A Work Project, presented as part of the requirements for the Award of a Master's degree in Finance from the Nova School of Business and Economics.

Banco Invest Consulting Project
«Delta-Gamma Value-at-Risk model for - »

Tim Gajewski (49154) - Portfolio of Call Digital options

Work project carried out under the supervision of:

Daniele D'arienzo

11/01/2022

Abstract (100 words maximum)

Banco Invest offers various over-the-counter (OTC) derivatives to institutional clients as part of its structured investment solutions. These derivatives are managed within the bank's Proprietary Trading Book. The focus of this consulting project is developing a Delta-Gamma Value-at-Risk (VaR) model that Banco Invest can implement to actively manage its equity derivative portfolio`s underlying risks. The first part contains the estimation of the portfolio delta and gamma. The second part consists of the quadratic approximation to calculate the portfolio standard deviation. In the last section, the authors calculate the Delta-Gamma Value-at-Risk and provide recommendations to Banco Invest.

Keywords: Value-at-Risk, Call digital option, Portfolio Delta, Portfolio Gamma, Delta-Gamma Value-at-Risk

This work used infrastructure and resources funded by Fundação para a Ciência e a Tecnologia (UID/ECO/00124/2013, UID/ECO/00124/2019 and Social Sciences DataLab, Project 22209), POR Lisboa (LISBOA-01-0145-FEDER-007722 and Social Sciences DataLab, Project 22209) and POR Norte (Social Sciences DataLab, Project 22209).

Contents

1 Value-at-Risk – Group part

Market risk describes the risk of a possible loss in a risk position due to collective adverse movements of market rates and prices. It is one of the most critical risks for institutions that actively trade in financial markets; quantifying and monitoring this risk is crucial for allocating capital and reserves needed to cover potential losses and assess their overall solvency. Market risks are determined by institutions using standard procedures or internal risk models; one of these procedures is the Value-at-Risk model. (Deutsche Bundesbank 2022)

1.1 Defining Value-at-Risk

The Value-at-Risk expresses the maximum potential loss, in absolute terms or as a percentage in the respective currency the asset is held, that results under normal market conditions from an adverse movement in the relevant market of an investment over a specified time horizon (H) at a given degree of confidence (α) during a fixed holding period of a risk position. The estimated maximum potential loss of the model, the VaR estimate, is only expected to be exceeded (1-α) % of the time. (Castellacci and Siclari 2003, pp. 531-532) (Fallon 1996, p. 2) The time horizon of interest for a VaR estimate can be one day or even months and is determined by the nature of the portfolio. The horizon should correspond to the most prolonged period needed for an orderly liquidation or the time to hedge an investment portfolio. (Bodie, Kane, and Marcus 2021, p. 138) The VaR estimate's horizon is determined by the liquidity profile of the assets in the underlying investment portfolio; the length relates to the time needed to sell these assets at average transaction volumes so that they have little impact on the market. Since the market impact of the liquidation scenario is not disregarded when choosing the horizon, the VaR estimate will be an estimate of a realizable loss and not only a loss on paper. (Wilmott 1998, p. 548) The confidence (α) level for a VaR estimate corresponds to the institution's risk profile, determined by its degree of risk aversion or regulatory requirements. (Fallon 1996, p. 2)

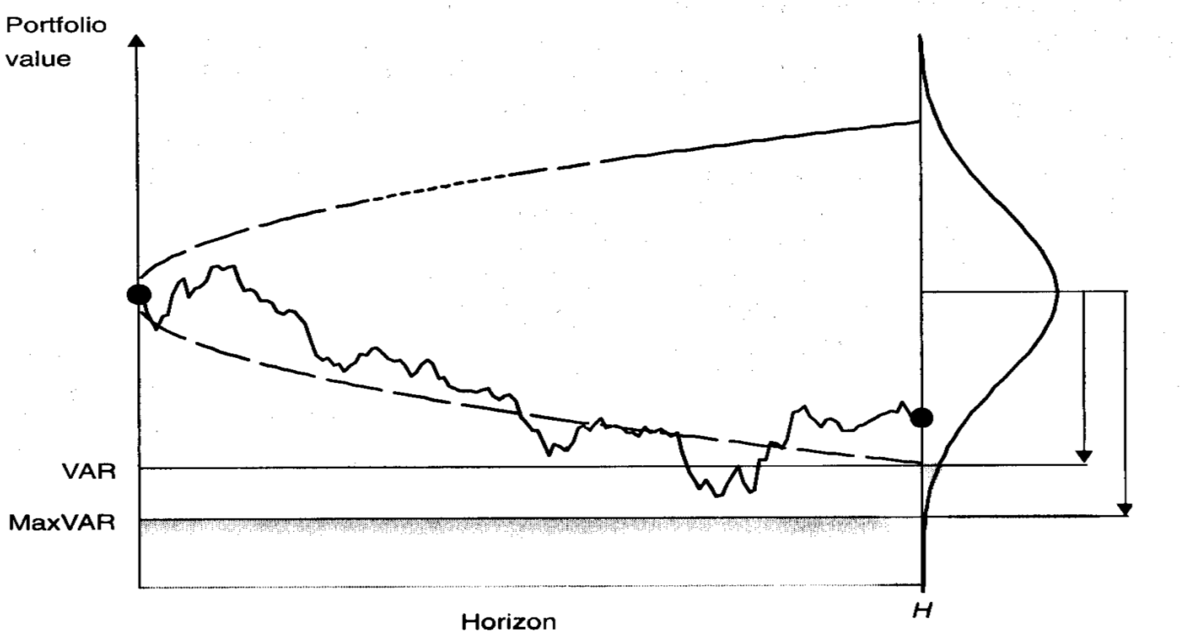


Figure 1: Development of VaR over time horizon H (Jorion 2007, p. 118)

A VaR calculation applies to all types of risky assets and can be applied to a single position and a whole portfolio of risky assets. Assessing VaR helps institutions evaluate the profitability of an investments in relation to the risk and identify investments with a higher-than-acceptable risk profile, allowing them to make changes or liquidate such investments. The VaR is used for active and passive risk measurement and defensive risk control. Ideally, it suits financial and non-financial institutions that engage in proprietary trading with significant exposure to market risks. (Jorion 2007, pp. 379-389) VaR estimates typically focus on 'tail events' where liquidity and large jumps are essential, as illustrated in ***Appendix 1*** below. (Wilmott 1998, p. 337) Therefore, confidence levels are typically set at 95%, 97.5%, and 99%. (Wilmott 1998, p. 547) An overview of which confidence levels translate into which z statics of the confidence interval can be found in ***Appendix 2***. The VAR statistic on portfolio losses is defined as a one-sided confidence interval:

$$Prob\ [\Delta\tilde{P}(\Delta t, \Delta\tilde{x}) > \ -VAR] = 1 - \alpha \tag{1}$$

In the above equation, $\Delta\tilde{P}(\Delta t, \Delta\tilde{x})$ stands for the change in the value of a portfolio that results from a function consisting of the forecasting period Δt and the vector $\Delta\tilde{x}$ of the random variables, with α being the confidence level. The equation can be interpreted as the portfolio's value will not fall by more than VAR over Δt number of trading days with α % confidence. (Fallon 1996, p. 2) The degree of complexity and the computational requirements of the calculation of a VaR estimate depends in particular on how the price of the instrument changes in relation to the underlying. ***Appendix 3*** depicts the two different relationships. (Romano 2017) The calculation of a VaR estimate for non-linear (i.e., derivatives) assets is more complex than for a linear asset (i.e., a stock or bond). In the context of an option: nonlinearity implies that a price movement in the underlying asset causes a non-linear change in the option price. There are three major methodologies to calculate Value-at-Risk, the historical approach, the parametric or model-building approach, and performing a Monte Carlo simulation. ***Figure 2*** below provides an overview of the different methodologies and their advantages and disadvantages. (Hull 2021, pp. 293-297 & 317-340)

