POLITECNICO
MILANO 1863

Financial Engineering

Assignment 3, Group 16

2023-2024

Alice Vailati - CP: 10683600 - MAT: 222944
Andrea Tarditi - CP: 10728388 - MAT: 251722
Jacopo Stringara - CP: 10687726 - MAT: 222456
Nicolò Toia - CP: 10628899 - MAT: 247208

Contents

1. Introduction and Data

The aim of this assignment is to compute on MATLAB, given the discounting curve vs Euribor 3m on the 15th of February 2008 at 10:45 C.E.T., multiple spreads, intensities (with different types of approximation) and to price a First To Default. In the end, we have to perform some basic exercises on Python.
In order to evaluate these exercises we have to use the following data:

Data for Asset Swap Spread exercise

Settlement Date: 19th February 2008 Maturity: 3 years
Price of the Bond: $\bar{C}$ (0)= 101 (101% of the face value)
Annual Coupon: $\bar{C} = 3.9\%$
Coupon paid on the swap dates

Data for Obligor ISP

Settlement date: 19th February 2008
Recovery π : 40%
CDS spreads (with spreads in Basis Point):

Years	1	2	3	4	5	7
Spreads	29	32	35	39	40	41

Data for Obligor UCG

Settlement date: 19th February 2008
Recovery π : 45%
CDS spreads (with spreads in Basis Point):

Years	1	2	3	4	5	7
Spreads	34	39	45	46	47	47

Data for exercise 3

Maturity: 20th February 2012
Correlation ρ : 20%

2. Asset Swap Spread Over Euribor 3m

The aim of this point was to evaluate the Asset Swap Spread Over Euribor3m. We consider as settlement date the 19th of February 2008 and as maturity the 21st of February 2011.
The idea in order to find the Asset Swap Spread is to represent the Asset Swap Package as an IB Coupon Bond and a Floater. First of all we have to find both the fixed dates and the floating leg dates: in the first case we have one payment per year while the others happen every three months and we have to check that all of these are working days.

When we move the dates to business days, we have to pay close attention to the calendar used. In Matlab, the function *busdate*, by default, uses the US calendar: unfortunately the 21st of February 2011 was "President's day", a closing date for the NYSE. Thus, we must not consider American holidays.

After this, we compute as follows the price of the IB coupon bond that replicates exactly the coupons payed by the Asset swap each year:

$$C(0) = \sum_{i=1}^{N} \delta(t_{i-1}, t_i)\bar{C}B(t_0, t_i) + B(t_0, t_N)$$

where we have $\overline{C} = 3.9\%$, the year fraction is evaluated with a European 30/360 day convention, the discounts are computed from the bootstrap and for ease of notation we assumed the facevalue to be 1.

At this point, we need to compute the BPV of the floating leg, in order to be able to compute the Asset Swap Spread. We evaluate it as:

$$BPV^f(0, N^f) = \sum_{i=1}^{N^f} \delta(t^f_{i-1}, t^f_i) B(t_0, t^f_i)$$

where, again, the discounts are computed from the bootstrap but the day count convention is ACT/360. Finally, we are able to compute the the Asset Swap Spread Over Euribor 3m as:

$$s^{ASW} = \frac{C(0) - \bar{C}(0)}{BPV_{float}}$$

This yields a spread of -34.1027 basis points.
A negative Asset Swap Spread is a sign that the underlying firm on which the contract is written has a rating higher than AA. Indeed the AA rating class is taken as a point of reference, with a spread over LIBOR of zero bps.
In other words the underlying company has a rating of either AA+ or AAA.

3. CDS bootstrap

In this case study, we undertook an analysis and constructed the CDS curve for Intesa San Paolo (ISP) as of February 15th, 2008, utilizing a bootstrapping approach. Notably, the settlement date, marking the starting of curve computation, was set on February 19, 2008.

3.1. CDS via spline

In addressing the point **a** , we were tasked with constructing a complete set of CDS via spline interpolation of the provided spreads. Given the available data, we derived the spread for the 6-year maturity.
It is noteworthy that upon plotting and observing the results, there is an unexpected behaviour in the curve's shape. Specifically, the spread for the 6-year maturity was found to be lower than that of the 5-year maturity. This unusual behavior arises from the cubic approximation inherent in spline interpolation, which considers the general concavity of preceding points. Indeed the spreads for CDS are usually monotonic. As a remedy, we advocate for employing linear interpolation, to rectify the unexpected behaviour observed in the curve's shape, ensuring a more intuitive and consistent representation of credit spreads over time.

