FlexYCF: Flexible Yield Curve Framework

Cash and Inflation Derivatives Quantitative Research

September 20, 2012

Revision history		
Author	**Date**	**Comment**
Manlio Trovato	3-Aug-2009	Initial Version
Nicolas Maury	18-Aug-2009	Rectified cross currency basis swap section
Nicolas Maury	05-Oct-2009	Monotonicity enforcement for convex splines
Nicolas Maury	15-Oct-2009	Added “Design and Implementation” and “Equivalent Curve” parts
Manlio Trovato	17-Nov-2009	Changed document structure
Nicolas Maury	23-Nov-2009	Added zero-coupon inflation swap
Nicolas Maury	03-Dec-2009	Added Future Convexity Model and Jacobian part.
Nicolas Maury	08-Dec-2009	Removed implementation details from “Curve Building Algorithm” and “Models” sections
Nicolas Maury	08-Dec-2009	Removed implementation details from “Curves and Interpolation” and “EquivalentCurve” sections
Nicolas Maury	15-Dec-2009	General Review
Nicolas Maury	17-Dec-2009	Updated initial guess in Initialization from instruments for currency and tenor basis swaps
Nicolas Maury	07-Jan-2010	Corrected typos
Nicolas Maury	17-Mar-2010	Updated knot placement part.
Manlio Trovato	23-Aug-2010	Added Fx Forwad instrumennts
Manlio Trovato	24-June-2011	Added Libor OIS Basis instrumennts and funding and index spread models
Manlio Trovato	20-July-2011	More on spread curve models
Jarek Solowiej	11-Aug-2011	Stripper’s CashSelectionRule and Fx Quote conventions
Jarek Solowiej	31-Aug-2011	MultiTenorOIS and MultiTenorOISFunding models
Jarek Solowiej	31-Aug-2011	Generic ccy basis swap and Index Stripper models
Chongxian Zhu	13-Sept-2011	Updated zero rate instrument
Jarek Solowiej	21-Sept-2011	Generic ccy basis swap and OIS instruments
Jarek Solowiej	23-April-2012	MultiTenorFormulation model
Jarek Solowiej	12-Sept-2012	”RunDown” and ”AllTenor”

Contents

Chapter 1

Abbreviations and Notation

df_F	The Funding discount factor	(16)
df_I	Index discount factor	(16)

Chapter 2

Instruments

2.1 Yield curve instruments

A *yield curve instrument* is a contract satisfying the following:

1. it has a present value
2. its present value is a continuously differentiable function of the discount factors:

$$PV = f(df_0^I, \ldots, df_{n_i-1}^I, df_0^F, \ldots, df_{n_F-1}^F), f \in \mathcal{C}^1$$

where n_i is the number of index discount factors, n_F is the number of funding discount factors, $n = n_i + n_F$.

2.1.1 Cash

A cash instrument is an index instrument. Recall the definition of libor rate between T_1 and T_2:

$$F_2 := \frac{1}{\alpha_{1,2}^R}\left(\frac{df_1^I}{df_2^I} - 1\right)$$

2.1.2 Future

A Future instrument is an index instrument as it has no sensitivity to the funding curve. A future can be defined by providing a start date, a term and a price.

The present value of a Future is defined as follows:

$$PV = \left[\left(\frac{df_1^I}{df_2^I} - 1\right)\frac{1}{\alpha_{1,2}^R} + C - R\right] \; = \; f(df_1^I, df_2^I),$$

where C is the convexity adjustment and R is the implied future LIBOR rate[1].

[1] see section 8 for exact definitions

2.1.3 Swap

A swap is an index instrument, as its primary sensitivities is with respect to the index curve. However, a swap has also some funding sensitivities due to the cash flow discounting. The dependency of the swap *PV* over both types of instruments is made clear by explicitly deriving its PV. Assume the fix side pays *K*; the *PV* of the fix side is

$$PV_{fix} = K\alpha^A_{0,1}df^F_1 + K\alpha^A_{1,2}df^F_2 + K\alpha^A_{2,3}df^F_3 + \cdots + K\alpha^A_{p-1,p}df^F_p = \tag{2.1}$$

$$= K\sum_{j=1}^{p}\alpha^A_{j-1,j}df^F_j$$

where *p* is the number of fixed payments.
The float side *PV* is

$$PV_{float} = F_1\alpha^A_{0,1}df^F_1 + F_2\alpha^A_{1,2}df^F_2 + \cdots + F_q\alpha^A_{q-1,q}df^F_q \tag{2.2}$$

$$= \frac{1}{\alpha^R_{0,1}}\left(\frac{df^I_0}{df^I_1} - 1\right)\alpha^A_{0,1}df^F_1 + \cdots + \frac{1}{\alpha^R_{q-1,q}}\left(\frac{df^I_{q-1}}{df^I_q} - 1\right)\alpha^A_{q-1,q}df^F_q$$

where F_j is the forward rate observed at time *j* and *q* is the number of floating payments.
The PV of a swap is computed by discounting all individual cash flows, for both the fixed and the floating side, and by netting the two side's PV. The swap has *PV*

$$PV = PV_{fix} - PV_{float} = f(df^I_0, df^I_1, \ldots, df^I_q, df^F_1, \ldots, df^F_p)$$

LIBOR fixings: it is possible to specify LIBOR fixings to calculate the PV of a swap. In this case, F_1, the first LIBOR rate of the float side, is the value of the LIBOR fixing provided rather than being calculated in terms of index discount factors.

Forward starting swap could be specified by providing a start date in the future.

2.1.4 Currency Basis Swap

A currency basis swap can be used both as index and funding instrument, since it is sensitive to both curves. Contrary to the other swaps, the interest rate payments on the two legs are in different currencies.

Forward starting basis swaps could be specified by providing a start date in the future.

A basis swap is composed by two legs:

- A floating leg in the same currency as the other instruments, with a given spread over the rate and initial and final exchange of notional; call $PV^B_{floatSpread}$ the PV of this leg.
- A floating leg in a "foreign currency", usually USD, with initial and final exchange of notional.

The *PV* of the latter leg is zero assuming, in approximation, there are no reset and accrual dates mistmaches and that the funding and index discount factors of the "foreign currency" currency are the same. Therefore the *PV* of a basis swap is computed by discounting the individual currency floating cash flows, with the given spread added to the libor rates, and by adding the present value of the initial and final notional exchange flows. Call *x* the spread of the basis over the libor rate, F_j the libor rate observed at time *j*, then the *PV* is

$$\begin{aligned} PV &= PV^B_{floatSpread} \\ &= -df^F_0 + df^F_n + (F_1 + x)\,\alpha^A_{0,1} df^F_1 + (F_2 + x)\,\alpha^A_{1,2} df^F_2 + \cdots + (F_n + x)\,\alpha^A_{n-1,n} df^F_n \\ &= -df^F_0 + df^F_n + \left[\frac{1}{\alpha^R_{0,1}}\left(\frac{df^I_0}{df^I_1} - 1\right) + x\right]\alpha^A_{0,1} df^F_1 + \\ &\quad + \left[\frac{1}{\alpha^R_{1,2}}\left(\frac{df^I_1}{df^I_2} - 1\right) + x\right]\alpha^A_{1,2} df^F_2 + \cdots + \\ &\quad + \left[\frac{1}{\alpha^R_{n-1,n}}\left(\frac{df^I_{n-1}}{df^I_n} - 1\right) + x\right]\alpha^A_{n-1,n} df^F_n \\ &= f(df^I_0, \ldots, df^I_n, df^F_0, \ldots, df^F_n). \end{aligned} \tag{2.3}$$

PV depends both on index and funding discount factor since $F_j = F_j(df^I_{j-1}, df^I_j)$; in (2.3) the first term of the sum $-df^F_0$ is the present value of the initial notional payment, the second term df^F_n is the present value of the final notional exchange. Notice that in $PV^B_{floatSpread}$ we do not make the approximation $\alpha^R = \alpha^A$.

In order to evaluate the approximation done in the valuation of the "foreign currency", consider that the flow schedule internally generated by the swap instrument can take into account, or disregard, the reset and accrual date schedule mismatches, due to any eventual different market convention. If the foreign leg is approximated to an exchange of notionals, which implies that $\alpha^R = \alpha^A$, (2.2) simplifies in

$$PV^{floatForeign} = \left(\frac{df^I_0}{df^I_1} - 1\right) df^F_1 + \cdots + \left(\frac{df^I_{q-1}}{df^I_q} - 1\right) df^F_q$$

An additional approximation can be done assuming that $df^I = df^F$; in this case, the previous formula simplifies in:

$$\begin{aligned} \widetilde{PV}^{SwapMarket}_{float} &= \left(\frac{df^F_0}{df^F_1} - 1\right) df^F_1 + \cdots + \left(\frac{df^F_{q-1}}{df^F_q} - 1\right) df^F_q = \\ &= df^F_0 - df^F_q \end{aligned} \tag{2.4}$$

This says that the foreign float side of a currency swap is equivalent to an exchange of notional at time T_0 and a notional payment at time T_q and this exchange of notional is exactly the opposite of the one that occurs in the foreign currency in the swap. This simplification applies for any frequency of the forward rates between T_0 and T_q.

Leg2 Type Controls

The form of the second leg (foreign leg) of currency swap can be controlled using *Leg2Type*

1. if *Leg2Type=None* (default value), then foreign leg is not present
2. if *Leg2Type=LiborFlat*, then foreign leg and exchange of notionals are present
3. if *Leg2Type=Resettable* then foreign leg is present; in addition, its notionals are reseted according to fx spot rate at foreign leg payment dates (creating additional cashflows), i.e.,

$$PV_{\text{foreign leg}} = (F_1\alpha^A_{0,1}X_0+(X_0-X_1))df^F_1+(F_2\alpha^A_{1,2}X_1+(X_1-X_2))df^F_2+\cdots$$

$$+(F_{q-1}\alpha^A_{q-2,q-1}X_{q-2}+(X_{q-2}-X_{q-1}))df^F_{q-1}+(F_q\alpha^A_{q-1,q}X_{q-1})df^F_q+X_{q-1}df^F_q-X_0df^F_0$$

where X_i means fx spot rate at time t_i.

Pricing Controls

One can control domestic and foreign leg index computations (source of df^I for domestic and foreign leg) using *Leg1DepositRateType/Leg2DepositRateType*;

1. if *Leg1DepositRateType=IBOR*, then an index is computed from appropriate tenor curve (default value);
2. if *Leg1DepositRateType=ON*, then an index is computed from 1D-tenor curve;
3. if *Leg1DepositRateType=FUNDING*, then an index is computed from a discount curve of a dependent model.

2.1.5 Special Types of Currency Basis Swaps

Currency OIS Swap

In cross-currency swap by specifying

Leg1DepositRateType=ON and *Leg2DepositRateType=ON*,

one can create currency OIS swap with the following domestic leg

$$PV_{\text{domestic leg}} = (F^{OIS}_1+s)\alpha^A_{0,1}df^F_1+(F^{OIS}_2+s)\alpha^A_{1,2}df^F_2+\cdots+(F^{OIS}_p+s)\alpha^A_{p-1,p}df^F_p+df^F_p-df^F_0,$$

where F^{OIS}_i is computed from 1D-curve; with similar formula for foreign leg.

Currency Funding Swap

Available only in Funding and Funding Spread stripper models, and equivalent curve functor. In cross-currency swap by specifying

Leg1DepositRateType=FUNDING and *Leg2Type=NONE*,

one can create a currency funding swap with the following domestic leg

$$PV_{\text{domestic leg}} = (F_1+s)\alpha^A_{0,1}df^F_1+(F_2+s)\alpha^A_{1,2}df^F_2+\cdots+(F_p+s)\alpha^A_{p-1,p}df^F_p+df^F_p-df^F_0,$$

where $F_i = 1/\alpha^A_{i-1,i}(df^F_{i-1}/df^F_i - 1)$ and df^F comes from discount curve of dependent model.

