Advanced Derivatives: Problem set 11

Tommaso Farnararo, Nicolo' Taroni, Girolamo Vurro
Financial Engineering, EPFL, Switzerland

***Abstract*—In this project we implement the Andersen Algorithm to price a *receiver* Bermudean swaption.**

I. Initial settings

For this problem we consider a flat term structure at 5% and a *receiver* Bermudean option with the following characteristics:

- The first exercise date is $T_s = 1$ (year) and the last exercise date is $T_x = 2$.
- The underlying swap has quarterly parameters ($\tau = 0.25$) and the final payment is at $T_N = 2$.

Therefore, we have the set of dates $[T_0, T_1, \ldots, T_{16}]$ with $T_i = i \cdot 0.25$, for $i \in \{0, 1, \ldots, 16\}$. The set of Forward rates F_k, $k = 1, \ldots, 16$ (F_1 which is the spot rate) is characterized by the forllowing constant volatilities:

- $\sigma_1 = 0$;
- $\sigma_k = 0.2$ for $k = 2, \ldots, 7$
- $\sigma_k = 0.22$ for $k = 8, \ldots, 11$
- $\sigma_k = 0.24$ for $k = 12, \ldots, 16$

Moreover, we have that every dW_k driving the corresponding F_k is a linear combination of two *independent* Brownian motions $dW^{(1)}$ and $dW^{(2)}$. This implies that $dW^{(1)}dW^{(2)} = 0$. For every $k > 1$:

$$dW_k = cos(\theta_k)dW^{(1)} + sin(\theta_k)dW^{(2)} \tag{1}$$

where

$$\theta_k = \frac{\pi}{2}\frac{k-2}{14}$$

Moreover, for each $j, k \in \{2, \ldots, 16\}$ we have that:

$$\begin{aligned} dW_j dW_k = \rho_{jk}dt &= (cos(\theta_j)cos(\theta_k) + sin(\theta_j)sin(\theta_k))dt \\ &= cos(\theta_j - \theta_k)dt \end{aligned} \tag{2}$$

and in particular, in the last equation, we used the *second* and *third Werner's formulas*. The equation (1) and the result (2) imply the following:

$$\begin{aligned} dW_2 &= dW^{(1)} \\ dW_{16} &= dW^{(2)} \\ dW_{j-1}dW_j &= cos(\pi/28)dt = 0.9937dt. \end{aligned}$$

II. Methodologies

A. Simulation of several paths of Forward rates

To be able to implement the Andersen Algorithm, we need to simulate several paths of Forward rates under the LR-Spot measure. We know from the lecture that for a generic Forward rate

$$F_k(t) = F(t, T_{k-1}, T_k)$$

the dynamic follows under the LR Spot measure:

$$dF_k(t) = F_k(t)\left(\sigma_k \sum_{j=\beta(t)}^{k} \frac{\tau\rho_{jk}\sigma_j F_j(t)}{1+\tau F_j(t)}dt + \sigma_k dW_k(t)\right).$$

Note that in our case σ_k does not depend on the time t but only on the maturity of the Forward rate we are considering, as specified in the Problem Set description. For simplicity, we can rename the drift part as

$$\mu_k(t) = \sigma_k \sum_{j=\beta(t)}^{k} \frac{\tau\rho_{jk}\sigma_j F_j(t)}{1+\tau F_j(t)}.$$