Type	Description	Advantages	Disadvantages
Historical	Estimates VaR using past distribution of returns to predict future returns	- Easy way to calculate VaR - Takes into account possible skeweness and fat tails - Accurate for non-linear products - No distributional assumptions necessary	- Assumes future returns dependend on the past (impractical) - Large amount of daily rate history required - Slow reaction to recent market events
Parametric	Estimates VaR using prespecified variables (volatility & correlation)	- Quick and easy to compute - Accurate for simple & linear products	- Assumption of normal distribution impractical - Less quick and accurate for non-linear derivatives
Monte Carlo	Estimates VaR by simulating random scenarios	- Accurate for linear & non-linear products - Flexibility to choose different distributions - Flexibilty on the choice of variables - Outputs full distribution of potential product values	- Massive computational power required to revalue the portfolio in each scenario - Accuracy dependend on number of simulation performed

Figure 2: Overview of different approaches for VaR calculation (Hull 2021, pp. 293-297 & 317-340)

1.2 Pitfalls and limitations of Value at Risk

Despite the widespread use of the Value-at-Risk model, it has several drawbacks that will be briefly discussed in the following. First and foremost, all methods require making assumptions and using them as inputs for the mode; this can result in different outcomes even if the same modelling approach is used. Assumptions have to be made, e. g. about the applicable horizon and confidence level and the appropriate number of simulations. (Jorion 2007, pp. 542-557) Furthermore, all methods rely to some extent on historical data as a proxy to forecast future estimates. What has happened in the past does not necessarily imply that it will happen again in the future, so that estimation can be Inaccurate. (Jorion 2007, pp. 542-557) Second, there is yet to be an industry-wide standard to model VaR. The different approaches and models to calculate VaR can also lead to different estimates for the same portfolio. Hence, the correct interpretation is vital. (Jorion 2007, pp. 542-557) This brings us to the next limitation: a VaR estimate is calculated assuming normal market conditions, meaning extreme and rare events, such as so-called black swans, are not considered by the estimate. Because VaR only allows the risk manager to make statements about which value will not be exceeded with what degree of certainty, it does not tell anything about the worst outcome in case the VaR number is ex (Hull 2018, pp. 273-274) Additionally, the traditional VaR disregards intervening losses. These occur when the portfolio's value falls below VaR during the time horizon but eventually rises above it at the end of it. This can be an essential aspect for management if the portfolio is marked to market daily and faces potential margin calls that could result in liquidation in the worst-case scenario. (Jorion 2007, pp. 117-119) A VaR estimate provides the "big picture" of what is at risk regarding market risk effects. However, as it only accounts for this specific risk type, it has a narrow focus on what is really at risk. There are also risks which are not incorporated in the VaR framework, commonly referred to as "risks not in Value-at-risk" (RNIV): This can result in the actual Value at Risk of an investment being much higher than what the VaR model is predicting when capturing many of the other existing risk variables such as (geo-)political risks, liquidity risks, and regulatory risk. (Jorion 2007, pp. 542-557)

2 The “Greeks” - Group part

In option pricing, as well as for other derivatives, the "Greeks" are commonly used to measure the sensitivity of a derivative's value to factors that might affect the price of an options contract. ***Appendix 4*** gives an overview of the existing Greeks and their definitions. (Leoni 2014) Within the frame of this work, the focus will be set on two risk metrics, delta (Chapter 3.1) and gamma (Chapter 3.2) risk, in relation to option pricing, as they are the most fundamental.

2.1 Delta Risk

The delta, designated with the symbol Δ, is the first-order partial derivative of the option pricing function c with respect to the underlying asset S. Therefore, it expresses the sensitivity of the option contract’s price to changes in the price of the underlying asset while leaving all else constant (ceteris paribus). (Taleb 1997, p. 224) (Bouzoubaa and Osseiran 2010, p. 66)

$$\Delta = \frac{\partial c}{\partial S} \tag{2}$$

For vanilla options, the delta for long calls and short puts on standard options varies between 0 and 1. Vice versa, short calls and long puts have a delta ranging between 0 and -1. Graphically expressed is it the slope of the curve that links the option price to the underlying asset price. The higher the slope, the higher the delta and the more the derivative contract will change in response to price fluctuations of the underlying asset. ***Figure 3*** below depicts the change in delta with respect to the Strike price K and the time to maturity T for a European call option. With the option increasingly getting out of the money (OTM), a higher Strike K, and/or the option approaching its maturity date T, the delta tends to move towards 0. Conversely, with lower Strike K, the option being more in the money (ITM), and/or longer time until maturity T, delta approaches 1. (Hilpisch 2015, p. 78) The most significant change in delta can be observed with the option being at the money (ATM), S = K, close to its maturity date T. This is because theoretically, with the option being ATM a few seconds before it matures, one small move in either direction would result in the option being either in the money or out of the money, hence the considerable variation in delta. (Hilpisch 2015, p. 78)

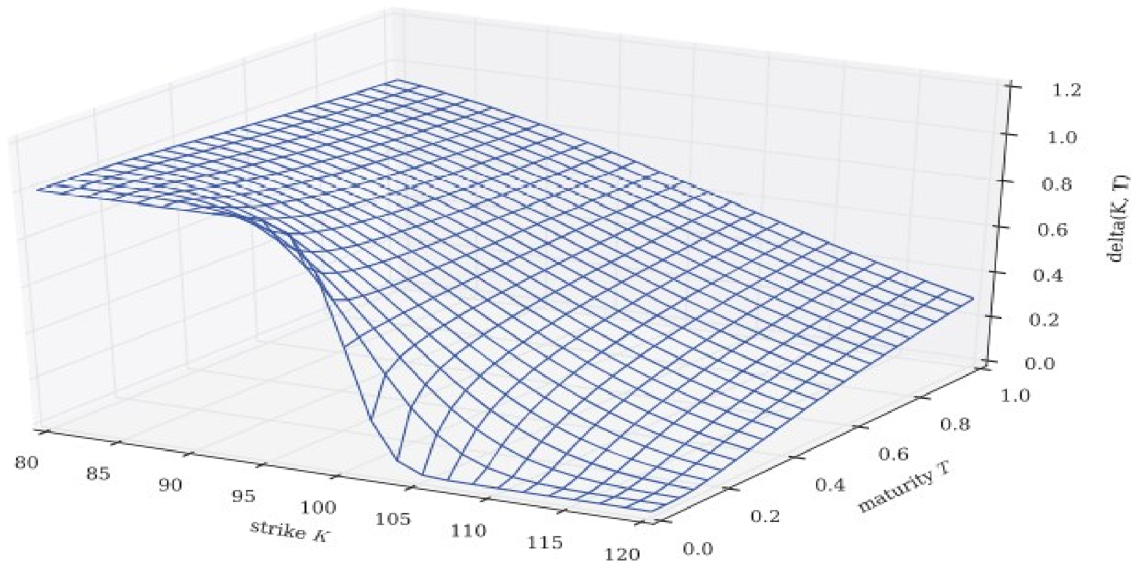


Figure 3: Delta of a European Call Option (Hilpisch 2015, p. 78)

Delta risk can be hedged to obtain a neutral position ($\Delta = 0$). How this can be achieved for a portfolio of derivatives will be explained in more detail in section 3.3, Hedging the Greeks.

