Cross-Currency Swap Leg2 Coupon Engine

Release 1.0.0

Yichi Zhang

Apr 24, 2020

BLOOMBERG

Author	Yichi Zhang
Version	1.0.0 of 2020/*03*/*15*
Dedication	To the Global ALM Team

CHAPTER

ONE

INTRODUCTION

This Cross-Currency Swap Leg2 Coupon Engine is designed and implemented by Yichi Zhang to provide an automated solution to calculate leg2 coupon instead of using Bloomberg Swap Manager.

This documentation includes Bloomberg setting, cross-currency swap leg2 coupon engine documentation, foreign spreads in local currency template documentation and ADB all-in template documentation.

For cross-currency swap leg2 coupon engine, user can link cross-currency swap information to the cross-currency swap leg2 coupon template, after running engine, it will output leg2 coupon value in the separate Excel file. The engine can be run in the office Bloomberg Terminal (Office version only).

For foreign spreads in local currency template and ADB all-in template, user can run Python script on the remote Bloomberg Terminal and laptop (WFH version) or in the office Bloomberg Terminal (Office version).

CHAPTER

TWO

TABLE OF CONTENT

2.1 Swap Curve Defaults Settings

2.1.1 Curve Settings

User need to set up curve setting in the Bloomberg account before running cross-currency swap leg2 coupon engine. Please *save* after making any change.

- In the Bloomberg, enter *{SWDF DFLT <GO>}*
  - **Curve Defaults**

    Pay=Mid / Receive=Mid

  - **Cross Currency Basis Defaults**

    Basis side always at mid

  - **Interpolation Method**

    3 - Step-function forward

  - **Brazilian Curve Interpolation Method**

    2 - Exponential

  - **Enable OIS Discounting/Dual-Curve Stripping** (Check)

2.1.2 Curve Number & Index

In the Bloomberg Swap Manager, CSA curve is used as discount factor in the cross-currency swap leg2 coupon calculation. However, Bloomberg Excel and Python API don't support to download discount factor by CSA curve number. Fortunately, discount factor can be downloaded by basis curve number and index, it has same value with CSA curve.

In the foreign spreads in local currency template, ADB all-in template and cross-currency swap leg2 coupon template, there is `Discount Factor BBG` tab showing these curve number and index information.

- **Basis Curve Number**

  Basis curve number is used to download discount factor through Bloomberg Excel API

- **Basis Curve Index**

  Basis curve index is used to download discount factor through Bloomberg Python API

- **CSA Curve Number**

  CSA curve number is used to get discount factor through Bloomberg Swap Manager

Currency	Basis Curve Number	Basis Curve Index	CSA Curve Number
USD	S42	YCSW0042	S42
SGD	S98	YCSW0098	S416
CAD	S135	YCSW0135	S401
AUD	S95	YCSW0095	S406
JPY	S97	YCSW0097	S404
HKD	S96	YCSW0096	S409
IDR	S227	None	S425
EUR	S92	YCSW0092	S403

2.1.3 How to Check Curve Number & Index

In the future, user may add new currency in the template and need to know curve number and index.

For example, user wants to add **GBP**

In the Bloomberg, enter *{SWPM -FXFX USD GBP <GO>}*

- **Basis Curve Number**

  Go to `5) Curves` tab, and check `Curve #`, the `GBP vs. USD basis` number is 91

- **Basis Curve Index**

Type `YCSW0091` in the blue search bar (number is same), **don't enter <GO>**, the `basis curve index` will be shown automatically in the `SECURITIES` tab. If not shown, it does not exist

- **CSA Curve Number**

Go to `5) Curves` tab, and check `Curve #`, the `GBP Cashflow CSA Curve` number is 405

The result is:

Currency	Basis Curve Number	Basis Curve Index	CSA Curve Number
GBP	S91	YCSW0091	S405

2.1.4 Curve Source Settings

In the initial setting, curve source is 1 or 8 randomly. However, curve can be only downloaded from source 8 through Bloomberg Excel API. Source 8 is the curve in function *{ICVS}*. In general, it would include more liquid instruments like futures or FRA in the middle section of the curve. And whenever possible, it will choose the instruments with same underlying (for example, 3M libor, futures with 3M libor, and swaps vs 3M libor) so as to make sure the entire curves is based on 3M libor. It will utilize our ICVS price to bootstrip the curve, so it would be more accurate in reflecting the interest rates.

Normally, CSA curve is based on OIS and basis curve. In order to get same discount factor value compared by CSA curve, user need to set up OIS and basis curve source into Source 8.

For USD, change `USD OIS` into Source 8

For any other currencies, say XYZ, normally, change `XYZ OIS` and `XYZ vs. USD basis` into Source 8

- In the Bloomberg, enter *{SWDF <GO>}*
  - **United States**

Curve Name	Number	Source
USD OIS	42	8

  - **Singapore**

Curve Name	Number	Source
Ohsh. SGD (vs. 6M SOR)	44	8
SGD vs. USD Basis	98	8

  - **Canada**

Curve Name	Number	Source
CAD OIS	147	8
CAD vs. USD Basis	135	8

  - **Australia**

Curve Name	Number	Source
AUD OIS	159	8
AUD vs. USD Basis	95	8

– **Japan**

Curve Name	Number	Source
JPY OIS	195	8
JPY vs. USD Basis	97	8

– **Hong Kong**

Curve Name	Number	Source
HKD OIS	145	8
HKD vs. USD Basis	96	8

– **Indonesia**

Curve Name	Number	Source
IDR OIS	158	8
IDR vs. USD Basis	227	8

– **Euro**

Curve Name	Number	Source
EUR OIS	133	8
EUR vs. USD Basis	92	8

2.1.5 How to Check Curve Source

In the future, user may add new currency in the template and need to know which curve should be changed source.