To be able to simulate the value of the Forward rate at T_j using the formula:

$$F_k(T_j) = F_k(T_{j-1})e^{((\mu_k(T_{j-1}) - \frac{1}{2}(\sigma_k(T_{j-1}))^2)\tau + \sigma_k\Delta W_k(T_j))}$$

we need to define some functions that calculate $\rho_{jk}, \mu_k(t)$ and $\Delta W_k(T_j)$. As specified before, we assume

$$dW_k = cos(\theta_k)dW^{(1)} + sin(\theta_k)dW^{(2)}$$

with $dW^{(1)}$ independent from $dW^{(2)}$ and it follows as showed before that $\rho_{jk} = cos(\theta_j - \theta_k)$, so we build a function for computing it. To describe how we simulate the increments of the Forward, we only need to describe how to simulate ΔW_k. Since the increment of a Brownian Motion can be assumed to be Normal Distributed with zero mean and variance equal to the increment in time, we use the definition of dW_k written above and integrate it to build a function that calculates this increment as

$$\Delta W_k = cos(\theta_k)Z_1\sqrt{\tau} + sin(\theta_k)Z_2\sqrt{\tau}$$

where Z_1 and Z_2 are two simulations of standard normal random variables independent of each other. Using the formulas that we have described, we build a 16×16 matrix, where the $i-th$ row represents the value of the Forward rates with different maturities at a given instant of time T_i, and the $k-th$ column gives the different values of the Forward rate with T_k maturity for each instant of Time T_i. We can observe that each Forward Rate lives until one period before its maturity.

III. Implementation of the Andersen Algorithm

A. Initial set-up

In order to implement the *Andersen Algorithm* we first created the following functions:

- A function that enables us to compute

$$P(T_j, T_k) = \prod_{i=j+1}^{k} \frac{1}{1 + \tau F_i(T_j)}$$

which is the "horizontal discount" factor. In particular, given an exercise date T_j, it enables us to discount **at T_j** the future payoffs generated by the exercised swaption. For example, the rates used to compute $P(T_4, T_{16})$ are those in red in the following matrix:

$$\begin{array}{ccccccc}
F_1(T_0) & F_2(T_0) & \dots & \dots & \dots & \dots & F_{16}(T_0) \\
 & F_2(T_1) & \dots & \dots & \dots & \dots & F_{16}(T_1) \\
 & \ddots & \dots & \dots & \dots & \dots & \dots \\
 & & F_4(T_3) & \dots & \dots & \dots & F_{16}(T_3) \\
\hline\hline
 & & & F_5(T_4) & \dots & \dots & F_{16}(T_4) \\
 & & & & F_9(T_8) & \dots & F_{16}(T_8)
\end{array}$$

- A function that computes

$$\mathcal{S}_{T_j, T_{16}} = \tau \sum_{k=j+1}^{16} P(T_j, T_k)(K - F_k(T_j))$$

which is the value of the underlying swap at T_j for a *receiver* swaption. Note that $j \in \{4, 5, 6, 7, 8\}$ are indexes of the possible exercise dates.
- A function that computes

$$D(0, T_e) = \prod_{j=1}^{e} \frac{1}{1 + \tau F_j(T_{j-1})}$$

namely the **stochastic discount factor** computed by using the **simulated spot rates**. This is similar to the stochastic discount factor in the risk-neutral rate

$$P(0, T_e) = \exp\left\{ - \int_0^{T_e} r_s ds \right\}$$

and for every time s, r_s is the instantaneous *spot rate* on each path. In our case, $e \in \{4, 5, 6, 7, 8\}$ and represents the "diagonal discount" rate which we use to compute the NPV (discount at T_0) of future cash flows on each path, provided that we exercise at T_e:

$$D(0, T_e)\mathcal{S}_{T_e, T_{16}}$$

For example, we use the red rates from the matrix below to compute $D(0, T_5)$:

$$\begin{array}{ccccccc}
F_1(T_0) & F_2(T_0) & \dots & \dots & \dots & \dots & F_{16}(T_0) \\
 & F_2(T_1) & \dots & \dots & \dots & \dots & F_{16}(T_1) \\
 & \ddots & \dots & \dots & \dots & \dots & \dots \\
 & & F_4(T_3) & \dots & \dots & \dots & F_{16}(T_3) \\
\hline\hline
 & & & F_5(T_4) & \dots & \dots & F_{16}(T_4) \\
 & & & & F_9(T_8) & \dots & F_{16}(T_8)
\end{array}$$

