BTRM

The Certificate
of Bank Treasury
Risk Management

SOFR OIS Pricing and Riskless USD Curve Construction in light of the Impending USD LIBOR Discontinuation

BTRM Working Paper Series #15

Ioannis Rigopoulos

December 2020

Abstract

USD LIBOR *(London Inter-bank Offered Rate)* is not just elderly. It lies in a palliative station waiting its almost certain death by the end of 2021. But everybody seems to be blissfully ignorant about its deteriorating state of health and keep doing business as usual. Although the successor prince SOFR (Secured Overnight Financing Rate) has been already crowned since June 2017 by the ARRC *(Alternative Reference Rates Committee)* of the US Federal Reserve Bank, the wider buy-side market still clings on USD LIBOR for valuation and hedge management purposes.

This article attempts to shed some light in the definition of Overnight Index Swaps referencing SOFR and the standard methodology used by practitioners for their pricing. It also looks in the construction of their implied yield curve.

Important Note

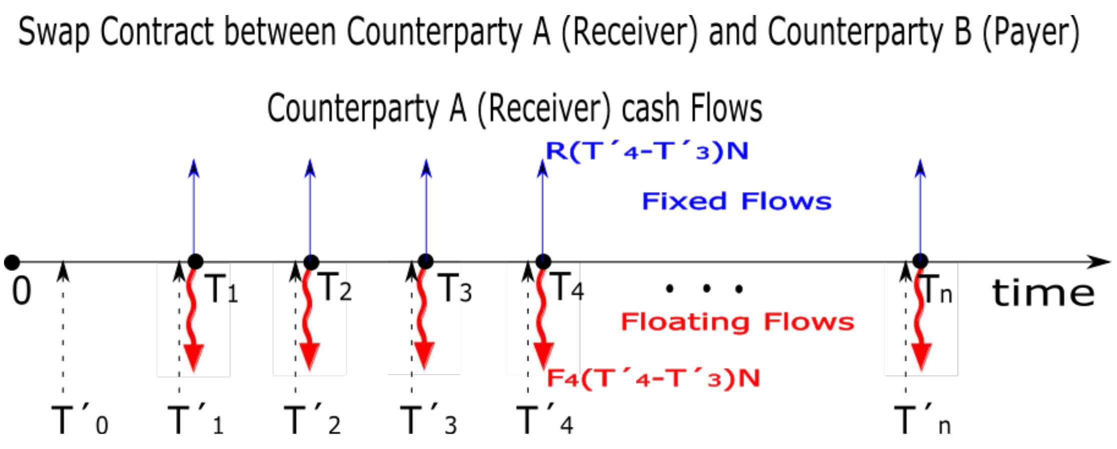
This article aims at explaining the mechanics of SOFR swaps by combining theory with practice. All spreadsheet formulas discussed here are part of the free downloadable spreadsheet curve.xlsx

Overnight Index Swap referencing SOFR

An OIS *(Overnight Index Swap)* is a swap consisting of two legs: a fixed leg that pays a fixed rate over regular intervals and a *floating* (or *overnigh*t) leg that pays a variable rate over the same intervals as the fixed leg.

More precisely, the OIS contract specifies n payment times T_1, T_2, ..., T_n in increasing order, where n is the number of periods in the swap. In exchange traded swaps, the payment frequency is usually annual. This implies n = 1 when the swap's maturity is less than a year and n = K for a swap with a maturity of K years.

The cash flow schedule is sketched in the following diagram:


Swap Contract between Counterparty A (Receiver) and Counterparty B (Payer)
Counterparty A (Receiver) cash Flows
R(T´4-T´3)N
Fixed Flows
0
T1
T2
T3
T4
. . .
Tn
time
Floating Flows
F4(T´4-T´3)N
T´0
T´1
T´2
T´3
T´4
T´n

Given an agreed swap notional of N (eg 10M $), at each payment time T_u, u = 1,...,n, counterparty A (the *Receiver*) receives a fixed amount of $R\Delta_u N$ from counterparty B (the *Payer*) and pays a floating amount of $F_u\Delta_u N$. The fixed amount is always known because the rate R is specified in the swap contract, but the floating amount can be calculated only when the time T_u arrives because it depends on the floating rate F_u described below.

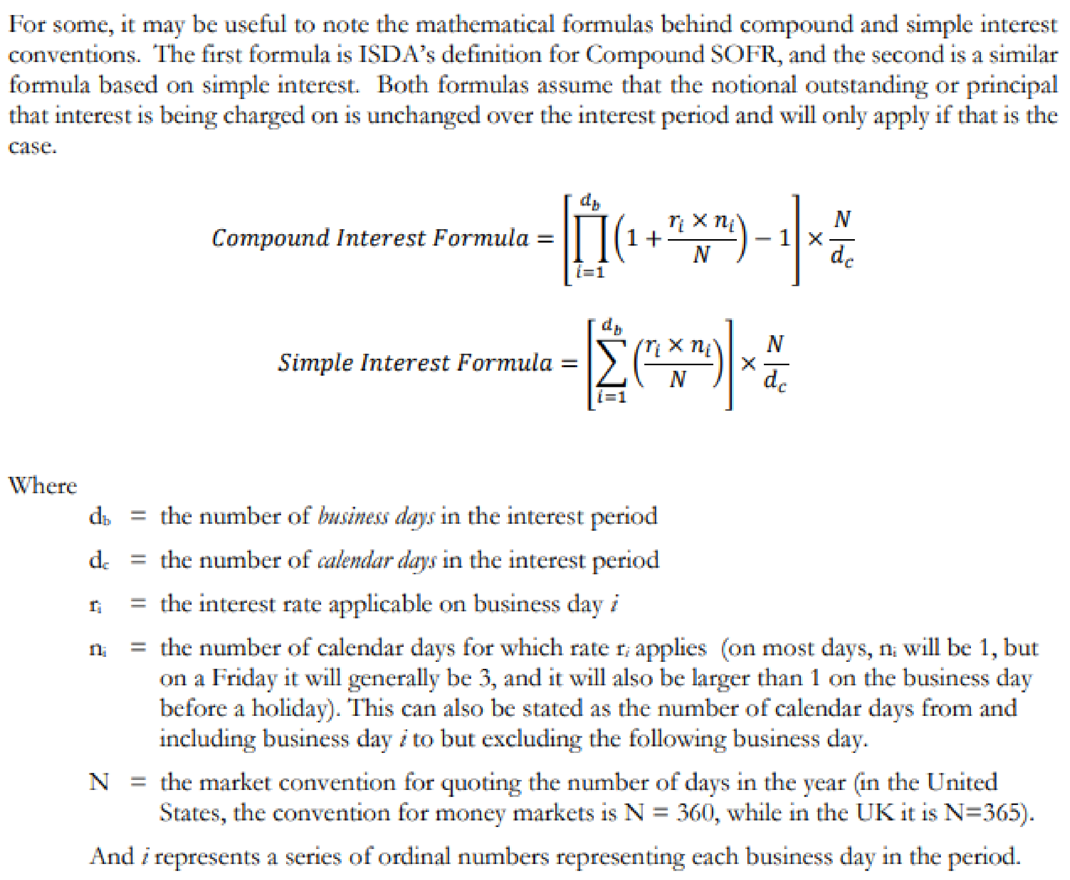
T'_0, T'_1, T'_2, ..., T'_n are the times in increasing order that define the accrual periods of the swap. These times do not generally coincide with the payment times T_1, T_2, ..., T_n. Concretely, T'_1 is the end of the first accrual period, T'_2 is the end of the second accrual period and T'_n is the end of the nth accrual period. Δ_u is the time difference T'_u-T'_{u-1} expressed in number of years and calculated according to the agreed day count convention, which in the case of SOFR OIS is ACT/360, meaning that Δ_u = (Number of calendar days from T'_{u-1} to T'_u) / 360.

In SOFR OIS the number of business days between T'_u and T_u is referred as *payment lag* and equals 2. This lag is intentional, so that the calculating agent to be by T_u in possession of all the SOFR indices referencing the days in the respective accrual period (see description of the floating payment below). There is also an initial settlement period of two business days, meaning that T'_0 is two business days in the future.

The floating rates F_u, u = 1,...,n, determine the floating payment amounts $F_u\Delta_u N$.
Each F_u is defined as the compounded average of the SOFR indices r_i , i = 1,...,d_b, published by the Federal Reserve Bank of new York every business day throughout the time range between T'_{u-1} and T'_u. Here d_b is the number of business days in the interval (T'_{u-1} , T'_u).

The precise formula for F_u is shown below, where N stands for the number 360. The image has been copied from this Federal Reserve page.

Formulas for Compounded and Arithmetic Average SOFR Index



For some, it may be useful to note the mathematical formulas behind compound and simple interest conventions. The first formula is ISDA's definition for Compound SOFR, and the second is a similar formula based on simple interest. Both formulas assume that the notional outstanding or principal that interest is being charged on is unchanged over the interest period and will only apply if that is the case.

$$\textit{Compound Interest Formula} = \left[\prod_{i=1}^{d_b}\left(1+\frac{r_i \times n_i}{N}\right)-1\right] \times \frac{N}{d_c}$$

$$\textit{Simple Interest Formula} = \left[\sum_{i=1}^{d_b}\left(\frac{r_i \times n_i}{N}\right)\right] \times \frac{N}{d_c}$$

Where

d_b	=	the number of *business days* in the interest period
d_c	=	the number of *calendar days* in the interest period
r_i	=	the interest rate applicable on business day *i*
n_i	=	the number of calendar days for which rate r_i applies (on most days, n_i will be 1, but on a Friday it will generally be 3, and it will also be larger than 1 on the business day before a holiday). This can also be stated as the number of calendar days from and including business day *i* to but excluding the following business day.
N	=	the market convention for quoting the number of days in the year (in the United States, the convention for money markets is N = 360, while in the UK it is N=365).

And *i* represents a series of ordinal numbers representing each business day in the period.

Actual versus Implied Calculation of the Compounded SOFR

While the above formula must be employed for calculating the actual present and past settlement amounts of the overnight leg, it is not helpful when it comes to future settlements, as the latter are based on – at least partially - not-yet-observed SOFR indices r_i, i = 1,...,d_b. For simplicity, assume that all indices r_i are not yet known and should therefore be treated as random variables from a today's perspective. It follows that the product $\Pi(1 + r_i n_i/360)$ over all i, appearing in the Compound Interest Formula above, is a product of random variables and therefore a random variable itself. This makes the corresponding floating amount a random variable as well.

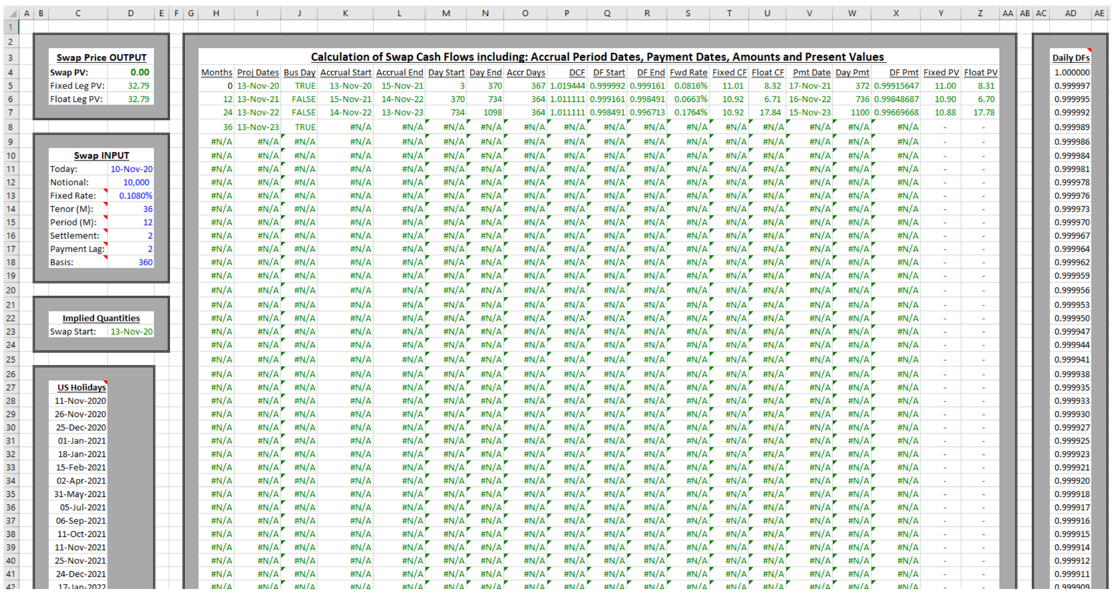
If the payment lag is ignored, one can prove that the present value of the floating payment at T'_0 based on the random compound factor $\Pi(1 + r_i n_i/360)$ is the same as the present value of a fixed payment at T'^{u} based on the fixed rate $R_u = [\Pi(1 + r'_i n_i/360) – 1] / \Delta_u$, where r'_i is the non-random, today-observed forward rate corresponding to r_i. This is a beautiful result that holds independently of the stochastic nature of the rates r_i.