2.2 Gamma Risk

For minor variations in the price of the underlying asset, delta proves to be good at estimating the change in the option's price. However, as soon as price changes become more severe, delta is extremely sensitive to changes in the underlying asset's price. This is because delta graphically represents a linear estimate for a non-linear option function. Hence, the actual option value might significantly differ from the proportion predicted by delta. (de Weert 2008, pp. 14-16) Gamma, Γ, measures by how much or how often a position or a portfolio of options needs to be re-hedged to maintain a delta-neutral position: it expresses by how much the Delta might change if the price of the underlying changes. It is the second-order derivative of the option pricing function c with respect to the underlying asset S.

$$\Gamma = \frac{\partial^2 c}{\partial^2 S} \tag{3}$$

The more curvature the option function entails, the higher the gamma and the more sensitive the delta is towards changes in the underlying's price. An increase in the underlying's price could significantly increase the delta and vice versa for a low gamma. Considering plain vanilla options, the gamma is always positive for long positions, whereas for short positions, it is negative. (Bouzoubaa and Osseiran 2010, p. 72) ***Figure 4*** below shows that the gamma value is stable for most of the option's life as it hovers near zero. The most notable value changes in gamma happen around ATM options close to maturity. As previously stated in the preceding section, it is for at-the-money options close to maturity where one move in either direction has the most significant influence on delta as it determines whether the option is exercised. Hence, the high value in gamma. (Yen Jerome and Lai 2015, pp. 84-85).

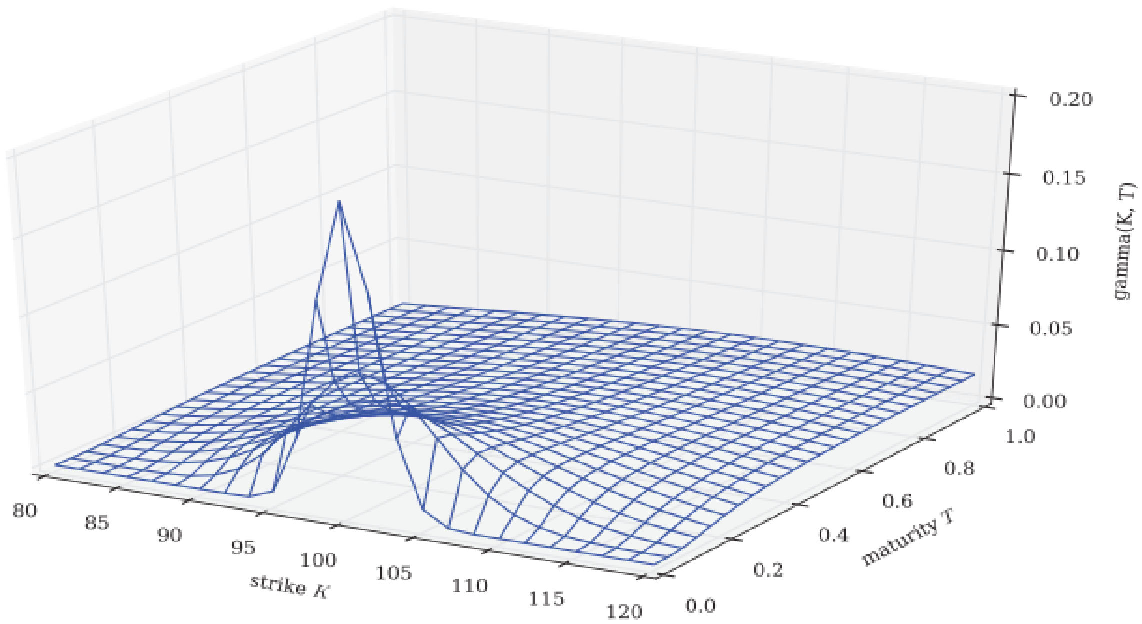


Figure 4: Gamma of a European Call option (Hilpisch 2015, p. 79)

How gamma is incorporated when hedging the respective portfolio's VaR will be explained in more detail in the next section.

2.3 Hedging the Greeks

As previously described, a portfolio's sensitivity to such is captured by the "Greek letters". The risk framework captures thresholds for each to ensure that these risks stay within the company's tolerance. Exceeding the limits initializes a process known as hedging. This is where counter positions in the market are established to ensure that the exposure to a particular risk factor stays within its predefined limit. In the following, it will be presented how a portfolio is hedged against delta and gamma. (Hull 2018, p. 161) Hedging delta consists of establishing a counter position equal to Δ amount of the underlying. By combining the existing portfolio and the hedging trade, the new portfolio's exposure to delta is neutralized. (Hull 2018, pp. 161-162) For linear products, hedging delta turns out to be static as it protects against both small and large changes in the value of the underlying. Further, once a linear hedge is implemented, there is no need to adjust it over time. The delta for a linear portfolio stays constant. (Hull 2018, pp. 163-164) Neutralizing delta exposure for non-linear products such as options proves to be a more complex procedure due to the non-linear relationship between the price of the underlying and the options contract. As mentioned earlier in this work, eliminating a portfolio's delta only offers protection from small fluctuations in the price of the underlying. Additionally, once it is set up, the delta hedge has to be adjusted frequently, also known as dynamic hedging or "rebalancing". This is because Delta constantly evolves throughout a non-linear product's lifetime. (Hull 2018, pp. 165-168) In practice, rebalancing is costly as, e.g., hedging a long position on an option involves buying the underlying when its price increased and selling it when it dropped to consistently create a synthetical position opposite of that to neutralize the option's delta. This is usually reflected in the premiums that option buyers have to pay. (Hull 2018, p. 169) With more significant changes in the prices of the underlyings, a portfolio's gamma comes into play. There are two ways of adjusting for the additional gamma exposure of a non-linear portfolio that will be briefly described below. (Hull 2018, pp. 169-170) Firstly, the portfolio is made gamma neutral by trading options with opposite gammas on the same underlyings as the options in the existing portfolio. Non-linear products are needed as linear products do not have exposure to gamma. By doing this, the new and combined portfolio's delta also changes and would have to be re-adjusted by trading opposite positions in the underlyings (Hull 2018, pp. 170-171) Implementing this in practice can be challenging as trading non-linear derivatives in the amounts needed often is impossible. Further, re-adjusting for the new delta of the combined portfolio is costly as it involves many transactions. (Hull 2018, p. 177) However, as described earlier, it makes economically more sense to see the gamma as a determinant of how often a portfolio needs to be re-hedged. In general, a portfolio with larger gamma would imply more frequent delta neutralization, whereas a smaller gamma results in less often adjustments to the portfolio, as changes in delta only tend to be small. (Hull 2018, pp. 169-170) Banco Invest hedges its equity derivatives portfolio with underlyings (delta neutralization) rather than options (gamma neutralization). The Bank does not take directional market risk, keeping the difference between the deltas (theoretical quantities) and the quantities held in the portfolio as close to zero as possible. These portfolio quantities are adjusted daily, at 30-minute intervals, based on market conditions, namely the evolution of the underlying shares.