2.1.6 OIS Swap

An OIS swap has a fixed leg (usually with annual payments) and floating leg paying compounded overnight rates (with payment dates matching fixed leg payment dates).

$$PV_{\text{fixed leg}} = K\alpha^A_{0,1}df^F_1 + K\alpha^A_{1,2}df^F_2 + K\alpha^A_{2,3}df^F_3 + \cdots + K\alpha^A_{p-1,p}df^F_p$$
$$PV_{\text{floating leg}} = F^{OIS}_1\alpha^A_{0,1}df^F_1 + F^{OIS}_2\alpha^A_{1,2}df^F_2 + \cdots + F^{OIS}_p\alpha^A_{p-1,p}df^F_p,$$

where $F^{OIS}_i = 1/\alpha^A_{i-1,i}(df^{1D}_{i-1}/df^{1D}_i-1)$ and df^{1D} comes from 1D-curve. If $df^{1D} = df^F$, then

$$PV_{\text{floating leg}} = df^F_0 - df^F_p. \tag{2.5}$$

2.1.7 Libor OIS Basis Swap

A Libor OIS basis swap is expressed as a combination of swap and OIS swap, i.e., it has the following four legs

$$\text{swap floating leg} - \text{swap fixed leg} + \text{OIS swap fixed leg} - \text{OIS swap floating leg},$$

where

$$PV^{swap}_{\text{floating leg}} = F_1\alpha^A_{0,1}df^F_1 + F_2\alpha^A_{1,2}df^F_2 + \cdots + F_p\alpha^A_{p-1,p}df^F_p$$
$$PV^{swap}_{\text{fixed leg}} = K_1\alpha^A_{0,1}df^F_1 + K_1\alpha^A_{1,2}df^F_2 + \cdots + K_1\alpha^A_{p-1,p}df^F_p$$
$$PV^{OIS}_{\text{fixed leg}} = K_2\alpha^A_{0,1}df^F_1(t^p_1) + K_2\alpha^A_{1,2}df^F_2(t^p_2) + \cdots + K_2\alpha^A_{p-1,p}df^F_p(t^p_p)$$
$$PV^{OIS}_{\text{floating leg}} = F^{OIS}_1\alpha^A_{0,1}df^F_1(t^p_1) + F^{OIS}_2\alpha^A_{1,2}df^F_2(t^p_2) + \cdots + F^{OIS}_p\alpha^A_{p-1,p}df^F_p(t^p_p),$$

where $F^{OIS}_i = 1/\alpha^A_{i-1,i}(df^{1D}_{i-1}/df^{1D}_i - 1)$. If swap's and OIS swap's payment dates agree, i.e., ignoring payment lags, which can be up to 2 business days, and $df^{1D} = df^F$, then the value of an OIS floating leg reduces to initial and final notional exchanges:

$$PV^{OIS}_{\text{floating leg}} = df^F_0 - df^F_p. \tag{2.6}$$

The Libor leg is computed similarly to eq. 2.3, ie:

$$PV^B_{LiborOISSpread} = (F_1 + x)\,\alpha^A_{0,1}df^F_1 + (F_2 + x)\,\alpha^A_{1,2}df^F_2 + \cdots + (F_p + x)\,\alpha^A_{p-1,p}df^F_p \tag{2.7}$$

Subtracting the two legs it follows that the value of a Libor OIS basis swap is given by eq 2.3, hence Libor OIS basis swaps can be modelled in the same way as currency swaps.

2.1.8 Tenor Basis Swap

A tenor basis swap is similar to a currency basis swap. Recall that (2.2) simplifies in (2.3) with the assumptions $\alpha^r = \alpha^A$ and $df^I = df^F$. We used these assumptions in section 2.1.7 to represent PV^B_{float}. A tenor basis swap has two floating legs

- a "long tenor" floating leg, where the floating rate is the observed forward rate; call PV^{TB}_{float} its PV. We have

$$PV^{TB}_{float} = F_1\alpha^A_{0,1}df^F_1 + F_2\alpha^A_{1,2}df^F_2 + \cdots + F_p\alpha^A_{p-1,p}df^F_p$$

- a "short tenor" floating leg, which PV we call $PV^{TB}_{floatSpread}$; this PV is obtained modifing (2.2) and adding a spread as follows:

$$PV^{TB}_{floatSpread} := (\hat{F}_1 + x)\alpha^A_{0,1}df^F_1 + (\hat{F}_2 + x)\alpha^A_{1,2}df^F_2 + \cdots + (\hat{F}_q + x)\alpha^A_{q-1,q}df^F_q$$

  where the forward rate $\hat{F}$ is computed from the index discount factor adjusted with the relevant tenor basis spread:

$$\hat{F}_h := \frac{1}{\alpha^R_{h-1,h}}\left(\frac{df^I_{h-1}}{df^I_h} - 1\right)$$

Note that as names suggest the tenor in the long tenor floating leg is longer than the one in the short tenor floating leg. The spread x is always over the forward rates of the smaller tenor floating leg.

The Tenor Basis instrument has PV

$$PV = PV^{TB}_{floatSpread} - PV^{TB}_{float} \tag{2.8}$$

$$\begin{aligned}
&= (\hat{F}_1 + x)\,\alpha^A_{0,1}\,df^F_1 + (\hat{F}_2 + x)\,\alpha^A_{1,2}\,df^F_2 + \cdots + (\hat{F}_q + x)\,\alpha^A_{q-1,q}\,df^F_q \\
&\quad - \left\{F_1\,\alpha^A_{0,1}\,df^F_1 + F_2\,\alpha^A_{1,2}\,df^F_2 + \cdots + F_p\,\alpha^A_{p-1,p}\,df^F_p\right\} \\
&= \sum_{h=1}^{q}\left[\frac{1}{\alpha^R_{h-1,h}}\left(\frac{df^I_{h-1}}{df^I_h} - 1\right) + x\right]\alpha^A_{h-1,h}\,df^F_h \\
&\quad - \sum_{l=1}^{p}\frac{1}{\alpha^R_{l-1,l}}\left(\frac{df^I_{l-1}}{df^I_l} - 1\right)\alpha^A_{l-1,l}\,df^F_l
\end{aligned}$$

2.1.9 Zeros

A zero instrument represents a zero coupon compounded interest rate instrument. In this case, the discount factor is given by:

$$df^F(t) = \exp(-z(t)t) \tag{2.9}$$

where z is the zero rate associated with maturity t, and t is the year fraction between the valuation date and the "end date" on an actual365 basis.

The "end date" of the zero rate is computed by the optional zero rate instrument parameters, namely Spot Days(default: "0C"), Spot Calendar, Spot Roll Convention, Spot Roll Rule Convention, Calendar, Roll Convention and Roll Rule Convention. First, an internal spot date is computed by rolling the valuation date with Spot Days, based on Spot Calendar, Spot Roll Convention and Spot Roll Rule Convention. Finally, the end date is computed by rolling the spot date with the zero rate tenor, based on Calendar, Roll Convention and Roll Rule Convention.

2.1.10 Turn

A turn is a structure curve instrument. Turn instruments define the turn correction to be applied at a given turn date. A turn instrument can be specified by providing a turn date and a spread.

$$\theta(T) = \begin{cases} \exp(-r \cdot \alpha_{turn}) & \text{if } T > T_{turn} \\ 1 & \text{if } T \leq T_{turn} \end{cases}$$

where α_{turn} is the year fraction of 91 days in Act/365 convention.

Given a turn instrument, a constant drop is applied to the turn curve at the given date. To gain an intuition of the effects of the turn instrument, we note that for a flat continuous forward interpolation method, the turn instrument can be interpreted as a sudden change in the continuously compounded forward rate, which reverts back after one calendar day.

2.1.11 Step

A step is a structure curve instrument. The step instrument is applied at a given step date and applies to all dates after the step date. The step instrument is defined as follows:

$$\theta(T) = \begin{cases} \exp(-r \cdot (T - T_{step})) & \text{if } T > T_{step} \\ 1 & \text{if } T \leq T_{step} \end{cases}$$

We note that the step change in the turn correction factor $\theta(T)$ is applied at the given turn date and applies linearly in time thereafter. To gain an intuition of the effects of the turn instrument, we note that for a flat continuous forward interpolation method, the step instrument can be interpreted as a sudden step change in the continuously compounded forward rate, which is constantly applied for all $T > T_{step}$.

2.1.12 Bump

A bump is a structure curve instrument. The bump instrument is applied at a given bump start date and applies to all dates until a bump end date. The bump instrument is defined as follows:

$$\theta(T) = \begin{cases} \exp(r(T - T_{bump\_start})) & \text{if } T_{bump\_start} < T < T_{bump\_end} \\ \exp(-r(T_{bump\_end} - T_{bump\_start})) & \text{if } T_{bump\_end} \leq T \\ 1 & \text{if } T \leq T_{bump\_start} \end{cases}$$

We note that the bump change in the turn correction factor $\theta(t)$ is applied at the given bump start date and applies linearly in time until bump end date. To gain an intuition of

the effects of the turn instrument, we note that for a flat continuous forward interpolation method, the bump instrument can be interpreted as a sudden step change in the continuously compounded forward rate, which is constantly applied for all t $>$ t_bump_start and it reverts back for t $>$ t_bump_end.

2.1.13 Fx Forward

Fx Forward Points

The market quotes the fx spot rate and fx forward points for an fx pair. An fx pair is typically referred to as CCY1CCY2, meaning CCY2 amount for one unit of CCY1. There are various ways of referrring to CCY1 and CCY2, but the standard practice is the following:

- CCY1 = Foreign = Main = Base
- CCY2 = Domestic = Money = Term

Let R_t the fx forward point for delivery at t and S the fx spot rate. Let also define:

- $R_{O/N}$ = overnight fx forward point
- $R_{T/N}$ = tomorrow next fx forward point
- S = fx spot rate
- $R_{S/N}$ = spot next fx forward point
- R_t = fx forward point for delivery at $t > t_{spot}$

Let also F_t the fx rate for delivery at t, with today = t_0. Then, the following relationships hold:

$$F_0 = S - \left(R_{O/N} + R_{T/N}\right) \tag{2.10}$$
$$F_1 = F_0 + R_{O/N} = S - R_{T/N} \tag{2.11}$$
$$F_2 = S \tag{2.12}$$
$$F_t = S + R_t \tag{2.13}$$

Fx Forward yield curve instrument

An fx forward is typically a funding instrument. The funding discount factor is solved such that the given fx rate for delivery at t is recovered. The instrument is used to solve for the foreign curve and it requires the domestic curve and the fx spot rate to be available. The present value of an fx forward instrument for delivery at t is:

$$PV = F_0 \left[\frac{df_t^f}{df_t^d}\right] - F_t \tag{2.14}$$

where f stands for foreign and d stands for domestic.

A curve with only Fx Foward points can be easily solved for by imposing that forward points between today and spot are provided in a contiguous date sequence, so that the fx rate between today and spot is given and defined. Assume spot is $t_0 + 2$, then the curve can be solved as follows:

At $t = t_0$:

$$F_0 = S - \left(R_{O/N} + R_{T/N}\right) \tag{2.15}$$
$$df_0^f = df_0^d = 1.0 \tag{2.16}$$
$$\tag{2.17}$$

At $t = t_1 = t_0 + 1$:

$$F_1 = S - \left(R_{T/N}\right) \tag{2.18}$$
$$df_1^f = \frac{F_1}{F_0} df_1^d = \frac{S - R_{T/N}}{S - \left(R_{O/N} + R_{T/N}\right)} df_1^d \tag{2.19}$$
$$\tag{2.20}$$

At $t = t_2 = t_{spot} = t_0 + 2$:

$$F_2 = S \tag{2.21}$$
$$df_2^f = \frac{F_2}{F_0} df_2^d = \frac{S}{S - \left(R_{O/N} + R_{T/N}\right)} df_2^d \tag{2.22}$$

At $t > t_{spot}$:

$$F_t = S + R_t \tag{2.23}$$
$$df_t^f = \frac{F_t}{F_0} df_t^d = \frac{df_{spot}^f}{df_{spot}^d} \frac{F_t}{F_{spot}} df_t^d \tag{2.24}$$
$$= \frac{df_{spot}^f}{df_{spot}^d} \frac{S + R_t}{F_{spot}} df_t^d \tag{2.25}$$

Fx forwards can be quoted as an outright rate F_t or as a spread premium R_t to spot, where $F_t = S + R_t$ or as a spread discount to spot R_t, where $F_t = S - R_t$. For O/N and T/N, spread premium is defined by equations (2.10) and (2.11). Spread premium is a default quotation method.