A mathematical proof of this result using forward measure martingales is provided in the appendix. There is also shown the following formula:
$\Pi(1 + r'_i n_i/360) = 1 + F'_u \Delta_u$
that expresses the product of forward overnight compound factors as a single forward term compound factor. The rate F'_u is defined as the today-observed forward rate spanning the time interval referenced by the random rate F_u. In other words:
$F'_u = [P(T'_{u-1}) / P(T'_u) - 1] / \Delta_u$
where P(t) is the discount factor with maturity t.

This is an important result that allows us - in a context where the discount factor curve is known - to replace the compounded average of curve-implied forward overnight rates with a single curve-implied forward term rate. In other words, in each accrual period the curve-implied OIS floating rate does not differ from the usual curve-implied Libor rate. This fact not only simplifies the valuation of OIS contracts off a given curve but also the reverse process, i.e. the curve generation off given OIS rates.

Spreadsheet Calculation of the Present Value of an Overnight Index Swap

It is possible and indeed quite straightforward to setup a spreadsheet using only built-in Excel formulas that can calculate the exact fair price of a generic OIS with any underlying overnight index, maturity and coupon period length. The image below shows the full set of required formulas, including the input and output data. No external dependencies are involved. The sheet is part of the Excel workbook that is downloadable through the link at the beginning of this article.



Swap Price OUTPUT

Swap PV:	0.00
Fixed Leg PV:	32.79
Float Leg PV:	32.79

Swap INPUT

Today:	10-Nov-20
Notional:	10,000
Fixed Rate:	0.1080%
Tenor (M):	36
Period (M):	12
Settlement:	2
Payment Lag:	2
Basis:	360

Implied Quantities

Swap Start:	13-Nov-20

US Holidays

US Holidays
11-Nov-2020
26-Nov-2020
25-Dec-2020
01-Jan-2021
18-Jan-2021
15-Feb-2021
02-Apr-2021
31-May-2021
05-Jul-2021
06-Sep-2021
11-Oct-2021
11-Nov-2021
25-Nov-2021
24-Dec-2021
17-Jan-2022

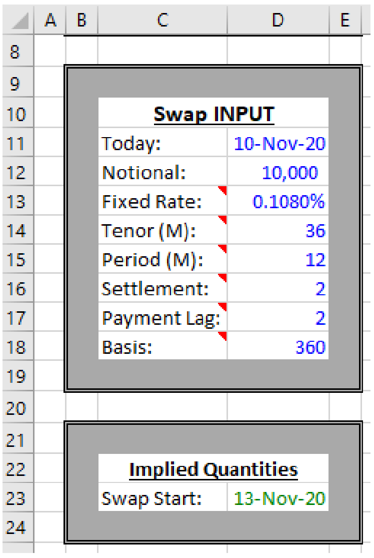
Calculation of Swap Cash Flows including: Accrual Period Dates, Payment Dates, Amounts and Present Values

Months	Proj Dates	Bus Day	Accrual Start	Accrual End	Day Start	Day End	Accr Days	DCF	DF Start	DF End	Fwd Rate	Fixed CF	Float CF	Pmt Date	Day Pmt	DF Pmt	Fixed PV	Float PV
0	13-Nov-20	TRUE	13-Nov-20	15-Nov-21	3	370	367	1.019444	0.999992	0.999161	0.0816%	11.01	8.32	17-Nov-21	372	0.99915647	11.00	8.31
12	13-Nov-21	FALSE	15-Nov-21	14-Nov-22	370	734	364	1.011111	0.999161	0.998491	0.0663%	10.92	6.71	16-Nov-22	736	0.99848687	10.90	6.70
24	13-Nov-22	FALSE	14-Nov-22	13-Nov-23	734	1098	364	1.011111	0.998491	0.996713	0.1764%	10.92	17.84	15-Nov-23	1100	0.99669668	10.88	17.78
36	13-Nov-23	TRUE	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	-	-

Daily DFs
1.000000
0.999997
0.999995
0.999992
0.999989
0.999986
0.999984
0.999981
0.999978
0.999976
0.999973
0.999970
0.999967
0.999964
0.999962
0.999959
0.999956
0.999953
0.999950
0.999947
0.999944
0.999941
0.999938
0.999935
0.999933
0.999930
0.999927
0.999925
0.999923
0.999921
0.999920
0.999918
0.999917
0.999916
0.999915
0.999914
0.999912
0.999911
0.999909

The rows in the middle area correspond to the accrual periods and the respective cash flows. Only the top 3 rows contain numbers because the input on the left specifies a 3-year swap with annual frequency. The calculation logic unwraps from left to right.

Everything starts with the *Swap INPUT* area at the left:

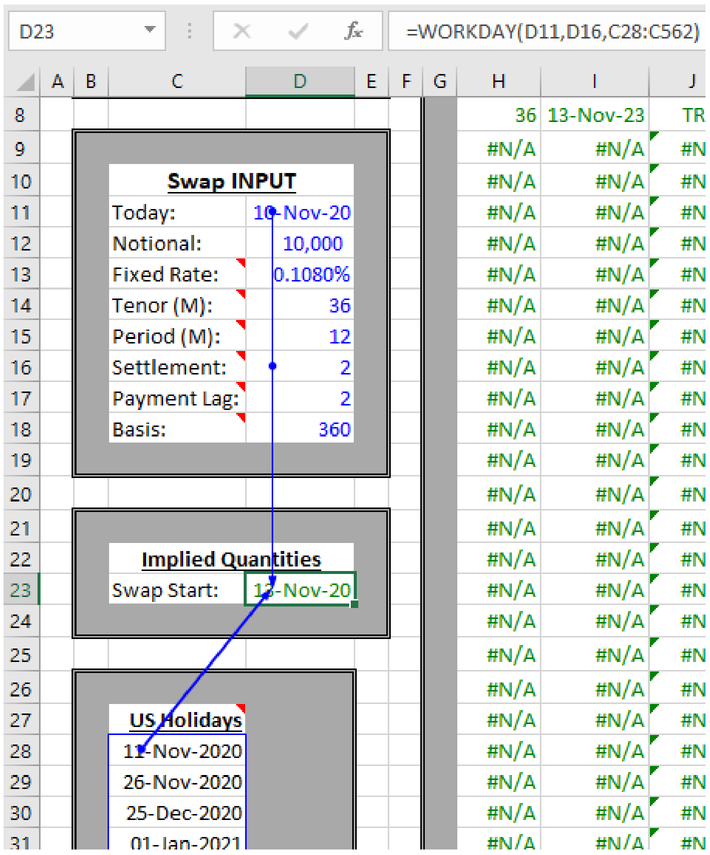


	A	B	C	D	E
8					
9					
10			**Swap INPUT**		
11			Today:	10-Nov-20	
12			Notional:	10,000	
13			Fixed Rate:	0.1080%	
14			Tenor (M):	36	
15			Period (M):	12	
16			Settlement:	2	
17			Payment Lag:	2	
18			Basis:	360	
19					
20					
21					
22			**Implied Quantities**		
23			Swap Start:	13-Nov-20	
24					

The color convention is:
Blue for data representing user input
Green for formulas
Black for text labels and fixed values

The red dotes represent cell notes that explain the meaning of the respective entries. Both the *Tenor* and *Period* are specified by integers that represent the corresponding number of months. The *Tenor* refers to the swap's maturity and the *Period* refers to the accrual interval of each cash flow. The user may enter any plausible values under the constraint that the *Tenor* cannot exceed 36 because the column at the far right (no political statement intended!) contains discount factors for only the next three years. The *Settlement* and *Payment Lag* are defined in business days. Note that a SOFR swap always has *Settlement* = 2 and *Payment Lag* = 2, but the sheet allows a different setting. Same with the *Basis*, which should be 360 for SOFR swaps.

Below the *Input* area, there is a box titled *Implied Quantities* that contains the *Swap Start*, which appears as 13-Nov-20. This is the date when the first accrual period starts. The formula in cell D23 is shown below:

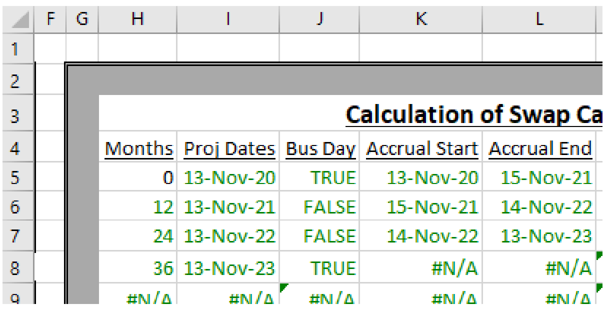


D23 =WORKDAY(D11,D16,C28:C562)

	A	B	C	D	E	F	G	H	I	J
8								36	13-Nov-23	TR
9								#N/A	#N/A	#N
10			**Swap INPUT**					#N/A	#N/A	#N
11			Today:	10-Nov-20				#N/A	#N/A	#N
12			Notional:	10,000				#N/A	#N/A	#N
13			Fixed Rate:	0.1080%				#N/A	#N/A	#N
14			Tenor (M):	36				#N/A	#N/A	#N
15			Period (M):	12				#N/A	#N/A	#N
16			Settlement:	2				#N/A	#N/A	#N
17			Payment Lag:	2				#N/A	#N/A	#N
18			Basis:	360				#N/A	#N/A	#N
19								#N/A	#N/A	#N
20								#N/A	#N/A	#N
21								#N/A	#N/A	#N
22			**Implied Quantities**					#N/A	#N/A	#N
23			Swap Start:	13-Nov-20				#N/A	#N/A	#N
24								#N/A	#N/A	#N
25								#N/A	#N/A	#N
26								#N/A	#N/A	#N
27			**US Holidays**					#N/A	#N/A	#N
28			11-Nov-2020					#N/A	#N/A	#N
29			26-Nov-2020					#N/A	#N/A	#N
30			25-Dec-2020					#N/A	#N/A	#N
31			01-Jan-2021					#N/A	#N/A	#N

This is the built-in Excel formula =WORKDAY(*start*, *n*, *calendar*) that returns the date occurring n business days after start according to the supplied calendar. Here *start* = 10-Nov-20, *n* = 2 and *calendar* is the array C28:C562 that contains all holidays according to the US Government Bond holiday schedule. It returns 13-Nov-20 because the 11-Nov-20 is the US Veterans Day.

Going over to the middle area, the most crucial part consists of the left 5 columns that produce the dates defining the accrual periods:

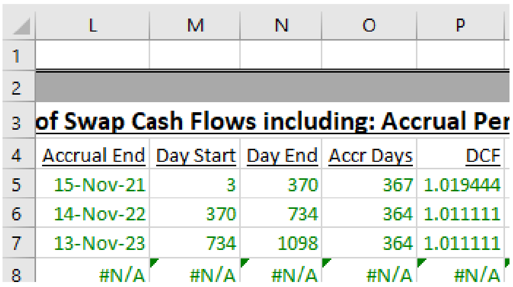


Calculation of Swap Ca

Months	Proj Dates	Bus Day	Accrual Start	Accrual End
0	13-Nov-20	TRUE	13-Nov-20	15-Nov-21
12	13-Nov-21	FALSE	15-Nov-21	14-Nov-22
24	13-Nov-22	FALSE	14-Nov-22	13-Nov-23
36	13-Nov-23	TRUE	#N/A	#N/A
#N/A	#N/A	#N/A	#N/A	#N/A

The projected dates in column I do not yet equal the dates that define the accrual periods because they may fall on non-business days. The next column calculates a Boolean that supplies exactly this information. The final two columns K and L use that Boolean to bump the projected date to the next good business day, if needed.

