MATCHING THE BLOOMBERG CURVE S45 WITH QUANTLIB

PETER CASPERS AND ANDREA PALERMO

Abstract. In this short note we summarise one of the results achieved in a project conducted by SoftSolutions! with the goal of aligning some of the bond analytics result in their system nexRates with those that are produced in Bloomberg. By adding a global bootstrapper to the QuantLib library [1, 2] we are able to apply a certain smoothing algorithm to the Euribor 6M curve bootstrapping. Thereby we achieve a very close match of the corresponding Bloomberg S45 curve, which is the basis for the computation of the Z-Spread for bonds.

1. Overview and Motivation

SoftSolutions! is an Italian fintech company providing, among other products, an e-trading system named nexRates for fixed income instruments. nexRates covers a wide part of the whole process involved in the market making activities: pricing analytics, prices contribution to market platforms, trading actions (most notably RFQ management), risk aggregation, STP. Pricing analytics are based on C++ QuantLib libraries, see [1]. Being nexRates developed in Java, C++ QuantLib libraries are embedded using JNI (Java Native Interface), specifically leveraging the SWIG project [3] that simplifies the creation of wrapper interfaces.

A key financial analytic provided by nexRates is the yield-to-price conversion, that is the base for the bonds pricing structure. But an important role is also played by the Zero-volatility spread (a.k.a. Z-spread): Z-Spread is the flat spread that, added to the yield at each tenor of the curve chosen for Z-Spread calculation related to bonds cash flows, makes the price of the bond equal to the present value of all its cash flows.

This analytic has two main practical applications for a fixed-income desk: as a risk comparison metric among bonds of same currency and as a driver for the pricing generation. In the first case, along with yield and Asset Swap Spread, the Z-Spread allows to evaluate how two bonds are relatively perceived as risky by the market. Instead, when used as a driver, the price of the bond is directly linked to the live movements of the Z-Spread curve related to the currency of the bond. Usually Z-Spread is chosen as the pricing driver for short-maturing fixed-coupon bonds; moreover for some currencies, like Norwegian Crown (NOK), the market practice is to use Z-Spread for the entire maturity spectrum of fixed-coupon bonds.

The algorithm for calculating the Z-Spread of a bond given its price (or yield) is based on a number of conventions that makes it a non unique figure among diverse calculation providers. Said that, given its predominance in the market among both the sell-side and buy-side traders, Bloomberg is the reference for analytic calculations: when the user of a trading platform communicates the Z-Spread of

Date: 28 June 2020.

a bond to a counterpart and the counterpart uses Bloomberg Terminal as his/her main financial tool, the relation between Z-Spread and price must be coherent, with possible misalignments not leading to difference in the settlement values derived by the price figure. This condition can be translated into a maximum acceptable difference in the Z-Spread to price conversion of 0.0009.

One core component to reach the desired accuracy w.r.t. Bloomberg reference results is to match the daily zero rates of the rate curves constructed in Bloomberg underlying the Z-Spread calculation. While some of the curves are relatively straightforward to reproduce in QuantLib (like NOK 313, the reference curve for computing Z-Spread for NOK bonds), others requires smoothing techniques that were not supported in QuantLib so far. One such curve is the EUR S45 (Euribor 6M) curve, which serves as the benchmark for the Z-Spread computation of EUR bonds.

This paper is organised as follows. In section 2 we describe the methodology we employed to match the Bloomberg EUR S45 curve. Section 3 describes the extensions that were added to QuantLib to implement the methodology. The usage of the extensions in Python are described in 4. We give numerical results based on real market data in 5.

2. Matching Bloomberg Calculations

In this section we give a detailed description of the methodology we devised to match the Bloomberg S45 (Euribor 6M) curve. We tested the methodology for the S45 curve, but we believe it will work for any other curve that is built from a similar instrument set. A special feature of the S45 curve is posed by the fact that it contains serial FRA instruments. To achieve a good matching with Bloomberg results, we decided to apply a smoothing algorithm, which we describe in detail in section 2.2.8.[1] and which requires a global bootstrapping approach rather than an iterative, by-instrument approach.

