FIN 503: Advanced derivatives **(Due: 15/12/20)**

Assignment #11

Professor: Elena Perazzi *Assistant:* Marc-Aurèle Divernois

Students: Hien Lê, Francesco Maizza, Anita Mezzetti

See Matlab code for implementation (3 files). We proceeded to calculate the swaption price as follows:

Given params:

$$r = K = 0.05 \tag{0.1}$$
$$T_s = 1 \tag{0.2}$$
$$T_x = 2 \tag{0.3}$$
$$\tau = dt = 0.25 \tag{0.4}$$
$$T_N = 4 \tag{0.5}$$
$$\sigma_k = \begin{cases} = 0.2 \text{ for } k = 2, ...7, \\ = 0.22 \text{ for } k = 8, ...11, \\ = 0.24 \text{ for } k = 12, ...16 \end{cases} \tag{0.6}$$
$$dW_k = cos(\theta_k)dW^{(1)} + sin(\theta_k)dW^{(2)} \ (W^{(1)} \perp W^{(2)}) \tag{0.7}$$
$$\theta_k = \frac{\pi}{2}\frac{k-2}{14} \tag{0.8}$$

In addition, we also have the parameter $N_{sim} = 10,000$ for the number of Monte-Carlo simulations.

1. **Simulation of Forward Rates**:

   For this part, we used the theories of Lecture 10 and Lecture 11. In particular, given t the index of an exercise date, and k the index of a swap payment date, we found the correlation between two forward rates to be:

   $$\rho_{tk} = cos(\theta_t - \theta_k)$$

   We then construct the risk factor dW_k driving each forward rate F_k as:

   $$dW_k = \cos(\theta_k)\sqrt{\tau}Z^{(1)} + \sin(\theta_k)\sqrt{\tau}Z^{(2)}$$

   with $Z^{(1)}, Z^{(2)}$ being drawn from the standard normal distribution with a sample of N_{sim} in size (using `randn`).

Now, using the formula from slide 6 of Lecture 11, we have that for each exercise date T_j and forward rate F_k:

$$\mu_k(t) = \sigma_k(t) \sum_{j=t}^{k} \frac{\tau \rho_{jk} \sigma_j(t) F_j(t)}{1 + \tau F_j(t)} dt \tag{0.9}$$

$$F_k(T_j) = F_k(T_{j-1}) \exp((\mu_k - 0.5\sigma_k^2)\tau + \sigma_k dW_k) \tag{0.10}$$

Now we have N_{sim} paths of forward rates over two dimensions - t (exercise dates of swaption) and k (payment dates of swap).

2. **Find best** $H(T_{x-i})$

Having had the forward rates' paths, we can now apply Andersen algorithm.

- Using the forward rates calculated, we computed the **value** of the underlying swap at each exercise date T_j (where $T_j \geq T_s/\tau = 4$, since $T_s = 1$ is the first exercise date). The formula can be found on slide 9 of Lecture 11:

$$P(T_j, T_N) = \Pi_{i=j+1}^{k} \frac{1}{1 + \tau F_i(T_j)} \tag{0.11}$$

$$S_{T_j,T_N} = \tau \sum_{k=j+1}^{N} P(T_j, T_k)(K - F_k(T_j)) \text{ (it's a \textbf{receiver} swaption)} \tag{0.12}$$

- Now that we have a matrix S of dimension $(N_{sim}, 5)$ (4 exercise dates + initial value), we applied the steps of the algorithm on slide 10 and 11 of Lecture 11 for each exercise date T_{x-i} ($i \in 0.25 \times \{1, 2, 3, 4\}$) to get the values $H(T_{x-i})$ that generated the greatest (discounted) values of the Swaption at each T_{x-i}. In this exercise we considered a large range of possible H for each exercise date, that is, $H(T_{x-i}) \in (1e^{-4}, 0.05)$ (0.05 being the strike/flat term structure) with a step of $1e^{-4}$.

  For each time that we step back from the last exercise date, we can determine the exercise time T_e as well as the discount factor from this date as:

$$D(0, T_e) = \Pi_{j=1}^{e} \frac{1}{1 + \tau F_j(T_{j-1})} \tag{0.13}$$

  and so the discounted future cash flow will be:

$$CF(0, T_e) = D(0, T_e) \times S_{T_e,T_N} \tag{0.14}$$

  where the matrix S has already been found in the substep above. To get the value of the swaption every time we step back, we took the mean value across all N_{sim} paths.

- For each time we step back to T_{x-i}, we can find the corresponding value of $H(T_{x-i})$ that generated the greatest value of the swaption. We store this in our array of optimal Hs for the exercise dates. These values were found to be, in the order of $x - i$:

```
0.0017
0.0035
0.0045
0.0054
```

3. **Price Swaption with optimal** $H(T_{x-i})$

   Now that we've got the optimal H for each exercise date, we used this to price the swaption: we generated new forward rates' paths as well as values of swap at last exercise dates, and ran the Andersen algorithm for all of the S matrix, which contains the intrinsic values (from first exercise date to last). This time, with each T_{x-i}, we compare the value of the swap (by looking up S_{T_{x-i},T_N}) with $H(T_{x-i})^*$ which we already found in the previous step.

   After doing this down to the first possible exercise date of the matrix S and hence determining the exercise date T_e for each path, we calculated the discounted cash flow using equations (0.13) and (0.14). To get the price of the swaption, we took the mean value across all N_{sim} paths. The swaption's price was found to be 0.0053037 (or 53.037 in money units).

