**iLib BuildCurve function**

Creates a named zero coupon discount curve. As many discount curves as PC memory allows can be created.

This function returns the name of the curve in the Excel cell. All other discount curve functions require this name. The cell containing the name should be passed to these functions, rather than coding the name as a string in the Excel formula. Then all dependent cells will update if any cells that the BuildCurve function depends on are changed.

Syntax:  
**BuildCurve(***Name, Currency, TodaysDate, Instruments, [CashDCT], [CashBDC], [CashDaysToSpot], [CashCentres], [SwapsDCT], [SwapsBDC], [SwapsDaysToSpot], [SwapsCentres], [CurrencyPair], [FxSpotRate], [SecondCurveName], [InterpolationMethod], [DecimalPlaceAccuracy], [InterolateParRates], [OverrideShortRates], [NoSensitivities]***)**

*Name* is the name to be given to this curve.

*Currency* is the currency code of the curve to be built.

*TodaysDate* is todays date. This will be rolled forward to the next business day of the default centre associated with *Currency* if this is not a business day for that centre.

*Instruments* is an Excel range with each row containing:

*Instrument,Maturity,CouponPeriod,RateOrPrice*

*Instrument* is either: 'Cash','Swap','Fra','FX' or a valid futures contract code as described on the control file.  
Abbreviations of the above are also translateable.

*Maturity* is one of the following:

|  |  |
| --- | --- |
| O/N | Overnight |
| T/N | Tom/Next |
| S/N | Spot/Next |
| nD | n Days from the spot date. BDC does not apply to nD. |
| nW | n Weeks from the spot date. BDC does not apply to weeks. |
| nM | n Months from the spot date. BDC does apply to months. |
| nY | n Years from the spot date. BDC does apply to years. |
| xVy | near date is x months from cash spot date, far date is y months from cash spot date. Would normally be used to describe FRA maturity, but will work with all instruments except Futures. |
| mmmyy or mmmyyyy or mm/yy or mm/yyyy or fy or 01/mm/yyyy | all describe the futures delivery month and year.Refer to function Futures/DeliveryDate for more details. |

CouponFreq is only required for instrument type Swaps.

Monthly  
Quarterly  
Semi-Annual  
Annual.  
Most abbreviations of the above can be translated.

RateOrPrice is the interest rate, where 1 = 100% and .05 = 5%, or  
a futures price, such as 95.00 or  
an FX forward price in points, such as -130.5

*Instruments* examples:

|  |  |  |  |
| --- | --- | --- | --- |
| Cash | o/n |  | 5.45% |
| Cash | t/n |  | 5.45% |
| Cash | 1w |  | 5.56% |
| Cash | 6m |  | 5.56% |
| Fra | 6v9 |  | 5.58% |
| Fra | 9v12 |  | 5.60% |
| Swap | 2y | Semi | 5.66% |
| Swap | 3y | Semi | 5.72% |
| Swap | 5y | Semi | 5.80% |

*CashDCT* is the cash day count type. Refer function DCF in Dates section for details. This parameter is used for Cash, Fra and FX instruments. If not supplied, defaults to value specified by currency in the control file. Note that Futures instrument does not use this parameter but rather takes the value specified for the contract in the iLib control file.

*CashBDC* is the cash business day convention used for instruments Cash, Fra and FX. Refer function FarDate in Dates section for details. If not provided, defaults to the value specified by currency in the iLib control file.

*CashDaysToSpot* is the number of business days from *TodaysDate* to cash spot date. Used for Cash and Fra instruments (FX always takes two days to spot). Defaults to the value specified by currency in the iLib control file.

*CashCentres* is the one or more financial centres which the function should take into account when checking for bank holidays for Cash and Fra instruments. If more than one centre is to be specified, separate the centres with commas. Defaults to the value specified by currency in the iLib control file.

*SwapsDCT* is the Swaps day count type. Refer function DCF in Dates section for details. This parameter is used for Swaps instruments only. If not supplied, defaults to value specified by currency in the control file.

*SwapsBDC* is the cash business day convention used for Swaps instruments only. Refer function FarDate in Dates section for details. If not provided, defaults to the value specified by currency in the iLib control file.