2.1. Settings in the SWDF Screen.

There are some settings in Bloomberg which control the curve building. These settings can be found in the so called "SWDF" screen. In what follows, we assume the following settings in this screen, which we found to be the default settings:

- Curve Defaults: Pay=Mid / Receive =Mid
- Interpolation Method: Piecewise linear (Simple-comp)
- Enable OIS Discount/Dual Curve Stripping: *not* enabled

The settings "Cross Currency Basis Defaults" and "Brazilian Curve Interpolation Method" are not relevant for the curves discussed here.

2.2. Bootstrap Methodology.

2.2.1. *Date to time conversion.* To be able to interpolate the zero rates of a rate curve for all calendar dates (or even in continuous time), one needs to define a function that converts calendar dates to non-negative real numbers. For this, let d_0 denote the current date. For a date $d \geq d_0$ the time $\tau(d_0, d)$ between d_0 and d is computed using a day count convention with the property that it assigns an equal weight to each calendar day, i.e., it is of the form

[1] notice that the smoothing is not applied for IMM FRA instruments, where it is not needed to match the Bloomberg results

$$\tau(d_0, d) = \frac{d - d_0}{N}$$

where $d - d_0$ should be read as the number of calendar days between d_0 and d and N is a fixed number. Typical examples for such day counters are ACT/365F or ACT/360 for which $N = 365$ or $N = 360$, respectively. In our concrete implementations we use ACT/365F as the curve day-count convention. However notice that the actual choice of N does not impact the results of the interpolation.

2.2.2. *Settlement dates.* The discount factors and zero rates displayed in Bloomberg associated to each quoted market instrument against which we compare our results are—we believe—*forward* discount factors and forward zero rates using the settlement delay of the respective instrument. We denote the settlement date for each instrument by d_s. Although d_s in general depends on the instrument, for the S45 curve we have $d_s = d_0 + 2\text{bd}$ w.r.t. the TARGET calendar for all instruments. Notice that by construction, the discount factor on the settlement date is 1.

2.2.3. *Simply compounded zero rate.* The simply compounded zero rate $z(d_0, d_2)$ for a period d_0 to d_2 is defined by the relation

$$D(d_0, d_2) = \frac{1}{1 + z(d) \cdot \tau(d_0, d_2)}$$

where $D(d_0, d_2)$ is the discount factor for the period d_0 to d_2. A similar definition can be given for the settlement date by replacing d_0 by d_s.

2.2.4. *Curve instruments, pillars and Interpolation.* The curve is built from a set of liquid market instruments. Each instrument defines a "pillar date" as outlined in sections 2.2.5 to 2.2.7. The set of pillar dates $d_1, \ldots, d_n$ is translated to curve times $t_1, \ldots, t_n$ as described in 2.2.1. Adding $t_0 = 0$ to this set we get an interpolation grid

$$(t_0, y_0), (t_1, y_1), \ldots, (t_n, y_n)$$

To match the Bloomberg interpolation method ("Piecewise linear (Simple-comp)"), the y_i are simply compounded zero rates (see 2.2.3) and we compute an arbitrary point (t, y) as follows:

- if $t < t_1$ we apply a "constant forward extrapolation":
$$y = (e^{-ht} - 1)/t$$
with
$$h = \frac{\ln(1 + y_1 \cdot t_1)}{t}$$
i.e., the *instantaneous* forward rate h is extrapolated flat from t_1 to the left
- if $t_1 < t < t_n$ we apply a linear interpolation: y is interpolated linearly between the adjacent interpolation grid points (t_i, y_i)
- if $t > t_n$ ("constant non-annualized zero extrapolation"):
$$y = y_n t_n / t$$
i.e., the product yt is kept constant right from t_n

This set of rules fully defines the interpolation of the curve in continuous time in terms of simply compounded zero rates.

2.2.5. *Cash Point (Euribor Fixing).* The first point in the S45 curve is given by the current Euribor 6M fixing, i.e., we have a cash deposit with

- fixing date = d_0
- start date = d_0 plus start delay (2 bd for Euribor)
- maturity date = start date plus tenor (6 months for Euribor 6M, modfol adjusted).

The pillar date for curve interpolation is equal to the maturity date.