B. Procedure adopted

Once we settled the previous functions and we simulated different paths for the Forward rates F_k under the LR-Spot measure, we were ready to apply the algorithm. In particular, we followed the following steps:

1) We started as if we had only the last exercise date (namely T_8). The exercise strategy here is simple:
   - Exercise if $\mathcal{S}_{T_8, T_{16}} > 0$
   - Do not exercise if the previous condition is violated.
2) Then we considered the case in which we have only the *last two* exercise dates. In this case, we would adopt the following strategy:
   - Exercise at T_7 if $\mathcal{S}_{T_7, T_{16}} > H$, where H is a threshold value to be determined.
   - Exercise at T_8 if the previous condition is not fulfilled and if $\mathcal{S}_{T_8, T_{16}} > 0$.
   - Do not exercise whenever none of the previous points is fulfilled.

   Therefore, with this strategy, we would have a NPV (for one path) computed as follows

$$V_0^{(7)} = D(0, T_7)\mathcal{S}_{T_7, T_{16}} \mathbb{1}_{\{\mathcal{S}_{T_7, T_{16}} > H\}} + D(0, T_8)\mathcal{S}_{T_8, T_{16}} \mathbb{1}_{\{\mathcal{S}_{T_8, T_{16}} > 0\}} \mathbb{1}_{\{\mathcal{S}_{T_7, T_{16}} \leq H\}}$$

   Thus, we selected the value of H that maximized the average of previous NPV over the different simulated paths and we saved it as $H(T_7)$ (we selected H from a set of values that ranged from 0 to H_{max}; the latter was settled after visualizing the maximum value assumed by $\mathcal{S}_{T_7, T_{16}}$ over our different simulations; we repeated this specific procedure also for the selection of H_{max} for the next iterations of the algorithm: the H_{max} for T_i with $i \in \{4, 5, 6, 7\}$ are shown in table I).

T_i	Max. Swaption value $\mathcal{S}_{T_i, T_N}$
$i = 4$	0.0274754
$i = 5$	0.03097767
$i = 6$	0.0292534
$i = 7$	0.03462074

TABLE I: Maximum Swaption value obtained through 1000 simulation for each Exercise date

3) Then, we proceeded considering the scenario in which we would have the possibility to exercise only at the *last three* exercise dates $[T_6, T_7, T_8]$. Following the same logic as before, the NPV of one path of this strategy is:

$$\begin{aligned}
V_0^{(6)} &= D(0, T_6)\mathcal{S}_{T_6, T_{16}} \mathbb{1}_{\{\mathcal{S}_{T_6, T_{16}} > H\}} \\
&+ D(0, T_7)\mathcal{S}_{T_7, T_{16}} \mathbb{1}_{\{\mathcal{S}_{T_7, T_{16}} > H(T_7)\}} \mathbb{1}_{\{\mathcal{S}_{T_6, T_{16}} \leq H\}} \\
&\quad + D(0, T_8)\mathcal{S}_{T_8, T_{16}} \mathbb{1}_{\{\mathcal{S}_{T_8, T_{16}} > 0\}} \cdot \\
&\qquad \cdot \mathbb{1}_{\{\mathcal{S}_{T_7, T_{16}} \leq H(T_7)\}} \mathbb{1}_{\{\mathcal{S}_{T_6, T_{16}} \leq H\}}
\end{aligned}$$

Therefore, also this time we computed the $H(T_6)$ that maximized the average of $V_0^{(6)}$ and we saved it.