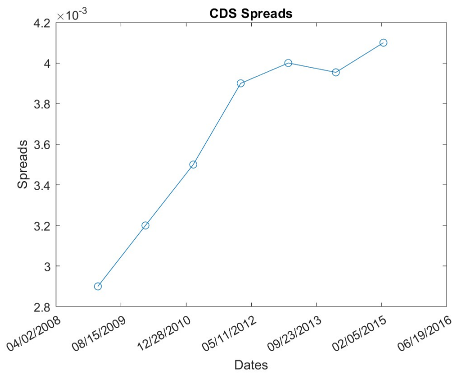


Figure 1: Cubic interpolation

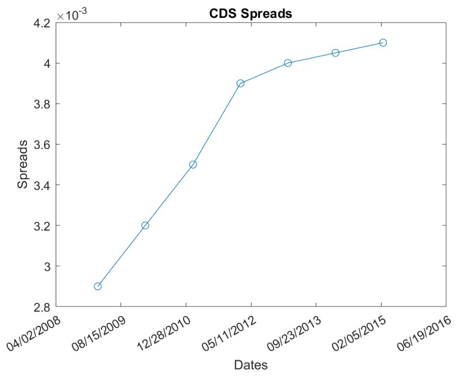
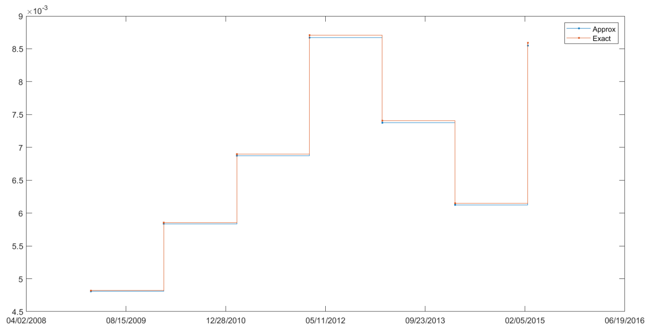


Figure 2: Linear interpolation

In the following we will nonetheless use the spline interpolation.

3.2. Intensities $\lambda(t)$

Moving on to point **b**, we proceeded to construct a piece-wise constant function for the intensity λ(t) by leveraging the quoted CDS spreads. This method ensures to capture variations in default risk over discrete time intervals. Throughout the bootstrapping process, we derived the intensity curve both with and without considering the accrual term. As anticipated by theory, the outcomes are closely aligned, affirming that the terms related to accrual can be effectively neglected. This assertion is substantiated by the minute discrepancy, less than 1 basis point, observed upon comparing the values obtained through the two methodologies. Furthermore, we computed survival probabilities, unveiling a declining trend over time, with the survival probability dwindling to 95.27% at the 7-year maturity mark.



Additionally, we explored the Jarrow-Turnbull (1995) simplified model, wherein payments are made continuously, and λ remains constant (flat). This model leads to the well-known rule of thumb provided in the following formula:

$$\lambda = \frac{\bar{s}}{(1-\pi)}$$

Qualitative plotting of the results, inclusive of the cumulative mean of the intensity, demonstrated alignment with the flat lambdas derived through J-T rules at each maturity point. Indeed:

$$\lambda_N^{JT} \approx \frac{1}{N}\sum_{i=1}^{N}\lambda_i$$

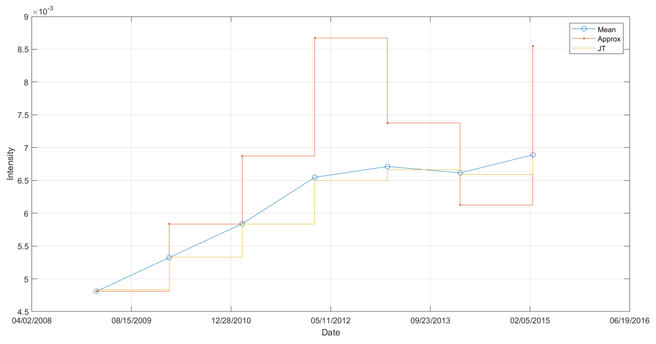


Figure 3: JT intensities were plotted as a step function, while they are simply flat estimates