3 Value-at-Risk for a Derivatives Portfolio - Group part

To begin with, calculating Value-at-Risk for a single asset is a straightforward process. Assuming linearity in the change of the portfolio's value to changes in the underlying and normally distributed returns, VaR is calculated as follows:

$$VaR = w_i S_i \left(\mu \, \delta t - \sigma_i \, (\delta t^{\frac{1}{2}}) \, \alpha(1-c) \right) \tag{4}$$

where w_i is the quantity of the asset i owned with price S_i. This is multiplied by the asset's drift over a predefined time horizon δt, with $\alpha(1-c)$ being the inverse cumulative distribution function of the standard normal distribution. This process is called delta approximation. (Wilmott 1998, pp. 548-550) Regarding a portfolio of assets, the calculation of VaR becomes more complex. First, the volatilities and covariances of all assets in the portfolio have to be computed. If this is done, the formula to calculate the VaR of a portfolio with M assets consisting of w_i amount of asset i and w_j amount of asset j is:

$$VaR_{Portfolio} = -M\left(\alpha(1-c)(\delta t^{\frac{1}{2}})\sqrt{\sum_{i=1}^{M}\sum_{j=1}^{M} w_i w_j \sigma_i \sigma_j \rho_{ij}}\right) \tag{5}$$

with σ_i being the volatility of asset i and ρ_{ij} the correlation between asset i and j. (Wilmott 1998, pp. 551) Estimating VaR for a portfolio of derivatives, as mentioned earlier, the delta approximation would only be sufficient for portfolios where the underlyings show small movements in price. This is because the relationship between the portfolio's value and price changes in the underlyings can no longer be regarded as linear. For non-linear portfolios, the sensitivity to gamma additionally has to be considered. This is visually demonstrated in ***Figure 5*** below. It depicts the relationship between the price of an underlying asset to the corresponding value of a long call option on the same. While the underlying’s price function is normally distributed, the option has a positively skewed probability distribution with a smaller tail on the left. (Hull 2018, pp. 333-334) This violates the initial premise that probabilities are normally distributed. If VaR were calculated based on this assumption, it would be excessively high. As a result, approximations for the portfolio's sensitivity to changes in the underlyings need to be reevaluated. (Wilmott 1998, pp. 550-551)

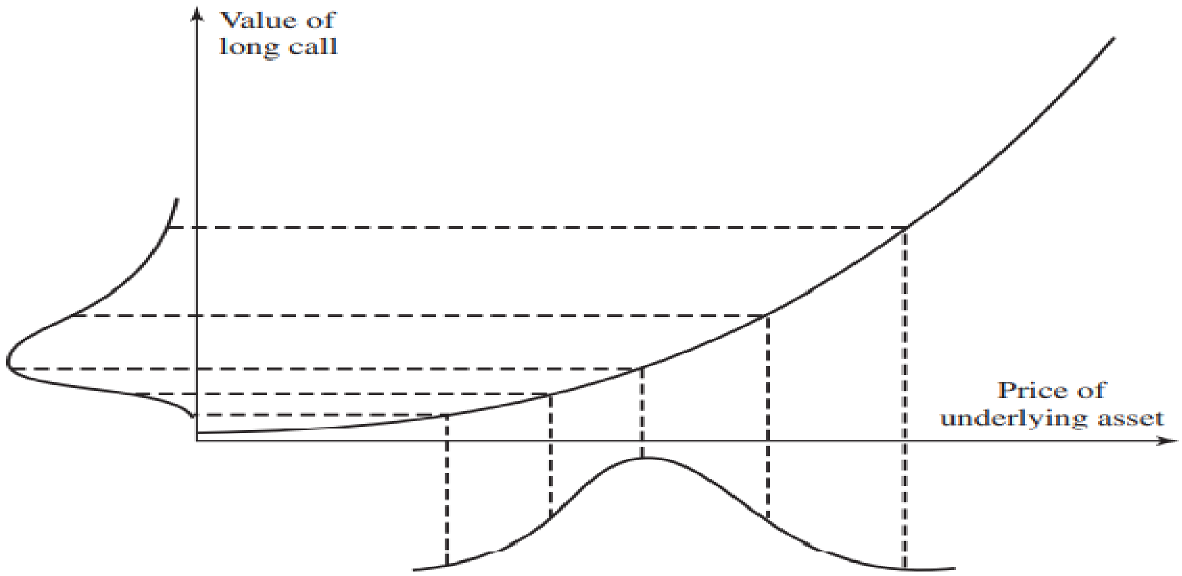


Figure 5: Translation of an Asset's normal probability distribution into that of a long call option (Hull 2018, p. 333)

To recapture, with larger swings in the prices of the underlyings of an options portfolio, the previous delta approximation to calculate VaR turns out to be inappropriate. A better estimation is achieved by incorporating the portfolio's sensitivity to gamma. Gamma exposure is particularly challenging as a second-order approximation is required. (Wilmott 1998, p. 551) This will be shown below. Assume a portfolio M consisting of a single option on an asset with price S. The change in the value of the portfolio δM compared to changes in the price of the underlying δS can be expressed as follows:

$$\delta M = \frac{\partial P}{\partial S}\delta S + \frac{1}{2}\frac{\partial^2 P}{\partial S^2}(\delta S)^2 + \frac{\partial P}{\partial \sigma}\delta t + \cdots \quad (6)$$

This can ultimately be reformulated into:

$$\delta M = \Delta\sigma S\, \delta t^{\frac{1}{2}}\, \phi + \delta t\left(\Delta\mu S + \frac{1}{2}\Gamma\sigma^2 S^2 \phi^2 + \Theta\right) + \cdots \quad (7)$$

where Θ is the time drift of the option (Theta). (Wilmott 1998, p. 551) The quadratic term, the portfolio's exposure to gamma, is of specific interest above. ***Figure 6*** shows three different distribution functions. The distribution of the underlying with a standard deviation of $\sigma S\ \partial t^{\frac{1}{2}}$ is considered to be normal. The projected distribution for the change in the value of the options portfolio according to the delta approximation. It is normally distributed with a standard deviation of $\Delta\sigma S\,\partial t^{\frac{1}{2}}$. Finally, the options portfolio's distribution using the delta-gamma approximation. (Wilmott 1998, pp. 551-552)

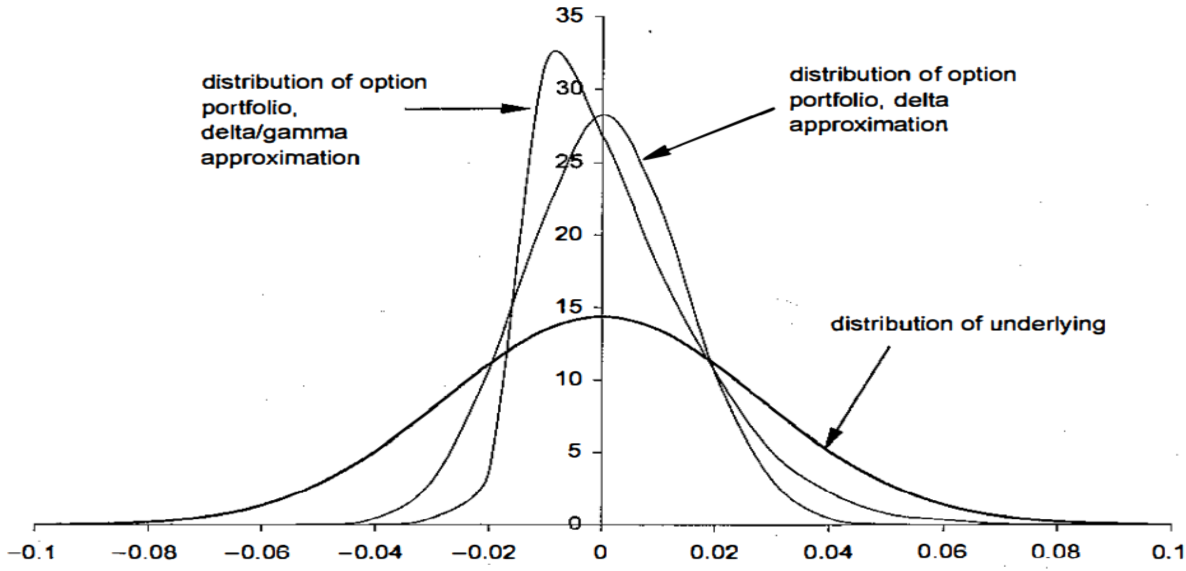


Figure 6: Relationship of an asset price's normal distribution to the distribution of an option portfolio according to the delta as well as the delta-gamma approximation (Wilmott 1998, p. 552)