2.2.6. *Serial FRA Points.* After the cash point, serial FRAs 1M-7M, 2M-8M, ..., 12M-18M with an Euribor 6M underlying are used to bootstrap the curve. The characteristic dates of a FRA are given by:

- spot date = d_0 plus settlement delay (2 bd for Euribor 6M FRAs)
- start date = spot date plus the period to start (i.e. 1M, 2M, 3M etc., modfol adjusted for Euribor 6M)
- maturity date = spot date plus the period to the FRA end (i.e. 7M, 8M, 9M, etc., modfol adjusted for Euribor 6M)

The pillar date for curve interpolation is equal to the maturity date. We notice that the forward rate for the FRA is estimated between the start date and the maturity date, which is often identical to the actual index estimation end date computed as the start date plus index period (adjusted), but not always. For example for d_0 = 16-Sep-2019, the Euribor 6M FRA 7x13 has

- spot date = 18-Sep-2019
- start date = 20-Apr-2020 (spot date plus 7M, adjusted)
- maturity date = 19-Oct-2020 (spot date plus 13M, adjusted)
- index estimation end date = 20-Oct-2020 (start date plus 6M, adjusted)

In this case the forward rate is estimated between the start and maturity date, which is a slight approximation to the "correct" estimation between the start and index estimation end date.

2.2.7. *Swap Points.* After the FRA section fixed vs. Euribor 6M interest rate swaps follow. The characteristic dates for a swap are given by:

- spot date = d_0 plus settlement delay (2 bd for Euribor 6M Swaps)
- maturity date = last payment date of the swap

Similarly to serial FRAs the maturity date is sometimes not identical to the index estimation end date of the last floating rate period of the swap. Nevertheless the pillar date is again chosen as the maturity date and the forward rate (intermediate + final periods) are estimated using the accrual periods of the swap, which again can introduce a slight approximation due to a mismatch of the true index estimation period end dates and accrual end dates for some of the periods.

2.2.8. *Smoothing for curves with serial FRAs.* For a curve which contains serial FRAs we found that the application of a smoothing algorithm is necessary in order to match the Bloomberg zero rates. Specifically for the S45 curve, we consider additional 5 serial FRA instruments

- FRA13x19, FRA14x20, FRA15x21, FRA16x22, FRA17x23

which are not quoted in the market. We require the implied quotes of these FRAs during the curve bootstrap to be linearly interpolated between the (quoted)

FRA12x18 and (not quoted) FRA18x24 implied FRA quotes. We assume that for the interpolation in the x-y-plane the x-grid is simply given by

$$0, 1, 2, 3, 4, 5, 6$$

i.e., we do not use exact dates of any kind for the interpolation of the implied FRA rates, but simply assume that the FRAs used for the interpolation have equal distance. To meet the addition 5 constraints we introduce 5 additional degrees of freedom to the interpolation grid of the curve by adding the start dates of the

- FRA 1M-7M, 2M-8M, 3M-9M, 4M-10M, 5M-11M

instruments as additional curve pillars. The bootstrap is implemented as a global optimisation in this case.

3. Implementation in QuantLib

In this section we describe the concrete implementation of the methodology described in 2 in QuantLib. To a great extend the existing code could be used without any modifications or extensions. We do not describe these parts here in detail, see [5], chapter 3, for a discussion.

3.1. **Rate Helpers.** The rate helpers we used are the standard helpers, for the S45 curve these are:

- `DepositRateHelper` to represent the current Euribor 6M fixing
- `FraRateHelper` to cover the FRA quotes section
- `SwapRateHelper` to cover the Swap quotes section

3.2. **The Use Indexed Coupon setting.** We observe that one should leave the `QL_USE_INDEXED_COUPON` setting *deactivated* to achieve an exact match of the pillar dates.