The next four columns, M to P, are shown below:

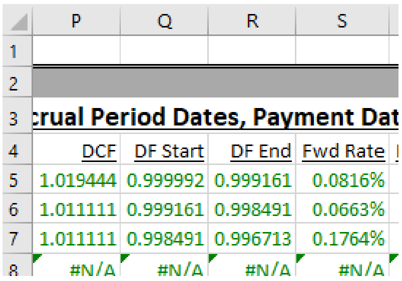


of Swap Cash Flows including: Accrual Per

Accrual End	Day Start	Day End	Accr Days	DCF
15-Nov-21	3	370	367	1.019444
14-Nov-22	370	734	364	1.011111
13-Nov-23	734	1098	364	1.011111
#N/A	#N/A	#N/A	#N/A	#N/A

They lead to the calculation of the DCF (*Day Count Fraction*) of each accrual period, which is simply the period's number of calendar days divided by the basis, which is the number 360 in the SOFR case.

The applicable discount factors are shown below in columns Q and R. Then the forward rate is calculated in column S:

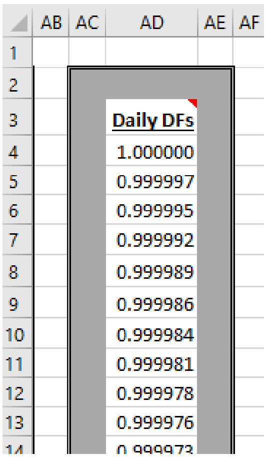


crual Period Dates, Payment Dat

DCF	DF Start	DF End	Fwd Rate
1.019444	0.999992	0.999161	0.0816%
1.011111	0.999161	0.998491	0.0663%
1.011111	0.998491	0.996713	0.1764%
#N/A	#N/A	#N/A	#N/A

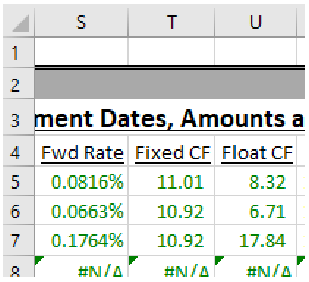
Question: Where do these discount factors come from?

Answer: They are supplied as input. Otherwise the fair price of the given OIS cannot be calculated. Normally, they will be supplied either by an external provider, such as Bloomberg or CME, or will have been extracted by a curve that has been previously built out of observed market rates. In this sheet, the shown discount factors are delivered through a curve that has been previously built out of Bloomberg swap rates observed at 10-Nov-20. They correspond to daily consecutive maturities starting with 10-Nov-20 and are pasted as values in a column as shown below.



	AB	AC	AD	AE	AF
1					
2					
3			Daily DFs		
4			1.000000		
5			0.999997		
6			0.999995		
7			0.999992		
8			0.999989		
9			0.999986		
10			0.999984		
11			0.999981		
12			0.999978		
13			0.999976		
14			0.999973		

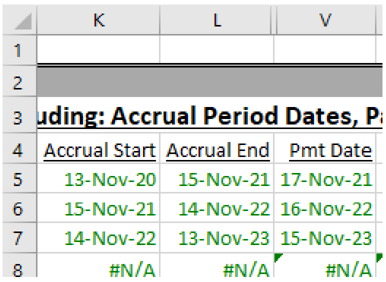
Now all ingredients are available for the calculation of the cash flow amounts. The fixed and floating amounts are given by the formulas $N*R*DCF_i$ and $N*F_i*DCF_i$ respectively, where N is the swap's notional and R is the swap's fixed rate. F_i and DCF_i are the forward rate and the Day Count Fraction of the i^{th} accrual period currently examined. Below are the results:



ment Dates, Amounts a

	S	T	U
4	Fwd Rate	Fixed CF	Float CF
5	0.0816%	11.01	8.32
6	0.0663%	10.92	6.71
7	0.1764%	10.92	17.84
8	#N/A	#N/A	#N/A

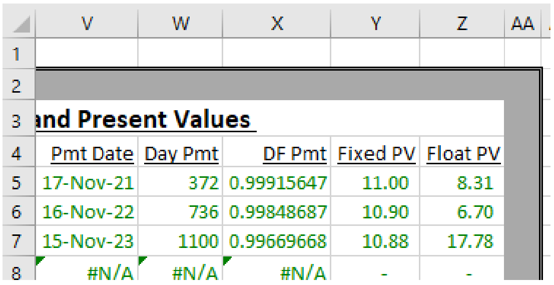
Below are the payment dates, seen together with the corresponding accrual dates for comparison:



uding: Accrual Period Dates, P

	K	L	V
4	Accrual Start	Accrual End	Pmt Date
5	13-Nov-20	15-Nov-21	17-Nov-21
6	15-Nov-21	14-Nov-22	16-Nov-22
7	14-Nov-22	13-Nov-23	15-Nov-23
8	#N/A	#N/A	#N/A

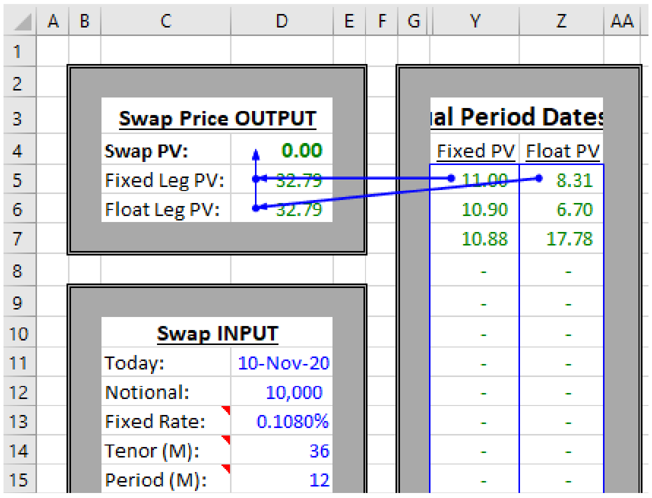
The last 4 columns complete the cash flow table with the discount factors as of the payment dates and the corresponding present values of the fixed and floating amounts:



and Present Values

	V	W	X	Y	Z	AA
4	Pmt Date	Day Pmt	DF Pmt	Fixed PV	Float PV	
5	17-Nov-21	372	0.99915647	11.00	8.31	
6	16-Nov-22	736	0.99848687	10.90	6.70	
7	15-Nov-23	1100	0.99669668	10.88	17.78	
8	#N/A	#N/A	#N/A	-	-	

Notice the present values of the fixed and floating amounts differ from each other. But – what a coincidence (!) – the two sums turn out to be exactly equal! This is shown in the *Output* area that displays the final swap value and the value of each leg:



	Swap Price OUTPUT
Swap PV:	0.00
Fixed Leg PV:	32.79
Float Leg PV:	32.79

	Swap INPUT
Today:	10-Nov-20
Notional:	10,000
Fixed Rate:	0.1080%
Tenor (M):	36
Period (M):	12

al Period Dates

Fixed PV	Float PV
11.00	8.31
10.90	6.70
10.88	17.78
-	-
-	-
-	-
-	-
-	-
-	-
-	-
-	-

The truth is that this result is not a coincidence. It is caused by the input data choice. The swap's fixed rate and tenor have been deliberately set to 0.1080% and 36 months in order for the swap here to match the market swap corresponding to the 3-year Bloomberg rate of 0.1080% that was part of the market rates used in bootstrapping the discount factors pasted in column AD!

In the sequel, native Excel formulas will be abandoned. The Deriscope Excel Add-In will be used instead to build the yield curve implied by market OIS rates. The pricing of any custom OIS will be also revisited.

The Deriscope Excel Add-In

Deriscope is a financial derivatives analytics application that includes an Excel interface in the form of a locally installed Excel Add-In that can perform such tasks as building several types of yield curves and calculating the price and risk of financial products. It delegates the bulk of its analytics routines to the well-known QuantLib C++ open source software library that is used directly or indirectly by all major investment banks and is continuously developed by a large team of developers since the year 2000.

Perhaps Deriscope's greatest advantage is its full and dynamic integration with Excel, where custom formulas can be easily set up to serve any conceivable task, such as scenario analysis and chart generation. This integration's style mirrors the architecture of OO (*object-oriented*) programming languages, such as C++, based on concepts, such as inheritance, abstract classes, static & local methods, virtual functions, objects and even objects' lifetimes and scopes.

OIS Curve Building in Excel using Bloomberg Rates and Deriscope

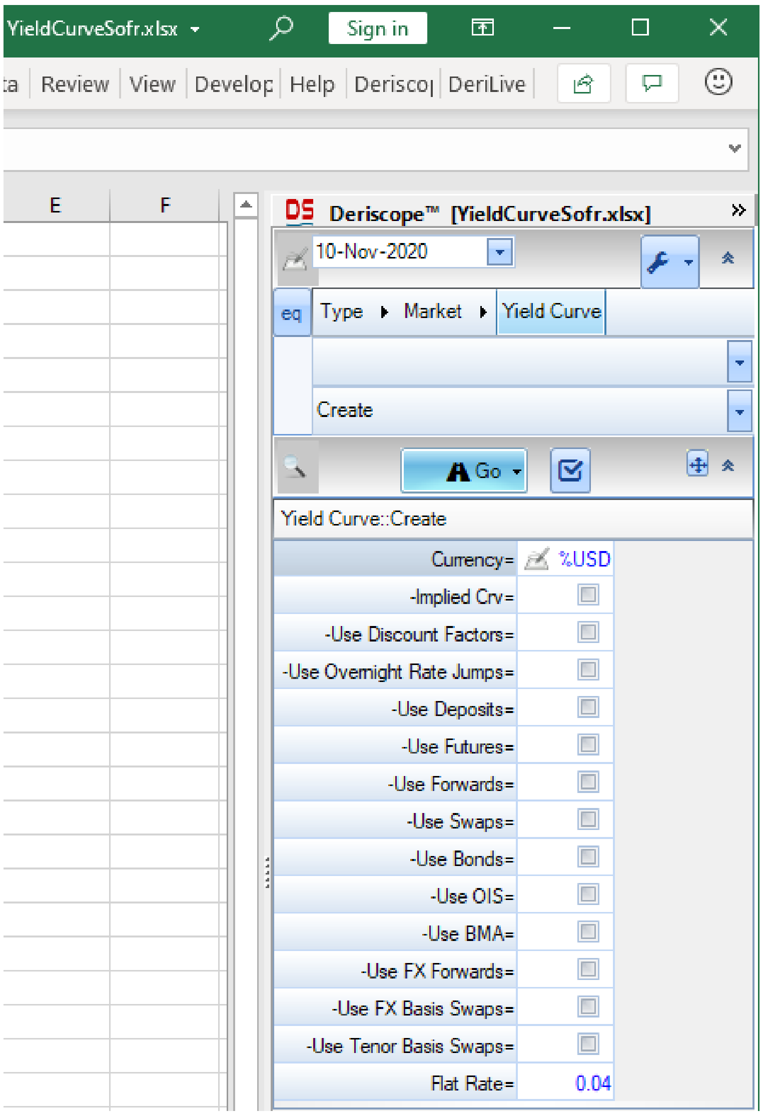
I will now use Deriscope to create a yield curve stripped out of given market SOFR OIS rates. The following video shows how the wizard creates the necessary formulas after 23 seconds with only a handful of mouse clicks:

deriscope.com/btrm/sofr/ois/ois.mp4

In pictures, the formula generation is accomplished in three steps:

Step 1:

Select the element labelled **Yield Curve** in the wizard's *Type Selector* entry bar:



10-Nov-2020

eq Type ▸ Market ▸ Yield Curve

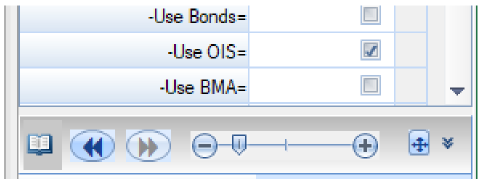
Create

Yield Curve::Create

Currency=	%USD
-Implied Crv=	☐
-Use Discount Factors=	☐
-Use Overnight Rate Jumps=	☐
-Use Deposits=	☐
-Use Futures=	☐
-Use Forwards=	☐
-Use Swaps=	☐
-Use Bonds=	☐
-Use OIS=	☐
-Use BMA=	☐
-Use FX Forwards=	☐
-Use FX Basis Swaps=	☐
-Use Tenor Basis Swaps=	☐
Flat Rate=	0.04

Step 2:

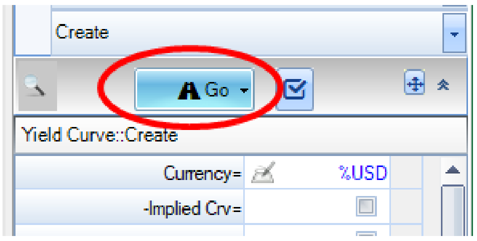
Check the -Use OIS box:



-Use Bonds=	☐
-Use OIS=	✓
-Use BMA=	☐

Step 3:

Click on the Go button:

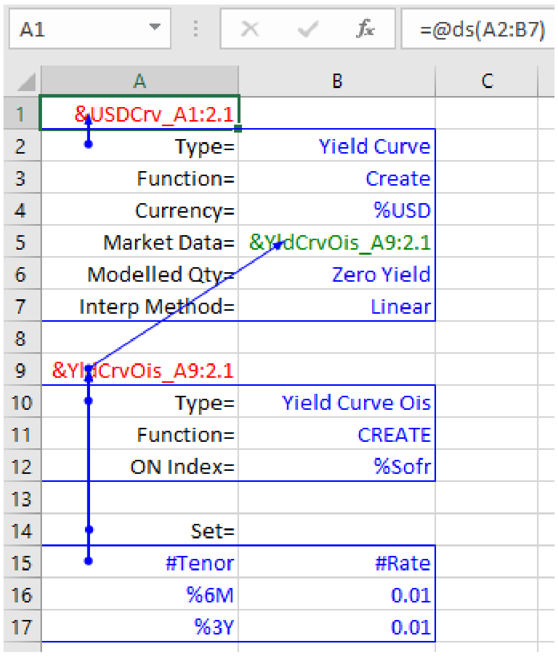


Yield Curve::Create

Currency= %USD

-Implied Crv=

The result is the two formulas shown below with their dependency arrows visible:



A1 =@ds(A2:B7)

	A	B	C
1	&USDCrv_A1:2.1		
2	Type=	Yield Curve	
3	Function=	Create	
4	Currency=	%USD	
5	Market Data=	&YldCrvOis_A9:2.1	
6	Modelled Qty=	Zero Yield	
7	Interp Method=	Linear	
8			
9	&YldCrvOis_A9:2.1		
10	Type=	Yield Curve Ois	
11	Function=	CREATE	
12	ON Index=	%Sofr	
13			
14	Set=		
15	#Tenor	#Rate	
16	%6M	0.01	
17	%3Y	0.01	

As you see above, cell A1 contains the formula =ds(A2:B7).

This is the universal Deriscope formula that has the syntax =ds(RANGE1, RANGE2, ...), where the input ranges contain data in the form of *key/value pairs*. A *key* is a text label ending with = and a *value* can be anything.

While a Deriscope formula may produce anything, most often returns a text that serves as the *handle name* of a corresponding *object* – a binary blob - living in Excel's memory. Here the two formulas have created two *objects* represented by the *handle names* &USDCrv_A1:4.1 and &YldCrvOis_A9:4.1 in cells A1 and A9. The first of these *objects* corresponds to the requested SOFR yield curve in the sense that every possible information one usually associates with a yield curve – for example, the curve of zero rates - can be retrieved from this *object* and displayed on the spreadsheet by means of its *handle name* and an appropriate *function*.

As has been mentioned above, Deriscope's Excel user interface parallels the syntax of the C++ object-oriented language. Just like in C++ every *function* (also known as *method*) is part of a *class*, every function in Deriscope is part of a *Deriscope type*. The two *key/value pairs* Type= Yield Curve and Function= Create identify this relationship. In fact, the value Create is special because it is an accepted *value* for the key Function= for all the *Deriscope types* that are capable of having corresponding *objects*. In effect, the *function* Create corresponds to the C++ constructor of a C++ *non-abstract class*, while *Deriscope types* without that *function* correspond to C++ *abstract classes*!

The *value* associated with the *key* Market Data= is expected to supply the OIS market rates and conventions. Such complex information cannot be conveyed through one or more numbers and should be obviously packaged as an *object*. Here this *object* is represented by the *value* &YldCrvOis_A9:4.1, which is that *object's handle name*.

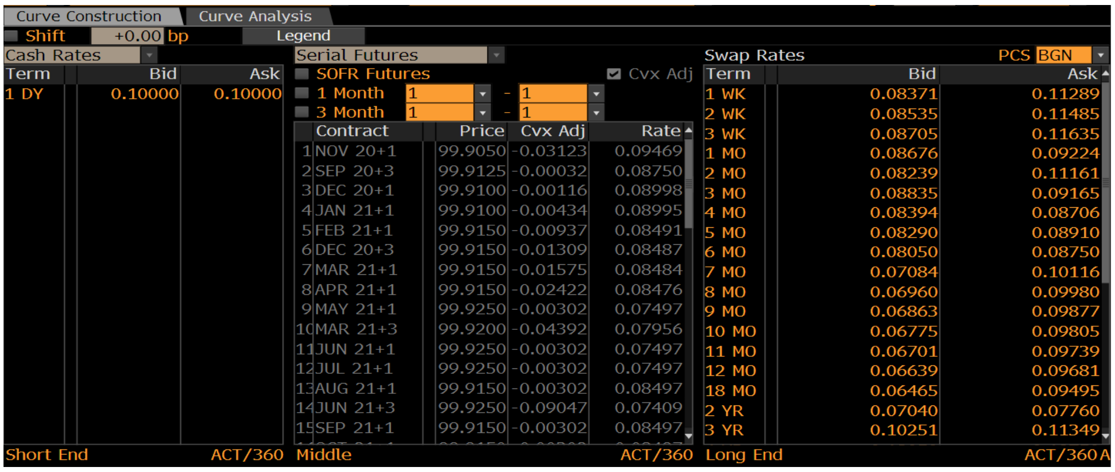
That *value* appears green because it is the output of a simple link to cell A9, where the *handle name* &YldCrvOis_A9:4.1 is produced by the formula =ds(A10:B12 , A14 , A15:B17). The corresponding *object* is of *type* Yield Curve Ois, as seen in cell B10.

The swap rates are supplied in the form of a two-column table containing the titles #Tenor and #Rate. The whole table acts as the *value* associated with the *key* Set=.

Note the shown rates of 0.01 are dummy *default values* generated by the wizard for the user's convenience and are not linked to any market data source. Their only utility is to be reasonable enough for allowing the Deriscope formulas to complete their task of creating a corresponding *Yield Curve object*. It is the user's task to replace these *default values* with real data received from a trusted data source, such as Bloomberg.

Incorporation of the Bloomberg Swap Rates

In the current case, the dummy Deriscope-generated swap rates will be replaced with Bloomberg data as of November 10, 2020. Below is the Bloomberg page with the SOFR OIS rates for maturities up to 3 years. The remaining rates are part of the downloadable spreadsheet at the beginning of this article.



Cash Rates

Term	Bid	Ask
1 DY	0.10000	0.10000

Short End ACT/360

Serial Futures

SOFR Futures ✓ Cvx Adj
1 Month 1 - 1
3 Month 1 - 1

	Contract	Price	Cvx Adj	Rate
1	NOV 20+1	99.9050	-0.03123	0.09469
2	SEP 20+3	99.9125	-0.00032	0.08750
3	DEC 20+1	99.9100	-0.00116	0.08998
4	JAN 21+1	99.9100	-0.00434	0.08995
5	FEB 21+1	99.9150	-0.00937	0.08491
6	DEC 20+3	99.9150	-0.01309	0.08487
7	MAR 21+1	99.9150	-0.01575	0.08484
8	APR 21+1	99.9150	-0.02422	0.08476
9	MAY 21+1	99.9250	-0.00302	0.07497
10	MAR 21+3	99.9200	-0.04392	0.07956
11	JUN 21+1	99.9250	-0.00302	0.07497
12	JUL 21+1	99.9250	-0.00302	0.07497
13	AUG 21+1	99.9150	-0.00302	0.08497
14	JUN 21+3	99.9250	-0.09047	0.07409
15	SEP 21+1	99.9150	-0.00302	0.08497

Middle ACT/360

Swap Rates PCS BGN

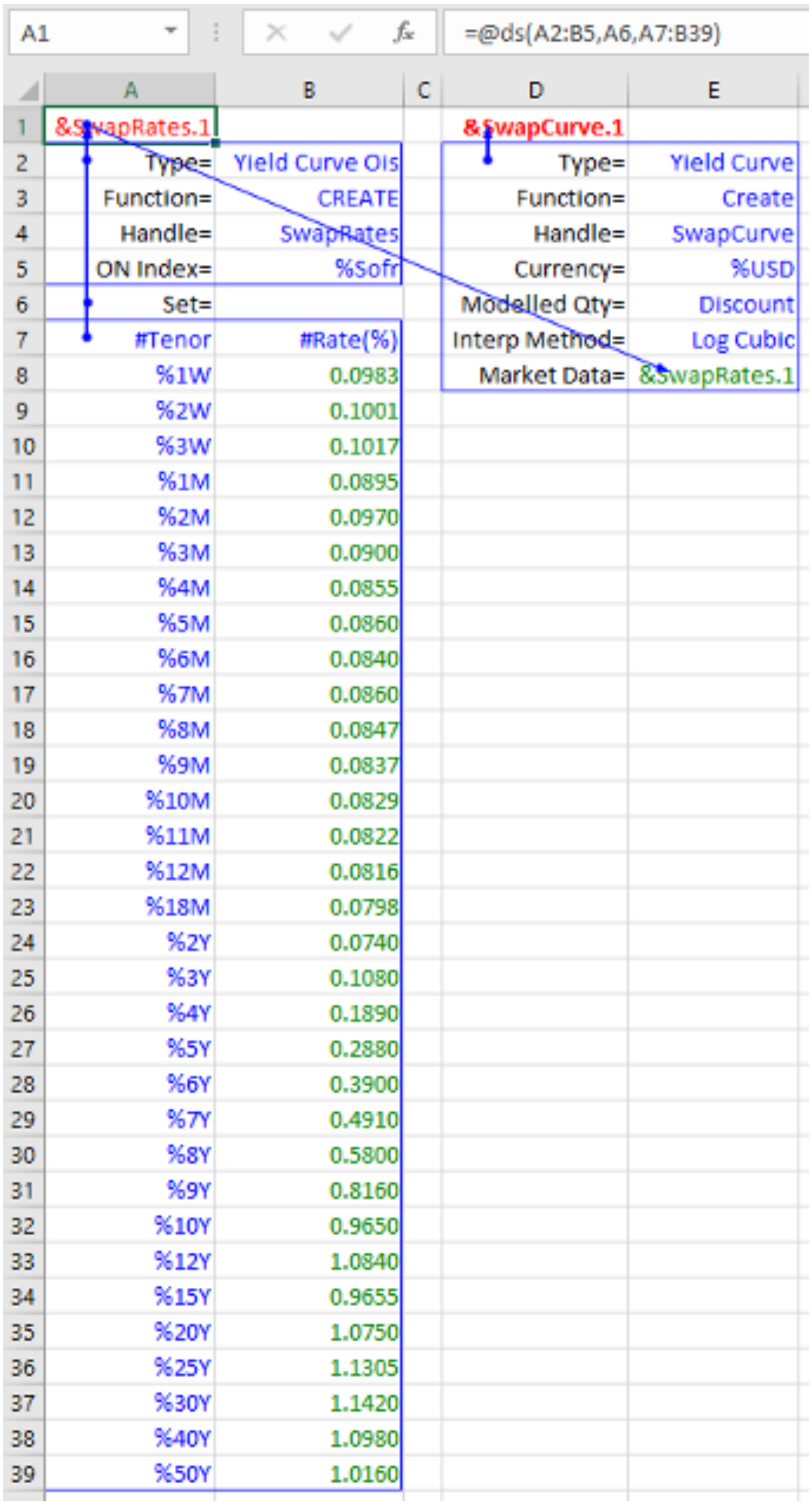
Term	Bid	Ask
1 WK	0.08371	0.11289
2 WK	0.08535	0.11485
3 WK	0.08705	0.11635
1 MO	0.08676	0.09224
2 MO	0.08239	0.11161
3 MO	0.08835	0.09165
4 MO	0.08394	0.08706
5 MO	0.08290	0.08910
6 MO	0.08050	0.08750
7 MO	0.07084	0.10116
8 MO	0.06960	0.09980
9 MO	0.06863	0.09877
10 MO	0.06775	0.09805
11 MO	0.06701	0.09739
12 MO	0.06639	0.09681
18 MO	0.06465	0.09495
2 YR	0.07040	0.07760
3 YR	0.10251	0.11349

Long End ACT/360A

The average of the Bid/Ask quotes will be used.