By looking at the three different distributions, it is evident that the one for the delta-gamma approximation is not normally distributed compared to the other two. (Wilmott 1998, pp. 551-552)

4 Methodology used in Python - Group part

In the following, the Assumptions used to calculate the Delta-Gamma VaR in Python, as well as the fundamental parts of the code, are presented and explained. As the basis for all calculations of the various input statistics of the VaR model, the authors assume one year consisting of 252 trading days. Because of their ease of use for time series modelling, such as symmetry, time-additivity, and the log-normal distribution assumption, the various underlyings performances are transformed into logarithmic returns. Next, **e**ach option's volatility is calculated using equally weighted implied volatilities of the option's underlyings. In the absence of implied volatility, the underlying's historical volatility on a 30-day basis is used. Furthermore, to determine the correlation, variance, and covariance of the different underlyings, a maximum lookback window of 2 years is assumed, the same as the option's time to maturity on the trade date. From there on, for each day that has progressed, the option's remaining time to maturity is used to calculate the above statistics until a predefined minimum of 30 days was reached. Below this, correlation, variance, and covariance are calculated on a 30-day basis until the option matures. At this point, it is referred to ***Appendix 5-6*** for the code example. The options in Banco Invest's portfolio are valued as of **30/06/2022** using Monte Carlo simulations. The first step of Monte Carlo involved calculating the geometric Brownian Motion. In finance, this is a stochastic process to model random behavior over a specific time frame (δt) that consists of two main components, drift, and a randomly generated variable. (Yan 2017, pp. 421-428) Drift indicates the direction of an asset's historical returns, allowing predictions on an asset's expected return. It is calculated as shown in ***equation (8)*** using the same receding time horizon as explained for the underlying's statistics, except for the time series' minimum requirement of 30 days.

$$Drift = \left(Mean\ (stock\ returns) - \frac{Variance\ (stock\ returns)}{2} \right) * \ \delta t \qquad (8)$$

Where underlyings are expected to pay dividends, the drift is adjusted further, as demonstrated in ***Appendix*** *7*. The next step is to obtain a random number by multiplying an asset's historical standard deviation with a random, standard normally distributed variable *(Z ([Rand (0;1)]))*.

$$Random\ variable = \left(Std.Dev.\ \ * \ Z([Rand(0;1)]) \right) * \sqrt{\delta t} \qquad (9)$$

As a result, the equation for predicting the future value of an asset (S_{t+1}) sums up to the following:

$$S_{t+1} = S_t * e^{Drift\ +\ Random\ variable} \qquad (10)$$

However, when pricing options comprised of baskets of underlyings, Cholesky Decomposition is performed as an extension of the Monte Carlo simulation to account for the correlation aspects between the various reference assets. A brief explanation of an example decomposition will be provided below. ***Appendix 8*** contains the code for the Cholesky decomposition performed for the different options. Assume a $2 * 2$ symmetric, positive definite correlation matrix Σ, where ρ is the correlation between $\boldsymbol{X_1}$ and $\boldsymbol{X_2}$.

$$\Sigma = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \tag{11}$$

The correlation matrix can then be decomposed into a $2 * 2$ lower triangular matrix L, where $LL^T = \Sigma$. (Wilmott 1998, pp. 682-683) This appears to be as follows:

$$L = \begin{pmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{pmatrix} \tag{12}$$

Following the generation of L, the random variables with desired correlation can be expressed as LZ, where Z is a column vector of the independent standard normal random variables:

$$Z = \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} \tag{13}$$

As a result, by setting $XL = Z$, we can sample from a bivariate normal distribution, indicating that: (Yen Jerome and Lai 2015, pp. 99-100)

$$X_1 = Z_1 \tag{14}$$

$$X_2 = \rho * Z_1 + \sqrt{1-\rho^2} * Z_2 \tag{15}$$

To generate a sufficient sample of possible future asset values for the different underlyings to calculate the option's payoffs appropriately, 200.000 simulations are run. Following this, the averaged payoffs are discounted using the respective's maturity Euribor 3-month forward. Where no forward for the maturity of the option's payoffs is readily available, linear interpolation is performed to compute the discount rate for the respective maturity's payoff, as shown in ***Appendix 9-10***. Further, each underlying's delta is estimated by changing its price by 1%, while leaving the other's prices constant, and calculating the new price of the option. The difference in both derivative prices is then divided by the relative changes in the prices of the underlying. The option's delta is estimated as the weighted average of the underlying's deltas, assuming an equally weighted portfolio of underlyings. To calculate gamma, the above calculation is done a second time to get the change in delta. The difference in both deltas is then divided by the relative adjustment to obtain the gamma value. The equations used and the respective code for this can be found in ***Appendix 12-26***. In terms of VaR, the confidence level was set to 99,9 %. Calculations are performed initially for a one-day time horizon and then later multiplied by the square root of 252 to get the annualized VaR, as this is the requirement from the risk management department at Banco Invest. Detailed calculations performed for this in Python can be found in ***Appendix 26***.

5 Call Digital – Individual part (Tim Gajewski)

Digital options, also called binary options are a type of product that guarantees the initial capital invested plus a conditional coupon, dependent on the performance of a single or basket of underlyings. (Bodie, Kane, and Marcus 2021, p. 689) With two possible scenarios, the payoff for a digital option is discontinuous. Either the investor receives a prespecified coupon if a certain exercise condition is met or the option expires worthless in case it is not. (Yan 2017, p. 497) The payoff function can be written as per below:

$$Cash - or - nothing_{payoff} = \begin{cases} C & if\ S(t) > K \\ 0 & otherwise \end{cases} \tag{16}$$

where the option pays coupon C if the asset price S(t) at time t exceeds the strike price K and 0 otherwise. (Bouzoubaa and Osseiran 2010, pp. 167-168) Just as with plain vanilla options, European style digitals are assessed only on the maturity date whereas American digitals are monitored each day during the life of the option.

Further can digitals be differentiated in being cash-or-nothing or asset-or-nothing. While the latter pays the price of the underlying asset in the event of the asset price exceeding the strike price, cash-or-nothing digitals pay a predefined coupon. In case the assets price does not exceed the strike, both only return the initial investment. (Hull 2021, pp. 600-601) Banco Invest solely trades European cash-or-nothing digital options. Hence, in the following when discussing digital options, the author is implicitly referring to European cash-or-nothing style digital options.

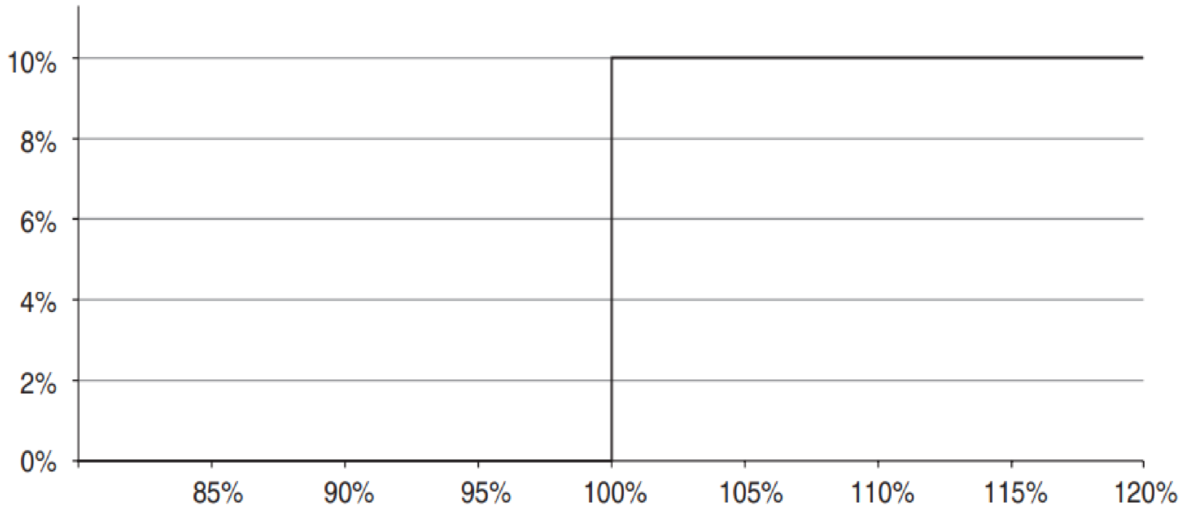


Figure 11: Example payoff of a 10 % digital call option with strike at 100 % (Bouzoubaa and Osseiran 2010, p. 168)