*SwapsDaysToSpot* is the number of business days from *TodaysDate* to Swaps spot date. Defaults to the value specified by currency in the iLib control file.

*SwapsCentres* is the one or more financial centres which the function should take into account when checking for bank holidays for Swaps instruments. If more than one centre is to be specified, separate the centres with commas. Defaults to the value specified by currency in the iLib control file.

*CurrencyPair* is mandatory if FX instruments are included in the curve, and not used otherwise. It is entered in the form "ccy1/ccy2", for example "USD/DEM". Currencies should be specified in the order that represents the spot rate quoting convention. USD/DEM, for instance, would indicate that the spot rate is quoted in terms of DEM per 1 USD. GBP/USD indicates quote in terms of USD per 1 GBP.

*FxSpotRate* is mandatory if FX instruments are included, and not used otherwise. See *CurrencyPair* for quoting conventions.

*SecondCurveName* is the name of the "other" curve from which FX yields will be implied. This curve has to be constructed and available previously. It would typically be, but doesn’t necessarily have to be, USD.

*InterpolationMethod* is a code dictating which of the three available interpolation methods to use:  
1 = Log linear interpolation of discount factors  
2 = Linear interpolation of continuously compounded interest rates  
3 = Linear interpolation of daily compounded interest rates  
Refer to the General Notes section below for details.

*DecimalPlaceAccuracy* is the accuracy to which the discount factors are calculated. This option is only effective when the *InterpolateParRates* option is false.

*InterolateParRates* dictates the method used to construct the curve. If this option is set to true, the curve is built using the "linear interpolation of par rates with bootstrapping" method, otherwise the "constant forward rates" method is used. Refer to the section on ‘How the curve is built’ below.

*OverrideShortRates* dictates the handling of the short dated instruments if futures and/or fras are used in the curve. If set to true, the futures/fra rate overrides all interest rates from it’s near date to far date. Refer to the section on ‘How the curve is built’ below.

*NoSensitivities* switches the calculation of discount factors for sensitivities on and off. If set to false, the sensitivity function can’t be used with this curve but calculation times will be a little quicker.

**How the curve is built:**

iLib will construct curves in either of two methods, linear interpolation of par rates with bootstrapping or constant forward rates.

**Linear interpolation of par rates**

This methodology uses the traditional approach of bootstrapping discount factors from today our to the furthest date:

http://www.wfe.com.au/images/BuildCurve1.gif

This process requires that all discount factors for all swap coupons except for the last one are known. BuildCurve will fail if rates are not supplied out to the near date of any instrument, or to the first swap coupon date. So, for instance, if the cash spot days to spot is 2, a cash 1m rate is supplied and the cash T/N rate is not supplied, BuildCurve will fail because it will not be able to calculate a discount factor for the 1m Cash instrument value date. Similarly, if a futures contract is supplied but the cash contracts do not extent to the futures delivery date, the function will fail.

BuildCurve will always build this type of curve using semi annual swap coupons. If swap rates other than semi annual are supplied, they will be converted to semi annual using a simple compounding approach, for example:

http://www.wfe.com.au/images/BuildCurve2.gif

The method requires that a swap par rate is known for each coupon date. We can typically observe the yearly swap rates in the market (2yr, 3yr etc) but equally typically cannot observe the mid year points (2.5yr, 3.5 yr etc). For all unknown swap par rates, we linearly interpolate the known rates.

**Constant forward rates**

The linear interpolation of par rates approach presents some difficulties. We most often do not have swap rates for all coupon dates and the linear interpolation of par rates does not necessarily give us a good result. It yields a curve with very irregular forward rates, for instance.

Market practice has moved towards logarithmic interpolation of the discount factors themselves to calculate the unknown but necessary discount factors in the bootstrapping process. If all but the last discount factor is known, that last discount factor is calculated through simple analytic process. If more than one discount factor is unknown, a numerical process is used. The last DF is guessed, the intervening discount factors are logarithmically interpolated, and the PV of the cash flows calculated. The final discount factor that provides a zero PV is then arrived at numerically by Newton-Raphson approximation.

This methology happily yields equal continuously compounded forward interest rates either side of the interpolated discount factors.