Below are the previous two formulas after the substitution has taken place. The added suffix **(%)** in the #Rate title indicates that all numbers in the respective column are in percentage scale.

Note also the new *key/value pairs* Modelled Qty= Discount and Interp Method= Log Cubic that result in a smoother curve that is more aligned with market practice. The optional *key/value pairs* Handle= SwapRates and Handle= SwapCurve fix the *handle names* of the respective created *objects* so that they are easier recognizable.

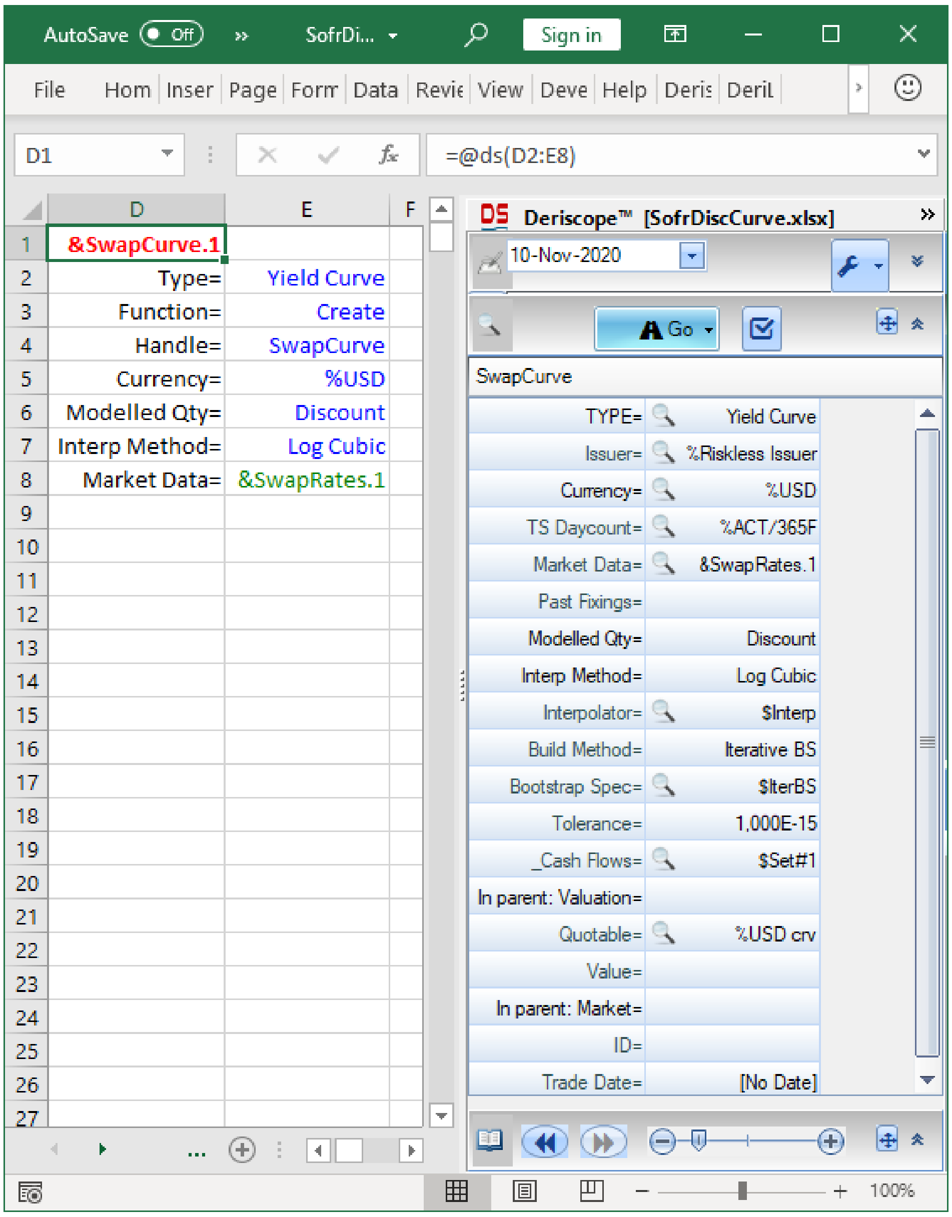


A1 =@ds(A2:B5,A6,A7:B39)

	A	B	C	D	E
1	&SwapRates.1			&SwapCurve.1	
2	Type=	Yield Curve Ois		Type=	Yield Curve
3	Function=	CREATE		Function=	Create
4	Handle=	SwapRates		Handle=	SwapCurve
5	ON Index=	%Sofr		Currency=	%USD
6	Set=			Modelled Qty=	Discount
7	#Tenor	#Rate(%)		Interp Method=	Log Cubic
8	%1W	0.0983		Market Data=	&SwapRates.1
9	%2W	0.1001			
10	%3W	0.1017			
11	%1M	0.0895			
12	%2M	0.0970			
13	%3M	0.0900			
14	%4M	0.0855			
15	%5M	0.0860			
16	%6M	0.0840			
17	%7M	0.0860			
18	%8M	0.0847			
19	%9M	0.0837			
20	%10M	0.0829			
21	%11M	0.0822			
22	%12M	0.0816			
23	%18M	0.0798			
24	%2Y	0.0740			
25	%3Y	0.1080			
26	%4Y	0.1890			
27	%5Y	0.2880			
28	%6Y	0.3900			
29	%7Y	0.4910			
30	%8Y	0.5800			
31	%9Y	0.8160			
32	%10Y	0.9650			
33	%12Y	1.0840			
34	%15Y	0.9655			
35	%20Y	1.0750			
36	%25Y	1.1305			
37	%30Y	1.1420			
38	%40Y	1.0980			
39	%50Y	1.0160			

Inspecting the Contents of the created Yield Curve object

Since a *handle name* acts as the spreadsheet ambassador of the memory-lived binary blob, it must facilitate the inspection of that blob's contents. Deriscope makes this possible as follows: The curious user needs only to select the cell that contains the *handle name* while the wizard is displayed. Then the wizard reacts by displaying in its so-called *Browse Area* the contents of the corresponding binary blob as a collection of *key/value pairs*. The image below shows how the wizard displays the contents of the *object* referenced by the *handle name* &SwapCurve.1:



=@ds(D2:E8)

	D	E
1	&SwapCurve.1	
2	Type=	Yield Curve
3	Function=	Create
4	Handle=	SwapCurve
5	Currency=	%USD
6	Modelled Qty=	Discount
7	Interp Method=	Log Cubic
8	Market Data=	&SwapRates.1

Deriscope™ [SofrDiscCurve.xlsx]

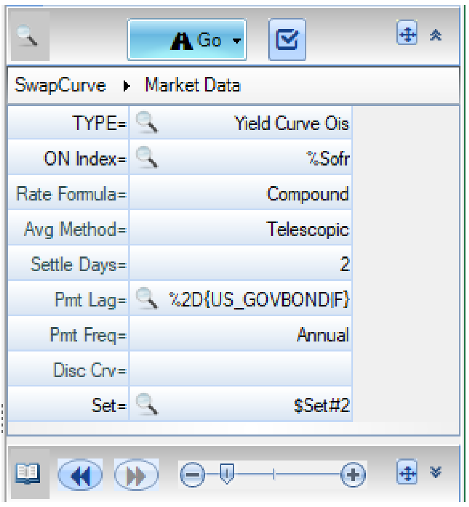
10-Nov-2020

SwapCurve

TYPE=	Yield Curve
Issuer=	%Riskless Issuer
Currency=	%USD
TS Daycount=	%ACT/365F
Market Data=	&SwapRates.1
Past Fixings=	
Modelled Qty=	Discount
Interp Method=	Log Cubic
Interpolator=	$Interp
Build Method=	Iterative BS
Bootstrap Spec=	$IterBS
Tolerance=	1,000E-15
_Cash Flows=	$Set#1
In parent: Valuation=	
Quotable=	%USD crv
Value=	
In parent: Market=	
ID=	
Trade Date=	[No Date]

The *keys* are shown on the left column as text labels ending with =. The corresponding *values* are shown on the second column. The *keys* appearing dim represent *optional input* that has not been explicitly defined in the input range D2:E8. When *optional keys* are not supplied as input to the ds formula, they are still considered as implied input with *default values* decided by Deriscope. Inspecting the *object* as above reveals the *default values* of the missing *optional keys*. This handling mirrors the C++ feature of *default function parameters*!

A few of the *values* are *handle names* of *objects* that can be inspected by clicking on the little lens sign on their left. For example, clicking on the lens sign of the *object* named &SwapRates.1 associated with the Market Data= *key*, the wizard displays the following:



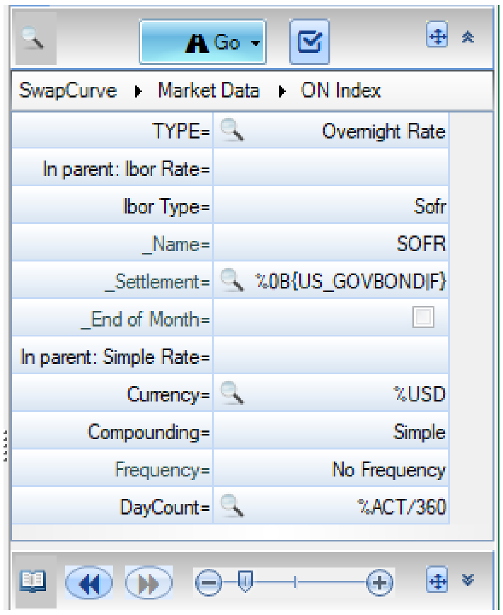
TYPE=	Yield Curve Ois
ON Index=	%Sofr
Rate Formula=	Compound
Avg Method=	Telescopic
Settle Days=	2
Pmt Lag=	%2D{US_GOVBOND|F}
Pmt Freq=	Annual
Disc Crv=	
Set=	$Set#2

This screen shows the contents kept in memory of the *object* &SwapRates.1 originally created by the spreadsheet formula in cell A1. Its shown *default values* can inform the user about the swap rates conventions.

Starting from the top, the *key/value pairs* are:

ON Index= %Sofr

This defines the overnight index underlying the OIS contracts as an *object* with the *handle name* %Sofr, the details of which can be shown by clicking on its lens sign. Below is the respective screenshot that assures the user that this overnight rate has the correct conventions, i.e. spot settlement, US Government Bond calendar and ACT/360 daycount.

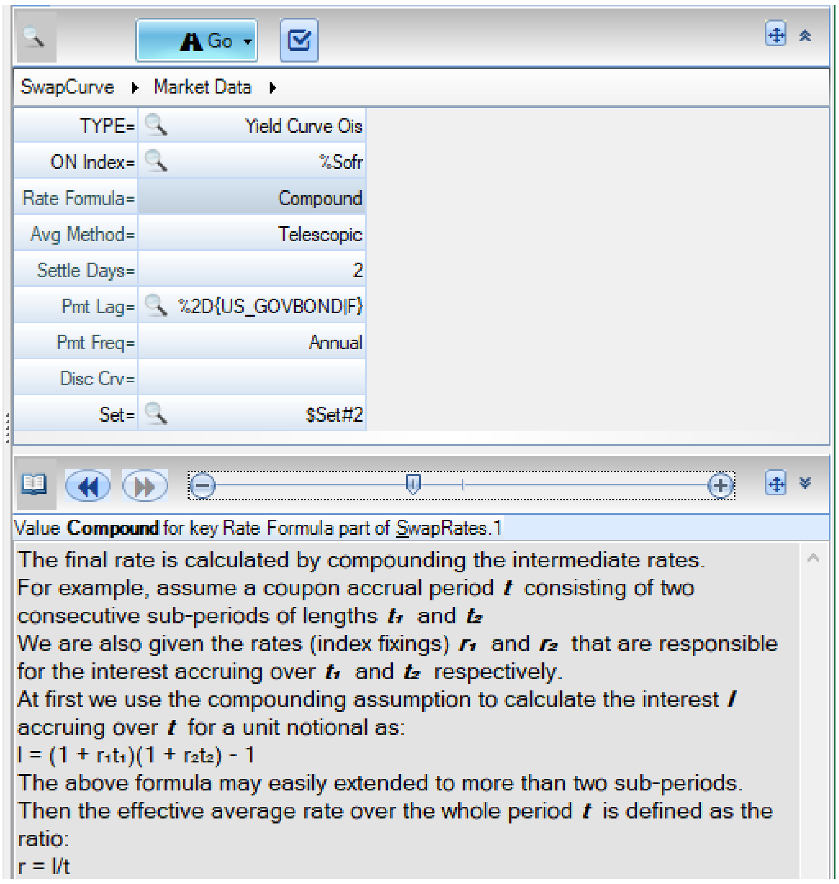


TYPE=	Overnight Rate
In parent: Ibor Rate=	
Ibor Type=	Sofr
_Name=	SOFR
_Settlement=	%0B{US_GOVBOND|F}
_End of Month=	☐
In parent: Simple Rate=	
Currency=	%USD
Compounding=	Simple
Frequency=	No Frequency
DayCount=	%ACT/360

Rate Formula= Compound

This defines the effective index on the OIS floating legs as a *compounded average* of the daily overnight rates.