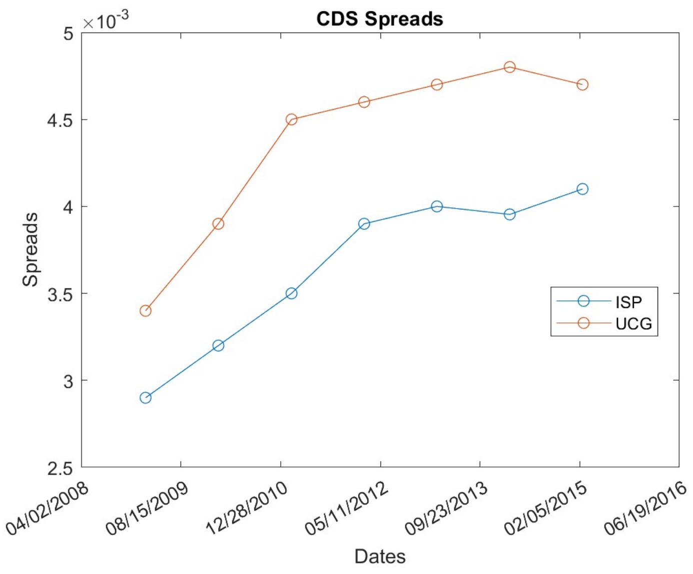
Even though the Jarrow-Turnbull is largely used by traders to efficiently compute the intensities, it presents some limitations. For instance, its assumption of constant default intensity may oversimplify the dynamics of credit risk, particularly during periods of economic stress.

4. Price first to default

In this section, we take into the analysis of the First to Default (FtD) contract involving two distinct companies, namely IntesaSanPaolo (ISP) and Unicredit (UCG).

4.1. CDS via spline

Initially, we embark on computing the complete set of Credit Default Swap (CDS) spreads for UCG, employing spline interpolation, as previously demonstrated. Notably, the spreads curve exhibits a non-traditional pattern, with the 6-year spread surpassing expectations, peaking at a higher value than its subsequent 7-year spread. To facilitate a comparative understanding between the two firms, a visualization of the CDS curves is presented.



4.2. Pricing

Moving on, we proceed to price the FtD contract, set to mature on the 20th of February 2012, involving both obligors. This pricing is executed utilizing the Li model, under the assumption of a Gaussian copula and a correlation coefficient (ρ) set at 20%.
To compute the FtD spread we employed a Monte Carlo simulation approach. For each simulation we compute the time of the first default τ (if any), then we computed the fee leg plus the accrual up to τ of the last fee and the contingent leg as the loss given default (1-π) paid at default time (τ), if applicable. Then we take the average of both legs to compute their NPV and take the contingent leg by the fee leg to obtain the spread.
This yields an FtD spread value amounting to 81.8942 bp. This computed value closely aligns with theoretical expectations.
Considering the computations above, it becomes evident that when the correlation coefficient remains relatively modest, the FtD price tends to approximate the sum of the individual CDS spreads of the obligors, resulting in an 85-basis point spread over 4 years.

Furthermore, to enhance the reliability of our findings, simulations are conducted within a 95% confidence interval framework, yielding a definitive range: [0.0075 - 0.0088].

4.3. First to default with different correlations

To capture the full spectrum of potential outcomes, simulations are iterated across the entire range of possible correlation coefficients (-1 to 1). This iterative process yields a revealing plot, elucidating the FtD spread's behaviour in response to varying correlation coefficients. Theoretical underpinnings suggest that the FtD price fluctuates in a continuous fashion between the maximum spreads of the obligors and the sum of their respective spreads. Thus, distinct scenarios emerge:

1. In instances where obligors exhibit negligible or negative correlation, the price impact remains consistent, as evidenced by the flat curve spanning from -1 to 0, stabilizing around 85 basis points.
2. Conversely, in scenarios characterized by positive correlation, the FtD price experiences a progressive decrease, eventually converging to the maximum spread between obligors for ρ equal to 1, settling at 45 basis points (UCG).

To complement our analysis, a visual representation in the form of a figure is included, to clearly show the relationship between FtD price and correlation coefficient.