For example, user wants to add **GBP**

In the Bloomberg, enter *{SWPM -FXFX USD GBP <GO>}*

1. Go to `5) Curves` tab, and check `Curve #`, choose `GBP Cashflow CSA Curve`, the `Dependent Crvs` is 91, which is `GBP vs. USD basis` number, user need to change `GBP vs. USD basis` into Source 8
2. Change `GBP OIS` curve into Source 8

2.1.6 How to Add 30 Years Term

For some currencies which don't have 30 years term through Bloomberg Excel API, user can add it in the Bloomberg account. After saving, 30 years term will be shown automatically through Bloomberg Excel API.

For example: **SGD**

- Check `SGD vs. USB basis` curve number, it is 98. User may refer `How to Check Curve Number & Index` section
- In the Bloomberg, enter *{ICVS 98 <GO>}*
  - `Modes` -> `Customize` -> `Add Instrument`
  - Type `Ticker` and `Term` value
  - `Actions` -> `Save As` -> Name: SGD vs. USD Basis Modified
  - Click `Make Default`
  - Save

For more details, please check Bloomberg *{LPHP ICVS:0:1 2774392 <GO>}*

2.2 Day Count & Pay Frequency

The cross-currency swap leg2 coupon engine includes all day counts and pay frequency associated with all currencies. Hence, user can get leg2 coupon of cross-currency swap with any currencies.

2.2.1 Day Count

- ACT/365
- ACT/365.FIXED
- 30I/360
- 30U/360
- ACT/360
- 30/360
- ACT/ACT
- DU/252

For more details about description of day count, please check Bloomberg *{LPHP SWPM:0:1 2859968 <GO>}*

2.2.2 Pay Frequency

- Annual
- SemiAnnual
- Quarterly
- Monthly
- 28 Days

- Weekly

2.2.3 Currency Setup Configuration

The configuration file `CurrencySetupConfiguration.json` includes all currencies' day count and pay frequency

```
{
    "CAD": {
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/365"
    },
    "CNY": {
        "pay_freq":"Quarterly",
        "day_count":"ACT/365"
    },
    "KWD": {
        "pay_freq":"Annual",
        "day_count":"ACT/365"
    },
    "RON": {
        "pay_freq":"Annual",
        "day_count":"ACT/360"
    },
    "CHF": {
        "pay_freq":"Annual",
        "day_count":"30/360"
    },
    "COP": {
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/360"
    },
    "KZT": {
        "pay_freq":"Annual",
        "day_count":"ACT/365"
    },
    "RUB": {
        "pay_freq":"Annual",
        "day_count":"ACT/ACT"
    },
    "EUR": {
        "pay_freq":"Annual",
        "day_count":"30U/360"
    },
    "COU": {
        "pay_freq":"Quarterly",
        "day_count":"ACT/360"
    },
    "MMK": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "SAR": {
        "pay_freq":"Annual",
        "day_count":"ACT/360"
    },
    "GBP": {
```

(continues on next page)

(continued from previous page)

```
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/365"
    },
    "CRC": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "MNT": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "SGD": {
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/365"
    },
    "JPY": {
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/365.FIXED"
    },
    "CZK": {
        "pay_freq":"Annual",
        "day_count":"ACT/360"
    },
    "MUR": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "THB": {
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/365"
    },
    "SEK": {
        "pay_freq":"Annual",
        "day_count":"30/360"
    },
    "DKK": {
        "pay_freq":"Annual",
        "day_count":"30/360"
    },
    "MWK": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "TJS": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "USD": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "DOP": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "MXN": {
        "pay_freq":"28 Days",
```

(continues on next page)

(continued from previous page)

```
        "day_count":"ACT/360"
    },
    "TRY": {
        "pay_freq":"Annual",
        "day_count":"ACT/360"
    },
    "AED": {
        "pay_freq":"Annual",
        "day_count":"ACT/360"
    },
    "EGP": {
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/360"
    },
    "MYR": {
        "pay_freq":"Quarterly",
        "day_count":"ACT/365"
    },
    "TWD": {
        "pay_freq":"Quarterly",
        "day_count":"ACT/365"
    },
    "ARS": {
        "pay_freq":"Quarterly",
        "day_count":"ACT/360"
    },
    "GEL": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "MZN": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "TZS": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "AUD": {
        "pay_freq":"Quarterly",
        "day_count":"ACT/365"
    },
    "GHS": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "NGN": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "UAH": {
        "pay_freq":"Annual",
        "day_count":"ACT/ACT"
    },
    "AZN": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
```

(continues on next page)

(continued from previous page)

```
    },
    "HKD": {
        "pay_freq":"Quarterly",
        "day_count":"ACT/365"
    },
    "NOK": {
        "pay_freq":"Quarterly",
        "day_count":"30/360"
    },
    "UGX": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "BDT": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "HUF": {
        "pay_freq":"Annual",
        "day_count":"ACT/365"
    },
    "NZD": {
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/365"
    },
    "UZS": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "BGN": {
        "pay_freq":"Annual",
        "day_count":"30/360"
    },
    "IDR": {
        "pay_freq":"Quarterly",
        "day_count":"ACT/360"
    },
    "OMR": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "VND": {
        "pay_freq":"Quarterly",
        "day_count":"ACT/360"
    },
    "BHD": {
        "pay_freq":"Annual",
        "day_count":"ACT/360"
    },
    "ILS": {
        "pay_freq":"Annual",
        "day_count":"ACT/365"
    },
    "PEN": {
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/360"
    },
```

(continues on next page)

(continued from previous page)

```
    "XAF": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "BRL": {
        "pay_freq":"SemiAnnual",
        "day_count":"DU/252"
    },
    "INR": {
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/365"
    },
    "PHP": {
        "pay_freq":"Quarterly",
        "day_count":"ACT/360"
    },
    "XOF": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "CLF": {
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/360"
    },
    "ISK": {
        "pay_freq":"Annual",
        "day_count":"ACT/360"
    },
    "PKR": {
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/365"
    },
    "ZAR": {
        "pay_freq":"Quarterly",
        "day_count":"ACT/365"
    },
    "CLP": {
        "pay_freq":"SemiAnnual",
        "day_count":"ACT/360"
    },
    "KES": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "PLN": {
        "pay_freq":"Annual",
        "day_count":"ACT/ACT"
    },
    "ZMW": {
        "pay_freq":"SemiAnnual",
        "day_count":"30I/360"
    },
    "CNH": {
        "pay_freq":"Quarterly",
        "day_count":"ACT/360"
    },
    "KRW": {
```

(continues on next page)

(continued from previous page)

```
        "pay_freq":"Quarterly",
        "day_count":"ACT/365"
    },
    "QAR": {
        "pay_freq":"Annual",
        "day_count":"ACT/360"
    }
}
```

2.3 Model & Code

The Cross-Currency Swap Leg2 Coupon Engine can calculate leg2 coupon of multiple cross-currency swap with any currencies in the multiple given date.