The characteristic discontinuity payoff feature can best be seen by looking at the graphical illustration of the payoff function shown in Figure 11 above. If the asset's price ends below the strike price of 100 %, the investor receives only the initial amount invested back. In case the price of the asset finishes above the strike price, he receives the full coupon of 10 % plus the initial investment. This discontinuity is also what makes them particularly hard to hedge using regular instruments. (Taleb 1997, 503)

Portfolio - Payoff Type: Call Digital

Product ID	Name	Effective date	Maturity date	Cap	Floor
1023	Invest 5G Jul-20	31/07/2020	08/08/2022	2.40%	0%
1058	BIC Healthcare Set-20	16/09/2020	23/09/2022	1.60%	0%
1060	Invest Green Deal Out-20	30/10/2020	07/11/2022	2.75%	0%
1079	BIC Energia Nov-20	16/11/2020	23/11/2022	1.50%	0%
1115	Invest Smart Cities Jan-21	29/01/2021	06/02/2023	2.10%	0%
1134	BIC Cabaz Global Fev-21	16/02/2021	23/02/2023	1.50%	0%
1149	BIC Multisector Mar-21	16/03/2021	23/03/2023	1.20%	0%
1154	BIC Sustentabilidade Abr-21	16/04/2021	24/04/2023	1.20%	0%
1182	BIC Cabaz Multinacionais Jun-21	16/06/2021	23/06/2023	1.20%	0%
1251	BIC Tecnologia 5G Nov-21	16/11/2021	23/11/2023	1.20%	0%
1316	Invest Health & Biotech Fev-22	28/02/2022	06/03/2024	1.95%	0%
1386	BIC Value Mar-22	16/03/2022	25/03/2024	1.20%	0%

Figure 12: Overview of Banco Invest's Call Digital options portfolio

5.1 Portfolio Delta

The bank is short on the underlying when Banco Invest sells Digital options to the market. Therefore, hedging the spot risk with delta is of utmost importance. The calculated delta for each digital option and the total portfolio of digital options at Banco Invest are shown in Figure 5 below. The notional value of delta was also calculated to determine how much euros would need to be invested in each option's underlying in order to offset the notional value of delta across the entire digital portfolio.

Delta (Δ) - Call Digital Products

Product ID	w_i	Delta (Δ)	Numerical value (€)
1023	5,62%	0,0936	233.138,76 €
1058	9,67%	0,0766	328.468,05 €
1060	5,05%	0,0383	85.703,68 €
1079	8,34%	0,0045	16.516,77 €
1115	2,43%	0,0062	6.691,44 €
1134	12,44%	0,0074	40.666,83 €
1149	10,74%	0,0035	16.808,65 €
1154	10,66%	0,0023	11.081,23 €
1182	9,37%	0,0035	14.529,32 €
1251	9,48%	0,0081	33.877,77 €
1316	4,95%	0,0167	36.692,69 €
1386	11,26%	0,0103	51.347,65 €

Delta (Δ) - Aggregated Call Digital Portfolio

Product Type	Notional	Delta (Δ)	Numerical value (€)
Indicap	44.363.647,75 €	0,0197	875.522,81 €

Figure 13: Delta of Banco Invest's Call Digital options portfolio

Being first said, the delta of a Digital call is behaving fundamentally different to a plain Call. As after a certain strike the investor receives a fixed payoff and below the strike nothing, the highest value of delta is at the strike and staying elevated around the strike. The further the underlying is distant from the strike (both sides!), the smaller the expected change in payoff resulting from a move in the underlying, the smaller the delta. Looking at the figure, it is noticeable that the majority of deltas is very small, below 0.01. This is, for example, the case for the delta for digital ID 1079. Considering, at the time of pricing, the option only had a couple of months left until its maturity date and additionally looking at the strike prices of each underlying in relation to the prices the underlying were trading as of 30/06/2022, the chances of the underlying to catch up to missing 68% is pretty small, hence the delta value. But we see the same dynamic in for deep in the money options: The digital call with the ID 1115 has a delta of 0.0062 and is deep in the money, with around eight months until maturity.

The highest delta can be observed for option ID 1023 (0.0936) and 1058 (0.0766), respectively. Both the option’s underlyings are in a reasonable range to their strikes, with an increasing likelihood to be called. In terms of the overall portfolio delta, multiplying each option's weight with its respective delta it was estimated to be 0.0197. Hence, buying the different underlying worth a total of 875,522,81 € would neutralize the portfolio’s delta.

Although, as we said, it is in the nature of digital options, the estimated deltas are still very small in comparison to standard or other exotic options. To prevent trading in unexpectedly large amounts of the underlying, Delta must be carefully monitored on these dates. Remember that the digital option is harder to hedge because there is no continuous movement with the underlying and there is a jump in the payoff at the strike. In actuality, it is advised to adjust the hedging more frequently the closer the options are to the observation date or maturity. Digitals that trade closely around the strike in the final days before maturity present the biggest challenge. A further dive into this topic will be below, as it takes gamma into consideration.

5.2 Portfolio Gamma

In Figure 14 below, the gamma for each option and also for the whole portfolio of digital options is summarized. For options with a very small delta, we have seen before, one can see significantly higher gamma values. As the options are deep out of the money or in the money, gamma seems to be still elevated compared to a plain call option. In terms of rebalancing, these options do need to be frequently delta hedged to neutralize directional exposure in the market. As we did not look at an absolute change to calculate the Greeks but rather a relative change, we can see for two digital options that are closer to the strike a Gamma even higher than 1. The biggest gamma value can be observed for the digital with ID 1023. This is due to all of the option's underlying trading around the strike, and therefore we see the strongest change in delta here. With decreasing time of maturity gamma is also increasing for options with underlying trading around the strike price as the change in delta is the highest. It also has the consequence that ID 1023 should be delta hedged the most frequent, as of 30/06/2022.

Gamma (Γ) - Call Digital Products

Product ID	w_i	Gamma (Γ)
1023	5,62%	1,1995
1058	9,67%	0,4786
1060	5,05%	1,0929
1079	8,34%	0,1489
1115	2,43%	0,4136
1134	12,44%	0,2948
1149	10,74%	0,3526
1154	10,66%	0,1172
1182	9,37%	0,3495
1251	9,48%	0,2302
1316	4,95%	0,1858
1386	11,26%	0,2284

Gamma (Γ) - Aggregated Call Digital Portfolio

Product Type	Notional	Gamma (Γ)
Indicap	44.363.647,75 €	0,3678

Figure 14: Gamma of Banco Invest's Call Digital options portfolio

For the overall portfolio, the gamma, calculated by taking the weighted average, was estimated to be 0,3678. As mentioned earlier, significantly higher compared to the regarding deltas.

5.3 Non-linear Delta-Gamma-VaR

Figure 15 below provides an overview of the different Value-at-Risk metrics calculated for each option within Banco Invest's Call Digital options portfolio.

