**Assignment 2 Yield Curve Calibration**

# **Introduction**

There are 3 main objectives in this assignment

1. Understand the popular market instrument and its pricing
2. Understand the logic and sequence of yield curve calibration
3. Enhance coding skill, familiar with a small library set up, and able to contribute new functionality in existing library.

Students are required to spend fair amount of time to understand the library code thoroughly. Best place to start is to run “UnitTest.py”, through different test cases, familiar with market instrument pricing and curve functions. Understand how a basis swap is priced from the code, and how to use related curve, swap, cashflow class object.

# **Part 1: Yield Curve Calibration**

**Task 1:**

Run Calibration.py, in the main function, keep ‘*calib set 2 USD.LIBOR.6M’* block disabled, the existing code is to calibrate OIS and LIBOR3M curve from scratch, the input info for calibration is saved under “curve\_calibration.csv”.

No action is required, understand the calibration logic and code.

**Task 2:**

Implement *save\_calibration\_results(curve\_constructor, output\_file\_path\_name)* function, save calibration result from task 1 to output csv file, on top of original info from curves\_info\_dict, add 3 more columns, curve maturities in year fraction, curve zero rates, recomputed curve targets. These 3 columns can be taken directly or computed quickly from *“curve\_constructor”* class.

**Task 3:**

Take the calibration result saved from task 2 as input, calibrate LIBOR.6M curve and save result csv file. In main function, disable ‘calib set 1’ block, and enable ‘calib set 2’ block. Read class “*curveconstructor \_\_init\_\_”* function carefully, understand how to pass parent curves as input.

**Task 4:**

JPY\_calibration.csv contains all JPY curves and its building instruments with market quotes. Calibrate JPY domestic curves: JPY.TONAR, JPY.LIBOR.6M, JPY.LIBOR.3M

What o submit:

* Calibration.py, this contains “*save\_calibration\_results”* function; in main function, add similar code block calib set 3, calib set 4 etc for JPY curves calibration.
* USDLIBOR6M\_result.csv FROM task 4
* Write a simple word document, what is the sequence to calibrate JPY curves and why?
* Calibrated JPY curve results in a csv file

# **Part 2: Resettable Cross Currency Swap**

In a resettable notional cross currency swap, the notional of one leg resets on every coupon period based on prevailing FX exchange rate to rebalance the FX exposures in the transaction. The mark to market difference at each reset is settled from OTM to the ITM party.

**Task 1:**

Study the spreadsheet mtmxccyswap.xlsx, cross check notes and understand how pricing works for this payoff, especially the periodic notional exchange. No action is needed.

**Task 2:**

Implement new instrument in class XccySwap, save it in MTMXccySwap.py.

Especially you need to implement function compute\_mtm(), compute\_target\_rate()

Hint, you may need a new XccyLeg class to compute different types cashflow, notional exchange, coupon payment, intermediate mtm cashlow etc.

In UnitTest.py, please enable the test case *“test\_mtmsccyswap”,* after I run the new test case test\_mtmxccyswap(val\_date), I expect to see mtm 6,176,825.11, the same as spreadsheet.

FYI, the constructor in test case MTMXccySwap.XccySwap, which takes on the parameters with financial meaning below, ignore the default value.

*def \_\_init\_\_(self, val\_date, tenor, leg1\_freq, leg2\_freq, is\_basis\_swap, target\_rate., notional\_dom., notional\_for, leg1\_discount\_curve, leg1\_forward\_curve, leg2\_discount\_curve, leg2\_forward\_curve, fx\_spot, is\_notional\_exchange, is\_MTM):*

**Task 3:**

Write a new test case, test\_mtmxccyswap\_readcurve(val\_date)

The new test case is quite similar as the one in task 2, but instead of applying flat dummy curve, the next test case reads real curve information from an input csv file, you need to manually add 2 dollar curves and JPY.LIBOR.3M curve calibrated previously in the csv file, and add JPY.XCCY.DICS curve from JPYXCCYDISC\_result.csv in the same file. Construct 4 curves accordingly, and pass to mtmxccyswap object for pricing. Still the same 5y mtm xccy swap, and please print the same output.

*print('5Y mtm xccy swap mtm is:', str(basis\_swap.compute\_mtm()))*

*print('5Y mtm xccy implied margin is:', str(basis\_swap.compute\_target\_rate()))*

What o submit:

* MTMXccySwap.py, this contains new instrument implementation together with new leg implementation.
* Your new test case code “*test\_mtmxccyswap\_readcurve(val\_date)”*, put it in UnitTest.py
* Report mtm and implied margin from task 2 and 3.

(After I add your py file and run the same test case, I expect to see the same mtm and implied margin that you reported above).

* Write a paragraph, discuss the challenge you are facing in this assignment, and how you conquer in the end.