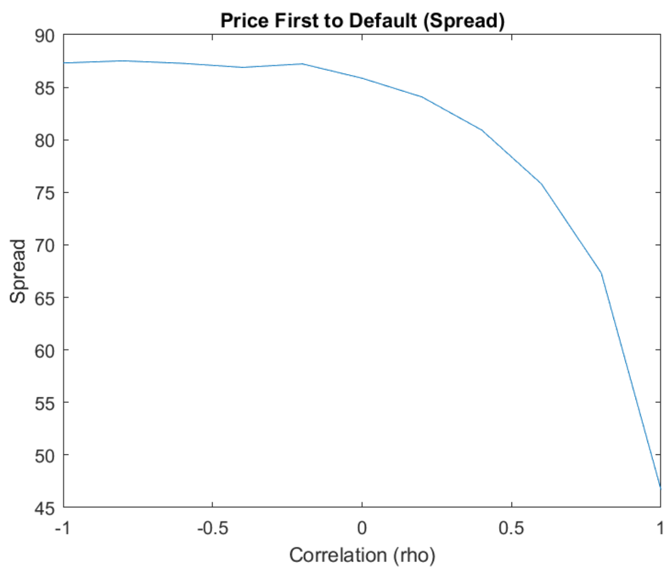


Figure 4: FtD spreads computed with $2 \cdot 10^6$ number of simulation

4.4. Drawbacks

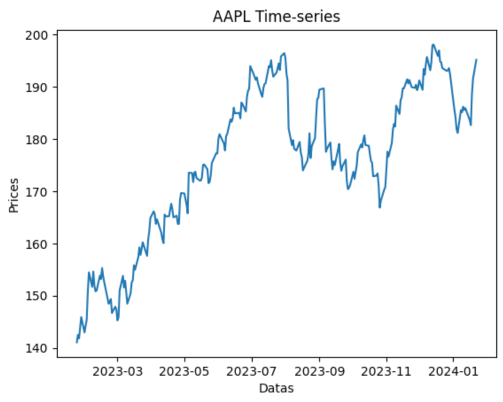
It is imperative to note that achieving robust results necessitates a significant number of simulations, thus incurring substantial computational costs.
Indeed, to use the Gaussian copula and compute the times of default τ, an inversion of the survival probability curve is required.
Parallelisation of this operation has proven to be quite challenging. The exponential step behaviour of the survival probability function and the usage of the year fraction do not allow to an easy inversion of a function. Furthermore, finding the payment dates before each τ is also a challenging operation in matlab. The problem can probably be vectorized if tensors are applied correctly but this would negatively affect the interpretability of our code.
In a practical context, whether to invest time into vectorizing the code must be weighed by how often the computation is run. Perhaps a variance reduction technique, such as Antithetic Variables, can be applied fruitfully to reduce the number of needed simulations.
Nonetheless we observed that the increase in simulations gives diminishing returns in accuracy. Indeed, after $3 \cdot 10^5$ we reached an IC of 2 bp while for $2 \cdot 10^6$ we have an IC of ≈ 1 bp.

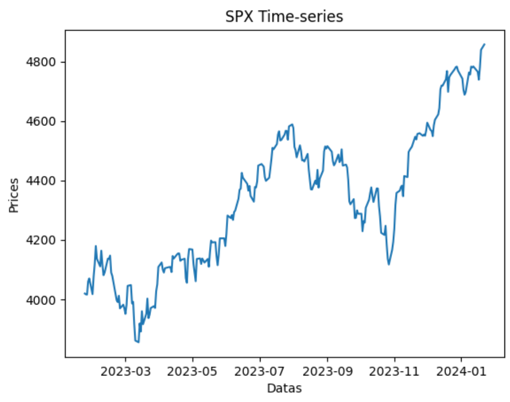
5. Python

In this section, we have to perform some basic Python exercise.

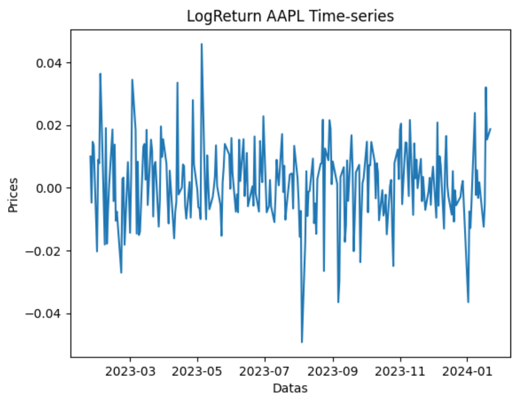
Time-series

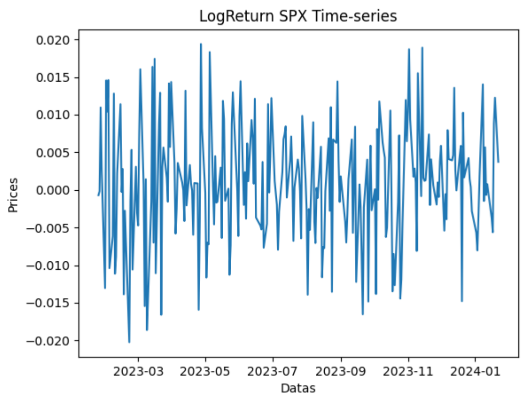
In the first part, we have to plot two different time-series: AAPL and SPX. After loading the data on Python, we plot them and the results are the following:





