Exercise 1

Write a class implementing the interface

`com.andreamazzon.exercise3.Swap`

you find in the exercises repository.

In particular, you have to write methods that calculate the value of an interest rate Swap (for given fixed swap rates S_i, $i = 0, \dots, n-1$) as well as the par swap rate S at evaluation time $t = 0$, in correspondence to a tenure structure $0 = T_0 < T_1 < \dots < T_n = T$ given in the constructor of your class.

Take into account the fact that the user can provide to the constructor of your class both the forward rates $L(T_i, T_{i+1}; 0)$, $i = 0, \dots, n-1$ or the initial zero-bond curve $P(T_i; 0)$, $i = 0, \dots, n-1$, both as array of doubles. To deal with this issue, you can use the methods written for Exercise 1, Handout 2.

Also try to give a different implementation to your methods in the case when the settlement dates $0 = T_0 < T_1 < \dots < T_n = T$ are evenly distributed, i.e., $\Delta_i := T_{i+1} - T_i = \delta > 0 \ \forall i = 1, \dots, n$: in this case, you can save time avoiding to compute Δ_i more than once.

Check then if the value for the Swap you get by choosing $S_i = S \ \forall i$ is actually (close to) 0. For this purpose, take $T = 3$, $\Delta_i = \delta = 0.5$ (half year), and the following zero coupon bond curve:

Maturity	Price
6m	0.9986509108
1yr	0.9949129829
1.5yr	0.9897033769
2 yr	0.9835370208
2.5 yr	0.9765298116
3 yr	0.9689909565

Exercise 2

Suppose you have a tenure structure

$$T_0 = 0 < T_1 < \dots < T_{n-1} < T_n = T$$

of evenly distributed points (i.e., $T_i - T_{i-1} = \Delta$ for every $i = 1, \dots, n$), and call S_i, $i = 3, \dots, n$, the par swap rate for the partial tenure structure

$$T_0 = 0 < T_1 < \dots < T_{i-1} < T_i,$$

i.e., the value S_i such that

$$\sum_{j=1}^{i-1} \left(L(T_j, T_{j+1}; 0) - S_i\right) P(T_{j+1}; 0)(T_{j+1} - T_j) = 0.$$

Also suppose that you are given the first two bonds $P(T_1; 0)$, $P(T_2; 0)$.

(a) Similarly to what you had to do for Exercise 2, Handout 2, write a class `BootstrapFromParSwapRate` in order to do the following:

  (i) get the value of $P(T_i; 0)$ knowing the value of $P(T_1), \dots, P(T_{i-1}; 0)$ and the par swap rate S_i, $i > 2$;

  (ii) get the value of $P(T_i; 0)$ and $P(T_{i-1}; 0)$ knowing the value of $P(T_1; 0), \dots, P(T_{i-2}; 0)$ and the par swap rate S_i, $i > 2$.

Try to give an implementation of this class such that these calculations can be done recursively, i.e., the newly computed values can be stored and used to compute the next ones.

(b) Write a test where you call the methods of this class in order to compute the value of the bonds $P(T_i;0)$, $i > 2$ when you have $\Delta = 0.5$, $P(T_1;0) = 0.98$, $P(T_2;0) = 0.975$, and

$$S_3 = 0.0086, \quad S_4 = 0.0077, \quad S_5 = 0.0073, \quad S_6 = 0.0084,$$
$$S_8 = 0.0075, \quad S_{10} = 0.0085, \quad S_{12} = 0.0095, \quad S_{14} = 0.0092.$$

You should get the following results:

$$P(0.5;0) = 0.9800, \quad P(1;0) = 0.9750, \quad P(1.5;0) = 0.9716, \quad P(2;0) = 0.9688$$
$$P(2.5;0) = 0.9658, \quad P(3;0) = 0.9597, \quad P(3.5;0) = 0.9572, \quad P(4;0) = 0.9547,$$
$$P(4.5;0) = 0.9490, \quad P(5;0) = 0.9433, \quad P(5.5;0) = 0.9366, \quad P(6;0) = 0.9301,$$
$$P(6.5;0) = 0,9266, \quad P(7;0) = 0,9231.$$

Hints:

- Note that, whereas you can address point (a) just looking at the formulas in the script, things are more difficult for point (b): this is for example the case when you have semi-annual bonds but you are given only annual par swap rates.
- You can solve this point by using a *root finder* algorithm, as the one implemented in `net.finmath.rootfinder.BisectionSearch`, by which you can find the zero of a function f.
- In particular, here $f(x)$ can be the difference $S - S(x)$, where S is the given par swap rate and $S(x)$ is the value of the par swap rate if the value of the new bond is x: at every iteration of the algorithm, $f(x)$ can be computed by a method which returns the value of the difference above.
- However, there are here two bonds whose value you don't know, i.e., $P(T_{i-1};0)$ and $P(T_i;0)$. For this reason, in order to compute $f(x)$ at the point above, you have to get the value of $P(T_{i-1};0)$ if $P(T_i;0) = x$. In order to do this, you can write a method which performs an interpolation between the values of two bonds, and call this method inside the method computing $f(x)$. A nice interpolation is for example

$$P(T_{i-1};0) = e^{0.5(\ln P(T_{i-2};0)+\ln P(T_i;0))}.$$