Appendix: Matlab code

```
As11.m %main file
%% parameters
% for swaption
t = 0; % first date
r = 0.05; % term structure (flat)
T_s = 1; % first ex date
T_x = 2; % last ex date

% for underlying swap
tau = 0.25;
T_N = 4; % last payment date
payment_dates = (0:16).*tau;

% corresponding forward rates
vols_fw = zeros(16,1);
vols_fw(2:7) = 0.2;
vols_fw(8:11) = .22;
vols_fw(12:16) = .24;

thetas = pi/2 * ((1:16) - 2)./14;

% other params
N_sim = 10000;
K = .05;
N_x = length(tau:tau:T_x);
N_n = length(tau:tau:T_N);
Hs = 1e-4:1e-4:r;

%% computation for best H

[F,S] = ForwardRates(r,vols_fw,thetas,K,tau,N_sim,N_x,N_n);
%disp(S)
first_ex_date = T_s/tau;
S = S(:,first_ex_date+1:end);
%disp(S)
[~,num_ex_dates] = size(S);
best_Hs = zeros(num_ex_dates-1,1);

% price of swaption at 4 dates
for t=1:4
    avg_swaption_prices = zeros(length(Hs),1);
    S_t = S(:,end-t:end);

    % discounted factor (slide 12 Lecture 11)
    D_t = cumprod(1./(1+tau*F(:,end-t:end,end-t)),2);
```

```
    %disp(D_t);
    %if (t==4)
    %    disp(D_t)
    %end

    % experimenting with different H
    for i=1:length(Hs)
        [~,dim2_S] = size(S_t);
        H = Hs(i).*ones(dim2_S,1);
        %disp(H)
        CF_H = Swaptions(S_t,H,D_t);
        %disp(size(D_t))
        avg_swaption_prices(i) = mean(CF_H);
    end
    [~,argmax] = max(avg_swaption_prices);
    best_Hs(t) = Hs(argmax);
end

%% Generate new paths and compute prices with optimal Hs
[F,S] = ForwardRates(r,vols_fw,thetas,K,tau,N_sim,N_x,N_n);

first_ex_date = T_s/tau;
S = S(:,first_ex_date:end);
[~,num_ex_dates] = size(S);

SPs = S;
S_t = S(:,end-4:end);
D_t = cumprod(1./(1+tau*F(:,end-4:end,end-t)),2);
CF_t = Swaptions(S_t,best_Hs,D_t);
fprintf("The swaption price is %.7f", mean(CF_t))
%The swaption price is 0.0053037
```

ForwardRates.m

```
function [F, S] = ForwardRates(r, sigma, theta, K, dt, N_sim, N_x, N_n)
% Dynamics of Forward rates under Spot-LIBOR measure

% N_x: Number of exercise dates
% N_n: Number of swap payment dates

%dW_k = cos(theta_k) dW_1 + sin(theta_k) dW_2

F = zeros(N_sim, N_x+1, N_n);
S = zeros(N_sim, N_x+1);
F(:, 1, :) = r*ones(N_sim, 1, N_n);

for t=2:N_x+1
    temp = 0;
```

```
    for k=t:N_n
        dW_1 = randn(N_sim, 1);
        dW_2 = randn(N_sim, 1);

        dW_k = (cos(theta(k))*dW_1 + sin(theta(k))*dW_2)*sqrt(dt);

        % slide 19 Lecture 10
        rho_tk = cos(theta(t) - theta(k));

        % slide 6 Lecture 11
        temp = temp + dt*rho_tk*sigma(k)*F(:,t-1,k)./(1+dt*F(:,t-1,k));
        mu_k = sigma(k)*temp;
        F(:,t,k) = F(:,t-1,k).*exp((mu_k - 0.5*sigma(k)^2)*dt +...
           sigma(k)*dW_k);
    end
    %disp(F)

    % compute value of underlying swap (intrinsic val) at each ...
    % exercise date
    % Andersen algo: slide 9 Lecture 11
    P = zeros(N_sim, N_n);

    if (t > 1/dt)
        P(:, :) = cumprod(1./(1+dt.*F(:,t,:)),3);
        P(:,1:t-1) = 0;

        % Value of underlying swap at t (exercise date)
        S(:, t) = dt*diag(P * (reshape(K-F(:,t,:), [N_sim,N_n]))');
    end
end
end
```

Swaptions.m

```
function [CF] = Swaptions(S,H,D)
   % slide 10 to 12: Goal is to compute discounted cashflows as...
   %  D(0,T_e)*S(T_e, T_N)
   [~,num_dates] = size(S);
   CF_t = zeros(size(S));
   %disp(num_dates)
   for i=1:num_dates-1
       ex_idx = logical(S(:,end-i) > H(end-i+1));
       CF_t(ex_idx,end-i) = S(ex_idx,end-i).*D(ex_idx, end-i+1);
       CF_t(ex_idx,end-i+1:end) = 0;
       for j=1:i
           ex_idx_next = logical(~(S(:,end-i+j-1) > H(end-i+j)).*...
               (S(:,end-i+j)>H(end-i+j)));
           CF_t(ex_idx_next, end-i+j) = S(ex_idx_next,end-i+j).*...
```

```
                D(ex_idx_next,end-i+j);
        end
    end
     %disp(CF_t);
%      if(num_dates ==3)
%           disp(CF_t)
%      end
    CF = sum(CF_t,2);
    %disp(CF)
end
```