GROUP 13
Silvia Alessandra Michela Fiecchi, 10702310
Paolo Pellegrinotti, 10628812
Luca Pizzo, 10614734
Alberto Polidori, 10574439

POLITECNICO
MILANO 1863

Financial Engineering

Academic Year 2023-2024

Assignment 4

Class Usage Overview

To enhance the capabilities of Python, we decided to adopt a more object-oriented programming approach. Nonetheless, we have implemented all functions with the required signatures and integrated them into the methods of classes in a more straightforward manner. The combination of these two approaches may not be particularly elegant, but the benefits of utilizing classes are significant. They improve the readability and efficiency of the main program and make the concepts involved more concrete.

Specifically, the key class we implemented is called "Portfolio". Its purpose is to model a simple portfolio based on the provided dataset and the parameters we have specified. This class simplifies the process of selecting assets of interest and determining the dates for our analysis.

Alongside the Portfolio class, we've developed the OptionDerivative class, which serves as a superclass for specific types of options, in particular is implemented EuropeanOption. The simple implementation chosen allows us to model option derivatives with minimal complexity, focusing on key attributes such as the time to maturity rather than the exact maturity date. This design choice avoids the need for the OptionDerivative class to access other attributes of the Portfolio class, thereby maintaining a clean separation of concerns. The Portfolio class manages these option derivatives effectively, incorporating the maturity date into its calculations to derive the time to maturity as needed.

Variance-Covariance Method for VaR & ES

In order to compute the daily VaR and ES via t-student parametric approach, we used the Portfolio methods `VaR` and `ES` respectively, specifying the t-student approach, since these methods can be used also for the Gaussian parametric approach. We obtained the following results:

VaR	561412.0284318669 €
ES	785964.6205037709 €

These results are coherent with what we expected since the value of the portfolio is €15M

Historical (HS & WHS) Simulation, Bootstrap and PCA for VaR & ES

For PortfolioA we used the constructor that takes as input the number of shares and computed the weights of the portfolio from there. We, then, computed the daily VaR and ES with a 5y estimation via Historical Simulation applying the Portfolio method `HSM`, which calls the function `HSMeasurements`. This function simply computes the losses, orders them and finds the VaR and ES coherently with the theory behind. We obtained the following results:

HS VaR	320174.50591103407 €
HS ES	441490.14562011603 €

We, then, repeated this computation via Statistical Bootstrap approach. To do so, we used the Portfolio method `BootstrapStat` that uses the functions `bootstrapStatistical` and `HSMeasurements`. The first function extract 200 random numbers, i.e. the indices of the losses to be computed, while the second one computes the VaR and ES via Historical Simulation approach. The results obtained are:

BS VaR	329796.2003200735 €
BS ES	447353.9503545874 €

Finally, we used the Portfolio method `pCheck`, that calls the funciton `plausibilityCheck`, to perform a Plausibility check of the obtained results. We obtained:

PC VaR: 302429.28388494335 €

Since the order of magnitude it's consistent with the results obtained, we consider it to be right.

For PortfolioB we computed the daily VaR and ES with a 5y estimation via Weighted Historical Simulation, applying the Portfolio method `WHSM` that calls the function `WHSMeasurements`. The results obtained are:

WHS VaR	0.015657144550171156 €
WHS ES	0.021257334277160485 €

Again, we checked the order of magnitude via Plausibility check:

PC VaR: 0.018735272285670932 €

As before, the result obtain confirms the order of magnitude of our results.

For PortfolioC we computed the 10 days VaR and ES with a 5y estimation via a Gaussian parametric PCA approach with $n = 1, ..., 5$. In order to do so, we used the Portfolio method `PCA`, that initializes the parameters and calls the function `PrincCompAnalysis`. The results obtained for each n are:

	VaR	ES
n = 1	0.05287715213131647 €	0.06712222032657257 €
n = 2	0.05284026806354508 €	0.06710018333769202 €
n = 3	0.05283730048518992 €	0.06709722452345276 €
n = 4	0.05298220940244455 €	0.06724599132009983 €
n = 5	0.05298169433044860 €	0.06724557972257911 €

Observing the results, we consider that $n = 4$ principal components are enough. Indeed, the VaR and ES slightly decrease for $n = 3$, increasing again for $n = 4$. Additionally, the following increase for $n = 5$ seems negligible.

We checked the correctness of the order of magnitude via Plausibility check and obtained:

PC VaR: 0.0546465998858422 €

Again, the plausibility check confirms that the order of magnitude of our results is correct.

Full Monte-Carlo and Delta normal VaR

To calculate the Value at Risk (VaR) of a nonlinear portfolio, it is necessary to employ a specific evaluation technique. In particular, we utilized two methods: the Full Monte Carlo and the Delta-Normal method.