Value-at-Risk (1d, 99,9%) - Call Digital Products		
Product ID	VaR in %	Numerical value (€)
1023	5,76%	143.610,78 €
1058	5,25%	225.391,38 €
1060	0,76%	16.960,15 €
1079	0,01%	518,30 €
1115	0,41%	4.443,51 €
1134	0,27%	14.651,00 €
1149	0,42%	19.906,29 €
1154	0,24%	11.195,35 €
1182	0,03%	1.151,10 €
1251	0,20%	8.559,03 €
1316	1,05%	22.995,42 €
1386	0,30%	15.121,31 €
	Undiversified VaR:	484.503,62 €

Value-at-Risk (1d, 99,9%) - Aggregated Call Digital Portfolio			
Product Type	Notional	Diversified VaR (in %)	Diversified VaR (in EUR)
Call Digital	44.363.647,75 €	0,25%	111.408,72 €

Figure 15: Delta-Gamma VaR for the Call Digital options portfolio of Banco Invest

Due to diversification effects, we end up with a Value at Risk for the Portfolio of around 111,408.72€, around 0.25% less than summing up the value at risk for the options each. As each option is focused on a specific industry, individual VaR tends to be relatively high. Considering a mix of these different industries in the overall digital options portfolio, the combined VaR is much lower. We see for the IDs 1023 and 1058 a very high VaR of over 5%, as we look at 1d and the 99.9% interval. In contrast we have also two digitals with a VaR of less than 0,05%, ending up in a portfolio VaR of 111,408.72€ (around 0,25%).

6 Recommendation - Group part

This chapter address how the bank's management should deal with the risk associated with the derivatives Portfolio. ***Figure 35*** below summarizes the delta, gamma, and Delta-Gamma Value-at-Risk for Banco Invest's overall options portfolio. The total derivatives portfolio of the bank has a notional of EUR 157.067.916, consisting of 53 different options. The 1-day Value-at-Risk at 99,9% confidence level for the bank's overall derivatives portfolio is EUR 372.773, implying a 99,9 % probability the portfolio will not lose more over the next trading day.

Banco Invest - Aggregated Derivatives Portfolio	
Notional	157.067.916,00 €
No. of option positions	53
Delta (Δ)	0,0163
Gamma (Γ)	0,3166
Volatility	4,91%
VaR (1d, @ 99,9%)	0,24%
VaR (1d, @ 99,9%)	372.773,00 €

Figure 35: Aggregated Portfolio Delta-Gamma VaR

As the bank does not take a directional risk on the market, the delta on combined option's portfolio must be neutralized with an appropriate hedging strategy. All five option types in the Banco Invest derivatives portfolio are basket options. The challenge of hedging, when facing options with a basket of underlying's, becomes evident in their correlated structure. This makes the evaluation of the contract's price but also the risks, e.g., delta, gamma, and their hedging a complex procedure. (Su 2006, pp. 3-5) This is because it is difficult to detangle the underlying basket's distribution. The correlation between the underlying tends to be volatile and can only be estimated. This further complicates the "perfect" hedging of basket options. As a result, in many cases, only a part of the underlying basket is used for hedging, or the payoffs of the basket are replicated "super-hedged". (Su 2008, pp. 19-23) Another difficulty arises from the number of underlying assets: When following a standard dynamic hedging strategy, a hedging portfolio for the basket options should be related to the underlying assets in the basket. The larger the amount of underlying's the more difficult it is to implement such a dynamic strategy and the larger the transactions cost, caused by the continuous rebalancing, become. Since most of the options are "near-zero-gamma", which means that the directionality, the delta of the option is not greatly affected by changes in the underlying market prices, a dynamic hedging strategy can be implemented as major changes in the delta are not expected to be caused by changes in the underlying market prices. Transaction costs for rebalancing will occur but will be manageable as they do not occur very frequently. Lamberton and Lapeyre (1992) showed that a dynamic hedge on even a subset of the underlying's works well: they developed a method using multiple regression analysis to create a dynamic approximate hedging portfolio of plain-vanilla options on only a subset of the underlying's. For our "near-zero-gamma" options, such a dynamic hedge could further reduce the already low cost of rebalancing. A static hedging strategy has the advantage that transaction costs caused by continuous rebalancing can be avoided, and therefore this strategy could have a better hedging performance. (Su 2008, pp 2-4) Su (2006) used the Principal Components Analysis (PCA) to demonstrate that also a static hedge on a subset of the underlying's performs well: The PCA was used to determine a dominant subset of assets of the basket. Since a dynamic hedge of a basket option often only approximates the optimal hedge, the complete neutralization of the delta can only be achieved by a static hedge. Since Banco Invest instructs it takes no directional risk in the market, the only hedging strategy that fits this case is a static strategy as described above. Moreover, since the assets in the respective basket options are all in the same thematic investment universe, it is worthwhile to follow the approach of Su (2006) to determine whether it is sufficient to apply a static hedge only to a subset of the underlying assets, due to the high correlation between them.

Bibliography

Banco Invest. 2022. "Risk Management on the Equity Derivatives."

Bodie, Zvi, Alex Kane, and Alan J. Marcus. 2021a. "Investments." New York.

———. 2021b. *Investments*. New York.

Bouzoubaa, Mohamed, and Adel Osseiran. 2010a. "Exotic Options and Hybrids." Chichester.

———. 2010b. *Exotic Options and Hybrids*. Chichester.

Castellacci, Giuseppe, and Michael J. Siclari. 2003. "The Practice of Delta–Gamma VaR: Implementing the Quadratic Portfolio Model." *European Journal of Operational Research* 150 (3): 529–45. https://doi.org/10.1016/S0377-2217(02)00782-8.

Chuan, T. 2008. Capital Protected is not Capital Guaranteed. *Financial Planning Central*

Deng, Geng, Joshua Mallett, and Craig McCann. 2011. "Modeling Autocallable Structured Products." *Journal of Derivatives and Hedge Funds* 17 (4): 326–40. https://doi.org/10.1057/jdhf.2011.25.

Deutsche Bundesbank. 2022. "Deutsche Bundesbank: Marktrisiko." 2022. https://www.bundesbank.de/de/aufgaben/bankenaufsicht/einzelaspekte/eigenmittelanforderungen/marktrisiko/marktrisiko-598476.

Fallon, William. 1996. "Calculating Value-at-Risk." Philadelphia. https://www.researchgate.net/publication/2428988_Calculating_Value-at-Risk.

Gharavi, Reza K. 2010. "Encyclopedia of Quantitative Finance" 4.

Guillaume, Tristan. 2015a. "Autocallable Structured Products." *THE JOURNAL OF DERIVATIVES* 73. www.iijournals.com.

———. 2015b. "Analytical Valuation of Autocallable Notes." *International Journal of Financial Engineering* 02 (02): 1550016. https://doi.org/10.1142/s2424786315500164.

Hilpisch, Yves. 2015. "Derivatives Analytics with Python." Chichester.

Hull, John C. 2018. "Risk Management and Financial Institutions." New York.

Hull, John C. 2021a. "OPTIONS, FUTURES, AND OTHER DERIVATIVES." New York.

———. 2021b. *OPTIONS, FUTURES, AND OTHER DERIVATIVES*. New York.

Jorion, Philippe. 2007. *Philippe Jorion - Value at Risk, 3rd Ed._ The New Benchmark for Managing Financial Risk (2006, McGraw-Hill)*. Third. The McGraw-Hill Companies, Inc.

Leoni, Peter. 2014. *The Greeks and Hedging Explained*.

Romano. 2017. "Options Trading." May 2017. https://romanornr.medium.com/options-trading-fd4d0bffb2c5.

Shefrin, Hersh. 2002. *Beyond Greed and Fear*. Oxford University PressNew York. https://doi.org/10.1093/0195161211.001.0001.

