QF 3101: Computing Assignment

Rudhra Prakash Raveendran

I. Part 1: Eurodollar Futures and Deferred Interest Rate Swaps

In this part of the assignment, settlement prices and expiry dates of Eurodollar futures on Friday, 15 March 2019 were collected and used to construct various forward LIBOR term structures. These term structures were then used to determine fair swap rates for deferred K-year interested rate swaps on half-yearly exchanges.

A. Stage 1: Data Collection

10 years' worth of settlement prices and expiry dates of Eurodollar futures contracts with delivery months of March, June, September, and December were collected from the Chicago Mercantile Exchange's website. With this information, the corresponding 3-month LIBOR was calculated: $(100-P)\%$ (where P is a given futures price). Graphing these rates over time shows the general trend for 3-month LIBOR over the next 10 years, and provides a useful comparison to results obtained in later stages.

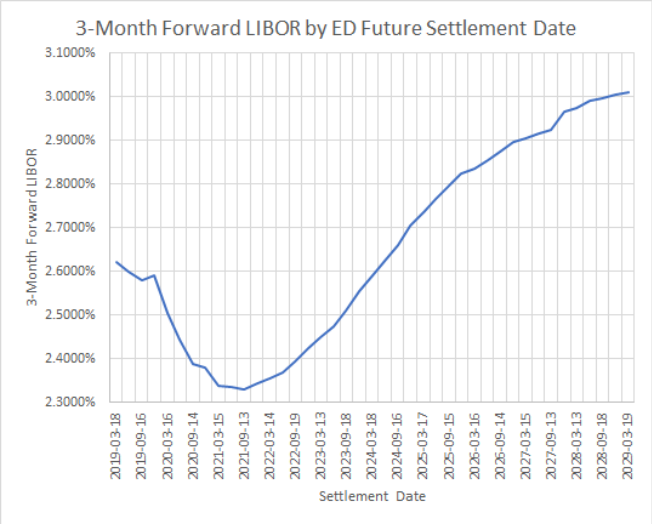

Fig. 1. 3-Month LIBOR over the next 10 years

B. Stage 2: Implied Forward LIBOR term structure

Using the 3-month LIBOR rates obtained during data collection, as well as the number of actual days in each 3-month period (calculated by subtracting settlement dates), one can calculate a forward LIBOR term structure (using the **actual/360** day count convention).

For example, to compute a 6-month forward LIBOR using two 3-month LIBOR rates, one can use the following formula:

$$(1+\frac{d_1+d_2}{360}r) = (1+\frac{d_1}{360}r_1)(1+\frac{d_2}{360}r_2)$$

$$r = [(1+\frac{d_1}{360}r_1)(1+\frac{d_2}{360}r_2) - 1] * \frac{360}{d_1+d_2}$$

where r is the 6-month LIBOR we want, r_1, r_2 are the two 3-month LIBOR rates, and d_1, d_2 are their respective days in term. This can be generalized for any N-month forward LIBOR (where N is a multiple of 3):

$$r = [\prod_{i=1}^{n}(1+\frac{d_i}{360}r_i) - 1] * \frac{360}{\sum_{i=1}^{n} d_i}$$

This can easily be translated to a VBA function where the product of the rates $\prod_{i=1}^{n}(1+\frac{d_i}{360}r_i)$ and the sum of the days $\sum_{i=1}^{n} d_i$ are continually built as a for-loop iterates over the vector of 3-month LIBOR rates (the exact code is found in the *GetTermStructure* function in the accompanying Excel file).

Using this method, I computed the implied forward LIBOR term structures for the reference dates of 18 March 2019, 17 June 2019, 16 September 2019, 16 December 2019 and 16 March 2020 (these dates simply determine which 3-month LIBOR rate to start the computation with).

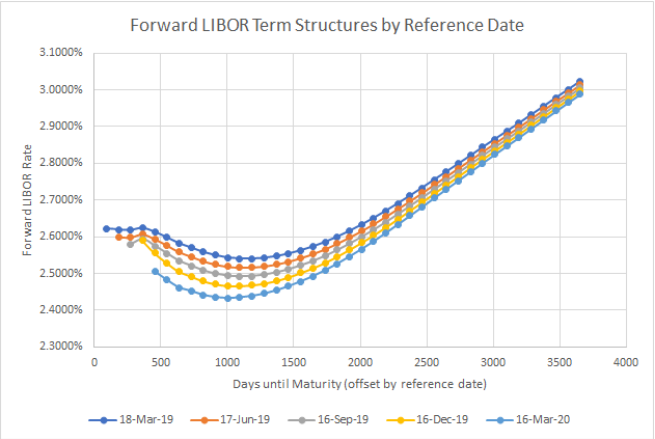

Fig. 2. Forward LIBOR Term Structures

Each dot in Figure 2's graph is a N-month LIBOR rate (the first in each series is a 3-month, then 6-month, etc.) Offsetting the term structures by their reference date lines up LIBOR rates by their end date, allowing a comparison to be drawn to the underlying 3-month LIBOR rates collected in Stage 1.