The conventions used to specify an fx spot rate include domestic currency, foreign currency, domestic calendar, foreign calendar, and spot days. The value of an fx index is computed as follows:

$$F_t = F_0 df_t^f / df_t^d.$$

2.2 Inflation curve instruments

2.2.1 Zero-Coupon Inflation Swap

The PV of a zero-coupon inflation swap is the difference between the PV of its fixed and inflation legs, respectively calculated as follows:

$$PV_{Fxd} \;=\; \left(\left[1 \;+\; (\frac{1 \wedge yf}{f})\, R \right]^{(1 \vee yf)} - 1 \right) df_U^F$$

$$PV_{IL} \;=\; \left(\frac{I(T)}{I(S)} \;-\; 1 \right) df_U^F$$

where:

- S is the inflation reference start date
- T is the inflation reference end state
- U is the maturity date
- $I(t)$ is the inflation swap breakeven index at date t
- R is the fixed rate of the inflation swap
- y is the number of years between the accrual start date and the accrual end date, calculated as the number of months between those two dates divided by 12.
- f is the frequency of compounding a year

Hence the breakeven inflation rate is:

$$R^* \;=\; \frac{f}{1 \wedge yf} \left[\left(\frac{I(T)}{I(S)} \right)^{\frac{1}{1 \vee yf}} - 1 \right]$$

Chapter 3

Components

3.1 Spine curves

We define the following spine curves:

- $df_i(t)$: the base tenor index spine curve
- $\theta(t)$: the structure spine curve
- $df_\tau(t,\tau)$: the tenor basis spread curve
- $df_f(t)$: the funding spread curve

3.2 Financial curves

We define the following financial curves:

- $df_I(t,\tau) \;=\; df_i(t)\, df_\tau(t,\tau)\, \theta(t)$; the index financial curve
- $df_F(t) \;=\; df_i(t)\, df_f(t)\, \theta(t)$; the funding financial curve

3.3 Jacobian

The spine curve values (except those of the structure spine curve) are functions of some variables $\mathbf{x} = (x_1, \ldots, x_n)$, so we write, for instance, $df_i(t) = df_i(t;\mathbf{x})$. Consequently, the index and funding financial curves, $df_I(t,\tau;\mathbf{x})$ and $df_F(t;\mathbf{x})$ also depend on the variables $\mathbf{x}$. In practice, each spine curve is only sensitive to a mutually exclusive subset of variables.

The financial curve gradients $\nabla df_I(t,\tau;\mathbf{x})$ and $\nabla df_F(t;\mathbf{x})$ are calculated from the spine curve gradients $\nabla df_i(t,\mathbf{x})$, $\nabla df_\tau(t,\tau;\mathbf{x})$ and $\nabla df_f(t;\mathbf{x})$.

Using the formulas in 2.1, it is then possible to calculate the gradient of the present value of each instrument relative to the variables: ∇PV.

The jacobian is the matrix whose rows are the instrument present value gradients ∇PV

Chapter 4

Curve Building Algorithm

4.1 Overview

At a conceptual level, the yield curve building algorithm can be split into the following steps:

1. Loading of the calibration instruments.
2. Creation of the model, based on some model parameters.
3. Creation of the knot-point placement algorithm.
4. Placement of the knot-points on the spine curves of the model.
5. Creation of the initialization algorithm.
6. Initialization of the unknowns according to this initialization algorithm.
7. Creation of the least squares solver based on a set of parameters.
8. Solving of the least squares problem.

Most of the available choices at each step can be changed independently of choices at the other steps.

4.2 Knot-point Placement

`FlexYCF` provides considerable flexibility in how to place knot-points on the time axis. The level (or value) of an unknown knot-point can still vary though, only its time is fixed once placed.

The problem of knot-point placement can be stated as follows: given a set of instruments and a given model containing one or several curves, where should the knots be placed? There is no definite answer to this question, as the placement ultimately depends on a trade-off between locality, solvability, interpretability, etc ...

The knot-point placement algorithm is made of two steps:

1. Instrument selection: the set of instruments is filtered so as to only retain those that are of interest or makes sense for the model at hand.

2. Knot-point creation: this is responsible to creating the knot-points, based on the subset of instruments just selected.

While models and knot-point placements are totally independent in the code, they are not wholly interchangeable in practice. Rather, each model, due to its internal representation of the curves, has a preferred knot-point placement. The beginning of the name of a placement class indicates which model it was planned to be used with.

Most of the knot-points are placed at a time corresponding to the time to maturity of an instrument.

Note that due to their nature, the so-called "structure" instruments, namely steps, turns and bumps, are handled separately: knot-point placements do not apply to them.

Here is a list of the knot-point placements currently implemented in `FlexYCF`, and some explanation about them and with which models they can be used.

4.2.1 Stripper Model Placements

This section presents knot placements that apply to the Stripper model.

As far as instrument selection is concerned, both placements introduced below discard instruments other than cash instruments, futures or interest rate swaps. They also filter expired futures out.

Additionally, if at this point there remains at least one futures that has not expired at the value date, non-base tenor cash instruments and interest rate swaps with maturity less than the last futures end date are discarded. This means that if there is no futures specified as input or if they all have expired, all cash instruments and interest rate swaps are selected.

All Dates Placement

After having removed some of the instruments as just described above, the "All Dates" placement places a knot on the time axis at the corresponding maturity of each selected instrument.

The issue with this placement is that this often results in a downward kink at the very short end of the forward curve. This is due to the first futures and base tenor cash rate overlapping, and is generally undesirable as it does not correspond to what is observed on the market.

No Cash End Date Placement

After the instrument selection step presented above, this knot-point placement places knots at the maturity of all instruments, except for the base tenor cash rate, for which the knot is placed at the effective date of the first futures. This avoids the undesirable kink as with the "All Dates"' placement.

The "No Cash End Date" placement is the default knot placement for the Stripper model. Moreover, parameter CashSelectionRule can be set to the following values.

FutureStartLinear Cash points before the start date of the first future are placed (i.e., any cash after the first future is discarded). Moreover all cash instruments are used to bootstrap zero rates and the zero rate of a stripper curve at the start date of the first future is linearly interpolated from them.

FutureStartBlending Cash points before the start date of the first future are placed, additionally the first cash after the first future is placed at the first future start date.

BaseRate Only the cash with tenor equal to base curve tenor is used.

4.2.2 Multi-Tenor Model Placement

This section presents knot placements that apply to the Multi-Tenor model.

Stripper Placement

This placement places one cash instrument, futures and swaps on the base spine curve, that is, the curve identified with "Base Rate" as follows:

- one knot is placed at the first futures start date for the cash instrument whose tenor corresponds the base rate
- a knot-point for each futures is placed at the related futures end date
- a knot-point for each swap is placed at the swap end date

In addition, knots on the other curves are placed as follows:

- a knot for each cross-currency swap is placed on the funding spine curve at each cross-currency swap end date
- a knot for each tenor basis swap is placed on the index spine curve other than the base rate tenor that matches the tenor leg of the basis swap
- a knot is placed at each end date of those cash instruments whose tenor is different from the base rate, on the

For consistency, the base rate should be the tenor against which tenor basis swap spreads are quoted.

It is an extension of the "No Cash End Date" placement for the Stripper model in the sense that if no cross-currency basis or tenor basis swap spread is provided as input, the "Stripper" placement will place knots from the same instruments at the same location according to the same rules.

Additional parameters can be specified in the "Kpp Parameters" table to drive the knot placement:

1. The "Tenor Surface" parameter specifies whether cash instruments ("Cash") or tenor basis swaps ("Tenor Basis Swaps") should have precedence on the short end of index spread curves. The default behaviour is cash precedence.

2. The "Multiple Only" parameter specifies whether to place a tenor basis knot-point on the appropriate index spine curve only if its maturity is a multiple of its non-reference tenor.

   For example, assuming a reference tenor of 3M, the 1M vs 3M basis swap with maturity 1M is placed on the 1M spine curve, but the 3M vs 6M basis swap with maturity 1M (and in general less than 6M) is discarded.

   The default behaviour is not to use this feature.

Back Stub Rate Calculation

Recall that the forward rate for a tenor τ between dates S and T is calculated as:

$$F(S,T) \;=\; \frac{B}{T-S}\left(\frac{df_I(S,\tau)}{df_I(T,\tau)} - 1\right),$$

where B is the appropriate basis.

In general, the maturity date T is calculated from the value date S by adding the tenor τ to it, accounting for Bank holidays and conventions. For a stub period however, this would result in a date T that is after the maturity date.

This occurs in one of following cases:

1. The maturity of an instrument is not a multiple of the tenor of (one of) its leg(s).
2. The maturity of an instrument is less than the tenor of (one of) its leg(s).

We are mostly interested by the second case, it allows specifying a fine term structure on the short end of a spine curve.

To address this issue, the end date the back stub period can be adjusted for "synthetic" cross-currency and tenor basis swaps with the following parameters in the "Curve Parameters" table:

1. "Tenor Basis Back Stub End Date". This parameter drives the calculation of the end date of the back stub of tenor basis swaps, when applicable.
2. "Ccy Basis Back Stub End Date". This parameter drives the calculation of the end date of the back stub of cross-currency swaps, when applicable.

See 4.2.2 for an example of the applicability of the back stub end date adjustments.

Both parameters can be set to one of the following values:

1. "NotAdjusted": no adjustment is made to the end date. This is the default value for cross-currency swaps.
2. "FromAccrualEndDate": the end date is set to the accrual end date. This the default value for tenor basis swaps.
3. "FromStartDatePlusTenor": the end date is calculated by adding the closest tenor corresponding to the accrual period to the start date.

Adjusting the end date of synthetic currency and tenor basis swaps avoids stability issues in model calibration and risk calculation.

Table 4.1: Example of "multiple only" and back stub end date adjustment for Tenor Basis Swaps

Maturity	1M vs 3M	3M vs 6M	3M vs 12M
2D	N†*	N†*	N†*
1W	N†*	N†*	N†*
1M	Y†	N†*	N†*
2M	Y†	N†*	N†*
3M	Y	N*	N*
6M	Y	Y	N*
9M	Y	N*	N*
12M	Y	Y	Y
18M	Y	Y	N*
2Y	Y	Y	Y

Y/N : with the "multiple only" option, is the instrument selected ? (Y)es/(N)o
† : back stub on the reference (3M) tenor leg
* : back stub on the non-reference leg

Natural Swap Tenor Placement

This placement places one cash instrument, instrument futures and swaps on the natural swap tenor curve (e.g.: 6M for GBP).

For consistency, the natural swap tenor should be the one against which tenor basis swap spreads are quoted.

4.2.3 Inflation Placement

This is the knot-point placement to build an inflation curve out of zero-coupon inflation swaps.

4.3 Initialization

Initialization algorithms provide the first guess from which multidimensional solvers iterate.

Initialization algorithms can be compared against each other by looking at their respective:

- accuracy: the norm of the residuals once variables have been initialized should be as small as possible
- time: initialization should be as fast possible

The following initialization algorithms have been implemented in FlexYCF:

4.3.1 Break-out Initialization

This initialization only applies to multi-curve models. It breaks out the initial problem into several subproblems (one for each curve). The underlying idea is that is faster to

solve each subproblem, and that the combined solutions of those is close to the solution of the whole problem.

It is possible to specify the order in which each subproblem should be solved.

4.3.2 Constant Initialization

This initialization initializes each unknown so that the corresponding spot rates matches a specified value.

4.3.3 Initialization from instrument rates

This uses the market rate of each instrument to initialize the variable.

Let R be the quoted rate of the instrument. The initial guess for the discount factor associated to the instrument can be computed as described below.

Cash

The rate R is applied to the accrual period. The initial guess of the base index spine discount factor at the end date t_n of the cash instrument is:

$$df_{ig}^{i}(t_n) \ := \ \frac{1}{1 + R\alpha^R} \tag{4.1}$$

Futures

The rate R is applied to the accrual period. The initial guess of the base index spine discount factor at the end date t_n of the future is:

$$df_{ig}^{i}(t_n) \ := \ \frac{df_{ig}^{i}(t_s)}{1 + (1 - R)\alpha^R} \tag{4.2}$$

where $df_{ig}^{i}(t_s)$ is the initial guess for the discount factor at the future start date t_s.