2.3.1 File Configuration

There are two configuration files. One file is called `CurrencySetupConfiguration.json`, which is shown in the `Day Count & Pay Frequency` section. It includes all day counts and pay frequency associated with all currencies.

Another file is called `FileConfiguration.json`. User need to set up before running Python script.

- `template_file_name`: template file name
- `bloomberg_file_name`: Bloomberg file name
- `output_file_name`: output file name
- `file_path`: template path
- `override_currency`: override currency which does not have 30 years term by Bloomberg Python API

```
{
    "template_file_name": "Cross-Currency_Swap_Leg2_Coupon_Engine_Template.xlsx",
    "bloomberg_file_name": "bloomberg_data.xlsx",
    "output_file_name": "Cross-Currency_Swap_Leg2_Coupon_Engine_Output.xlsx",
    "file_path": "C:/Automation/Cross-Currency_Swap_Leg2_Coupon_Engine/",
    "override_currency": ["HKD"
    ]
}
```

2.3.2 Instruction (Office Version)

User need to run Python script in the office Bloomberg Terminal.

Before following the instruction, please make sure the Bloomberg account setting is correct.

1. Go to `C:\Automation\Cross-Currency_Swap_Leg2_Coupon_Engine`, open `Cross-Currency_Swap_Leg2_Coupon_Engine_Template.xlsx`
2. Go to `Data` tab, load cross-currency swap information for every `Curve_Date`
3. Link `Rec_Coupon` from input template
4. Go to `Main` tab, modify curve date, swap name and run

5. Save and close
6. Run `Run_Engine_Template_Script.bat`

The output file is `Cross-Currency_Swap_Leg2_Coupon_Engine_Output.xlsx` (data value only), which contains `Pay_Coupon` result.

2.3.3 How to Add Cross-Currency Swap with New Currency

In the future, user may add cross-currency swap with new currency in the template.

For example, user wants to add **ADB Spread in GBP**

In the `Cross-Currency_Swap_Leg2_Coupon_Engine_Template.xlsx`:

- Go to `Data` tab, load `ADB Spread in GBP` information for every `Curve_Date`
- Go to `Discount Factor BBG` tab:
  - Type following information into table, user may refer `How to Check Curve Number & Index` section

Currency	Basis Curve Number	Basis Curve Index	CSA Curve Number
GBP	S91	YCSW0091	S405

  - Keep same format of discount factor and insert same formula in the row 4, but link S91 cell into formula instead
  - Check **GBP**'s 30 years term is loaded or not
    - If 30 years term is loaded, done!
    - If 30 years term is not loaded, add 30 years term in the Bloomberg account. User may refer `How to Add 30 Years Term` section
    - If there is no ticker for 30 years term in the Bloomberg, check **GBP** `Basis Curve Index`
      - If `Basis Curve Index` is available, type `"GBP"` into `"override_currency"` in the `FileConfiguration.json`
      - If `Basis Curve Index` is not available, load 30 years term manually

2.3.4 Related Main Python Libraries

1. blpapi
2. openpyxl
3. QuantLib
4. pandas
5. numpy
6. tqdm
7. openpyxl
8. xlsxwriter
9. xlwings
10. Tkinter

2.3.5 Code Documentation

class BloombergAutomation.**ExampleOption**(**opts*, ***attrs*)

BloombergAutomation.**GUI**()
The GUI function is to pop-up a window to show the Python script is done

Returns None

Return type None

BloombergAutomation.**SWPMRequest**(*session*, *currency_list*, *curve_date*)
The SWPMRequest function is to request discount factor in the given currency list and curve date

Parameters

- **session** (*session*) – session
- **currency_list** (*list*) – list of currency
- **curve_date** (*datetime*) – curve date

Returns None

Return type None

BloombergAutomation.**checkDateTime**(*option*, *opt*, *value*)
The checkDateTime function is to check date is in the correct format

Parameters **value** (*datetime*) – date

Returns datetime

Return type datetime

BloombergAutomation.**check_curve_load**()
The check_curve_load function is to check whether curve is loaded or not

Returns True (loaded) or False (unloaded)

Return type bool

BloombergAutomation.**check_excel_data**(*bloomberg_data_address*)
The check_excel_data function is to check excel data is loaded or not

Parameters **bloomberg_data_address** (*str*) – Bloomberg data excel file path

Returns None

Return type None

BloombergAutomation.**check_sheet_data**(*bloomberg_data_wb*, *sheet_name*)
The check_sheet_data function is to check excel sheet data is loaded or not

Parameters

- **bloomberg_data_wb** (*workbook*) – Bloomberg workbook
- **sheet_name** (*str*) – sheet name

Returns None

Return type None

BloombergAutomation.**currency_df**(*currency*, *dt*, *df_dict*)
The currency_df function is to return discount factor in the given currency

Parameters

- **currency** (*str*) – currency
- **dt** (*datetime*) – date
- **df_dict** (*ordered dictionary*) – ordered dictionary of discount factor

Returns discount factor in the given currency

Return type float

BloombergAutomation.**datetime_to_ql**(*dt*)

The datetime_to_ql function is to convert datetime into QuantLib datetime format

Parameters **dt** (*datetime*) – datetime

Returns QuantLib datetime

Return type QuantLib datetime

BloombergAutomation.**day_count_30360**(*start_date*, *end_date*)