Both Figure 1 and 2 experience a small bump in rates around 16 December 2019 (364 days in), which is due

to a .01% increase in the 3-month LIBOR rate between September and December 2019. Then there is a fall in 3-month LIBOR rates until the lowest rate of 2.33% for 13 September 2021 (≈1000 days in), after which the rates continue to raise for the rest of the time period. This trend is reflected in the forward LIBOR term structures, as the term structures for all reference days decrease until September 2021, after which they recover and gain steadily.

However, the term structures in Figure 2 are noticeably more consistent, with none falling below 2.4% despite the lowest 3-month LIBOR being 2.33%. Furthermore one can see that the earlier the reference date, the higher the rates in the overall term structure. Despite this, the term structures for all reference dates seem to converge as time goes on. All of these characteristics can be explained by the product and sum terms: $\prod_{i=1}^{n}(1+\frac{d_i}{360}r_i)$ and $\sum_{i=1}^{n} d_i$.

The consistency of rates in the term structures is due to the fact that the rate for each N-month forward LIBOR is calculated via a product. Since each 3-month LIBOR rate changes by only a fraction of a percent, as we reach rates for longer terms the product will change less and less as each term will make up a smaller portion of the overall product.

The product term also explains why earlier reference dates have higher rates overall in their term structures. Going back to Figure 1, we know that 3-month LIBOR rates are falling for the first 3 years. Since the reference dates are all within a year of each other and within this 3 year falling rate period, each one will start with a lower 3-month LIBOR. This means that their respective term structures won't be taking into account the higher 3-month LIBOR rates of earlier dates, thus explaining the lower rates for later reference dates.

Despite the variation in early periods of the term structures, they converge as time goes on regardless of reference date, due to a combination of product of rates and sum of days in each term. The greater the N in a N-month LIBOR, the more terms are included in the product of rates and summation of days. Aside from the terms between the first and last reference date, these rates and days will all be the same. So as $n \to \infty$, $\prod_{i=1}^{n}(1+\frac{d_i}{360}r_i)$ and $\sum_{i=1}^{n} d_i$ will approach the same number across the different term structures.

C. Stage 3: Swap Rate for Deferred Interest Rate Swap

Using the forward LIBOR term structures obtained in Stage 2, we can model swap rates for a supposed interest rate swap between 2 parties with them swapping payments tied to a fixed rate (**actual/360**) for floating payments based on 6-month LIBOR at half-yearly intervals.

For example, the fair swap rate in such a swap (with a 1 year tenor, i.e. constructed with two 6-month LIBOR rates) can be determined via the following formula:

$$\frac{r*\frac{d_1-0}{360}}{1+r_1*\frac{d_1}{360}}+\frac{1+r*\frac{d_2-d_1}{360}}{1+r_2*\frac{d_2}{360}}=1$$

$$r\frac{\frac{d_1-0}{360}}{1+r_1*\frac{d_1}{360}}+r\frac{\frac{d_2-d_1}{360}}{1+r_2*\frac{d_2}{360}}+\frac{1}{1+r_2*\frac{d_2}{360}}=1$$

$$r=\frac{1-\frac{1}{1+r_2*\frac{d_2}{360}}}{\frac{\frac{d_1-0}{360}}{1+r_1*\frac{d_1}{360}}+\frac{\frac{d_2-d_1}{360}}{1+r_2*\frac{d_2}{360}}}$$

where r is the swap rate we want, r_1, r_2 are the first and second 6-month LIBOR rates, and d_1, d_2 are their respective days until maturity. This can be generalized to a K-year swap (constructed out of n 6-month LIBOR rates):

$$r=\frac{1-\frac{1}{1+r_n*\frac{d_n}{360}}}{\sum_{i=1}^{n}\frac{\frac{d_i-d_{i-1}}{360}}{1+r_i*\frac{d_i}{360}}}$$

where $d_0 = 0$. Similarly to Stage 2, this can easily be translated to a VBA function by using a for-loop for the summation (the exact code can be found within the *GetSwapRate* function).