Swaps

A constant rate R is applied up to swap maturity with annual compounding. The initial guess of the base index spine discount factor at the maturity date t_n of the interest rate swap is:

$$df_{ig}^{i}(t_n) \ := \ (1 + R)^{-t_n} \tag{4.3}$$

$$bpv_{ig} = \frac{\partial PV}{\partial R} = t_n(1 + R)^{-t_n+1} \tag{4.4}$$

Currency basis swaps

A constant rate R is applied up to swap maturity with annual compounding. In this case the rate R represents the spread to the index financial curve; also, the rate is quoted as a spread on the paying leg. Empirical tests have shown that the funding spine discount factor at the maturity of the currency basis swap is about $(1 - R)^{t_n}$. Hence, we use this proxy as an initial guess:

$$df_{ig}^{f}(t_n) \;:=\; (1 - R)^{t_n} \tag{4.5}$$

Tenor basis swaps

A constant rate R is applied up to swap maturity with annual compounding. In this case the rate R represents the spread to the index spine curve; also, the rate is quoted as a spread on the receiving leg. As for cross-currency swaps, empirical tests showed that a good proxy of the tenor basis spine discount factor at the maturity date t_n of a tenor basis swap is $(1 \pm R)^{t_n}$, depending on whether its (non-base) tenor τ is respectively shorter or longer than the base tenor τ^*. Thus, we set the initial guess as follows:

$$df_{ig}^{\tau}(t_n, \tau) \;:=\; \begin{cases} (1 + R)^{t_n} & \tau < \tau^* \\ (1 - R)^{t_n} & \tau > \tau^* \end{cases} \tag{4.6}$$

4.3.4 Inflation Initialization

This algorithm initializes the unknowns of an inflation curve from the inflation-linked zero-coupon inflation swap rates.

4.3.5 Composite Initialization

This allows to combine several initializations in a row. This makes sense only for break-out.

4.4 Solvers

Least squares solvers are separated from FlexYCF but have been developed to closely fit the needs of FlexYCF.

The formulation of the least squares problem can be stated as follows. Given m real functions $r_1, \ldots, r_m$ of n variables $\mathbf{x} = (x_1, \ldots, x_n)$ each, and $m \geq n$, find $\mathbf{x}^*$ a (local) minimizer for the sum of squares:

$$F(\mathbf{x}) \;=\; \frac{1}{2} \sum_{i=1}^{m} r_i^2(\mathbf{x}) \;=\; \frac{1}{2} \,||\, \mathbf{r}(\mathbf{x}) \,||_2^2$$

In our yield curve building context, the residuals r_i correspond to the PVs of the input instruments and the x_j to the spine curve variables so solve.

It is possible to specify a function for the calculation of the gradients of the residuals will be used to calculate the jacobian $\mathbf{J}$ of $\mathbf{r}$ analytically by the solvers. If such a function

is not known by the user, a proxy of the jacobian will be calculated by a first order approximation of the derivatives.

4.5 Least Squares Solvers

Non-linear Least squares solvers are iterative algorithms that, starting from an initial guess $\mathbf{x}^{(0)}$ provided by the user, repeat a series of instructions until one of a series of stopping criteria is met.

4.5.1 Gauss-Newton

It is possible to specify whether to calculate the jacobian only at the first iteration or at each iteration. The advantage of computing only the jacobian once is that it requires less calculations. In this case the algorithm is likely to converge only if the user can provide a good initial guess.

The Gauss-Newton is based on a linear approximation of $\mathbf{r}$ around the current point $\mathbf{x}$ and relies, at each step, on the solving of a linear least squares problem (the so-called *normal equations*): $\mathbf{J}^{\mathrm{T}}\mathbf{J}\,\mathbf{p} = -\mathbf{J}^{T}\,\mathbf{r}$, where $\mathbf{p}$ is the unknown direction to solve for so as to set $\mathbf{x}^{(k+1)} := \mathbf{x}^{(k)} + \mathbf{p}$ for next iteration.

Such a linear least squares problem is solved using an in-house implementation of QR decomposition.

4.5.2 Levenberg-Marquardt

The Levenberg-Marquardt algorithm can be seen as an improvement over the Gauss-Newton solver as it is more likely to converge even when the starting point is far from the solution.

4.5.3 Composite

A composite solver is a solver made of a several other least squares solvers. One after the other, each of the inner solvers will try to minimize a squares problem in the same order as the one according which they were added to the composite solver. This sequence of solving attempts will stop once a criterion has been satisfied: for now it is that the norm of the residuals is under some specified threshold.

4.5.4 Stopping Criteria

The iterative algorithms presented above all share some common stopping criteria. All these stopping criterias can be retrieved from tables. They will stop when one of the following condition is satisfied:

Maximum number of iterations

Stop when the number of iterations has reached a specified number.

Maximum number of function evaluations

Stop when the number of evaluations of the objective functions has reached a specified number. This is useful when evaluating the objective function is expensive.

Residuals Norm Threshold

Stop when the euclidean norm of the residuals, $|| \mathbf{r}(\mathbf{x}) ||_2$, is less than a specified threshold: $\varepsilon_{\mathbf{r}}$. As $|| \mathbf{r} ||_\infty \leq || \mathbf{r} ||_2$, when the norm of the residuals is less the threshold, so is each of the residuals. In other words $|| \mathbf{r}(\mathbf{x}) ||_2 < \varepsilon_{\mathbf{r}}$ implies $| r_i(\mathbf{x}) | < \varepsilon_{\mathbf{r}}$.

Often in practice, we even have $| r_i(\mathbf{x}) | \ll \varepsilon_{\mathbf{r}}$.

x Tolerance

Stop when the relative improvement error between two consecutive iterates is less than a specified tolerance $\varepsilon_{\mathbf{x}}$:

$$|| \mathbf{p} ||_2 \ < \ || \mathbf{x} ||_2 \, \varepsilon_{\mathbf{x}}$$

F Tolerance

Stop when the relative improvement error between two consecutive sum of squares is less than a specified threshold ε_F:

$$| F(\mathbf{x} + \mathbf{p}) - F(\mathbf{x}) | \ < \ \varepsilon_F \, F(\mathbf{x})$$

Gradient Norm Threshold

Stop when the norm of the gradient of the objective function F is less than a specified threshold: ε_G.

Noticing that $\nabla F(\mathbf{x}) \ = \ \mathbf{J}^T(\mathbf{x}) \, \mathbf{r}(\mathbf{x})$, the stopping condition can be written:

$$|| \nabla F(\mathbf{x}) ||_2 \ = \ || \mathbf{J}^T(\mathbf{x}) \, \mathbf{r}(\mathbf{x}) ||_2 \ < \ \varepsilon_G$$

Chapter 5

Models

Models can calculate funding discount factors, index discount factors and their gradients relative to the unknowns at any time. Index discount factors are used to calculate forward rates.

Models are made of one or more "financial curve(s)" (see 3.2). `FlexYCF` distinguishes *single curve models* that are made of one financial curve only and *multi-curve models* that are made of several curves. At the moment, a financial curve can either be a funding curve, that is a curve to be used for discounting, or an index curve. There are index curves for tenors ON, 1W, 1M, 2M, ..., 12M.

Each financial curve can be a single curve or a combination of several curves. For instance, the spot rate curves of the multi-tenor model are the sum of the base rate curve, the tenor spread curve and the structure curve spot rates. This model has a base rate curve which is the combination of a curve for this rate plus the structure curve. Other curves (discount and tenors) are internally represented as a spread curve "over" the base rate curve.

At the time of writing, the models implemented in FlexYCF are:

5.1 Stripper Model

This is the model to use to build a yield curve out of a set of instruments that include cash rates, futures, swaps and structure instruments (turns, steps and bumps). Cross-currency and tenor basis spread will be ignored if passed as inputs to build the yield curve of a stripper model. Reciprocally, a stripper model is used if the input market data contain no cross-currency of tenor basis spreads.

The unique financial curve of a stripper model is a combination of a base and a structure spine curves. Funding and index discount factors are calculated from this financial curve: $df_I(t,\tau) = df_F(t),\ \forall\, t,\ \tau$ (using notations as in 3.2).

5.2 Multi-Tenor Model

This is the model of choice to build a yield curve that fits to a set instruments that include cross-currency and/or tenor basis swaps.

Contrary to the stripper model, it is made of several financial curves, as follows:

1. A funding financial curve made of the base index spine curve, the funding spread curve and the structure spine curve:

$$df_F(t) \;=\; df_i(t)\, df_f(t)\, \theta(t)$$

   Funding discount factors are calculated from this financial curve.

2. A collection of index financial curve (one per tenor τ supported), made of the base index spine curve, the tenor spread curve and the structure spine curve:

$$dF_I(t,\tau) \;=\; df_i(t)\, df_\tau(t,\tau)\, \theta(t)$$

   Index discount factors are calculated from this financial curve.

Note that for the base tenor τ^*, we have $df_{\tau^*}(t,\tau^*) \;=\; 1$.

5.3 Multi-Tenor Formulation Model

This the collection of discount curve and tenor curves build using zero rates and tenor zero rates (each curve is formulated as a stripper curve).

1. A funding financial curve made of the funding curve and the structure spine curve:

$$df_F(t) \;=\; df_f(t)\, \theta(t)$$

   Funding discount factors are calculated from this financial curve.

2. A collection of index financial curve (one per tenor τ supported), made of the tenor curve and the structure spine curve:

$$dF_I(t,\tau) \;=\; df_\tau(t,\tau)\, \theta(t)$$

   Index discount factors are calculated from this financial curve.

Note that there is no base curve.

5.4 Multi-Tenor OIS Model

This is a multi curve model that is compatibile with the optimized representation of OIS swaps and OIS-Libor basis swaps as described in equations (2.5) and (2.6). Like a multi-tenor model, it is made of several financial curves:

1. A funding financial curve made of the base spine curve and the structure spine curve:

$$df_F(t) \;=\; df_f(t)\, \theta(t)$$

   Funding discount factors are calculated from this financial curve.

2. A collection of index financial curves (one per tenor τ supported), made of the base spine curve, the tenor spread curve and the structure spine curve:

$$dF_I(t,\tau) \;=\; df_f(t)\, df_\tau(t,\tau)\, \theta(t)$$

   Index discount factors are calculated from this financial curve.

5.5 Multi-Tenor OIS Funding Model

This is a multi curve model that is compatibile with the optimized representation of OIS swaps and OIS-Libor basis swaps as described in equations (2.5) and (2.6). In addition the discounting is done using overnight rate. As a multi-tenor model, it is made of several financial curves:

1. A funding financial curve made of the base spine curve and the structure spine curve:
$$df_F(t) \; = df_b(t)\, df_s(t)\, df_{ON}(t, ON)\, \theta(t)$$
Funding discount factors are calculated from this financial curve.

2. A collection of index financial curves (one per tenor τ supported), made of the base spine curve, the tenor spread curve and the structure spine curve:
$$dF_I(t,\tau) \; = \; df_b(t)\, df_\tau(t,\tau)\, \theta(t)$$
Index discount factors are calculated from this financial curve.

5.6 Spread Models

On top of any model, one can create the following spread models using generic currency basis swaps as calibration instruments (see section 2.1.4 for possible choices).

Let $df^{F_{base}}(t)$ and $df^{I_{base}}(t,\tau)$ denote discount factor and tenor discount factor (respectively) from the base model.

5.6.1 Funding Spread Stripper Model

The Funding Spread Stripper Model builds a spread curve which is then used in pricing for the calculation of funding discount factors as a spread over a base funding curve, whereas the index discount factors are resolved from the base index curve, as follows

$$df^F(t) = df^{F_{base}}(t) df^f(t) \tag{5.1}$$
$$df^I(t,\tau) = df^{I_{base}}(t,\tau) \tag{5.2}$$

where $df^{F_{base}}(t)$ is the funding discount factor of the base curve and $df^f(t)$ is the funding spread discount factor.

The Funding Spread Stripper Model is typically bootstrapped with single currency or cross currency Libor basis instruments or zero rates, where in the calibration the domestic funding discount factor is replaced with $df^F(t) = df^{F_{base}}(t) df^f(t)$.

Single currency Libor basis swaps are represented as currency basis instruments composed of a Libor leg vs an exchange of notionals. In this case, the model discounts at a spread over Libor. In fact, if input spreads are zero, the spread discount factors $df^f(t)$ are solved such that pricing recovers the base index discount factors.

The model also supports single currency basis OIS swaps as calibration instruments. In this case, the model discounts at a spread over OIS rates. In addition, the model

supports basis swaps over a base funding curve. In this case, the model discounts at a spread over the base funding curve, whatever the base funding curve is.

