QFGB8960 Advanced C++ for Finance
Homework 6

Spring 2025

Problem 1 (20 points) Yield Curve Check

In a Python notebook, create a yield curve with name "USD-LIBOR" using the function `qf.ycCreate` from qflib-0.6.0 and the maturities (in years) and spot rates (cont. cmpd) below.

Maturity	SpotRate
1/12	1.00%
1/4	2.00%
1/2	3.00%
3/4	3.50%
1	4.00%
2	4.75%
3	5.25%
5	6.00%

Create a list of maturities from 1 month to 3 years with a monthly step. This means each maturity in years is equal to the previous maturity plus 1/12. For each maturity in the list, calculate the forward rate from the previous maturity to this maturity in two ways:

a) using the `qf.fwdRate` function and the USD-LIBOR curve

b) manually, using the relation between the forward rate and the two spot rates, eq. (19) in the lecture notes

Compute the difference between the two calculations and show that the results are the same.

Plot the above 36 spot and forward rates against the correspoding maturities.

Problem 2 (40 points) Coding the Volatility Term Structure Class

Create a `VolatilityTermStructure` class to be declared in the file
`qflib/market/volatilitytermstructure.hpp`
and implemented in the file
`qflib/market/volatilitytermstructure.cpp`.

The class should support the following:

1) An enumeration `VolType` with values `SPOTVOL` and `FWDVOL`.

2) A templetized constructor of the form

```
template<typename XITER, typename YITER>
VolatilityTermStructure(XITER tMatBegin, XITER tMatEnd,
                        YITER volBegin, YITER volEnd,
                        VolType vtype = SPOTVOL);
```

3) Methods for reading out spot and forward annualized volatilities as

```
double spotVol(double tMat) const;
double fwdVol(double tMat1, double tMat2) const;
```

Internally the class should maintain the forward starting variances from maturity to maturity in a `PiecewisePolynomial` class of order 0 (piecewise constant), to be called `fwdvars_`. Use the `YieldCurve` class in qflib 0.6.0 as an example.

Problem 3 (40 points) Volatility Term Structure as a Market Object

Part A

1) Extend the `Market` class to allow it to contain objects of type `VolatilityTermStructure` and to return a reference to their map via the method `Market::volatilities()`.

2) Extend the `Market::clear()` method to also clear the volatility map.

3) Implement and register three Python callable functions

```
qf.volCreate(volname, tmats, vols, voltype)
```

creates a volatility term structure from spot or forward vols and gives it the name “volname”

```
qf.spotVol(volname, tmat)
```

returns the spot volatility to maturity using the named vol term structure object

```
qf.fwdVol(volname, tmat1, tmat2)
```

returns the forward volatility from tmat1 to tmat2 using the named vol term structure object

All volatilities should be reported in annualized terms.

4) Extend the `qf.mktList()` function to return a second key called "Volatilities" with the names of the volatility objects in the market.

The Python callable functions should be implemented in the file `pyqflib/pyfunctions2.hpp`.

Part B

On the Python notebook for problem 1 add the following:

1) Create a volatility term structure with the name "VOLTS", using the `qf.volCreate` function and the ranges below

Maturity	SpotVol
0.25	20%
0.50	22%
0.75	23%
1	25%
1.5	27%
2	30%
3	35%
4	40%

2) For each maturity above, calculate the forward volatility between this and the previous maturity in two ways:

a) using the `qf.fwdVol` function and the VOLTS term structure

b) manually, using the relation between forward and spot variances, eq. (20) in the lecture notes

3) Show that the two calculations match for each maturity.

Submit your source code and the Python notebook.