The approach provides a very flexible and powerful curve construction methodology. If, for instance, a 3 month cash rate is known but O/N and T/N rates aren’t, the curve will still be built and O/N and T/N discount factors interpolated. In the extreme case, a curve can be constructed using only a 20 year swap rate. One significant benefit for the Australian market is that an accurate curve can be constructed with the mix of quarterly and semi-annual swap rates typically quoted in the interbank market.

**General Notes**

The curve is always built with TodaysDate having a discount factor of 1.

Interpolation of discount factors is by one of three similar approaches:

LogLinear interpolation is generally considered the best approach:

BuildCurve3.gif (1381 bytes)

Linear interpolation of continously compounding interest rate is favoured by some institutions:

BuildCurve4.gif (1162 bytes)

BuildCurve5.gif (1170 bytes)

BuildCurve6.gif (1852 bytes)

BuildCurve7.gif (1236 bytes)

BuildCurve8.gif (1259 bytes)

BuildCurve9.gif (1656 bytes)

**Instrument Priorites and Short Date Overrides**

All instruments are sorted by Far Date. If there are overlaps in interest rate periods, the following priorities are applied:

FX overrides Swaps and Cash rates of the same maturity.

Swaps override cash rates of the same maturity.

The treatment of Futures and Fras depends on the *OverrideShortRates* option. If *OverrideShortRates* is true, the cash, FX or swap rate nearest the future/fra will be "stubbed", meaning a discount factor will be calculated using that rate to its maturity date, and then that discount factor will be interpolated back to the near date of the future/Fra. The future/Fra rate will then be applied to the whole period of the future/Fra.

This approach tends to lead to problems when the futures contract approaches expiry. The contract will typically be out of step with the cash market due to market imperfections, and on the day of expiry a sharp jump in short term rates is seen as the future dissappears and the 1, 2 and 3 month cash rates are again used.

To get around this problem, if *OverrideShortRates* is false, the first future/Fra does not override all rates between its near and far dates but merely adds another discount factor to the curve at its far date.

**Notes on Futures**

Futures dates can be entered in a variety of formats, as specified in the *Instruments* section. The futures interest rate period is deemed to be from Delivery Date of the future to the tenor specified in the Control File for that contract. So, for instance, and Sep 97 Eurodollar with tenor specified as "3m" and date type 1 (delivery on 3rd Wednesday of contract month), will have a near date of 17 September 1997 and far date of 17 December 1997. If the tenor were specified as "90d", the far date would have been 16 December 1997.

The rate immediately preceeding the first futures contract will be ‘stubbed’ to the delivery date of the first futures contract. In the above example, if a cash rate had a maturity of 15 November 1998, the discount factor for 17 September would be calculated via the specified interpolation methodology and stored. The futures interest rate would then apply from that discount factor.

**Notes on FRAs**

A FRA is treated in essentially the same manner as a future, taking into consideration the priority issues mentioned above. Near and Far dates are derived using the *CashDCT* parameter. A 3v6 Fra will therefore have a near date equivalent to *Dates.NearDate(TodaysDate, CashDBC, CashDaysToSpot, CashCentres,*"3m"*)* and a far date equivalent to *FarDate(TodaysDate, CashBDC, CashDaysToSpot, CashCentres,* "3m")

**Notes on FX**

An FX discount curve is constructed by implying yields from the FX spot and forward points and the other currency’s discount curve. These yields are then applied as would be cash rates over the appropriate near and far FX value dates. FX points will typically, but not necessarily, be out of the FX spot date.

FX instruments can be used in conjunction with any mix of other instruments, such as futures and swaps, noting the priority issues discussed above.

Centres for holiday purposes always default to the default centre for each currency of the currency pair specified.

**Control File Parameters**

The following parameters in the ilib control file file effect the BuildCurve function:

**[Curve Options]**  
InterpolationMethod={1|2|3}  
DecimalPlaceAccuracy=*NumberDecimals*InterpolateParRates={*True|False*}  
OverrideShortRates={*True|False*}  
NoSensitivities={*True|False*}

These values will be used as default unless overriden using the BuildCurve parameters when it is called from Excel.

The iLib control file comes supplied with the following values (and will default to them unless the control file is changed):

InterpolationMethod=1}  
DecimalPlaceAccuracy=8InterpolateParRates=*False*  
OverrideShortRates=*False*  
NoSensitivities=*False*