Finally, the model also supports cross currency Libor basis swaps, and in this case it discounts at cross currency rates.

5.6.2 Funding Stripper Model

df bootstrapped,

$$df^F(t) = df(t)$$

and $df^I(t,\tau) = df^{I_{base}}(t,\tau)$ for any τ.

The Funding Stripper Model model is similar to the Funding Spread Stripper model, but in this case we solve directly for the funding discount factor $df^F(t)$, as opposed to the spread discount factor $df^f(t)$. Whilst the Funding Stripper Model is equivalent to the Funding Spread Stripper Model for pricing, the two models differ in the risk decomposition. In the case of the Funding Spread Stripper Model, the risk is decomposed in the funding delta of the base curve and the funding spread delta, whereas in the case of the Funding Stripper Model the risk is expressed in terms of the funding delta only.

5.6.3 Index Spread Stripper Model

Calibration instruments: single currency basis swaps (*Leg2Type=LiborFlat* has to be used).

$$df^I(t,\tau) = df^{I_{base}}(t,\tau) df_s(t)$$

for any τ and $df^F(t) = df^{F_{base}}(t)$.

5.7 Inflation Model

The inflation model builds an inflation curve out of inflation-linked zero-coupon swaps and seasonality instruments.

5.8 Default Model Parameters

We now detail the model parameters used by default for the stripper and multi-tenor models.

5.8.1 Stripper Model Default Parameters

It uses a log FVF formulation with straight line interpolation method, the “no cash end date” stripper knot-point placement, the “from instruments” initialization and the Levenberg-Marquardt solver.

5.8.2 Multi-Tenor Model Default Parameters

By default, the base rate is 3M. It adds the following index spine curves to the tenor spread surface: “ON, 1M, 3M, 6M and 1Y” and use multi-tenor stripper knot-point placement. The interpolation of the base spine curve is composite: straight line up to the last future date and monotone convex spline after this. All spread curves use straight line interpolation. The formulation is hard-coded to log FVF. There is no initialization and the solver is Levenberg-Marquardt.

Chapter 6

Bespoke Models

QL has a set of bespoke yield curve models which embed calibration instrument definition, knot point placement scheme, interpolation scheme and solvers within the model itself. These bespoke models are also known as StandardIRCurve models.

The bespoke models implemented in QL are described in the following sub-sections.

6.1 Standard IR Curve

This model is conceptually similar to the MultiTenor model. It builds a main projection curve, a spread funding curve and a spread surface for projection at different tenors. The interpolation method is a proxy quadratic form and the model supports Cash, Swaps, Ccy Basis and Tenor Basis instruments only.

6.1.1 Model selection

This model is selected when a curve is built with at least Cash or Swaps instruments and no model parameters.

6.2 Funding Swap Spread Model

The Funding Swap Spread Model builds a spread curve over a base funding curve, for example a spread over OIS or Ccy Basis. The spread model is then used in pricing in order to calculate funding and index discount factors according to the following relationship:

$$df^F(t) = df^{F_{base}}(t) df^s(t) \tag{6.1}$$

$$df^I(t,\tau) = df^{I_{base}}(t,\tau) \qquad \forall \tau \tag{6.2}$$

where $df^{F_{base}}(t)$ is the funding discount factor of the base curve and $df^s(t)$ is the spread discount factor.

The Funding Swap Spread Model supports only Ccy Basis instruments, which are interpreted as swap basis instruments composed of a two 3m floating legs. The input

spread is added (or subtracted) to the forwards on one leg, and the spread discount factor curve is solved to project a term structure of spreads in order to reprice the calibration instruments to par. Also, in this model, the forwards are projected from the funding curve of the base curve. Hence, the governing equations of the Ccy Basis instrument in this model are:

$$\left(\frac{df_0^{F_{base}}}{df_1^{F_{base}}}\frac{df_0^s}{df_1^s} - 1\right) df_1^{F_{base}} + \cdots + \left(\frac{df_{n-1}^{F_{base}}}{df_n^{F_{base}}}\frac{df_{n-1}^s}{df_n^s} - 1\right) df_n^{F_{base}} \tag{6.3}$$

$$= \left(F_1^{F_{base}} + a \times x\right) \alpha_{0,1}^A df_1^{F_{base}} + \cdots + \left(F_n^{F_{base}} + a \times x\right) \alpha_{n-1,n}^A df_n^{F_{base}} \tag{6.4}$$

where $a = -1.0$ for Spread Type=Subtract or missing and $a = 1.0$ for Spread Type=Add, df_t^s is solved such that the basis instrument is priced to par, $df_t^{F_{base}}$ are the funding discount factors from the base curve and $F_i^{F_{base}}$ are the forward implied from the funding base curve for a $3m$ period,i.e.:

$$F_{t_i}^{F_{base}} = \left(\frac{df_{t_{i-1}}^{F_{base}}}{df_{t_i}^{F_{base}}} - 1\right) \frac{1}{\alpha_{t_{i-1},t_i}^A} \tag{6.5}$$

It is straightforward to verify that when $x = 0$ we have $df^s(t) = 1$ for all t, meaning that for zero spreads the model discounts and projects at the same discounting rates as the base funding curve. In particular, if the base funding curve is built with OIS rates, with $x = 0$ the discounting spread model discounts at OIS rates.

6.2.1 Model selection

This model is selected when a curve is built with Ccy Basis swaps instruments only, with no model parameters and the market is SGOVT (from QL350), IGOVT (from QL350) or FDG (from QL370).

The SGOVT and IGOVT curves are used to describe the OTR bonds spread discounting over the base discounting curve of the swap market. The FDG curve is used to describe a funding spread over the base discounting curve.

6.3 Index Swap Spread Model

The Index Swap Spread Model model builds a spread curve which is then used in pricing for the calculation of funding and index discount factors as a spread over a base index projection curve (i.e. as a spread over Libor) as follows:

$$df^F(t) = df^{I_{base}}(t, 3m) df^s(t) \tag{6.6}$$

$$df^I(t, \tau) = df^{I_{base}}(t, \tau) df^s(t) \tag{6.7}$$

where $df_{base}^I(t, \tau)$ is the τ index projection discount factor of the base curve and $df^s(t)$ is the spread discount factor.

Similarly to the Funding Swap Spread Model, the Index Swap Spread Model supports only Ccy Basis instruments, which are interpreted as swap basis instruments composed of a two 3m floating legs. The input spread is added (or subtracted) to the forwards on one leg, and the spread discount factor curve is solved to project a term structure of spreads in order to reprice the calibration instruments to par. But, in this case, the forwards are projected from the index curve of the base curve. Hence, the governing equations are:

$$\left(\frac{df_0^{I_{base}}}{df_1^{I_{base}}}\frac{df_0^s}{df_1^s} - 1\right) df_1^{F_{base}} + \cdots + \left(\frac{df_{n-1}^{I_{base}}}{df_n^{I_{base}}}\frac{df_{n-1}^s}{df_n^s} - 1\right) df_n^{F_{base}} \tag{6.8}$$

$$= \left(F_1^{I_{base}} + a \times x\right) \alpha_{0,1}^A df_1^{F_{base}} + \cdots + \left(F_n^{I_{base}} + a \times x\right) \alpha_{n-1,n}^A df_n^{F_{base}} \tag{6.9}$$

where $a = -1.0$ for Spread Type=Subtract or missing and $a = 1.0$ for Spread Type=Add, df_t^s is solved such that the basis instrument is priced to par, $df_t^{I_{base}}$ are the $3m$ index discount factors from the base curve and $F_t^{I_{base}}$ are the $3m$ forwards computed on the base index curve, i.e.:

$$F_{t_i}^{I_{base}} = \left(\frac{df^{I_{base}}(t_{i-1}, 3m)}{df^{I_{base}}(t_i, 3m)} - 1\right) \frac{1}{\alpha_{t_{i-1},t_i}^A} \tag{6.10}$$

The model can be applied as a funding spread curve, to discount at a spread over $3m$ Libor. In fact, it is straightforward to verify that when $x = 0$ we have $df_t^s = 1$ for all t, meaning that for zero spreads the model discounts at $3m$ Libor flat.

The model can also be applied to project indices, at different tenors, over and above the projected rates of the base index curve for each given tenor τ, which incorporate tenor basis spreads. Again, with $x = 0$ (and hence $df^s(t) = 1$ for all t), the model projects at Libor flat at the appropriate tenor τ.

6.3.1 Model selection

This model is selected when a curve is built with Ccy Basis swaps instruments only, with no model parameters and the (currency, market) pair is either EUR LIBOR, JPY TIBOR, AUD LIBOR or CAD LIBOR.

6.4 Index 3m Swap Spread Model

In some cases, it is useful to project the index discount factor for an index with tenor τ as a spread over $3m$ Libor forwards, as opposed to as spread over the τ Libor forwards. The calibration instruments are identical to the Index Swap Spread Model, but in this case the funding and index discount factor are computed as follows:

$$df^F(t) = df^{I_{base}}(t, 3m) df^s(t) \tag{6.11}$$

$$df^I(t) = df^I{}_{base}(t, 3m) df^s(t) \tag{6.12}$$

Hence, in this model, the index discount factor $df^I(t)$ does not depend on τ, i.e. $df^I(t, \tau) = df^I(t)$ for all τ.

This model is typically applied to project indices which are defined for a given tenor, as for example the daily OIS indices.

6.4.1 Model selection

This model is selected when a curve is built with Ccy Basis swaps instruments only, with no model parameters and the market is different from the main index of the given currency. In particular, this model is used for the projection of OIS indices, as for example UKBAS, SONIA, RONIA, EONIA and FEDF, or for discounting at a spread over $3m$ Libor.

Chapter 7

Curves and Interpolation

While intuitive, the meaning of a "curve" in FlexYCF is polysemantic. The requirements were:

- a curve goes through knot-points
- new "points" can be added to the curve dynamically.
- the shape of curves of calibrating models can be calibrated to fit market data
- the location of knots
- curves should not oscillate unnecessarily
- left extrapolation, interpolation and right interpolation can be specified independently
- it is possible to "mix" two interpolation methods

A "mathematical" curve is essentially a collection of knot-points $(x_i, y_i)_i$ ordered ascendingly according to their x coordinate, an *interpolation curve*, a left extrapolation method and a right extrapolation method.

An "interpolation curve" be either a UKP[1] curve, a tension spline, or a so-called composite curve. UKP curves are the simplest as their interpolation are close to the intuition of drawing a line that goes through knot-points.

7.0.2 Knot-point

A knot-point (x, y) can be either fixed or unknown. The y-coordinate of a fixed knot-point does not change once set, while it can vary for an unknown knot-point. The x's of all knot-points, whether fixed or unknown, are assumed to be constant once set.

Fixed knot-points are practical for enforcing certain requirements. For instance a discount curve has to go through the $(0, 1)$ knot-point.

Unknown knot-points, whose y-coordinate can vary, are useful to represent unknowns of an optimization problem to be solved. In general, the unknowns of a curve to solve are directly represented as those y's in FlexYCF. For tension splines though, the unknowns are the coefficients of its functional representation.

[1] UKP stands for Unknown Knot-Points

7.1 Classic Interpolation Methods

The interpolation methods detailed here are used by a UKP curve.

Given a collection of knot-points $(x_i, y_i)_{0 \leq i \leq N}$, such that $x_0 < \ldots < x_N$, an interpolation curve/method can calculate, for any x in (x_0, x_N) its interpolated value and its gradient relative to the unknowns.

To simplify gradient calculations, we will assume that all knot-points are unknown as it is always possible to extract the gradient relative to the unknowns from the gradient relative to all knot-points by removing those partial derivatives relative to fixed-knot-points. Note that as knot-points start at index 0 the positions of coordinates in gradients must be understood accordingly.

7.1.1 Flat Interpolation

In FlexYCF, piecewise constant curves are called flat interpolation methods.

Flat Right

Flat right interpolation is defined as: $f(x) = y_{i-1}$, if $x \in [x_{i-1}, x_i),\ 0 < i \leq N$.

If $x \in [x_{i-1}, x_i),\ 0 < i \leq N$, the gradient is

$$\nabla f(x) = \big(0, \ldots, 0, 1, 0, \ldots, 0\big),$$

the 1 being at position $i - 1$.