The day_count_30360 function is to return number of days between start_date and end_date, using 30/360 convention

Parameters

- **start_date** (*datetime*) – start date
- **end_date** (*datetime*) – end date

Returns number of days between start_date and end_date, using 30/360 convention

Return type int

BloombergAutomation.**day_count_30I360**(*start_date*, *end_date*)

The day_count_30I360 function is to return number of days between start_date and end_date, using 30I/360 convention

Parameters

- **start_date** (*datetime*) – start date
- **end_date** (*datetime*) – end date

Returns number of days between start_date and end_date, using 30I/360 convention

Return type int

BloombergAutomation.**day_count_30U360**(*start_date*, *end_date*)

The day_count_30U360 function is to return number of days between start_date and end_date, using 30U/360 convention

Parameters

- **start_date** (*datetime*) – start date
- **end_date** (*datetime*) – end date

Returns number of days between start_date and end_date, using 30U/360 convention

Return type int

BloombergAutomation.**dc_generation**(*day_cnt*, *pay_date*)

The dc_generation function is to return a list of year fraction in the specific convention

Parameters

- **day_cnt** (*str*) – day count
- **pay_date** (*list*) – list of payment date

Returns list of year fraction in the specific convention

Return type list

BloombergAutomation.**df_generation**()
The df_generation function is to generate discount factor

Returns an ordered dictionary of discount factor

Return type order dictionary

BloombergAutomation.**df_interp**(*dt*, *df_dict*, *leg*)
The df_interp function is to return discount factor in the given leg

Parameters

- **dt** (*datetime*) – date
- **df_dict** (*ordered dictionary*) – ordered dictionary of discount factor
- **leg** – leg

Returns discount factor in the given leg

Return type float

BloombergAutomation.**df_interp_30360**(*dt*, *df_dict*)
The df_interp_30360 function is to interpolate discount factor in the given date, using 30/360 convention

Parameters

- **dt** (*datetime*) – date
- **df_dict** (*ordered dictionary*) – ordered dictionary of discount factor

Returns discount factor in the given date, using 30/360 convention

Return type float

BloombergAutomation.**df_interp_30I360**(*dt*, *df_dict*)
The df_interp_30I360 function is to interpolate discount factor in the given date, using 30I360 convention

Parameters

- **dt** (*datetime*) – date
- **df_dict** (*ordered dictionary*) – ordered dictionary of discount factor

Returns discount factor in the given date, using 30I360 convention

Return type float

BloombergAutomation.**df_interp_30U360**(*dt*, *df_dict*)
The df_interp_30U360 function is to interpolate discount factor in the given date, using 30U/360 convention

Parameters

- **dt** (*datetime*) – date
- **df_dict** (*ordered dictionary*) – ordered dictionary of discount factor

Returns discount factor in the given date, using 30U/360 convention

Return type float

BloombergAutomation.**df_interp_ACT360**(*dt*, *df_dict*)
The df_interp_ACT360 function is to interpolate discount factor in the given date, using ACT/360 convention

Parameters

- **dt** (*datetime*) – date
- **df_dict** (*ordered dictionary*) – ordered dictionary of discount factor

Returns discount factor in the given date, using ACT/360 convention

Return type float

BloombergAutomation.**df_interp_ACT365**(*dt*, *df_dict*)

The df_interp_ACT365 function is to interpolate discount factor in the given date, using ACT/365 convention

Parameters

- **dt** (*datetime*) – date
- **df_dict** (*ordered dictionary*) – ordered dictionary of discount factor

Returns discount factor in the given date, using ACT/365 convention

Return type float

BloombergAutomation.**df_interp_ACT365FIXED**(*dt*, *df_dict*)

The df_interp_ACT365FIXED function is to interpolate discount factor in the given date, using ACT/365.FIXED convention

Parameters

- **dt** (*datetime*) – date
- **df_dict** (*ordered dictionary*) – ordered dictionary of discount factor

Returns discount factor in the given date, using ACT/365.FIXED convention

Return type float

BloombergAutomation.**df_interp_ACTACT**(*dt*, *df_dict*)

The df_interp_ACTACT function is to interpolate discount factor in the given date, using ACT/ACT convention

Parameters

- **dt** (*datetime*) – date
- **df_dict** (*ordered dictionary*) – ordered dictionary of discount factor

Returns discount factor in the given date, using ACT/ACT convention

Return type float

BloombergAutomation.**df_interp_DU252**(*dt*, *df_dict*)

The df_interp_DU252 function is to interpolate discount factor in the given date, using DU/252 convention

Parameters

- **dt** (*datetime*) – date
- **df_dict** (*ordered dictionary*) – ordered dictionary of discount factor

Returns discount factor in the given date, using DU/252 convention

Return type float

BloombergAutomation.**get_allIn_df_info**(*curve_date_index*)

The get_allIn_df_info function is to load discount factor in the ADB All-in Template [Discount Factor BBG]

Parameters **curve_date_index** (*int*) – curve date index

Returns ordered dictionary of currency information

Return type ordered dictionary

BloombergAutomation.**get_allIn_excel_info**()
The get_allIn_excel_info function is to load swap information in the ADB All-in Template

Returns ordered dictionary of swap information

Return type ordered dictionary

BloombergAutomation.**get_currency_info**()
The get_currency_info function is to get currency information in the ALM Reporting Template [Discount Factor BBG]

Returns ordered dictionary of currency information

Return type ordered dictionary

BloombergAutomation.**get_currency_setup_info**()
The get_currency_setup_info is to get every country's calendar from QuantLib. Some small countries don't have calendar from QuanLib, United States calendar will be used by dufault

Returns ordered dictionary of every country's calendar from QuantLib

Return type ordered dictionary

BloombergAutomation.**get_date_info**()
The get_date_info function is to get date information in the ADB All-in Template [Dates]

Returns ordered dictionary of date

Return type ordered dictionary

BloombergAutomation.**get_df_info**()
The get_df_info function is to load discount factor in the ALM Reporting Template [Discount Factor BBG]