3.3. **PiecewiseYieldCurve, Global Bootstrap.** To build the curve, the (standard) `PiecewiseYieldCurve` can be used. The interpolation variable trait should be set to `SimpleZeroYield` which was previously not available to QuantLib. The interpolation method should be `Linear`. To apply the smoothing algorithm described above, a new Bootstrap class `GlobalBootstrap` was added to the QuantLib library. The complete constructor call is given in listing 1.

```
typedef PiecewiseYieldCurve<SimpleZeroYield, Linear, GlobalBootstrap> Curve;

ext::shared_ptr<Curve> curve = ext::make_shared<Curve>(
      2, TARGET(), helpers, Actual365Fixed(), std::vector<Handle<Quote> >(), std::vector<Date>(),
      Linear(),
      Curve::bootstrap_type(additionalHelpers, additionalDates(),
                            additionalErrors(additionalHelpers), 1.0e-12));
```

Listing 1: PiecewiseYieldCurve constructor call

Here, we construct the curve with 2 settlement days w.r.t. the $TARGET$ calendar. We pass in the rate helpers as a vector `helpers` as usual. We set the day counter (rather arbitrarily, see above) to Act/365 (fixed). The last parameter is the new global boostrap instance taking 4 parameters:

- `additionalHelpers`: This is a vector of rate helpers that will be available in the definition of the additional error terms `additionalErrors` (see below). In the case of the S45 curve these are FRA helpers for 12–18, 13–19, . . . , 18–24 FRAs.
- `additionalDates`: This is a functor returning a vector of dates that are used as additional curve pillars to compensate for the additional error terms. In the case of the S45 curve these pillars are given as today plus 1M, 2M, . . . , 5M. The code defining this functor is given in listing 2 for reference.
- `additionalErrors`: This is a function returning a an array of additional error terms used during the bootstrap. In the case of the S45 curve this is the deviation of the 13–19, 14–20, . . . , 17–23 fair FRA quotes (defined as `additionalHelpers` above) from linearly interpolated quotes between the 12–18 and 18–24 fair FRA quotes. The code defining this functor is given in listing 3 for reference.

```
struct additionalErrors {
    explicit additionalErrors(
        const std::vector<ext::shared_ptr<BootstrapHelper<YieldTermStructure> > >&
            additionalHelpers)
    : additionalHelpers(additionalHelpers) {}
    std::vector<ext::shared_ptr<BootstrapHelper<YieldTermStructure> > > additionalHelpers;
    Array operator()() {
        Array errors(5);
        Real a = additionalHelpers[0]->impliedQuote();
        Real b = additionalHelpers[6]->impliedQuote();
        for (Size k = 0; k < 5; ++k) {
            errors[k] = (5.0 - k) / 6.0 * a + (1.0 + k) / 6.0 * b -
                        additionalHelpers[1 + k]->impliedQuote();
        }
        return errors;
    }
};
```

Listing 2: Functor definition for additional dates

```
struct additionalDates {
    std::vector<Date> operator()() {
        Date settl = TARGET().advance(Settings::instance().evaluationDate(), 2 * Days);
        std::vector<Date> dates;
        for (Size i = 0; i < 5; ++i)
            dates.push_back(TARGET().advance(settl, (1 + i) * Months));
        return dates;
    }
};
```

Listing 3: Functor definition for additional errors

4. Python interfaces

There is a binder example available at [4] demonstrating the global bootstrap.

5. Numerical Results

In table 1 we provide a comparison between BBG zero rate and QuantLib zero rates for the S45 (Euribor 6M) curve using market data as of 26-Sep-2019. We display differences between QuantLib and Bloomberg zero rates with and without the application of the smoothing. All numbers are in percent. Without the smoothing the error in the zero rate reaches almost 1 basis point in the FRA segment, while with the treatment the error stays below 0.01 basis points.

The table is divided into the 3 segments made up from

- the cash point,
- the FRA instruments (1M-7M, ..., 12M-18M), and
- the swap instruments (2Y, 3Y, ..., 10Y, 11Y, 12Y, 15Y, 20Y, 25Y, 30Y, 35Y, 40Y, 45Y, 50Y)

Notice that the zero rates are expressed using 2 TARGET settlement days and the following conventions:

- for the cash point (1st segment) and the FRA points (2nd segment): Act/360 simply compounded annual rate
- the the swap points (3rd segment): 30/360 simple then compounded annual rate

Figure 1 shows a graph the zero curve with and without smoothing. In the graph the additional calibration points for the serial FRA treatment and the regular calibration points are marked by blue and yellow arrows, respectively. Furthermore the implied FRA rates for the 12M-18M, ..., 18M-24M FRA instruments are plotted as stars (with smoothing) and boxes (without smoothing).