Using this method, I calculated the swap rates for K-year swaps ($K = 1, 2...8$) for each of the deferred start-dates of 18 March 2019, 17 June 2019, 16 September 2019, 16 December 2019 and 16 March 2020, yielding the following results:

K	Mar 19	Jun 19	Sep 19	Dec 19	Mar 20
1	2.6067%	2.5772%	2.5384%	2.4907%	2.4377%
2	2.5233%	2.4878%	2.4545%	2.4228%	2.3916%
3	2.4651%	2.4423%	2.4225%	2.4065%	2.3918%
4	2.4475%	2.4361%	2.4277%	2.4225%	2.4193%
5	2.4585%	2.4559%	2.4562%	2.4591%	2.4637%
6	2.4889%	2.4924%	2.4981%	2.5058%	2.5144%
7	2.5285%	2.5348%	2.5427%	2.5520%	2.5617%
8	2.5679%	2.5754%	2.5841%	2.5938%	2.6043%

Fig. 3. Swap Rates of deferred K-year swaps

Graphing these values allows us to better visualize the trends in rates as K increases:

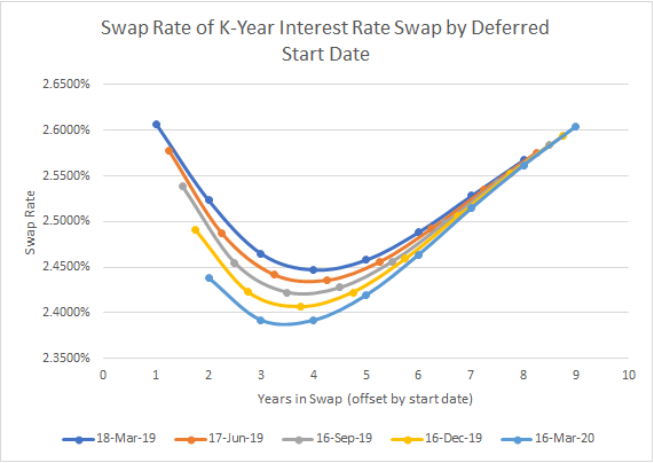

Fig. 4. Swap Rates of deferred K-year swaps (graph)

Offsetting the series by their start date shows trends that better represent the underlying LIBOR term structures that the swap rates are obtained from. Just like those LIBOR rates

from Stage 2, swap rates from later start dates have initially lower values, but as K increases the rates start to converge. Also like in Stage 2, these characteristics in the swap rate trends can be explained by the data used to construct them.

The swap rate trends follow the general shape of the LIBOR term structures from Stage 2, which themselves follow the underlying 3-month LIBOR rates originally collected. This is why all the swaps experience their lowest rates around $K = 3$, which is around September 2021 (the lowest 3-month LIBOR rate).

Just like the term structures in Stage 2, the reason why later start dates lead to lower rates (at least for small values of K) is because of the lower underlying rates used in the calculation. When K is small, the values used in calculating the swap rate are within the period of declining 3-month LIBOR rates between March 2019 and December 2021. The forward LIBOR rates for each successive start date will start lower and lower, resulting in lower swap rates.

Also like in Stage 2, the swap rates converge regardless of start date as K grows larger. As K grows larger, the number of terms in the summation part of the formula grows larger as well, and the more terms in total, the more terms are shared between the start dates. With less different terms, the summation will start to approach the same value for each of the start dates. Furthermore, the numerator will also start to converge as it is dependent on the forward LIBOR rates from Stage 2, which we already observed to converge as time goes on.

II. Part 2: Efficient Equity Portfolios

In this part of the assignment, seven US stocks belonging to different industrial/commercial sectors were selected for the construction of efficient portfolios. The chosen stocks were: LULU, DIS, TGT, FIZZ, MSFT, XOM, and V.

A. Stage 1: Data collection

Weekly closing prices (in the five year period between 1 January 2014 and 1 January 2019) of the seven stocks were collected from Yahoo Finance, from which weekly rates of return were obtained for each stock. Using those rates, a covariance matrix was constructed for the seven stocks. A graph of the prices of each stock over the five year period gives a sense of price trends for the individual stocks.