Returns ordered dictionary of currency information

Return type ordered dictionary

BloombergAutomation.**get_excel_info**(*main_dict*)
The get_excel_info function is to get all swap tabs information in the ALM Reporting Template swap tabs

Parameters **main_dict** (*ordered dictionary*) – ordered dictionary of main information

Returns ordered dictionary of swap information

Return type ordered dictionary

BloombergAutomation.**get_main_info**()
The get_main_info function is to get main information in the ALM Reporting Template [Main Tab]

Returns ordered dictionary of main information

Return type ordered dictionary

BloombergAutomation.**get_pay_freq_info**()
The get_pay_freq_info function is to get all pay frequency from QuantLib

Returns ordered dictionary of all pay frequency from QuantLib

Return type ordered dictionary

BloombergAutomation.**load_discount_factor**(*bloomberg_data_address*)
The load_discount_factor function is to load discount factor from excel API

Parameters **bloomberg_data_address** (*str*) – Bloomberg data excel file path

Returns None

Return type None

BloombergAutomation.**load_maturity_date**(*rec_coupon*, *val_dt*, *term*)
The load_maturity_date is to calculate maturity date

Parameters

- **rec_coupon** (*float*) – receive coupon
- **val_dt** (*datetime*) – valuation date
- **term** (*int*) – term

Returns ordered dictionary of local config

Return type ordered dictionary

BloombergAutomation.**main**()
The main function is to choose to run ALM Reporting Template or ADB All-in Template

Returns None

Return type None

BloombergAutomation.**payCouponCalc**(*rec_coupon*, *rec_df_vec*, *pay_df_vec*, *rec_dc_vec*, *pay_dc_vec*)
The payCouponCalc function is to calculate pay coupon

Parameters

- **rec_coupon** (*float*) – receive coupon
- **rec_df_vec** (*list*) – list of receive discount factor
- **pay_df_vec** (*list*) – list of pay discount factor
- **rec_dc_vec** (*list*) – list of receive day count
- **pay_dc_vec** (*list*) – list of pay day count

Returns pay coupon

Return type float

BloombergAutomation.**pay_date_gen**(*eff_dt*, *mat_dt*, *pay_freq*, *currency*)
The pay_date_gen function is to generate payment date

Parameters

- **eff_dt** (*datetime*) – effective date
- **mat_dt** (*datetime*) – maturity date
- **pay_freq** (*str*) – payment frequency
- **pay_freq** – currency

Returns a list of payment date

Return type list

BloombergAutomation.**payment_date_calc**(*eff_dt*, *mat_dt*, *pay_freq*, *currency*)
The payment_date_calc function is to calculate payment date

Parameters

- **eff_dt** (*datetime*) – effective date
- **mat_dt** (*datetime*) – maturity date
- **pay_freq** (*str*) – payment frequency

- **pay_freq** – currency

Returns a list of payment date

Return type list

BloombergAutomation.**processMessage**(*msg*, *currency_list*)
The processMessage function is to load discount factor in the given currency list

Parameters

- **msg** (*message*) – message
- **currency_list** (*list*) – list of currency

Returns None

Return type None

BloombergAutomation.**ql_to_datetime**(*d*)
The ql_to_datetime function is to convert QuantLib datetime format into datetime

Parameters **d** (*QuantLib datetime*) – QuantLib datetime

Returns datetime

Return type datetime

BloombergAutomation.**run_all_in_template**()
The run_all_in_template function is to run ADB All-in Template

Returns None

Return type None

BloombergAutomation.**run_bloomberg_python_api**(*currency_list*, *curve_date*)
The run_bloomberg_python_api function is to run Bloomberg Python API to download discount factor

Parameters

- **currency_list** (*list*) – list of currency
- **curve_date** (*datetime*) – curve date

Returns None

Return type None

BloombergAutomation.**run_excel_macro**(*file_path*, *file_name*)
The run_excel_macro function is to run Excel macro by Python

Parameters

- **file_path** (*str*) – file path
- **file_name** (*str*) – file name

Returns None

Return type None

BloombergAutomation.**run_pay_coupon_calc**(*rec_coupon*, *period*)
The run_pay_coupon_calc function is to return pay coupon from above functions

Parameters

- **rec_coupon** (*float*) – receive coupon
- **period** (*int*) – period

Returns pay coupon

Return type float

BloombergAutomation.**run_report_template**()
The run_report_template function is to run ALM Reporting Template

Returns None

Return type None

BloombergAutomation.**store_df**(*bloomberg_data_address*)
The store_df function is to store discount factor from excel

Parameters **bloomberg_data_address** (*str*) – Bloomberg data excel file path

Returns None

Return type None

BloombergAutomation.**year_fraction_30360**(*start_date*, *end_date*)
The year_fraction_30360 function is to return fraction in years between start_date and end_date, using 30/360 convention

Parameters

- **start_date** (*datetime*) – start date
- **end_date** (*datetime*) – end date

Returns fraction in years between start_date and end_date, using 30/360 convention

Return type float

BloombergAutomation.**year_fraction_30I360**(*start_date*, *end_date*)
The year_fraction_30I360 function is to return fraction in years between start_date and end_date, using 30I/360 convention

Parameters

- **start_date** (*datetime*) – start date
- **end_date** (*datetime*) – end date

Returns fraction in years between start_date and end_date, using 30I/360 convention

Return type float

BloombergAutomation.**year_fraction_30U360**(*start_date*, *end_date*)
The year_fraction_30U360 function is to return fraction in years between start_date and end_date, using 30U/360 convention

Parameters

- **start_date** (*datetime*) – start date
- **end_date** (*datetime*) – end date

Returns fraction in years between start_date and end_date, using 30U/360 convention

Return type float

BloombergAutomation.**year_fraction_ACT360**(*start_date*, *end_date*)
The year_fraction_ACT360 function is to return fraction in years between start_date and end_date, using ACT/360 convention

Parameters

- **start_date** (*datetime*) – start date
- **end_date** (*datetime*) – end date