A detailed description about the *value* Compound can be displayed by selecting the containing cell, as shown below:



TYPE=	Yield Curve Ois
ON Index=	%Sofr
Rate Formula=	Compound
Avg Method=	Telescopic
Settle Days=	2
Pmt Lag=	%2D{US_GOVBOND|F}
Pmt Freq=	Annual
Disc Crv=	
Set=	$Set#2

Value **Compound** for key Rate Formula part of SwapRates.1

The final rate is calculated by compounding the intermediate rates.
For example, assume a coupon accrual period t consisting of two consecutive sub-periods of lengths t_1 and t_2
We are also given the rates (index fixings) r_1 and r_2 that are responsible for the interest accruing over t_1 and t_2 respectively.
At first we use the compounding assumption to calculate the interest I accruing over t for a unit notional as:
$I = (1 + r_1t_1)(1 + r_2t_2) - 1$
The above formula may easily extended to more than two sub-periods.
Then the effective average rate over the whole period t is defined as the ratio:
$r = I/t$

Avg Method= Telescopic

This is a technical feature that optimizes the floating accrual calculation.

Settle Days= 2

This defines the OIS settlement to be 2 business days, meaning the two swap legs start accruing 2 business days after the swap's trading date.

Pmt Lag= %2D{USGOVBOND|F}

This defines the payment lag of each cash flow from the end of its respective accrual period as 2 business days in accordance with the US Government Bond calendar and the *Following* date bump convention.

Pmt Freq= Annual

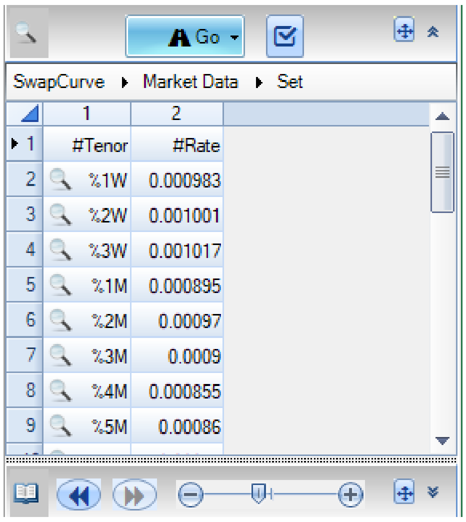
This defines the payment frequency on both legs.

Disc Curve= <blank>

This informs the user that no exogeneous discounting curve is considered during bootstrapping.

Set= $Set#2

This is the most important entry, since it contains the market swap rates. After clicking on the lens sign, all swap rates originally entered as input to the spreadsheet formula are displayed, albeit in standard (non-percentage) scale:



	1	2
1	#Tenor	#Rate
2	%1W	0.000983
3	%2W	0.001001
4	%3W	0.001017
5	%1M	0.000895
6	%2M	0.00097
7	%3M	0.0009
8	%4M	0.000855
9	%5M	0.00086

Sanity Check: Replicating the Market Price of an Overnight Index Swap

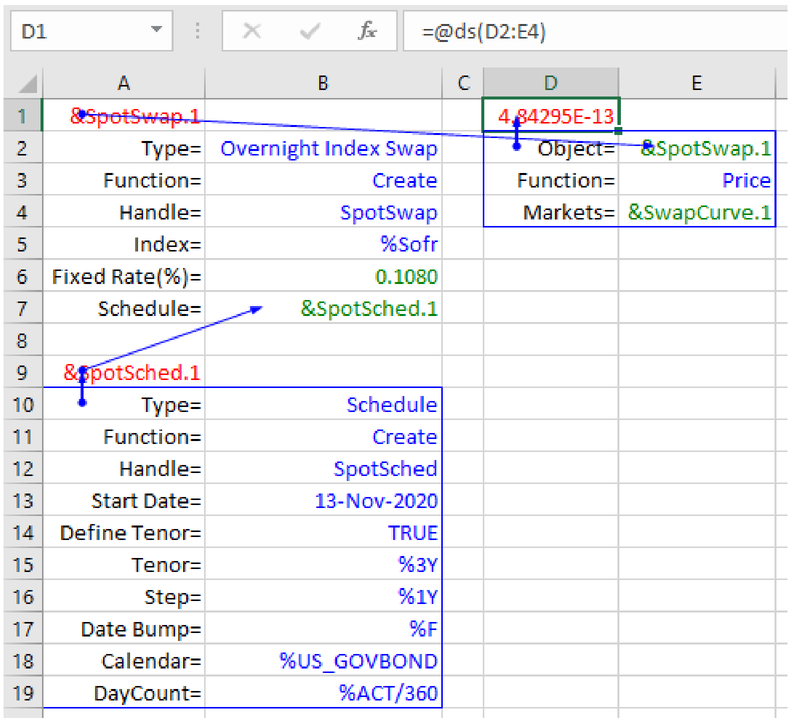
An important requirement that the produced *Yield Curve object* must fulfil is the ability to imply fair swap rates that match the Bloomberg market rates used as input in its construction. Rather than checking the equality of implied and input swap rates, it is easier to verify that the implied prices of the market swaps equal zero. This can be done in two ways:

The long and complicated way is to first produce the discount factors implied by the given *Yield Curve object* for all relevant maturities. Then a spreadsheet can be built that calculates the fair price of some selected market swap. In effect, this was done above with the spreadsheet pricing of a 3-year swap.

The second way is much simpler and relies on the Deriscope's *Price function* that is made available for all *objects* of *Deriscope type* equal to *Tradable*. All the test involves is creating an *object* of *type Overnight Index Swap* followed by running the *Price function* on that *object*. The following video shows how this is achieved with a few mouse clicks in 39 seconds:

deriscope.com/btrm/sofr/ois/price.mp4

The result is shown below:



D1 =@ds(D2:E4)

	A	B	C	D	E
1	&SpotSwap.1			4.84295E-13	
2	Type=	Overnight Index Swap		Object=	&SpotSwap.1
3	Function=	Create		Function=	Price
4	Handle=	SpotSwap		Markets=	&SwapCurve.1
5	Index=	%Sofr			
6	Fixed Rate(%)=	0.1080			
7	Schedule=	&SpotSched.1			
8					
9	&SpotSched.1				
10	Type=	Schedule			
11	Function=	Create			
12	Handle=	SpotSched			
13	Start Date=	13-Nov-2020			
14	Define Tenor=	TRUE			
15	Tenor=	%3Y			
16	Step=	%1Y			
17	Date Bump=	%F			
18	Calendar=	%US_GOVBOND			
19	DayCount=	%ACT/360			

The wizard has set most *values* in a way that they are compatible with the nature of the underlying overnight index SOFR. The only manual adjustments to the wizard-generated formulas have been the following three:

a) The original wizard-generated dummy curve *handle* in cell E4 has been replaced with a link to the existing curve *handle* &SwapCurve.1, since the objective is to calculate the swap's price with respect to that curve.

b) The Tenor *value* in cell B15 has been set to %3Y in order to setup a 3-year swap.

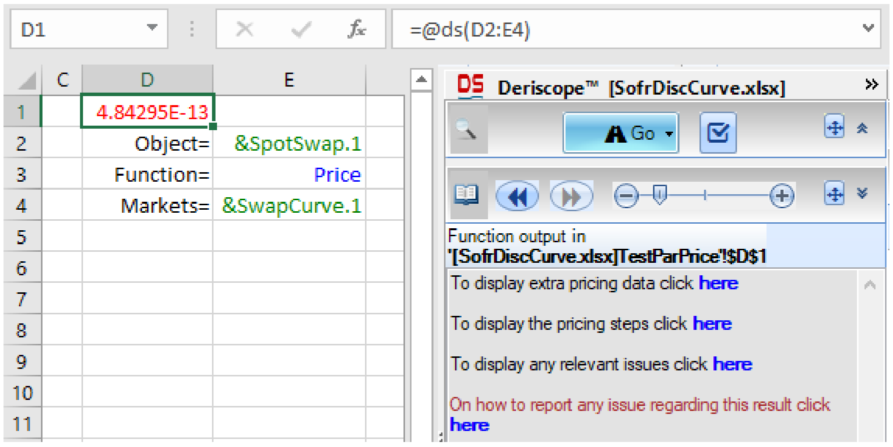
c) The Fixed Rate *value* in cell B6 has been set to the Bloomberg 3-year rate of 0.1080% so that the swap matches the market 3-year swap.

The calculated price shown in cell D1 is considered numerically equal to 0, which means the test has been successful.

Parenthetically, note that no *key* named *Type=* exists in the formula where the price is produced. Instead, the *key/value pair Object=* &SpotSwap.1 exists and is the one that defines the context of the *Price function*. In fact, every Deriscope *function* must be accompanied by either the *Type* or *Object key* in order to know where the function applies. This points to another great parallel with the C++ language, where the various *functions* (*methods*) can be either *static* or *local*. In Deriscope, the *functions* are also distinguished in being either *Static* or *Local*. A *Local function* (like *Price*) always references a specific *object* and therefore requires the concurrent specification of the *key* named *Object=*, while a *Static* function only needs the specification of the *key* named *Type=*.

Additional Output of the Price Function

Primarily, the *Price function* returns the … price. But as the pricing algorithm runs its course, several other quantities of interest are generated and can be optionally reported. A simple way to quickly glimpse over such additional quantities is by selecting the cell where the price is displayed, while the wizard is open. For example, selecting the cell D1, the following appears inside the *Info Area* at the bottom of the wizard:



D1 =@ds(D2:E4)

	D	E
1	4.84295E-13	
2	Object=	&SpotSwap.1
3	Function=	Price
4	Markets=	&SwapCurve.1

Deriscope™ [SofrDiscCurve.xlsx]

Function output in
'[SofrDiscCurve.xlsx]TestParPrice'!D1

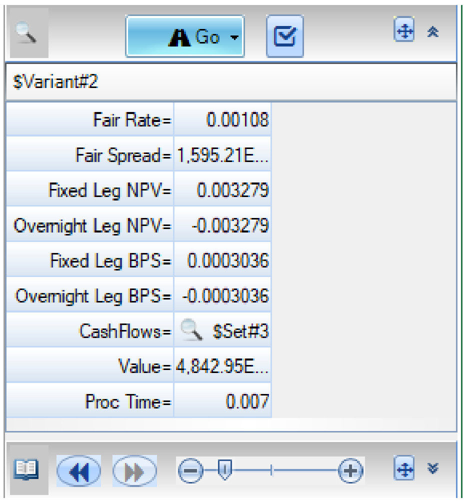
To display extra pricing data click here

To display the pricing steps click here

To display any relevant issues click here

On how to report any issue regarding this result click here

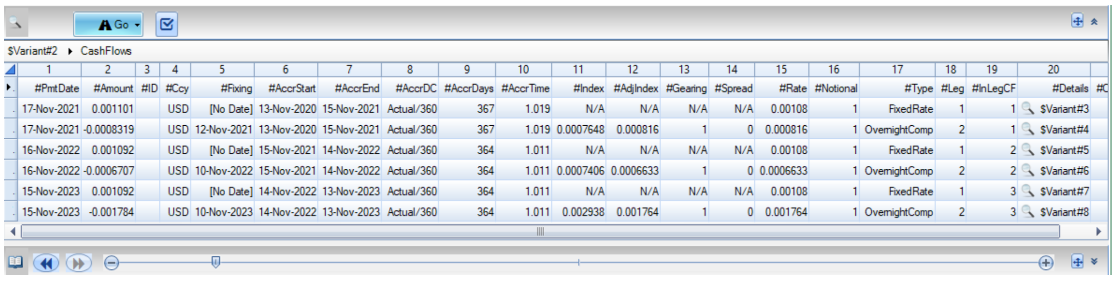
In effect, the wizard suggests to the user a few possible actions relating to the currently selected pricing outcome. The top sentence reads "*To display extra pricing data click* here". After clicking on here, the following appears:



$Variant#2

Fair Rate=	0.00108
Fair Spread=	1,595.21E...
Fixed Leg NPV=	0.003279
Overnight Leg NPV=	-0.003279
Fixed Leg BPS=	0.0003036
Overnight Leg BPS=	-0.0003036
CashFlows=	$Set#3
Value=	4,842.95E...
Proc Time=	0.007

Reported are the *Fair Rate, Fair Spread, NPV* and *BPS* of both legs, *Cash Flows, Value* and *Processing Time* in seconds. The *Cash Flows value* is itself an *object*, of which the contents can be displayed by clicking on the lens sign, as below:



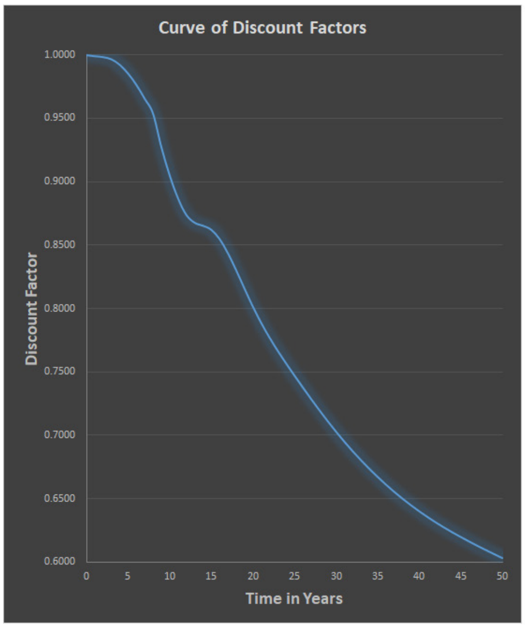
$Variant#2 ▸ CashFlows

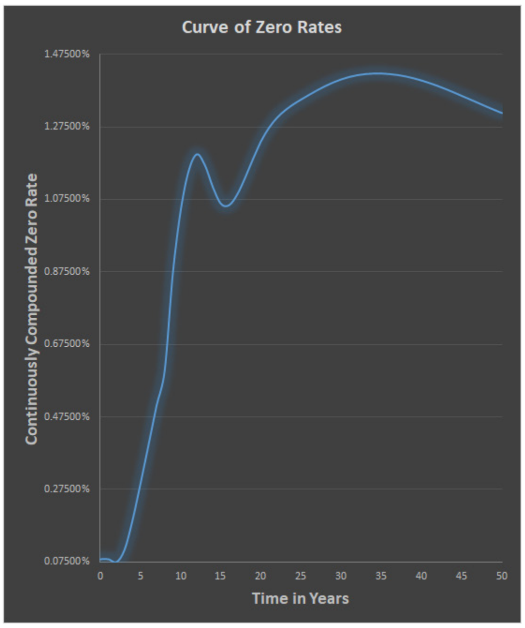
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
#PmtDate	#Amount	#ID	#Ccy	#Fixing	#AccrStart	#AccrEnd	#AccrDC	#AccrDays	#AccrTime	#Index	#AdjIndex	#Gearing	#Spread	#Rate	#Notional	#Type	#Leg	#InLegCF	#Details
17-Nov-2021	0.001101		USD	[No Date]	13-Nov-2020	15-Nov-2021	Actual/360	367	1.019	N/A	N/A	N/A	N/A	0.00108	1	FixedRate	1	1	$Variant#3
17-Nov-2021	-0.0008319		USD	12-Nov-2021	13-Nov-2020	15-Nov-2021	Actual/360	367	1.019	0.0007648	0.000816	1	0	0.000816	1	OvernightComp	2	1	$Variant#4
16-Nov-2022	0.001092		USD	[No Date]	15-Nov-2021	14-Nov-2022	Actual/360	364	1.011	N/A	N/A	N/A	N/A	0.00108	1	FixedRate	1	2	$Variant#5
16-Nov-2022	-0.0006707		USD	10-Nov-2022	15-Nov-2021	14-Nov-2022	Actual/360	364	1.011	0.0007406	0.0006633	1	0	0.0006633	1	OvernightComp	2	2	$Variant#6
15-Nov-2023	0.001092		USD	[No Date]	14-Nov-2022	13-Nov-2023	Actual/360	364	1.011	N/A	N/A	N/A	N/A	0.00108	1	FixedRate	1	3	$Variant#7
15-Nov-2023	-0.001784		USD	10-Nov-2023	14-Nov-2022	13-Nov-2023	Actual/360	364	1.011	0.002938	0.001764	1	0	0.001764	1	OvernightComp	2	3	$Variant#8

Each row corresponds to a fixed or floating cash flow. Several interesting data are displayed that include the *payment date*, the *start* and *end date* of the accrual period, the applicable *rate* (*compounded average* in the case of floating cash flows) and the projected *amount*.

Generating the Curve of Implied Discount Factors and Zero Rates

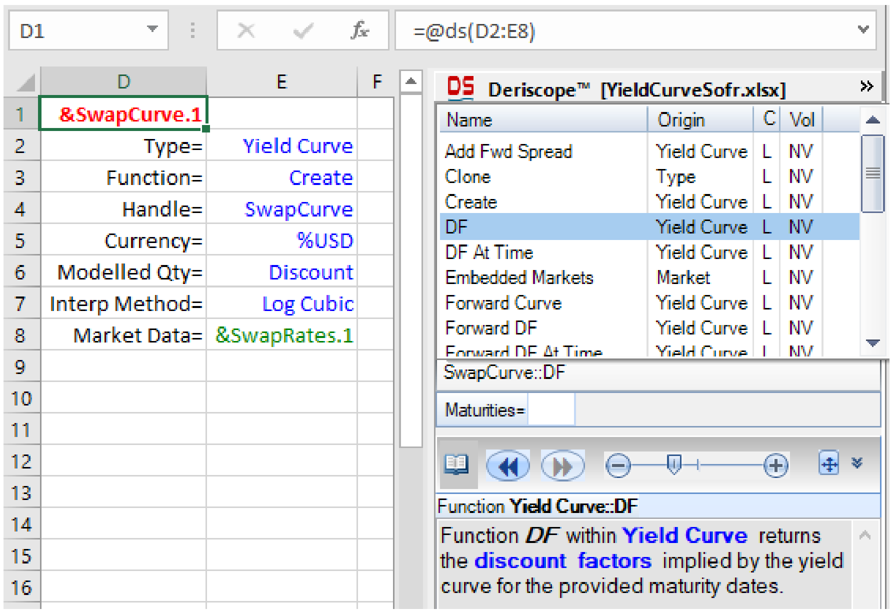
Most people in the industry understand by *yield curve* or *curve* a geometrical line like these two examples below:





From a technology perspective, the objective of curve generation is almost never the direct creation of such curves, as this would represent a very inefficient way of representing the thousands of *discount factors* or *zero rates* from today until the latest date of interest that often lies 50 years in the future. This is also Deriscope's approach, where the produced *Yield Curve object* contains neither *discount factors* nor *zero rates*! It is a *binary entity* that contains only the logic for calculating *discount factors* for specific *maturities* upon explicit request. In fact, this is how the object &SwapCurve.1 is used by the pricing algorithm launched through the formula in cell D1. When the algorithm hits a point where the *discount factor* for a specific *maturity* T is needed, a request is sent to the curve *object*, which responds with the corresponding *discount factor* for *maturity* T.

Nevertheless, it is often useful to display on the spreadsheet charts with time-dependent curve-implied quantities, such as *discount factors, zero rates* or *forward rates*. This is easily achieved by means of various functions that are *Local* (see above for the *Local* meaning in Deriscope) to *objects* of *type Yield Curve*. For example, one such *function* is called *DF* and can be accessed most easily through the wizard, by selecting it through the *Function Selector* as shown below. Note the *Function Selector* displays all *functions* applicable to the currently selected *object*, which is the &SwapCurve.1 in cell D1:


D1 =@ds(D2:E8)

	D	E	F
1	&SwapCurve.1		
2	Type=	Yield Curve	
3	Function=	Create	
4	Handle=	SwapCurve	
5	Currency=	%USD	
6	Modelled Qty=	Discount	
7	Interp Method=	Log Cubic	
8	Market Data=	&SwapRates.1	
9			
10			
11			
12			
13			
14			
15			
16			

DS Deriscope™ [YieldCurveSofr.xlsx]

Name	Origin	C	Vol
Add Fwd Spread	Yield Curve	L	NV
Clone	Type	L	NV
Create	Yield Curve	L	NV
DF	Yield Curve	L	NV
DF At Time	Yield Curve	L	NV
Embedded Markets	Market	L	NV
Forward Curve	Yield Curve	L	NV
Forward DF	Yield Curve	L	NV
Forward DF At Time	Yield Curve	L	NV

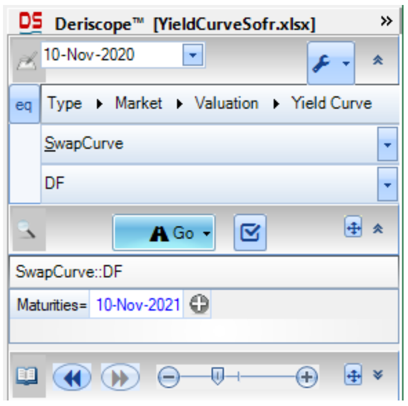
SwapCurve::DF

Maturities=

Function **Yield Curve::DF**

Function *DF* within **Yield Curve** returns the **discount factors** implied by the yield curve for the provided maturity dates.

Below is, how the wizard appears after the *DF* has been selected:


Deriscope™ [YieldCurveSofr.xlsx]

10-Nov-2020

Type ▸ Market ▸ Valuation ▸ Yield Curve

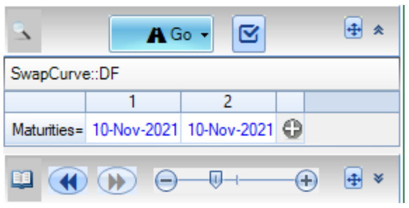
SwapCurve

DF

SwapCurve::DF

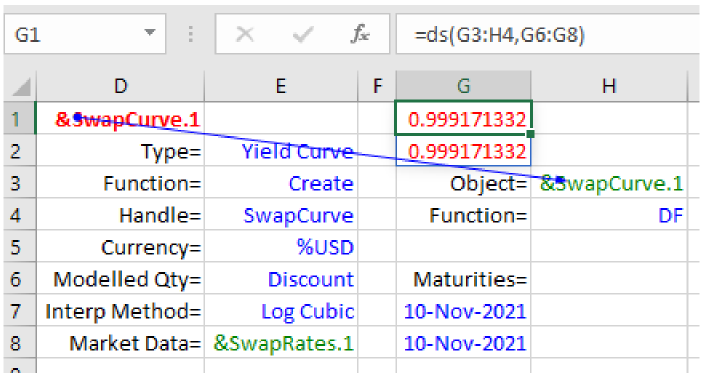
Maturities= 10-Nov-2021

The + sign indicates the *value* associated with the *key Maturity*= is an array. By default, this array is initialized by the wizard with only one element. It may be sensible to increase the array length by adding one more element so that the generated spreadsheet formula is pasted in the spreadsheet as a dynamic array. This can be accomplished by clicking on the + sign:


SwapCurve::DF

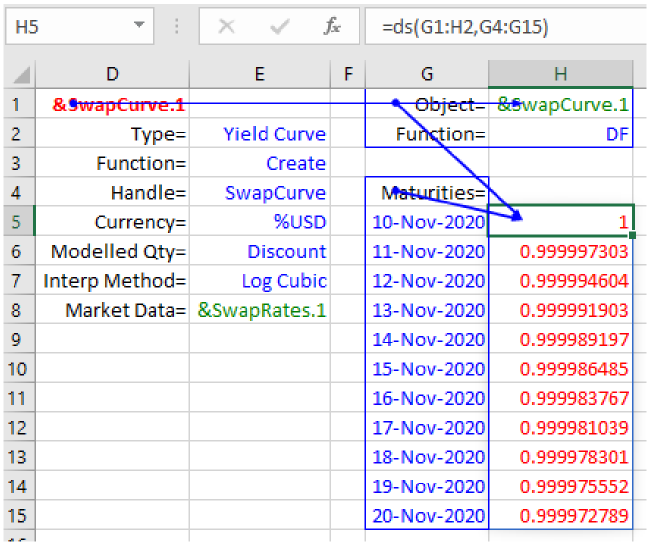
	1	2
Maturities=	10-Nov-2021	10-Nov-2021