The next point is about log-returns: we evaluate them applying the natural logarithm to the quotient obtained dividing a return and the previous one. After this computation, we obtain the following plots:



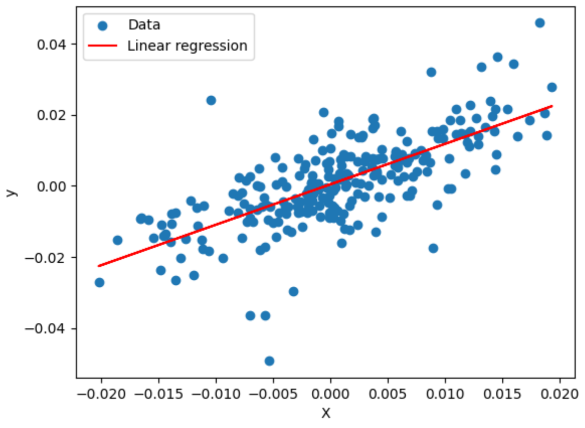


We can notice through the observation of the plots how, using the log-returns instead of the returns, we obtain a more stable variance and average price; due to the fact that there are no big differences between one value and the next one, the value is always around the value of zero (indeed it is the natural logarithm of a quotient near to one).

After this, we have to estimate the slope of the regression AAPL on SPX: we obtain, as result:

$$\beta_1 = 1.1352$$

The plot of this regression is the following:



The last request about this part is to evaluate the year fraction using ACT/365 between the first and the last date of the AAPL time-series:

$FirstDate$	$LastDate$	$YearFrac$
23/01/2023	22/01/2024	0.9973

Point 5: linear interpolation

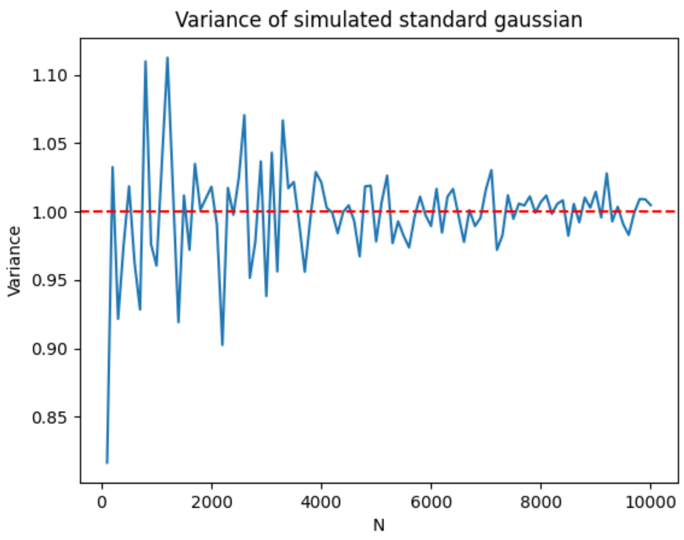
In this point we have to linearly interpolate in x=2.7 knowing the following data:

x	0	1	2	3	4
f(x)	1	2	3.5	4	2

The result find is f(2.7) = 3.85

Point 6: simulation of Normal Random Variables

First of all, we have to simulate a standard Normal Random Variable and to check that the variance, as the number of the simulation increases, converges to 1. In order to show this, we decide to draw the plot of the variance as N increases.



Observing the graph is easy to see that after around 5000 iterations the variance begin to be quite stable around a value of 1.
Finally we have to check that the Gaussian CDF evaluated in the quantile 0.9 of the simulated data gives the correct result. We compute the 0.9 quantile of the simulated data:

$$z^{s}_{0.9}im = 0.9028$$

We can see that this is quite close to the value 0.9.

Point 7: minimum of a multivariate function

The function that we have to minimize is $\mathbf{f(x, y) = (x - 3)^2 + (y - 7)^2}$.

First of all we have to find the minimum analytically: in order to do this we evaluate the first derivative in both of the variables and equal them to zero:

$$\Delta f(x) = \begin{bmatrix} 2x - 6 \\ 2y - 14 \end{bmatrix} = \underline{0} \quad H_f(x) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

The only possible solution is the point P(3, 7) which is a minimum. Evaluating this numerically using the function optimize.minimize, let us obtain exactly the same result.