Returns fraction in years between start_date and end_date, using ACT/360 convention

Return type float

BloombergAutomation.**year_fraction_ACT365**(*start_date*, *end_date*)

The year_fraction_ACT365 function is to return fraction in years between start_date and end_date, using ACT/365 convention

Parameters

- **start_date** (*datetime*) – start date
- **end_date** (*datetime*) – end date

Returns fraction in years between start_date and end_date, using ACT/365 convention

Return type float

BloombergAutomation.**year_fraction_ACT365FIXED**(*start_date*, *end_date*)

The year_fraction_ACT365FIXED function is to return fraction in years between start_date and end_date, using ACT/365.FIXED convention

Parameters

- **start_date** (*datetime*) – start date
- **end_date** (*datetime*) – end date

Returns fraction in years between start_date and end_date, using ACT/365.FIXED convention

Return type float

BloombergAutomation.**year_fraction_ACTACT**(*start_date*, *end_date*)

The year_fraction_ACTACT function is to return fraction in years between start_date and end_date, using ACT/ACT convention

Parameters

- **start_date** (*datetime*) – start date
- **end_date** (*datetime*) – end date

Returns fraction in years between start_date and end_date, using ACT/ACT convention

Return type float

BloombergAutomation.**year_fraction_DU252**(*start_date*, *end_date*)

The year_fraction_DU252 function is to return fraction in years between start_date and end_date, using DU/252 convention

Parameters

- **start_date** (*datetime*) – start date
- **end_date** (*datetime*) – end date

Returns fraction in years between start_date and end_date, using DU/252 convention

Return type float

2.4 Foreign Spreads in Local Currency Template

The foreign spreads in local currency template includes multiple cross-currency swaps with different currencies in one given curve date.

2.4.1 File Configuration

There are two configuration files. One file is called `CurrencySetupConfiguration.json`, which is shown in the `Day Count & Pay Frequency` section. It includes all day counts and pay frequency associated with all currencies.

Another file is called `FileConfiguration.json`. User need to set up before running Python script.

- `file_name`: template name
- `file_path`: template path
- `user`: user name
- `type`: type of foreign spreads in local currency template is A (by default)
- `version`: WFH or office
- `override_currency`: override currency which does not have 30 years term by Bloomberg Python API

```
{
    "file_name": "Foreign Spreads in Local Currency Template.xlsx",
    "file_path": "C:/Automation/ALM_Reporting/",
    "user": "Daodao",
    "type": "A",
    "version": "office",
    "override_currency":[
                    "HKD"
    ]
}
```

2.4.2 Instruction (WFH & Remote Bloomberg Terminal Version)

User need to remote Bloomberg Terminal and run Python script.

Before following the instruction, please make sure the Bloomberg account setting is correct.

1. Go to `C:\Automation\ALM_Reporting\Code`, modify `FileConfiguration.json`, type `"WFH"` or `"wfh"` into `"version"`
2. Go to `C:\Automation\ALM_Reporting\Template`, make a copy of `Foreign Spreads in Local Currency Template.xlsx` into `C:\Automation\ALM_Reporting`

   Please don't rename the template, also don't modify the original template directly!

3. Log in Bloomberg anywhere and open `Foreign Spreads in Local Currency Template.xlsx` from `C:\Automation\ALM_Reporting`
4. Load inputs from other input files
5. Go to `Discount Factor BBG` tab, refresh the sheet, the discount factor will be shown automatically.

If not, please check Excel `Formulas` -> `Calculation Options` -> `Automatic`

6. For some currencies which don't have 30 years term, load it manually

   For example: HKD

   - In the Bloomberg, enter *{SWPM -FXFX USD HKD <GO>}*
   - In the `Valuation Settings`, change `Curve date` (same with template)
   - Go to `5) curves` tab, choose `409 HKD Cashflow CSA Curve(s)`, copy 30 years term's discount factor into template

7. Check all input values. Save and close the template. **Don't open it again!**
8. Run `Run_Report_Template_Script.bat`

The output Excel is `Foreign Spreads in Local Currency_yyyymmdd.xlsx` (data value only)

2.4.3 Instruction (WFH & Local Version)

User can run Python script on the laptop.

Before following the instruction, please make sure the Bloomberg account setting is correct.

1. Go to `S:\corp\alm\BloombergAutomation\ALM_Reporting`, modify `FileConfiguration.json`, type `"WFH"` or `"wfh"` into `"version"`
2. Go to `S:\corp\alm\BloombergAutomation\ALM_Reporting\Template`, make a copy of `Foreign Spreads in Local Currency Template.xlsx` into `S:\corp\alm\BloombergAutomation\ALM_Reporting`

   Please don't rename the template, also don't modify the original template directly!

3. Log in Bloomberg anywhere and open `Foreign Spreads in Local Currency Template.xlsx` from `S:\corp\alm\BloombergAutomation\ALM_Reporting`
4. Load inputs from other input files
5. Go to `Discount Factor BBG` tab, refresh the sheet, the discount factor will be shown automatically.

   If not, please check Excel `Formulas` -> `Calculation Options` -> `Automatic`

6. For some currencies which don't have 30 years term, load it manually

   For example: HKD

   - In the Bloomberg, enter *{SWPM -FXFX USD HKD <GO>}*
   - In the `Valuation Settings`, change `Curve date` (same with template)
   - Go to `5) curves` tab, choose `409 HKD Cashflow CSA Curve(s)`, copy 30 years term's discount factor into template

7. Check all input values. Save and close the template. **Don't open it again!**
8. Run `BloombergAutomation.exe`

The output Excel is `Foreign Spreads in Local Currency_yyyymmdd.xlsx` (data value only)

2.4.4 Instruction (Office Version)

User can run Python script in the office Bloomberg Terminal.

Before following the instruction, please make sure the Bloomberg account setting is correct.

1. Go to `C:\Automation\ALM_Reporting\Code`, modify `FileConfiguration.json`, type `"office"` or `"Office"` into `"version"`
2. Go to `C:\Automation\ALM_Reporting\Template`, make a copy of `Foreign Spreads in Local Currency Template.xlsx` into `C:\Automation\ALM_Reporting`

   Please don't rename the template, also don't modify the original template directly!