The final act is clicking on the Go button in order to insert the corresponding wizard-generated formula in the currently selected cell G1 of the spreadsheet:


G1 =ds(G3:H4,G6:G8)

	D	E	F	G	H
1	&SwapCurve.1			0.999171332	
2	Type=	Yield Curve		0.999171332	
3	Function=	Create		Object=	&SwapCurve.1
4	Handle=	SwapCurve		Function=	DF
5	Currency=	%USD			
6	Modelled Qty=	Discount		Maturities=	
7	Interp Method=	Log Cubic		10-Nov-2021	
8	Market Data=	&SwapRates.1		10-Nov-2021	

As shown above, cell G1 contains the formula =ds(G3:H4,G6:G8) as a dynamic array formula. From this point on, it is the user's job to add additional maturities in the column below the *key Maturities*=. For example, 10 consecutive calendar dates have been added in the image below, while the output dynamic array column has been shifted and placed next to the column containing the maturities:

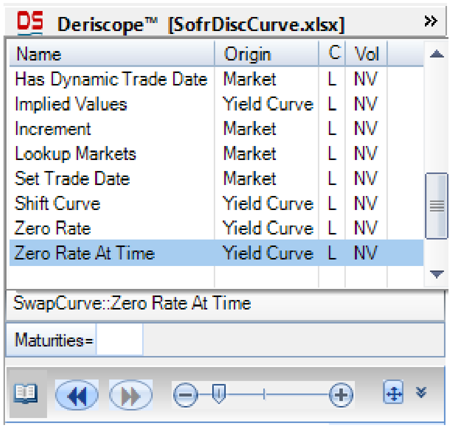


H5 =ds(G1:H2,G4:G15)

	D	E	F	G	H
1	&SwapCurve.1			Object=	&SwapCurve.1
2	Type=	Yield Curve		Function=	DF
3	Function=	Create			
4	Handle=	SwapCurve		Maturities=	
5	Currency=	%USD		10-Nov-2020	1
6	Modelled Qty=	Discount		11-Nov-2020	0.999997303
7	Interp Method=	Log Cubic		12-Nov-2020	0.999994604
8	Market Data=	&SwapRates.1		13-Nov-2020	0.999991903
9				14-Nov-2020	0.999989197
10				15-Nov-2020	0.999986485
11				16-Nov-2020	0.999983767
12				17-Nov-2020	0.999981039
13				18-Nov-2020	0.999978301
14				19-Nov-2020	0.999975552
15				20-Nov-2020	0.999972789

It is a trivial exercise to extend the maturities all the way to a desired horizon of X years and have the ds formula returning the corresponding discount factors, which could then be used as input to the company's pricing and risk management systems.

A quick chart of the long term implied *zero rates* can be produced by the function *Zero Rate at Time* shown in the wizard's *Function Selector* below:



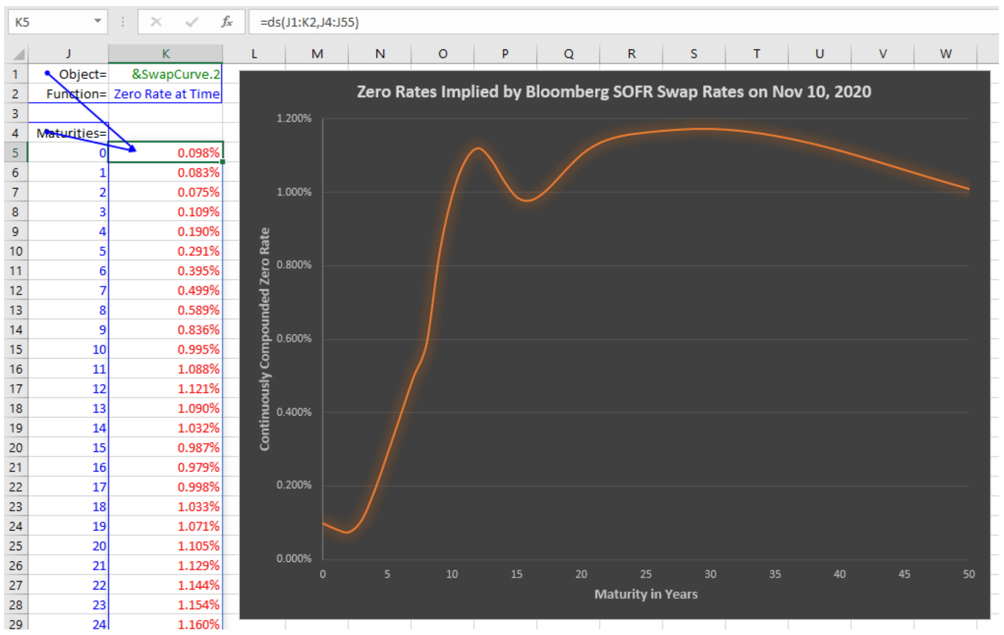
Deriscope™ [SofrDiscCurve.xlsx]

Name	Origin	C	Vol
Has Dynamic Trade Date	Market	L	NV
Implied Values	Yield Curve	L	NV
Increment	Market	L	NV
Lookup Markets	Market	L	NV
Set Trade Date	Market	L	NV
Shift Curve	Yield Curve	L	NV
Zero Rate	Yield Curve	L	NV
Zero Rate At Time	Yield Curve	L	NV

SwapCurve::Zero Rate At Time

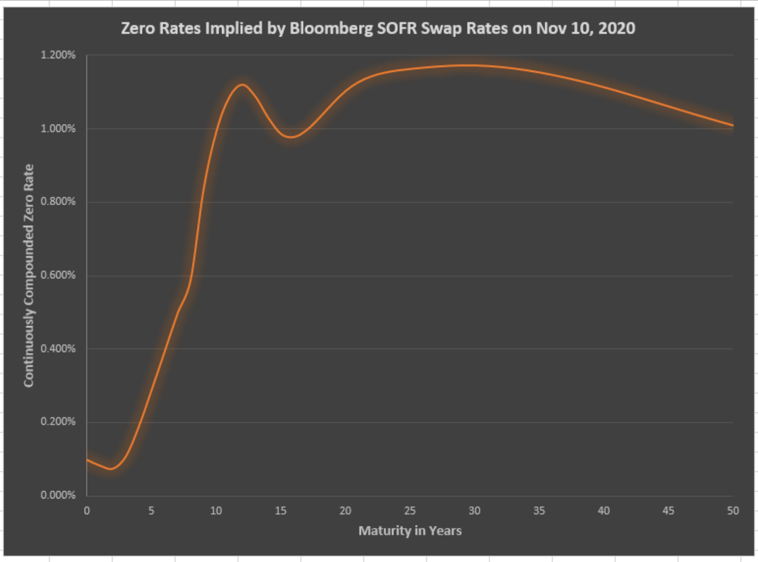
Maturities=

This *function* takes as input an array of maturities expressed as number of years and returns the corresponding *continuously compounded zero rates*. With the appropriate input, the following chart is produced:



K5 =ds(J1:K2,J4:J55)

	J	K
1	Object=	&SwapCurve.2
2	Function=	Zero Rate at Time
3		
4	Maturities=	
5	0	0.098%
6	1	0.083%
7	2	0.075%
8	3	0.109%
9	4	0.190%
10	5	0.291%
11	6	0.395%
12	7	0.499%
13	8	0.589%
14	9	0.836%
15	10	0.995%
16	11	1.088%
17	12	1.121%
18	13	1.090%
19	14	1.032%
20	15	0.987%
21	16	0.979%
22	17	0.998%
23	18	1.033%
24	19	1.071%
25	20	1.105%
26	21	1.129%
27	22	1.144%
28	23	1.154%
29	24	1.160%



Appendix

Proof that the Expectation of the Compounded Average of the Overnight Rates equals the Compounded Average of the Expectations of the Overnight Rates, where the Expectation is defined in Forward Measure.

It will be proven that:

$$E\left[\prod_{j=1}^{n}(1+I_j\tau_j)\right] \rightarrow \prod_{j=1}^{n}(1+E[I_j]\tau_j)$$

where the expectation operator E is defined with respect to the forward measure.

I_j represents the overnight rate (index) from t_{j-1} to t_j (one business day apart) as observed at t_{j-1}. Since t_{j-1} is in the future, I_j is a random variable from today›s perspective.

τ_j is the day count fraction of the interval from t_{j-1} to t_j according to a given basis convention.

I_j can be expressed in terms of the – also random variable - discount factor $P(t_{j-1},t_j)$, observed at t_{j-1} for maturity t_j:

$$I_j = \frac{\frac{1}{P(t_{j-1},t_j)} - 1}{\tau_j}$$

So, in terms of the t_{j-1}-observed discount factor $P(t_{j-1}, t_j)$ the above expectation takes the form:

$$E\left[\prod_{j=1}^{n}(1+I_j\tau_j)\right] = E\left[\prod_{j=1}^{n}\frac{1}{P(t_{j-1},t_j)}\right] = E\left[E\left[\prod_{j=1}^{n}\frac{1}{P(t_{j-1},t_j)} \mid F_{t_{n-2}}\right]\right]$$

where we used the tower law of conditional expectation, which states that the expectation of a random variable equals the expectation of its conditional expectation.

In the above case, the conditional expectation is taken with respect to the filtration $F_{t_{n-2}}$ as of time t_{n-2}, which the fancy name for the information available as of time t_{n-2}.

The first n-1 factors of the product are non-random with respect to $F_{t_{n-2}}$, so they can be taken out of the

$$E\left[\left(\prod_{j=1}^{n-1}\frac{1}{P(t_{j-1},t_j)}\right)E\left[\frac{1}{P(t_{n-1},t_n)} \mid F_{t_{n-2}}\right]\right]$$

We may now substitute:

$$\frac{1}{P(t_{n-1},t_n)} = \frac{P(t_{n-1},t_{n-1})}{P(t_{n-1},t_n)}$$

which shows that the conditionally integrated quantity equals the ratio of the price as of t_{n-1} of the zero bond maturing at t_{n-1} divided by the value of our numeraire as of t_{n-1}.

It follows, this ratio must be a martingale, which means that its conditional expectation with respect to $F_{t_{n-2}}$ must equal its value as of t_{n-2}.

Therefore:

$$E\left[\frac{1}{P(t_{n-1},t_n)} \mid F_{t_{n-2}}\right] = E\left[\frac{P(t_{n-1},t_{n-1})}{P(t_{n-1},t_n)} \mid F_{t_{n-2}}\right] = \frac{P(t_{n-2},t_{n-1})}{P(t_{n-2},t_n)}$$

and the unconditional expectation reduces to

$$E\left[\left(\prod_{j=1}^{n-1}\frac{1}{P(t_{j-1},t_j)}\right)E\left[\frac{1}{P(t_{n-1},t_n)} \mid F_{t_{n-2}}\right]\right] = E\left[\left(\prod_{j=1}^{n-1}\frac{1}{P(t_{j-1},t_j)}\right)\frac{P(t_{n-2},t_{n-1})}{P(t_{n-2},t_n)}\right] =$$
$$E\left[\left(\prod_{j=1}^{n-2}\frac{1}{P(t_{j-1},t_j)}\right)\frac{1}{P(t_{n-2},t_n)}\right]$$

Proceeding recursively, we reach the result:

$$E\left[\prod_{j=1}^{n}(1+I_j\tau_j)\right] = \frac{1}{P(t_0,t_n)} = 1 + R(t_0,t_n)\tau$$

where R is the term rate observed at t_0 for the interval from t_0 to t_n and τ is the daycount fraction of that interval.

$$E\left[\prod_{j=1}^{n}(1+I_j\tau_j)\right] = \frac{1}{P(t_0,t_n)} = \frac{1}{P(t_0,t_1)}\frac{P(t_0,t_1)}{P(t_0,t_2)}\cdots\frac{P(t_0,t_{n-1})}{P(t_0,t_n)} =$$
$$(1+E[I_1]\tau)(1+E[I_2]\tau)\cdots(1+E[I_n]\tau) = \prod_{j=1}^{n}(1+E[I_j]\tau)$$

Above, $E[I_j]$ represents the forward of the overnight rate I_j as observed today.