The Full Monte Carlo method is highly generic and was applied for a comprehensive evaluation of the portfolio's VaR. For the underlying assets, we adopted a 2-year Weighted Historical Simulation (WHS) approach, which allows for a nuanced assessment that takes historical performance into account, weighted by relevance to the current assessment.

In the application of the Full Monte Carlo method, we randomly selected log returns for 10-day time windows. Each window had its weight, calculated with the WHS weighting formula, as probability of being selected.
Those randomly selected log returns were then used to get the price of the share after 10 days for each simulation. For each simulation, the price of the call option after 10 days was calculated by applying the Black and Scholes formula with the new price of the share as S_{t_0} and $(TimeToMaturity - 10/256)$ as time to maturity in years.
We then calculated the loss of the total portfolio for each simulation and identified the 95th percentile of these losses as the VaR. This percentile provides a measure of the maximum unexpected loss with a confidence level of 95

In the Delta-Normal method, the changes in the share price are calculated in the same way as in the Full Monte Carlo method, while the difference lies in the way the losses are calculated. The portfolio is linearized so that we don't have to calculate the price of the derivative for each simulation, and this will give us a huge advantage in terms of total number of calculations. The loss for each option in each simulation is then calculated as the sensitivity with respect to the change in the underlying price (the share).
The total loss in the single simulation will be calculated as:

$$L_{ptf} = L_{1share} \cdot (1 \cdot N_{shares} + \Delta \cdot N_{calls})$$

$$L_{ptf} = L_{sharesptf} \dot{(} 1 + \Delta \cdot \frac{N_{calls}}{N_{shares}})$$

Since the VaR is homogeneous, we can calculate the VaR of the shares portfolio first and then multiply it by $\dot{(}1 + \Delta \cdot \frac{N_{calls}}{N_{shares}})$. Now the result of the Delta Normal method should be very close to the one calculated with the Full Monte Carlo method. Since the moneyness of the call is greater than 3, we are dealing with some calls far deep in the money, so with a $\Delta_{call,t_0} \approx \Delta_{call,t_0+10d}1$. This means that calculating the loss for a call as the difference in prices between t_0 and $t_0 + 10d$, or as $\Delta_{call,t_0} \cdot L_{share}$ should be almost the same. In practice we found that there is a big difference between the two values of VaR calculated, and this is a criticality of our model.
The results that we got are the following:

FMC VaR	1804.6465 €
DN VaR	504.0799 €

Taking into consideration the gamma should give us some better results, since it gives some more information about how the delta is changing. This is extremely useful because we can then approximate the change in

the option price as a quadratic function of the change in the underlying price, which will be more robust to price changes.

The Full Monte Carlo method is particularly computation-intensive when used for pricing exotic derivatives that cannot be solved through closed-form formulas. This is primarily due to the necessity of simulating potential M_{sim} future paths all the way to the expiry of the derivative for each one of the N_{sim} simulations of the Full Monte Carlo, to get the price in $t_0 + 10d$ for each one of those N_{sim} simulations.

Pricing in presence of counterparty risk

Observing the payoff of the Cliquet Option and considering the definition of the option we are able to find a closed formula, via change of numeraire. In particular,

$$\mathbb{E}^{T_{i-1}}[(LS_{t_i} - S_{t_{i-1}})^+] = \mathbb{E}^{T_{i-1}}\left[S_{t_{i-1}}\left(L\frac{S_{t_i}}{S_{t_{i-1}}} - 1\right)^+\right] = \frac{1}{B(0,t_i)}\mathbb{E}^{T_{i-1}}\left[\left(L\frac{S_{t_i}}{S_{t_{i-1}}} - 1\right)^+\right]$$

We recognize the payoff of a call option, therefore

$$\mathbb{E}^{T_{i-1}}[(LS_{t_i} - S_{t_{i-1}})^+] = \frac{1}{B(0,t_i)}C^{blk}\left(L\frac{B(0,T_{i-1})}{B(0,T_i)}, 1, \sigma\right) =: C_i$$

The price of the Cliquet option, considering the counterparty risk, is:

$$P = notional \cdot \left\{ \sum_{i=1}^{7} \mathbb{P}(0,T_i) \cdot B(0,T_i) \cdot C_i + \pi \sum_{i=1}^{7} [1 - \mathbb{P}(0,T_i)] \cdot B(0,T_i) \cdot C_i \right\}$$

We implemented this formula in the function `CliquetOption`, and obtained the following result:

16602148.8038 €

Assuming that ISP issues the Cliquet option, it will not face any counterparty risk, since it receives all cash flows at the start of the contract. In order to profit, ISP must sell it at price higher or equal to the price of the option in case of no default. This is due to the fact that it has to discount all future cash flows that it must pay to the option buyer.