4) We applied this procedure also for the previous exercise dates, and we saved $H(T_5)$, $H(T_4)$ which completed the set of optimal thresholds for our exercise strategy. In order to better visualize the value of the $H(T_i)$ that maximized the value of each recursive strategy (average of the NPV at $t = 0$ of every single path), we reproduced the plots of the respective value of the strategy at T_0 as a function of the threshold H, and we present them respectively with the figures 1, 2, 3, 4.
5) Once we had our optimal exercise strategy, we **generated new paths** for the Forward rates and we priced the Bermudean swaption following the optimal strategy, i.e. for each path, determining the exercise time T_e and computing the time 0-NPV of future cash flows. In particular, for each simulation $m \in \{1, \ldots, 1,000\}$ we computed:
   - $\varepsilon_i = \{\mathcal{S}^m_{T_i,T_{16}} > H(T_i)\}$, for $i \in \{4, 5, 6, 7\}$
   - $\varepsilon_8 = \{\mathcal{S}^m_{T_8,T_{16}} > 0\}$

$$\begin{aligned} V_0(m) = {} & D_m(0, T_4)\mathcal{S}^m_{T_4,T_{16}} \mathbb{1}_{\{\varepsilon_4\}} \\ & + D_m(0, T_5)\mathcal{S}^m_{T_5,T_{16}} \mathbb{1}_{\{\varepsilon_5\}} \mathbb{1}_{\{\varepsilon_4^C\}} \\ & + \ldots + D_m(0, T_8)\mathcal{S}^m_{T_8,T_{16}} \mathbb{1}_{\{\varepsilon_8\}} \cdot \\ & \cdot \mathbb{1}_{\{\varepsilon_7^C\}} \mathbb{1}_{\{\varepsilon_6^C\}} \mathbb{1}_{\{\varepsilon_5^C\}} \mathbb{1}_{\{\varepsilon_4^C\}} \end{aligned}$$

   And therefore we computed the final price (assuming notional 1):

$$Price_0 = \frac{1}{N_{sim}} \sum_{m=1}^{1000} V_0(m)$$

IV. Result

After we implemented the Andersen Algorithm that we described in the previous section, we were able to compute the price of the *receiver* Bermudean swaption at the initial time:

$$Price_0 = 0.00461878110291$$

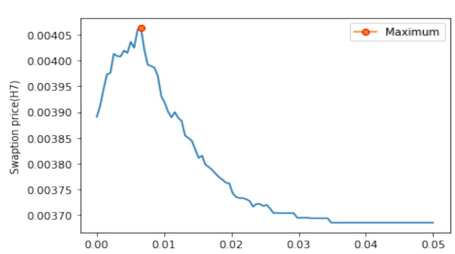

Fig. 1: Swaption price at 0 as a function of the Exercise threshold H for T_7

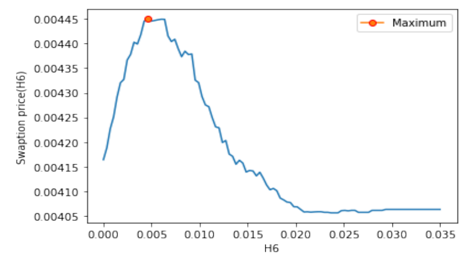

Fig. 2: Swaption price at 0 as a function of the Exercise threshold H for T_6

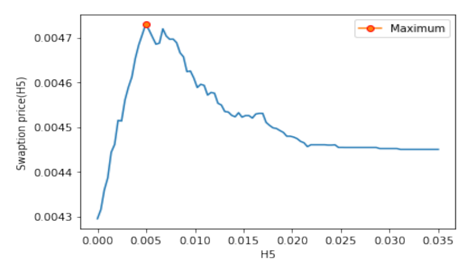

Fig. 3: Swaption price at 0 as a function of the Exercise threshold H for T_5

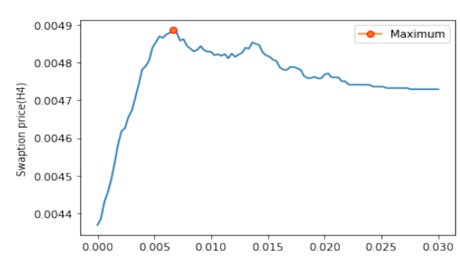

Fig. 4: Swaption price at 0 as a function of the Exercise threshold H for T_4