3. Log in Bloomberg Terminal and open `Foreign Spreads in Local Currency Template.xlsx` from `C:\Automation\ALM_Reporting`
4. Load inputs from other input files
5. Go to `Discount Factor BBG` tab, refresh the sheet, the discount factor will be shown automatically.

   If not, please check Excel `Formulas` -> `Calculation Options` -> `Automatic`

6. For some currencies which don't have 30 years term, user does not need to load 30 years term manually, just modify `FileConfiguration.json`

   For example: HKD

   - In the `FileConfiguration.json`, type `"HKD"` into `"override_currency"`
7. Check all input values. Save and close the template.
8. Run `Run_Report_Template_Script.bat`

The output Excel is `Foreign Spreads in Local Currency_yyyymmdd.xlsx` (data value only)

2.4.5 How to Add Cross-Currency Swap with New Currency

In the future, user may add cross-currency swap with new currency in the template.

For example, user wants to add **ADB Spread in GBP**

In the `Foreign Spreads in Local Currency Template.xlsx`:

- Type following information in the `Main Tab`

Tab	Run (Y/N)	Receive Currency	Pay Currency
ADB Spread in GBP	Y	USD	GBP

- Create new tab called `ADB Spread in GBP`, which is same name in the Tab
- In the `ADB Spread in GBP` tab, keep same table format and load value

- Double check column name is correct
- In the `Discount Factor BBG` tab:
  - Type following information into table, user may refer `How to Check Curve Number & Index` section

Currency	Basis Curve Number	Basis Curve Index	CSA Curve Number
GBP	S91	YCSW0091	S405

  - Keep same format of discount factor and insert same formula in the row 4, but link S91 cell into formula instead
  - Check **GBP**'s 30 years term is loaded or not
    - If 30 years term is loaded, done!
    - If 30 years term is not loaded, add 30 years term in the Bloomberg account. User may refer `How to Add 30 Years Term` section
    - If there is no ticker for 30 years term in the Bloomberg, check **GBP** `Basis Curve Index`
      - If `Basis Curve Index` is available, type `"GBP"` into `"override_currency"` in the `FileConfiguration.json`
      - If `Basis Curve Index` is not available, load 30 years term manually

2.5 ADB All-in Template

The ADB all-in template includes one cross-currency swaps with two currencies in the multiple given curve date.

2.5.1 File Configuration

There are two configuration files. One file is called `CurrencySetupConfiguration.json`, which is shown in the `Day Count & Pay Frequency` section. It includes all day counts and pay frequency associated with all currencies.

Another file is called `FileConfiguration.json`. User need to set up before running Python script.

- `template_file_name`: template file name
- `input_file_name`: input file name
- `file_path`: template path
- `user`: user name
- `type`: type of foreign spreads in local currency template is B (by default)
- `version`: WFH or office
- `override_currency`: override currency which does not have 30 years term by Bloomberg Python API

```
{
    "template_file_name": "ADB_Bloomberg_DownLoad_HKD.xlsm",
    "input_file_name": "ADB_Bloomberg_HKD_All-in_Input.xlsm",
    "file_path": "C:/Automation/ADB_All-in_Template/ABB/",
    "user": "Donghao",
    "type": "B",
```

(continues on next page)

(continued from previous page)

```
    "version": "office",
    "override_currency": ["HKD"
    ]
}
```

2.5.2 Instruction (WFH & Remote Bloomberg Terminal Version)

User need to remote Bloomberg Terminal and run Python script.

Before following the instruction, please make sure the Bloomberg account setting is correct.

For example: **HKD**

1. Go to `C:\Automation\ADB_All-in_Template\ABB`, modify `FileConfiguration.json`, type `"WFH"` or `"wfh"` into `"version"`
2. Open `ADB_Bloomberg_DownLoad_HKD.xlsm`, select `Month` and `Year` for report

   Please don't change Input Path, Output Path and Output Filename!

3. Verify the `TMY date`, then enter the `Curve Date` which corresponds to the last date in the source file before (or equal to) each `TMY Date`
4. Copy input files with same `Curve Date` into `S:\corp\alm\BloombergAutomation\ADB_All-in_Template\Input_Files`
5. Go to `Discount Factor BBG` tab, refresh the sheet, the discount factor will be shown automatically.

   If not, please check Excel `Formulas` -> `Calculation Options` -> `Automatic`

6. For some currencies which don't have 30 years term, load it manually

   For example: HKD

   - In the Bloomberg, enter *{SWPM -FXFX USD HKD <GO>}*
   - In the `Valuation Settings`, change `Curve Date` (same with template)
   - Go to `5) curves` tab, choose `409 HKD Cashflow CSA Curve(s)`, copy 30 years term's discount factor into template
7. Check all input values. Save and close the template. **Don't open it again!**
8. Run `Run_HKD_Template_Script.bat`

There are two output Excel files. One Excel is `ADB_Bloomberg_HKD_All-in_Input.xlsm`, which contains formulas but no result. Another file is `ADB_Bloomberg_HKD_All-in_yyyymmdd.xlsx`, which contains result but data value only.

2.5.3 Instruction (WFH & Local Version)

User can run Python script on the laptop.

Before following the instruction, please make sure the Bloomberg account setting is correct.

For example: **HKD**

1. Go to `S:\corp\alm\BloombergAutomation\ADB_All-in_Template\ABB`, modify `FileConfiguration.json`, type `"WFH"` or `"wfh"` into `"version"`
2. Open `ADB_Bloomberg_DownLoad_HKD.xlsm`, select `Month` and `Year` for report

   Please don't change Input Path, Output Path and Output Filename!

3. Verify the `TMY date`, then enter the `Curve Date` which corresponds to the last date in the source file before (or equal to) each `TMY Date`
4. Copy input files with same `Curve Date` into `S:\corp\alm\BloombergAutomation\ADB_All-in_Template\Input_Files`
5. Go to `Discount Factor BBG` tab, refresh the sheet, the discount factor will be shown automatically.