Flat Left

Flat left interpolation is defined as: $f(x) = y_i$, if $x \in (x_{i-1}, x_i],\ 0 < i \leq N$.

If $x \in (x_{i-1}, x_i],\ 0 < i \leq N$, the gradient is

$$\nabla f(x) = \big(0, \ldots, 0, 1, 0, \ldots, 0\big),$$

the 1 being at position i.

7.1.2 Straight Line

For $x \in [x_{i-1}, x_i),\ 0 < i \leq N$, straight line interpolation is given by the following formula:

$$f(x) \ = \ \big(1 - \lambda_i(x)\big)\, y_{i-1} \ + \ \lambda_i(x)\, y_i, \qquad \text{with} \quad \lambda_i(x) := \frac{x - x_{i-1}}{x_i - x_{i-1}}$$

Still assuming that $x \in [x_{i-1}, x_i),\ 0 < i \leq N$, the gradient is calculated as:

$$\nabla f(x) = \big(0, \ldots, 0, 1 - \lambda_i(x), \lambda_i(x), 0 \ldots, 0\big),$$

the non-zero coordinates being at position $i - 1$ and i.

7.1.3 Bi-Quadratic Interpolation

For $0 \leq i \leq N$, we define the polynomials q_i by $q_i(x) := \alpha_i\, x^2 + \beta_i\, x + \gamma_i$, such that:

1. for each $0 < i < N$, q_i is the quadratic that goes through the consecutive points (x_{i-1}, y_{i-1}), (x_i, y_i) and (x_{i+1}, y_{i+1}).

2. q_0 is either chosen to be linear ($\alpha_0 = 0$) and to go through the points (x_0, y_0) and (x_1, y_1), or is set to be constant ($\alpha_0 = \beta_0 = 0, \gamma_0 = y_0$)[2].

3. q_n is either chosen to be linear ($\alpha_N = 0$) and to go through the points (x_{N-1}, y_{N-1}) and (x_N, y_N), or is set to be constant ($\alpha_N = \beta_N = 0, \gamma_N = y_N$)[3]

The choice of the constant for extremal polynomials will become clear when the bi-quadratic interpolation is introduced in the next section.

For $0 < i < N$, it is easy to check that necessarily:

$$\alpha_i \;=\; \frac{1}{x_{i+1} - x_{i-1}} \left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i} \;-\; \frac{y_i - y_{i-1}}{x_i - x_{i-1}} \right) \tag{7.1}$$

$$\beta_i \;=\; \frac{y_i - y_{i-1}}{x_i - x_{i-1}} - \alpha_i\,(x_{i-1} + x_i) \tag{7.2}$$

$$\gamma_i \;=\; y_{i-1} + x_{i-1} \left(\alpha_i\, x_i - \frac{y_i - y_{i-1}}{x_i - x_{i-1}} \right) \tag{7.3}$$

Interpolation

The bi-quadratic interpolation is given by:

$$f(x) \;=\; \big(\,1 - \lambda_i(x)\,\big)\; q_{i-1}(x) \;+\; \lambda_i(x)\; q_i(x), \quad x \in [x_{i-1}, x_i),\; 0 < i \leq N,$$

where we set

$$\lambda_i(x) \;:=\; \frac{x - x_{i-1}}{x_i - x_{i-1}}$$

Note that $f(x_{i-1}) = y_{i-1}$ and $\lim_{x \uparrow x_i} f(x) = y_i$ for all $0 < i \leq N$, regardless of the choices for q_0 and q_N as long as they are chosen as above. This ensures f is continuous and goes through all knot-points as required.

For $x \in [x_{i-1}, x_i)$, we can write:

$$\begin{aligned} f(x) \;=\; \frac{1}{x_i - x_{i-1}} \big[\, &(\alpha_i - \alpha_{i-1})\, x^3 + (\beta_i - x_{i-1}\,\alpha_i + x_i\,\alpha_{i-1} - \beta_{i-1})\, x^2 \\ &+ (\gamma_i - x_{i-1}\,\beta_i + x_i\,\beta_{i-1} - \gamma_{i-1})\, x + (x_i\,\gamma_{i-1} - x_{i-1}\,\gamma_i) \big] \end{aligned} \tag{7.4}$$

[2] By default, q_0 is constant: q_0: $q_0(x) = y_0$

[3] By default, q_N is linear and goes through the points (x_{N-1}, y_{N-1}) and (x_N, y_N): $q_N(x) = \big(1 - \lambda_N(x)\big)\; y_{N-1} + \lambda_N(x)\; y_N$.

Gradient calculations

As the λ_i's do not depend on the unknowns, it is clear that:

$$\nabla f(x) \;=\; \lambda_i(x)\,\nabla q_i(x) \;+\; \big(1-\lambda_i(x)\big)\,\nabla q_i(x)$$

and

$$\nabla q_i(x) = x^2\,\nabla\alpha_i + x\,\nabla\beta_i + \nabla\gamma_i$$

But using the formulae above, and setting $\Delta_i := x_i - x_{i-1}$, we have:

$$\nabla\alpha_i \;=\; \Big(0,\,\ldots,\,0,\,\frac{1}{\Delta_i\,(\Delta_i+\Delta_{i+1})},\,\frac{-1}{\Delta_i\Delta_{i+1}},\,\frac{1}{\Delta_{i+1}\,(\Delta_i+\Delta_{i+1})},\,0,\,\ldots,\,0\Big) \tag{7.5}$$

$$\nabla\beta_i \;=\; \Big(0,\,\ldots,\,0,\,\frac{-(x_i+x_{i+1})}{\Delta_i(\Delta_i+\Delta_{i+1})},\,\frac{x_{i-1}+x_{i+1}}{\Delta_i\,\Delta_{i+1}},\,\frac{-(x_{i-1}+x_i)}{\Delta_{i+1}\,(\Delta_i+\Delta_{i+1})},\,0,\,\ldots,\,0\Big) \tag{7.6}$$

$$\nabla\gamma_i \;=\; \Big(0,\,\ldots,\,0,\,\frac{x_i\,x_{i+1}}{\Delta_i\,(\Delta_i+\Delta_{i+1})},\,\frac{-x_{i-1}\,x_{i+1}}{\Delta_i\,\Delta_{i+1}},\,\frac{x_{i-1}\,x_i}{\Delta_{i+1}\,(\Delta_i+\Delta_{i+1})},\,0,\,\ldots,\,0\Big), \tag{7.7}$$

where the only non-zero partial derivatives are at positions $i-1$, i and $i+1$. Not that those gradients depend only on the x_i's which are always known, so these $9\times(N-1)$ are calculated only once in the FlexYCF implementation. From there we can easily compute the $\nabla q_i(x)$'s and in turn $\nabla f(x)$, which will have at most 4 non-zero partial derivatives.

7.1.4 Monotone Convex Spline

Whereas all previous interpolation methods are independent from the representation of the financial curve (its `CurveFormulation`), the monotone convex splines relies a "logFvf" formulation. We follow [**?**] and give explicit gradient calculations in all cases. Let f be the instantaneous forward rate curve, and let F defined as $F(t) = \int_0^t f(s)\,ds$. We assume the knot-points are $(t_i, F_i)_{0\le i\le N}$ where $F_i := F(t_i)$. It is assumed that $t_0 = 0$ so that necessarily $F_0 = 0$: the knot-point $(0,0)$ is fixed.

The *discrete forwards* f_i^d are defined as:

$$f_i^d := \frac{F_i - F_{i-1}}{t_i - t_{i-1}}, \quad 0 < i \le N$$

Note that f_i^d is the average instantaneous forward rate over the interval $[t_{i-1}, t_i]$:

$$f_i^d = \frac{1}{t_i - t_{i-1}} \int_{t_{i-1}}^{t_i} f(s)\,ds$$

From a practical point of view, discrete forwards f_i^d are good approximations of the forward rates $F(t_{i-1}, t_i)$, because:

$$F(t_{i-1}, t_i) \;=\; \frac{1}{t_i - t_{i-1}}\left(\frac{P(t_{i-1})}{P(t_i)} - 1\right) \;\simeq\; \frac{F_i - F_{i-1}}{t_i - t_{i-1}}$$

Let us define the *modified discrete forwards* f_i:

$$\begin{aligned} f_i &:= \frac{t_i - t_{i-1}}{t_{i+1} - t_{i-1}}\, f^d_{i+1} \;+\; \frac{t_{i+1} - t_i}{t_{i+1} - t_{i-1}}\, f^d_i, \qquad 0 < i < N \\ f_0 &:= f^d_1 - \frac{1}{2}\,(f_1 - f^d_1) \\ f_N &:= f^d_N - \frac{1}{2}\,(f_{N-1} - f^d_N) \end{aligned}$$

Geometrically, f_i is just the linear interpolation at t_i of the line going through the points $(\frac{t_{i-1}+t_i}{2},\, f^d_i)$ and $(\frac{t_i+t_{i+1}}{2},\, f^d_{i+1})$.

Moreover, $\min(f^d_i,\, f^d_{i+1}) \;\le\; f_i \;\le\; \max(f^d_i,\, f^d_{i+1})$: the modified discrete forwards are smoother that the discrete forwards.

Thus, the idea is to approximate the instantaneous forward rate curve f on the interval $[t_{i-1}, t_i]$, by the quadratic q_i that satisfies the following three conditions:

$$q_i(t_{i-1}) \;=\; f_{i-1}, \qquad q_i(t_i) = f_i, \qquad \frac{1}{t_i - t_{i-1}} \int_{t_{i-1}}^{t_i} q_i(s)\, ds = f^d_i$$

Simple algebra gives:

$$q_i(t) = f_{i-1} - (4\, f_{i-1} + 2\, f_i - 6\, f^d_i)\, x_i(t) + (3\, f_{i-1} + 3\, f_i - 6\, f^d_i)\, x^2_i(t),$$

where

$$x_i(t) \;=\; \frac{t - t_{i-1}}{t_i - t_{i-1}}$$

For $1 < i \le N$, we define the function g_i on $[t_{i-1}, t_i]$ by $g_i(t) := q_i(t) - f^d_i$, and we write $g^L_i := g_i(t_{i-1})$ and $g^R_i := g_i(t_i)$ its values on the left and right bounds of its definition interval.

Note that:

$$\int_{t_{i-1}}^{t_i} g(s)\, ds \;=\; 0$$

We have:

$$g_i(t) \;=\; g^L_i \left(1 - 4\, x_i(t) + 3\, x^2_i(t)\right) \;+\; g^R_i \left(-\, 2\, x_i(t) + 3\, x^2_i(t)\right)$$

Interpolation

For $t \in [t_{i-1}, t_i)$, by definition:

$$F(t) \;=\; F_{i-1} + \int_{t_{i-1}}^{t} f$$

Replacing the instantanenous forward rate function f by q_i gives:

$$\begin{aligned} F(t) \;&\simeq\; F_{i-1} + \int_{t_{i-1}}^{t} q_i(s)\, ds \\ &=\; F_{i-1} + (t - t_{i-1})\, f^d_i + \int_{t_{i-1}}^{t} g_i(s)\, ds \\ &=\; \big(1 - x_i(t)\big)\, F_{i-1} \;+\; x_i(t)\, F_i \;+\; G_i(t), \end{aligned}$$

with

$$\begin{aligned} G_i(t) \ &:= \ \int_{t_{i-1}}^{t} g(s)\, ds \\ &= \ g_i^L(t - t_{i-1}) - (2g_i^L + g_i^R)(t_i - t_{i-1})\, x_i^2(t) + (g_i^L + g_i^R)(t_i - t_{i-1})\, x_i^3(t) \end{aligned}$$

And directly:

$$g_i^L = f_{i-1} - f_i^d \qquad g_i^R = f_i - f_i^d$$

So we can interpolate.
We have not implemented the monotonicity described in [**?**].