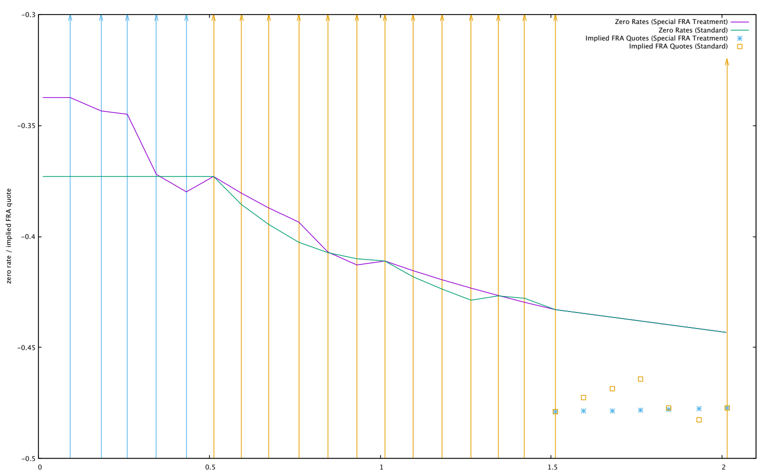

Figure 1. Zero rates up to 2Y with and without smoothing, the blue arrows mark additional calibration points, the yellow arrows mark the regular calibration points

References

[1] QuantLib: a free/open-source library for quantitative finance, `http://www.quantlib.org`, `http://doi.org/10.5281/zenodo.3724021` (version 1.18).

Curve Pillar Date	Diff QL/BBG With Smoothing	Diff QL/BBG Without Smoothing
2020-03-31	0.00000	-0.00000
2020-04-30	-0.00000	0.00513
2020-05-29	0.00000	0.00744
2020-06-30	0.00000	0.00910
2020-07-31	0.00000	0.00028
2020-08-31	0.00010	-0.00268
2020-09-30	0.00000	0.00000
2020-10-30	-0.00000	0.00275
2020-11-30	-0.00000	0.00420
2020-12-31	-0.00000	0.00542
2021-01-29	0.00001	0.00018
2021-02-26	0.00006	-0.00175
2021-03-31	-0.00000	-0.00000
2021-09-30	-0.00000	-0.00000
2022-09-30	0.00002	0.00002
2023-09-29	0.00000	0.00000
2024-09-30	-0.00000	-0.00000
2025-09-30	0.00006	0.00006
2026-09-30	-0.00000	-0.00000
2027-09-30	0.00001	0.00001
2028-09-29	0.00006	0.00006
2029-09-28	0.00000	0.00000
2030-09-30	0.00002	0.00002
2031-09-30	0.00003	0.00003
2034-09-29	0.00000	0.00000
2039-09-30	0.00000	0.00000
2044-09-30	0.00000	0.00000
2049-09-30	0.00000	0.00000
2054-09-30	-0.00000	-0.00000
2059-09-30	0.00000	0.00000
2064-09-30	-0.00000	-0.00000
2069-09-30	-0.00000	-0.00000

TABLE 1. Comparison between BBG and QuantLib Zero Rates as of 26-Sep-2019 for the Euribor 6M curve S45 with and without smoothing, all numbers are in percent

[2] QuantLib-SWIG: QuantLib wrappers to other languages, `http://doi.org/10.5281/zenodo.3724022` (version 1.18).
[3] SWIG project `http://www.swig.org/`
[4] Binder example global bootstrap in QuantLib, `https://mybinder.org/v2/gh/lballabio/QuantLib-SWIG/binder?filepath=Python%2Fexamples` (global-bootstrap.py)
[5] Luigi Ballabio: Implementing QuantLib, available at `https://leanpub.com/implementingquantlib`

(Peter Caspers) QUATERNION RISK MANAGEMENT
E-mail address: `peter.caspers@quaternion.com`

(Andrea Palermo) SOFTSOLUTIONS!
E-mail address: `andrea.palermo@softsolutions.it`