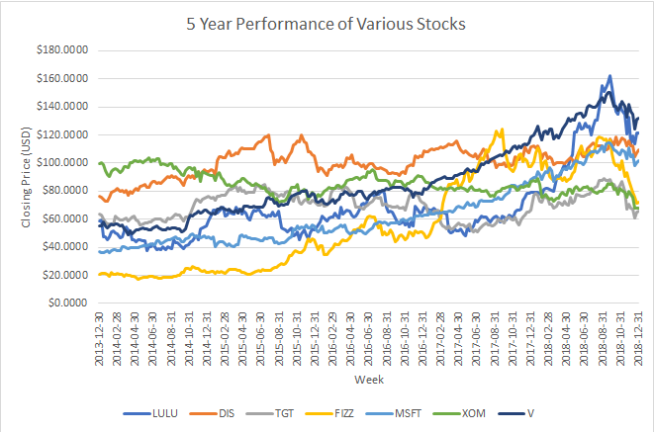

Fig. 5. 5 Year Stock Performance

The average weekly rate of return for each stock are as follows:

Stock	Average Rate of Return
LULU	0.4249%
DIS	0.1737%
TGT	0.0763%
FIZZ	0.5915%
MSFT	0.4315%
XOM	-0.1130%
V	0.3644%

Fig. 6. Average Weekly Rate of Return

With Figures 5 and 6, it's clear that for all the stocks save for XOM, value generally increased (even if only by a small amount). The covariances (scaled by 10^3) of the stocks can be seen in the following figure (exact values can be found in the accompanying Excel file):

	LULU	**DIS**	**TGT**	**FIZZ**	**MSFT**	**XOM**	**V**
LULU	2.904	0.297	0.434	0.213	0.298	0.232	0.282
DIS	0.297	0.676	0.266	0.268	0.334	0.261	0.336
TGT	0.434	0.266	1.207	0.233	0.282	0.233	0.249
FIZZ	0.213	0.268	0.233	2.178	0.323	0.065	0.219
MSFT	0.298	0.334	0.282	0.323	0.856	0.290	0.414
XOM	0.232	0.261	0.233	0.065	0.290	0.627	0.249
V	0.282	0.336	0.249	0.219	0.414	0.249	0.615

Fig. 7. Stock Covariance (scaled by 10^{-3})

The above covariance matrix makes it clear that, although the values are very low, there is positive covariance between all the stocks, implying that there is at least some weak relationship in their price movements.

B. Stage 2: Minimum Variance Point Portfolio And An Efficient Portfolio

This stage involved the creation of VBA functions to generate portfolio weights for a Minimum Variance Portfolio and an Efficient Portfolio (weights are simply the proportion of a portfolio that should be allocated to a stock, where negative weights indicate short positions). Both functions are based off simple formulas.

To obtain the weights for a Minimum Variance Portfolio, one simply needs to solve the following equation:

$$[cov] * [w] = [1]$$

where $[cov]$ is the covariance matrix generated in Stage 1, $[1]$ is a Nx1 vector of 1's, and $[w]$ is the vector of weights. The terms can be rearranged to get

$$[w] = [cov]^{-1} * [1]$$

Each weight in the weight vector is simply the sum of the corresponding row in $[cov]^{-1}$, and this can be easily be done in VBA through a for-loop (exact code is found in the *GetMinVarPortfolio* function).

Obtaining the weights for an Efficient Portfolio is a very similar process, the only difference is that the right-hand side of the equation is replaced with a vector of the average weekly rate of return for each stock:

$$[w] = [cov]^{-1} * [r]$$

This can be accomplished with built-in matrix multiplication functions in VBA (used in *GetAnEfficientPortfolio*).

C. Stage 3: Efficient Portfolios

Using the functions from Stage 2, I obtained the following portfolio weights for a Minimum Variance Portfolio and an Efficient Portfolio (portfolio rates of return are calculated by summing the Hadamard product of a portfolio weight vector and the vector of average rates of return per stock):

	Minimum Variance	Efficient
LULU	0.029036481	0.221487569
DIS	0.172936568	-0.119629245
TGT	0.108070251	-0.176512976
FIZZ	0.087622728	0.382270756
MSFT	0.047908328	0.816556168
XOM	0.320675924	-1.176546897
V	0.23374972	1.052374626
Rate of Return	0.1721%	1.1547%

Fig. 8. Portfolio Weights and Rates of Return

As expected, the Minimum Variance Portfolio has low variance in its weights, while the Efficient Portfolio varies greatly. This low variance leads to a less risky portfolio, but this comes with the downside of weaker performance. The Efficient Portfolio on the other hand has a much higher rate of return, but recommends shorting 3 of the stocks (a riskier move than a simple long). For a better visualization of each portfolio's breakdown, refer to the following stacked bar graph.