Gradient

To calculate the gradient $\nabla F(t)$ (relative to the F_i's) at a point $t \in [t_{i-1}, t_i)$, we can rearrange the formulas above to write:

$$G_i(t) \ = \ \big[(t - t_{i-1}) + (t_i - t_{i-1})\, x_i^2(t)\, \big((x_i(t) - 2\big)\big]\ g_i^L \ + \ (t_i - t_{i-1})\, x_i^2(t)\, \big(x_i(t) - 1\big)\ g_i^R \tag{7.8}$$

This requires expressing g_i^L and g_i^R in terms of the unknowns $(F_j)_{0<j\leq N}$
Using the definitions of g_i^L and g_i^R, f_i and f_i^d, straightforward calculations give:

$$\begin{aligned} g_1^L \ &= \ \frac{1}{2}\left(\frac{-1}{t_2 - t_0}\, F_0 \ + \ \frac{1}{t_2 - t_1}\, F_1 \ - \ \frac{t_1 - t_0}{(t_2 - t_0)(t_2 - t_1)}\, F_2 \right) \\ g_i^L \ &= \ -\frac{t_i - t_{i-1}}{(t_i - t_{i-2})(t_{i-1} - t_{i-2})}\, F_{i-2} \ + \ \frac{1}{t_{i-1} - t_{i-2}}\, F_{i-1} \ - \ \frac{1}{t_i - t_{i-2}}\, F_i\,, \qquad 1 < i \leq N \\ g_i^R \ &= \ \frac{1}{t_{i+1} - t_{i-1}}\, F_{i-1} - \frac{1}{t_{i+1} - t_i}\, F_i \ + \ \frac{t_i - t_{i-1}}{(t_{i+1} - t_{i-1})(t_{i+1} - t_i)}\, F_{i+1}\,, \qquad 1 \leq i < N \\ g_N^R \ &= \ \frac{1}{2}\left(\frac{t_N - t_{N-1}}{(t_N - t_{N-2})(t_{N-1} - t_{N-2})}\, F_{N-2} \ - \ \frac{1}{t_{N-1} - t_{N-2}}\, F_{N-1} \ + \ \frac{1}{t_N - t_{N-2}}\, F_N \right) \end{aligned}$$

So to calculate the gradients of the g_i^L's and g_i^R's, it suffices to fill the coordinates with the coefficients in front of the corresponding R_i, the others being zero.

7.2 Generalized Tension B-Spline

FlexYCF includes generalized tension splines. While they share a collection of knots with curves that have classic interpolation methods presented above, they do not manipulate directly the knot-points. Indeed, their natural representation is functional, the unknowns of the tension splines being the coefficients of each basis function. There is one basis function per knot, and this function is localized around the knot.

Tension splines are a generalization of cubic splines and allow to adjust the curvature of the curve by changing the tension parameter on each interval between knots, providing greater flexibility than other interpolation methods.

Tension splines have many interesting properties. For our purpose, suffice to mention that when the tension parameter σ_j tends to 0, the tension spline degenerates to a regular cubic spline on the interval $[t_j, t_{j+1}]$. When $\sigma_j \to +\infty$, the interpolation reduces to a linear interpolation on this interval.

Tension splines formulae can be expressed generically in terms of a collection of two related functions, the so-called B-spline *defining functions* Ψ and Φ. Assuming there are M knots placed at times $t_1 < \ldots < t_M$, the interpolation at time $t \in [t_j, t_{j+1})$ is:

$$g(t) = \sum_{k=j-1}^{j+2} b_k \, B_{k-2,4}(t)$$

where b_k are the unknown coefficients to solve and the basis functions $B_{j,4}$ are defined recursively by:

$$B_{j,2} \;=\; \begin{cases} \Psi''(t), & t_j \le t < t_{j+1} \\ \Phi''(t), & t_{j+1} \le t \le t_{j+2} \\ 0 & \text{otherwise} \end{cases}$$

and

$$B_{j,k} \;=\; \Lambda_{j,k-1}(t) - \Lambda_{j+1,k-1}(t), \quad k = 3,\, 4,$$

where:

$$\Lambda_{j,k}(t) \;=\; \begin{cases} 0 & t < t_j \\ \int_{t_j}^{t} B_{j,k}(u)\,du \,/\, \int_{t_j}^{t_{j+k}} B_{j,k}(u)\,du, & t_j \le t \le t_{j+k} \\ 1 & \text{otherwise} \end{cases}$$

$B_{j,4}$ can be shown to be positive on $[t_j, t_{j+4})$ and null elsewhere, ensuring locality of such basis functions.

Note that there are six extra knots added ($t_{-2} < t_{-1} < t_0$ before t_1 and $t_{M+1} < t_{M+2} < t_{m+3}$ after t_M) and two (unknown) coefficients b_0 and b_{M+1}. These are introduced for formula consistencies and boundary conditions.

By setting $z_j := \Psi_{j-1}(t_j) - \Phi_j(t_j), \quad z'_j := \Psi'_{j-1}(t_j) - \Phi'_j(t_j), \quad y_j := t_j - z_j \,/\, z'_j,$

It can be shown that:

$$\begin{aligned} g(t) \;=\; & b_{j-1} \frac{\Phi_j(t)/z'_j}{y_j - y_{j-1}} + b_j \left(1 - \frac{t - y_j + \Phi_j(t)/z'_j}{y_{j+1} - y_j} - \frac{\Phi_j(t)/z'_j}{y_j - y_{j-1}} + \frac{\Psi_j(t)/z'_{j+1}}{y_{j+1} - y_j} \right) \\ & + b_{j+1} \left(\frac{t - y_j + \Phi_j(t)/z'_j}{y_{j+1} - y_j} - \frac{\Psi_j(t)/z'_{j+1}}{y_{j+2} - y_{j+1}} - \frac{\Psi_j(t)/z'_{j+1}}{y_{j+1} - y_j} \right) + b_{j+2} \frac{\Psi_j(t)/z'_{j+1}}{y_{j+2} - y_{j+1}} \end{aligned}$$

Thus we can calculate interpolation for any pair of defining functions. Also, the gradient $\nabla_b g(t)$ of g at time t, relative to the unknowns $(b_0, \ldots, b_{M+1})$ is directly read

from the formula. One would have to rely on numerical analysis to invert a system of linear equations if the unknowns were the knot-point values as for UKP curves.

Fore more details over the derivation of the formulae, the interested reader is invited to consult [**?**].

Boundary conditions.

As of now, natural spline boundary conditions are hardcoded for tension splines in FlexYCF. See [**?**] for the exact formula.

B-Spline Defining Functions

The following three families of defining functions were implemented in `FlexYCF`:

1. Exponential B-spline defining functions

$$\begin{aligned}\Psi_j(t) &= \frac{(t-t_j)^3 \ \exp\big(-\sigma_j\,(t_{j+1}-t)\big)}{h_j\,(6+6\,\sigma_j\,h_j+\sigma_j\,h_j^2)}\\ \Phi_j(t) &= \frac{(t_{j+1}-t)^3 \ \exp\big(-\sigma_j\,(t-t_j)\big)}{h_j\,(6+6\,\sigma_j\,h_j+\sigma_j\,h_j^2)}\end{aligned}$$

2. Rational B-spline defining functions (with linear denominator)

$$\begin{aligned}\Psi_j(t) &= \frac{(t-t_j)^3}{h_j\,\big(1+\sigma_j\,(t_{j+1}-t)\big)\,(6+6\,\sigma_j\,h_j+2\,\sigma_j\,h_j^2)},\\ \Phi_j(t) &= \frac{(t_{j+1}-t)^3}{h_j\,\big(1+\sigma_j\,(t-t_j)\big)\,(6+6\,\sigma_j\,h_j+2\,\sigma_j\,h_j^2)}\end{aligned}$$

3. Hyperbolic B-spline defining functions

$$\begin{aligned}\Psi_j(t) &= \frac{\sinh\big(\sigma_j\,(t-t_j)\big)-\sigma_j\,(t-t_j)}{\sigma_j^2\,\sinh(\sigma_j\,h_j)},\\ \Phi_j(t) &= \frac{\sinh\big(\sigma_j\,(t_{j+1}-t)\big)-\sigma_j\,(t_{j+1}-t)}{\sigma_j^2\,\sinh(\sigma_j\,h_j)}\end{aligned}$$

7.3 Composite Curve

Composite curves allow to "glue" two interpolation curves sharing the same unknowns (ideally, sharing the same knots) at a so called separation point. When a composite curve must interpolate or compute the gradient at a point that at the left of the separation point, it delegates this task to one curve, and when this point is right to the separation point, it delegates to the other.

Rationale: composite curves were introduced to allow (quasi-)linear[4] interpolation on the short-end of the 3M curve, namely up to the last Futures maturity, while allowing a smooth interpolation afterwards.

[4] that is to say, straight-line

Noticing that the 3M forward rate starting at date S and paying at date T verifies:

$$\begin{aligned} F(S,T) &= \frac{1}{\delta(S,T)} \Big(\frac{P(S)}{P(T)} - 1 \Big) \\ &= \frac{1}{\delta(S,T)} \Big(\exp \big(\int_S^T f(t)\, dt \big) - 1 \Big) \\ &\simeq \frac{1}{\delta(S,T)} \big(\int_S^T f(t)\, dt \big) \;=\; \frac{1}{\delta(S,T)} \big(F(T) - F(S) \big) \end{aligned}$$

where the year fraction $\delta(S,T) \simeq 0.25$, the piecewise approximate linearity of the forward rate is ensured by taking straight-line interpolation for F or flat right or flat left interpolation on the instantaneous forward rate curve f.

Important remark: A restriction on composite curves is that at the moment they cannot mix a UKP curve and a tension spline. This is because the type of the unknowns are not the same. So even if interpolation could be possible on each time interval, the bridging between the two around the separation point is non-trivial and involves equations that depend on the interpolation and defining functions used.

This limitation could be addressed in the future by representing the unknowns of tension splines in the same way as the UKP curves. This would complicate the solving for the tension spline unknowns inverting a linear system, which would result in a performance hit. Of course, the unknowns could remain as they are now when the tension spline is used as a stand alone curve. However the current limitation has to be nuanced by the fact that tension splines already allow fine-grain control of the smoothness on each knot-delimited interval, from linear to cubic/smooth, which is the reason why composite curves were introduced in the first place.

7.4 Extrapolation Methods

As for interpolation methods, we assumed we have a series of knot-points $(x_i, y_i)_{\,0 \le i \le N}$.

7.4.1 Flat Extrapolation

Flat Left Extrapolation

If $x < x_0$, flat left extrapolation is given by:

$$f(x) \;=\; y_0$$

and the gradient for such an x is

$$\nabla f(x) \;=\; (\,1,\, 0,\, \ldots,\, 0)$$

Flat Right Extrapolation

If $x >= x_N$, flat right extrapolation is given by:

$$f(x) \;=\; y_N$$

and the gradient for such an x is

$$\nabla f(x) \;=\; (\,0,\,\ldots,\,0,\,1\,)$$

7.4.2 Straight Line Extrapolation

There must be at least two knot-points in the curve to perform a straight line extrapolation.

Left Straight Line Extrapolation

If $x < x_0$, left straight line extrapolation is given by:

$$f(x) \;=\; \big(1 - \lambda_1(x)\big)\, y_0 \;+\; \lambda_1(x)\, y_1$$

The gradient is easily calculated as:

$$\nabla f(x) \;=\; \big(\,1 - \lambda_1(x),\, \lambda_1(x),\, 0,\,\ldots,\,0\,\big)$$

Right Straight Line Extrapolation

if $x >= x_N$, right straight line extrapolation is:

$$f(x) \;=\; \big(1 - \lambda_N(x)\big)\, y_{N-1} \;+\; \lambda_N(x)\, y_N$$

The gradient is

$$\nabla f(x) \;=\; \big(\,0,\,\ldots,\,0,\,1 - \lambda_{N-1}(x),\, \lambda_N(x)\big)$$

7.5 Curve Formulation

The curve formulation applies a transformation on the discount factor, so that the an interpolation in the transformed quantity can result in desired properties on the curve solved for. The available curve formulations are detailed below.

7.5.1 Discount

The interpolation is done directly on the discount factors $df(t)$.

Note that independently of the interpolation, we must ensure that $df(0) = 1$.

7.5.2 LogFvf

The interpolated quantity is $Y(t)$, where $Y(t)$ is:

$$df(t) \;=\; e^{-Y(t)} \tag{7.9}$$

Note that independently of the interpolation, we must ensure that $Y(0) = 0$.

7.5.3 Spot Rate

The interpolated quantity is $R(t)$, where $R(t)$ is:

$$df(t) \;=\; e^{-\,R(t)\,t} \tag{7.10}$$

Chapter 8

Future Convexity Model

Future convexity adjustments are calculated externally and provided as input to the yield curve builder. This section describes the model used to calculate the future convexity spreads, the details of their calculation and reviews calibration choices.

