{5}$$

The scale factor for price moving up by $u$, or moving down by $u^{-1}$ is:

$$u = e^{\sigma\sqrt{2dt}} \tag{6}$$

Probabilities for price movement up ($p_u$), down ($p_d$) or staying the same ($p_m$) are:

$$p_u = \frac{\sqrt{u} \cdot e^{-(r-q)dt/2} - 1}{(u-1)^2} \tag{7}$$
$$p_d = \frac{\sqrt{u} \cdot e^{-(r-q)dt/2} - u}{(u-1)^2} \tag{8}$$
$$p_m = 1 - p_u - p_d \tag{9}$$

## 2.2 Backward induction

European option pricing is used in Flex Option calculation. The American and Bermudan options are calculated also for comparison.

**The value of an European option at the end of time period $\tau_k$** is:

$$V_{\tau_k,i} = \max\left[0, \pm(S_{\tau_k,i} - K)\right] \tag{10}$$

Here signs $\pm$ correspond to call/put options.
The connection between a derivative price $V_{j-1,i}$ at time $t_{j-1}$ (the node $i$) and derivative prices $V_{j,i+2}$, $V_{j,i+1}$, and $V_{j,i}$ at a time $t_j$ (nodes $i$, $i+1$ and $i+2$) is:

$$V_{ji} = e^{-r\Delta t}\left(p_u V_{j+1,i+2} + p_m V_{j+1,i+1} + p_d V_{j+1,i}\right) \tag{11}$$

The equation (11) provides the tool for backward derivative price calculation. Working backward from $j = j_k$ to $j = 0$ with equation (11) one can obtain the derivative price $V_{00}$ at $t = 0$ (here $j_k$ is the time point corresponding to the end $\tau_k$ of the $k^{th}$ period.

**American Option** In case of American Option the exercise can occure at any time. To take this into account we modify the node values (Equation 11) as follows

$$V^A_{ji} = \max\left[V_{ji}, \pm(S_{ji} - K)\right] \tag{12}$$

**Bermudan Option** Bermudan option can be exercised at any time $\tau_k$ (including maturity). The node values in this case are modified at exercise time points $\tau_k$ as follows:

$$V^B_{j_k i} = \max\left[V_{j_k i}, \pm(S_{j_k i} - K)\right] \tag{13}$$

This numerical procedure leads to the option value at $t = 0$:

$$V_{00} \tag{14}$$

## 2.3 Optimisation

The set of units exercised at times $\tau_k$ is derived from optimisation procedure using function 1 with constraint

$$\sum_{k=1}^{n} x_k = N \tag{15}$$

The optimisation used in this pricer can be one of: the downhill simplex method (Nelder-Mead), the Broyden–Fletcher–Goldfarb–Shanno algorithm using a limited amount of computer memory (L-BFGS-B), the Sequential Least Squares Programming method (SLSQP). At each iteration $s$ of the optimisation the following adjustment is performed (to satisfy constraint):

$$x_k^{(s+1)} = x_k^{(s)} \cdot \frac{N}{\sum\limits_k x_k^{(s)}} \tag{16}$$

# 3 Pricer prototype

The prototype of the option pricer has been developed in Python 3.5.2. The prototype interface (Fig.1) allows to set desired parameters of the option, to choose the optimisation method (the downhill simplex method: Nelder-Mead, the Broyden–Fletcher–Goldfarb–Shanno algorithm using a limited amount of computer memory: L-BFGS-B, the Sequential Least Squares Programming method SLSQP), and the option type (Flex, European, American, Bermudan). Parameter "Zero iteration" can be used to accelerate the optimization, it sets the exercise schedule with equal number of units for each period.

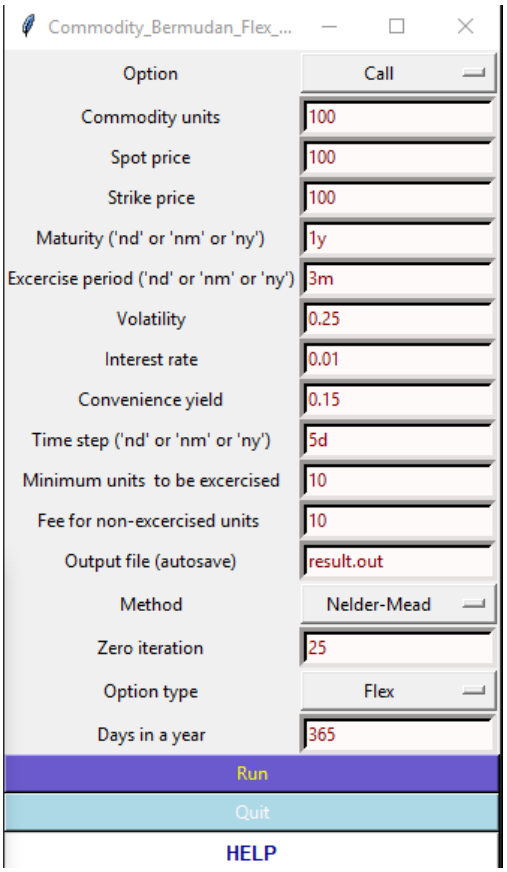


Figure 1: Interface

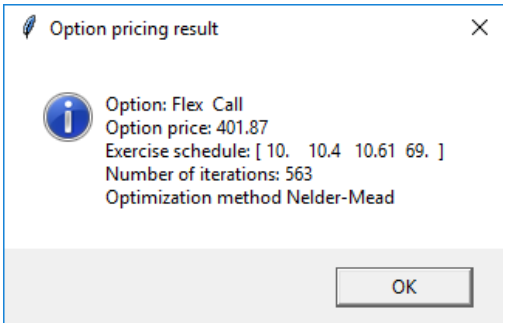


Figure 2: Result

The output is displayed as in Figure 2.

# 4 Pricing tests

Option pricer was tested with the following parameters:

Table 1: Parameters

| Parameters | Maturity | Time step | Period | S | K | r | $\sigma$ | q | w |
|---|---|---|---|---|---|---|---|---|---|
| Values | 1y | 5d | 3m | 100 | 100 | 0.01 | 0.25 | $0.02 \cdots 0.20$ | 10 |

Results are presented in tables 2 and 3 [1]

Table 2: Option prices (low convenience yield q = 0.02, different penalty fees $f$)

| | f = 0.0 | | f = 10 | | f=20 | |
|---|---|---|---|---|---|---|
| Option Style | Call | Put | Call | Put | Call | Put |
| European | 929.81 | 1028.32 | | | | |
| Bermudan | 929.81 | 1028.32 | | | | |
| American | 939.98 | 1028.32 | | | | |
| Flex | 929.81 | 1028.32 | 847.66 | 984.35 | 848.84 | 933.13 |
| n[k] | 0,0,0,100 | 0,0,0,100 | 10,11,10,69 | 0,10,10,80 | 10,10,10,70 | 10,10,10,70 |

Table 3: Option prices (high convenience yield q = 0.20, different penalty fees $f$)

| | f = 0.0 | | f = 10 | | f=20 | |
|---|---|---|---|---|---|---|
| Option Style | Call | Put | Call | Put | Call | Put |
| European | 287.05 | 2000.24 | | | | |
| Bermudan | 287.05 | 2000.24 | | | | |
| American | 476.50 | 2000.24 | | | | |
| Flex | 309.74 | 2000.24 | 304.53 | 1885.23 | 304.47 | 1885.23 |
| n[k] | 0,100,0,0 | 0,0,0,100 | 10,70,10,10 | 0,10,10,80 | 11,69,10,10 | 0,10,10,80 |

[1] n[k] is the exercise schedule