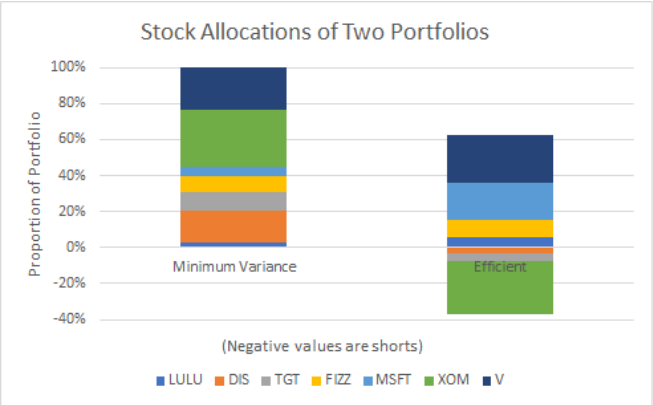

Fig. 9. Portfolio Configurations

Using the previous portfolio weights and the Two-Fund Theorem, we can create an infinite amount of portfolios by using various proportions of each portfolio. This can be represented via a formula:

$$[w'] = \alpha * [w_1] + (1 - \alpha) * [w_2]$$

where $[w']$ is the new portfolio weights we want, $[w_1], [w_2]$ are the original two portfolios, and α determines the proportion of each portfolio, $0 \leq \alpha \leq 1$.

Using the Minimum Variance Portfolio as $[w_1]$ and the Efficient Portfolio as $[w_2]$, and values of α from the set $\frac{1}{31}, \frac{2}{31} \cdots \frac{30}{31}$, we can obtain a series of 30 portfolios whose rates of return are greater than that of the Minimum variance Portfolio. Graphing the expected rates of return of each of these portfolios in respect to their standard deviation gives the Efficient Frontier for a portfolio comprised of the seven stocks. (Standard deviation of a portfolio with n stocks is given by $\sqrt{\sum_{i,j=1}^{n} w_i * w_j * cov_{i,j}}$, where w_i is the weight of a stock and $cov_{i,j}$ is the covariance of stocks i and j)

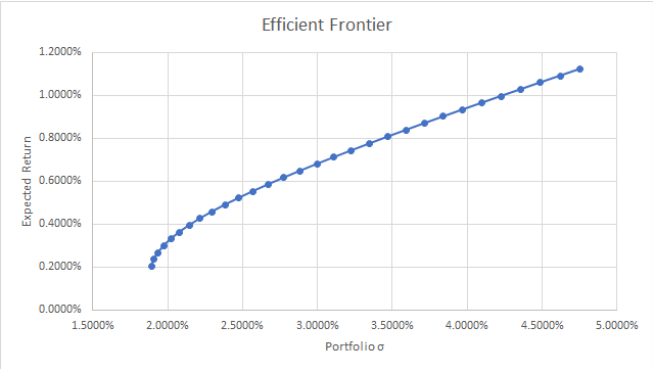

Fig. 10. Efficient Frontier of Our Portfolio

As we can see, the rates of return for the various portfolios follow the shape of the Efficient Frontier, which is the highest expected rate of return for a given level of risk (standard deviation). We can compare and confirm the accuracy of the graph in Figure 10 to a canonical example of an efficient frontier from the Luenberger text:

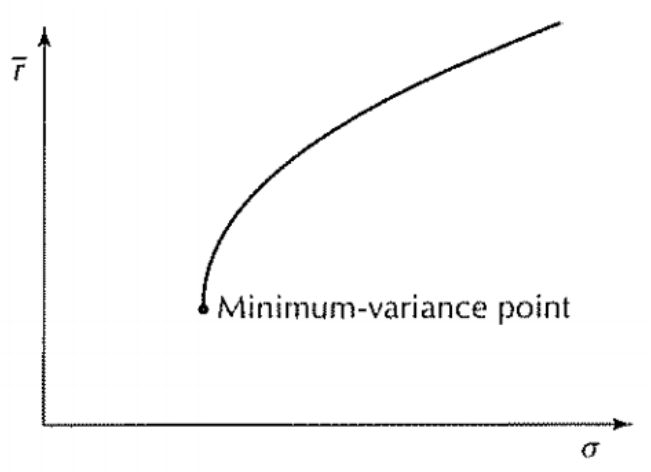

Fig. 11. Example Efficient Frontier

From Figure 10 we can see that as the standard deviation σ (risk) increases, so too does the expected rate of return. The lowest level of risk corresponds with $\alpha = 1$ i.e. a Minimum Variance Portfolio, while the highest level risk ($\alpha = 0$) is the Efficient Portfolio.

Knowing this, we can provide the best portfolio consisting of LULU, DIS, TGT, FIZZ, MSFT, XOM, and V for investors with different risk tolerances. Young investors with long time-horizons and high risk tolerance could use portfolios with low α, while investors closer to retirement may want to use a high α value.