8.1 The model

The model in which the futures convexity adjustments are calculated is the LIBOR Market Model.

Let $(T_k)_{0\leq k\leq 40}$ be the tenor structure of the futures. For the sake of simplicity, we assume that the maturity of any future perfectly matches the expiry date of the next one.

In this setting, each forward rate F_k expiring at T_{k-1} and maturing at T_k is assumed to follow the dynamics:

$$dF_k(T) \;=\; F_k(t)\,\sigma_k(t)\,dZ_k^{T_k}(t), \quad t \leq T_{k-1}$$

where $Z_k^{T_k}$ is a standard Brownian motion under the T_k-forward measure $\mathbb{Q}^{T_k}$ and the instantaneous correlation between Brownian motions is given by $d\langle Z_i^{T_k}, Z_j^{T_k}\rangle(t) \;=\; \rho_{i,j}\,dt$.

8.2 Calculation

Assuming a unit notional amount, a futures contract is worth $1-L(S,T)$ at time $S<T$.

It is known ([**?**]) that for the price at $t=0$ of such a futures contract is given by:

$$\begin{aligned}\texttt{Fut}\,(0;S,T) \;&=\; \mathbb{E}^*\big[\,1-L(S,T)\,\big]\\ &=\; 1 \;-\; \mathbb{E}^*\big[\,F(S;S,T)\,\big]\end{aligned} \tag{8.1}$$

The peculiarity of Futures pricing is that the forward rate expectation refers to the risk-neutral measure $\mathbb{Q}^*$ and not its natural T-forward measure under which it is a martingale. This will oblige us to express the dynamics of $F(\,.\,;S,T)$ under $\mathbb{Q}^*$. The pricing of a futures contract comes down to calculating the so called implied *futures LIBOR* $L^f(S,T) := \mathbb{E}^*[\,L(S,T)\,]$.

$F_k(t) := F_k(t; T_{k-1}, T_k)$ is the *forward rate* at time t. S_k is the *convexity spread* over the k-th forward rate to retrieve the futures rate (as of today: $t = 0$). $\texttt{Fut}_k$ is the price of the k-th futures at time $t = 0$. The *implied futures LIBOR* rate of the k-th futures, $\texttt{iL}_k$ is defined as the expectation of the k-th forward index under the risk neutral measure $\mathbb{Q}^*$:

$$\texttt{iL}_k \;:=\; \mathbb{E}^*[\, F_k(T_{k-1})\,]$$

We approximate it by calculating $\mathbb{E}^L[\, F_k(T_{k-1})\,]$, the expection under the *spot LIBOR measure* $\mathbb{Q}^L$. Roughly speaking, this amounts to neglecting the volatility of the instantaneous forward rate (see [**?**] pp 218-219). The numéraire associated with $\mathbb{Q}^L$ is the discretely rebalanced bank-account asset $B_d(t)$ defined as:

$$B_d(T) \;:=\; \frac{P(t, T_{\beta(t)-1})}{\prod_{i=0}^{\beta(t)-1} P(T_{i-1}, T_i)},$$

where $\beta(t) = 1$ if and only if $T_{m-2} < t \leq T_{m-1}$.

The spot LIBOR measure dynamics of forward LIBOR rates in in the LIBOR market model is given by:

$$dF_k(t) \;=\; \sigma_k(t)\, F_k(t) \sum_{i=\beta(t)}^{k} \rho_{i,k} \frac{\tau_i F_i(t)}{1 + \tau_i F_i(t)}\, \sigma_i(t)\, dt \;+\; \sigma_k(t)\, F_k(t)\, dZ_k^L(t), \quad t \leq T_{k-1},$$

where the Z_k^L are standard Brownian motions under $\mathbb{Q}^L$ and τ_i the year fraction between T_{i-1} and T_i (so roughly 0.25 in our case).

Freezing the drift part gives:

$$dF_k(t) \;=\; \sigma_k(t)\, F_k(t) \sum_{i=\beta(t)}^{k} \rho_{i,k} \frac{\tau_i F_i(0)}{1 + \tau_i F_i(0)}\, \sigma_i(t)\, dt \;+\; \sigma_k(t)\, F_k(t)\, dZ_k^L(t)$$

From which, using Ito formula:

$$F_k(t) \;=\; F_k(0)\; \exp\left\{ \int_0^t \sum_{i=\beta(s)}^{k} \rho_{i,k}\, \frac{\tau_i\, F_i(0)}{1 + \tau_i\, F_i(0)} \sigma_i(s)\, \sigma_k(s)\, ds \right\} \mathcal{E}(X_k)_t \qquad (8.2)$$

where

$$\mathcal{E}(X_k)_t := \exp\left(X_k(t) - \frac{1}{2}\, \langle X_k \rangle(t) \right)$$

is the Doléans-Dade exponential of the stochastic process X_k defined by $X_k(t) := \int_0^t \sigma_k(s)\, dZ_k^L(s)$, and $\langle X_k \rangle(t) = \int_0^t \sigma_k^2(s)\, ds$ denotes its quadratic variation.

Note that the lower bound of the discrete summation in formula 8.2 is time-dependent.

Defining

$$\psi_{i,k}(s) \;:=\; \rho_{i,k}\, \frac{\tau_i\, F_i(0)}{1 + \tau_i\, F_i(0)}\, \sigma_i(s)\, \sigma_k(s),$$

we calculate:

$$\begin{aligned}
\int_0^{T_{k-1}} \sum_{i=\beta(s)}^{k} \psi_{i,k}(s)\,ds &= \int_0^{T_{k-1}} \sum_{m=1}^{k} \mathtt{1}_{(T_{m-2},T_{m-1}]}(s) \sum_{i=m}^{k} \psi_{i,k}(s)\,ds \\
&= \sum_{m=1}^{k} \int_{T_{m-2}}^{T_{m-1}} \sum_{i=m}^{k} \psi_{i,k}(s)\,ds \\
&= \sum_{m=1}^{k} \sum_{i=m}^{k} \int_{T_{m-2}}^{T_{m-1}} \psi_{i,k}(s)\,ds \\
&= \sum_{i=1}^{k} \sum_{m=1}^{i} \int_{T_{m-2}}^{T_{m-1}} \psi_{i,k}(s)\,ds \\
&= \sum_{i=1}^{k} \int_0^{T_{i-1}} \psi_{i,k}(s)\,ds \\
&= \sum_{i=1}^{k} \rho_{i,k}\,\frac{\tau_i\,F_i(0)}{1+\tau_i\,F_i(0)} \int_0^{T_{i-1}} \sigma_i(s)\,\sigma_k(s)\,ds
\end{aligned}$$

Obviously[1], $\mathbb{E}^L[\mathcal{E}(X_k)_t] = 1$ for all t, so the approximation of the k-th implied futures LIBOR rate is:

$$\mathbb{E}^L[F_k(T_{k-1})] = F_k(0)\,\exp\left\{ \sum_{i=1}^{k} \rho_{i,k}\,\frac{\tau_i\,F_i(0)}{1+\tau_i\,F_i(0)} \int_0^{T_{i-1}} \sigma_i(s)\,\sigma_k(s)\,ds \right\}$$

Given that the implied Futures LIBOR verifies:

$$\begin{aligned}
\mathtt{iL}_k &= 1 - \mathtt{Fut}_k \\
&= F_k(0) + \mathtt{S}_k \\
&\simeq F_k(0)\,e^{\psi_k}
\end{aligned}$$

where

$$\psi_k = \sum_{i=1}^{k} \rho_{i,k}\,\frac{\tau_i\,F_i(0)}{1+\tau_i\,F_i(0)} \int_0^{T_{i-1}} \sigma_i(t)\,\sigma_k(t)\,dt$$

we calculate the k-th convexity spread at date T_k as the following aproximation:

$$\mathtt{S}_k \simeq F_k(0)\,(e^{\psi_k} - 1)$$

The convexity spreads between dates T_k is are linearly interpolated.

[1] $\mathcal{E}(X_k)$ is a $\mathbb{Q}^L$-martingale.

8.3 Calibration

The LIBOR market model used to calculate Futures convexity spreads is calibrated to a tenor structure of ATM caplet Black implied volatilities.

Using the Levenberg-Marquardt solver introduced in 4.5.2, the calibration minimizes the following least squares problem:

$$\sum_i \left(\sqrt{\frac{1}{T_{i-1}} \int_0^{T_{i-1}} \sigma_i^2(s;\theta)\, ds} - v_i \right)^2,$$

where $v_{T_{i-1}}$ are the T_{i-1}-caplet (Black) implied volatilities we want to fit and θ is the vector of parameters. Its dimension must be less than numbers of tenor structure dates.

In practice, the implied volatilities of the caplet with the following maturities are used to calibrate the model: 3M, 6M, 1Y, 2Y, 3Y, 5Y, 7Y, 10Y, 15Y, 20Y and 30Y.

8.3.1 Instantaneous Volatility Structure

The instantaneous volatility used is the so-called Rebonato volatility:

$$\sigma_i(t;\theta) := \left[a\,(T_{i-1} - t) + d \right] e^{-b\,(T_{i-1}-t)} \; + \; c$$

where $\theta := (a, b, c, d)$ are the parameters to be calibrated

Other instantaneous volatility structures could be used, but the proposed volatility structure compared favourably against others in terms of fitting.

8.3.2 Instantaneous Correlation

The so-called Rebonato correlation is hard-coded in the Futures convexity model.

It is given by:

$$\rho_{i,j} \; = \; \exp(-\beta\,|T_i - T_j|\,)$$

The parameter of the Rebonato correlation is not calibrated but to be chosen by the end-user. We have found that $\rho \simeq 10\%$ gives an acceptable fit in practice.

Chapter 9

Equivalent Curve

9.1 Overview

Conceptually, the algorithm can be described as follows: given a source model, calculate the rates of some target instruments such that they are priced at par in this source model.

9.2 Instrument Par Rates

We now explain how the instruments' par rates are calculated: each pricing formula of the instruments depends linearly on its market quote. It is thus straightforward to calculate them as a function the other model-dependent components of instruments. In each of the following equations, the term on the right is calculated using the source curve.

9.2.1 Cash

The par cash rate R of a cash instrument is calculated as:

$$R = F(S, T),$$

where:

- S is the start date of the cash instrument
- T is its end date
- $F(S, T)$ is the $T - S$ month forward rate that sets at date S and pays at date T

9.2.2 Future

Let $\texttt{iL}(S, T)$ be the *implied Futures LIBOR rate* of a Futures whose start date and end date are respectively S and T. $\texttt{iL}(S, T)$ is the rate such that the market quote of the futures is $1 - \texttt{iL}$. It can be priced at par as:

$$\texttt{iL}(S, T) \;=\; F(S, T) + \texttt{Cvx}(T),$$

where $\texttt{Cvx}(T)$, the convexity adjustment at date T, can either be a constant value directly specified in the target curve table or calculated if the source curve tables has the settings to build a convexity model.

9.2.3 Swap

It is well-known that the swap rate R is at par when calculated as:

$$R = \frac{\texttt{FltLeg}}{\texttt{FxdLeg}},$$

where `FltLeg` and `FxdLeg` are the floating leg and the fixed leg of the swap, respectively.

9.2.4 Cross Currency Basis Swap

The par spread X of a cross-currency swap is calculated as:

$$X = \frac{df_F(T_0) - df_F(T_n) - \texttt{FltLeg}}{\texttt{FxdLeg}},$$

where the addition of `FltLeg` and `FxdLeg` is equal to the present value of the foreign leg of the instrument, without exchange of notional. In other words, it is the decomposition of the discounted cash-flows made of the sum a forward rate and a fixed spread as a floating leg of discounted forward rates and a fixed leg.

9.2.5 Tenor Basis Swap

The par basis spread X of a tenor basis swap is:

$$X = \frac{\texttt{FltLeg}_{\tau_2} - \texttt{FltLeg}_{\tau_1}}{\texttt{FxdLeg}_{\tau_1}},$$

where $\tau_1 < \tau_2$, such that the tenor spread is paid over the forward rates of the leg of tenor τ_1, which can be decomposed as a pure floating leg of discount forward rates $\texttt{FltLeg}_{\tau_1}$ and a fixed leg $\texttt{FxdLeg}_{\tau_1}$.