   If not, please check Excel `Formulas` -> `Calculation Options` -> `Automatic`

6. For some currencies which don't have 30 years term, load it manually

   For example: HKD

   - In the Bloomberg, enter *{SWPM -FXFX USD HKD <GO>}*
   - In the `Valuation Settings`, change `Curve Date` (same with template)
   - Go to `5) curves` tab, choose `409 HKD Cashflow CSA Curve(s)`, copy 30 years term's discount factor into template
7. Check all input values. Save and close the template. **Don't open it again!**
8. Run `BloombergAutomation.exe`

There are two output Excel files. One Excel is `ADB_Bloomberg_HKD_All-in_Input.xlsm`, which contains formulas but no result. Another file is `ADB_Bloomberg_HKD_All-in_yyyymmdd.xlsx`, which contains result but data value only.

2.5.4 Instruction (Office Version)

User can run Python script in the office Bloomberg Terminal.

Before following the instruction, please make sure the Bloomberg account setting is correct.

For example: **HKD**

1. Go to `C:\Automation\ADB_All-in_Template\ABB`, modify `FileConfiguration.json`, type `"office"` or `"Office"` into `"version"`
2. Open `ADB_Bloomberg_DownLoad_HKD.xlsm`, select `Month` and `Year` for report

Please don't change Input Path, Output Path and Output Filename!

3. Verify the `TMY date`, then enter the `Curve Date` which corresponds to the last date in the source file before (or equal to) each `TMY Date`
4. Copy input files with same `Curve Date` into `S:\corp\alm\BloombergAutomation\ADB_All-in_Template\Input_Files`
5. Go to `Discount Factor BBG` tab, refresh the sheet, the discount factor will be shown automatically.

   If not, please check Excel `Formulas` -> `Calculation Options` -> `Automatic`

6. For some currencies which don't have 30 years term, modify `FileConfiguration.json`

   For example: HKD

   - In the `FileConfiguration.json`, type `"HKD"` into `"override_currency"`
7. Check all input values. Save and close the template.
8. Run `Run_HKD_Template_Script.bat`

There are two output Excel files. One Excel is `ADB_Bloomberg_HKD_All-in_Input.xlsm`, which contains formulas but no result. Another file is `ADB_Bloomberg_HKD_All-in_yyyymmdd.xlsx`, which contains result but data value only.

2.5.5 How to Add Cross-Currency Swap with New Currency

In the future, user may add cross-currency swap with new currency in the template.

For example, user wants to add **ADB Spread in GBP**

In the `ADB_Bloomberg_DownLoad_HKD.xlsm`:

- In the `Dates` tab
  - Change `Input Path`, `Output Path` and `Output Filename`
  - Type following information into table

Receive Currency	Pay Currency
USD	GBP

- Replace **HKD** as **GBP** from every table column in the `Formula` tab
- In the `Discount Factor BBG` tab:
  - Remove row 5 which contains **HKD** information from table
  - Type following information into table, user may refer `How to Check Curve Number & Index` section

Currency	Basis Curve Number	Basis Curve Index	CSA Curve Number
GBP	S91	YCSW0091	S405

  - In the row 4, change all **HKD** discount factor formula, link S91 cell into formula instead

  - Check **GBP**'s 30 years term is loaded or not
    * If 30 years term is loaded, done!
    * If 30 years term is not loaded, add 30 years term in the Bloomberg account. User may refer `How to Add 30 Years Term` section
    * If there is no ticker for 30 years term in the Bloomberg, check **GBP** `Basis Curve Index`
      · If `Basis Curve Index` is available, type `"GBP"` into `"override_currency"` in the `FileConfiguration.json`
      · If `Basis Curve Index` is not available, load 30 years term manually

2.6 Installation

2.6.1 Software Requirements

This Cross-Currency Swap Leg2 Coupon Engine is written by Python 2.7. For office version, in order to download discount factor from Bloomberg, Bloomberg Python API is required.

- Python 2.7
- Bloomberg Python API

2.6.2 How to Install Sphinx

- Open Command Prompt:
  - PIP install Sphinx

```
>>> C:\Users\bloomapi>pip install -U Sphinx --trusted-host pypi.org --trusted-
↪host files.pythonhosted.org
```

  - PIP install Sphinx theme

```
>>> C:\Users\bloomapi>pip install sphinx_rtd_theme --trusted-host pypi.org --
↪trusted-host files.pythonhosted.org
```

- Create documentation folder, in the documentation folder:
  - Hold `Shift` button and right click, `Open command window here`
  - Set up documentation

```
>>> C:\Automation\Documentation>sphinx-quickstart
```

2.6.3 How to Use Sphinx

- Modify `conf.py` first
- Create `.rst` file, for more details about how to write `.rst` file, please check Sphinx Cheat Sheet
- In the documentation folder
  - Hold `Shift` button and right click, `Open command window here`

```
>>> C:\Automation\Documentation>make html
```

- Go to `C:\Automation\Documentation\_build\html`, open `index.html`

2.6.4 How to Install Libraries

For example, one need to install **pyinstaller**

Open Command Prompt:

- Change the directory into Python 2.7 script folder

```
>>> C:\Users\bloomapi>cd "C:\Python27\Scripts"
```

- PIP install pyinstaller

```
>>> C:\Python27\Scripts>pip install pyinstaller --trusted-host pypi.org --trusted-
↪host files.pythonhosted.org
```

2.6.5 How to Generate EXE File

EXE file includes Python, Python script and all libraries required in the Python script. User can run EXE file without installing Python.

- **pyinstaller** is required
- In the Python script folder:
  - Hold `Shift` button and right click, `Open command window here`

```
>>> C:\Automation>pyinstaller your_Python_script.py
```

2.7 License

2.8 Contact

For any further questions, please contact zhangyichi19941030@gmail.com or message me at Yichi Zhang’s LinkedIn

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

b

INDEX

Q

R

S

Y