Su, Xia. 2006. "Hedging Basket Options by Using a Subset of Underlying Assets." 14. Bonn. https://www.econstor.eu/bitstream/10419/22959/1/bgse14_2006.pdf.

———. 2008. "Essays on Basket Options Hedging and Irreversible Investment Valuation." Rheinische Friedrich-Wilhelms-Universität Bonn. https://bonndoc.ulb.uni-bonn.de/xmlui/handle/20.500.11811/3322.

Taleb, Nassim Nicholas. 1997a. "Dynamic Hedging: Managing Vanilla and Exotic Options." New York.

———. 1997b. *Dynamic Hedging: Managing Vanilla and Exotic Options*. New York.

Wealth Focus Pty Ltd. 2010. "Guide to Capital Protected Products." Manly: Wealth

Focus Pty Ltd. https://www.fundsfocus.com.au/managed-funds/pdfs/Capital-Protection.pdf.
Weert, Frans de. 2008. “Exotic Options Trading.” Chichester.
Wilmott, Paul. 1998. *Derivatives : The Theory and Practice of Financial Engineering*. John Wiley & Sons Ltd.
Yan, Yuxing. 2017a. *Python for Finance : Financial Modeling and Quantitative Analysis Explained*. Birmingham: Packt Publishing.
———. 2017b. *Python for Finance : Financial Modeling and Quantitative Analysis Explained*. Birmingham: Packt Publishing.
Yen Jerome, and Kin Keung Lai. 2015. “Emerging Financial Derivatives.” New York.
Z. Tong, Kevin. 2019. “A Recursive Pricing Method for Autocallables under Multivariate Subordination.” *Quantitative Finance and Economics* 3 (3): 440–55. https://doi.org/10.3934/QFE.2019.3.440.

Appendix

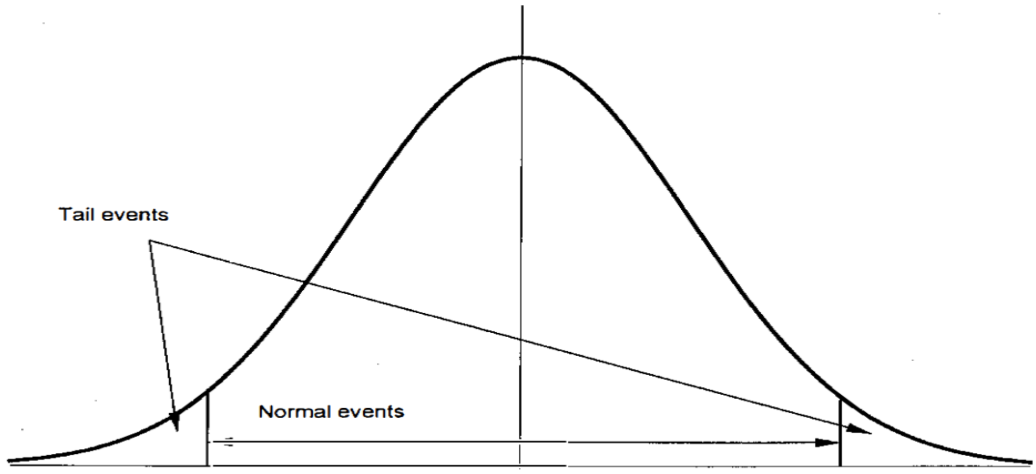
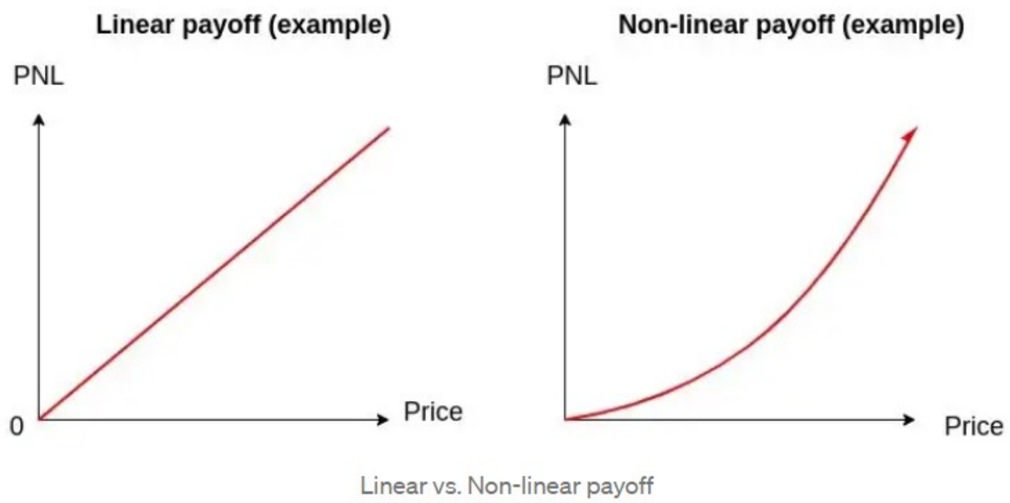


Figure 27.1 'Normal events' and 'tail events'.

Appendix 1: Value-at-Risk distribution showing possible tail events (Wilmott 1998, p. 338)

CI	Z
80%	1,282
85%	1,440
90%	1,645
95%	1,960
99%	2,576
99,5%	2,807
99,9%	3,291

Appendix 2: Overview of most common z-statistic for VaR calculation



Appendix 3: Linear / Non-linear VaR (Romano 2017)

Name	Symbol	Derivative	Measures	Definition
Delta	Δ	$\frac{\partial c}{\partial s}$	Equity Exposure	Measures how much an option's price is estimated to shift in response to a change of a one unit in the underlying security
Gamma	Γ	$\frac{\partial^2 c}{\partial^2 s}$	Payout Convexity	Measures the amount of change in Delta if the price of the underlying security changes by one unit
Theta	Θ	$\frac{\partial c}{\partial T}$	Time Decay	Measures the change in the option price induced by the decrease of 1 day of the remaining time to maturity
Vega	V	$\frac{\partial c}{\partial \sigma}$	Volatility Exposure	Measures how much an option's price will change in response to a 1% change in the volatility of the underlying securities
Rho	Ρ	$\frac{\partial c}{\partial r}$	Interest Rate Exposure	Measures how much the value of an option changes based on a 1% change in the interest rate

Appendix 4: Overview Greeks – In accordance with (Leoni 2014, pp. 85-97)

```python
def drift_calc(data, return_type='log'):
    if return_type == 'log':
        lr = log_returns(data)
    elif reutrn_type == 'simple':
        lr = simple_returns(data)
    u = lr.mean()
    var = lr.var()
    drift = u-(0.5*var)
    try:
        return drift.values
    except:
        return drift

drift = drift_calc(modified_data)

div = portfolio[self.id]['div']
# Drift adjusting if dividend paying (for Brownsche Motion)
if div > 0:
    drift = drift - div
```

Appendix 5: Drift calculation in Python

```python
covar = log_ret.cov() #covariance matrix.

chol = np.linalg.cholesky(covar) #create cholesky matrix from covariance matrix

uncorr_x = norm.ppf(np.random.rand(num_stocks, simulated_days)) #stocks, days

corr_x = np.dot(chol, uncorr_x)

corr_2 = np.zeros_like(corr_x) #Return an array of zeros with the same shape and type as a given array.
for i in range(num_stocks):
    corr_2[i] = np.exp(drift[i] + corr_x[i])
corr_2[0]

stock0 = pd.DataFrame()                                   #create new data frame
for s in range(len(ticks)):
    ret_reshape = corr_2[s]
    ret_reshape = ret_reshape.reshape(simulated_days) #Gives a new shape to an array without changing its data
    price_list = np.zeros_like(ret_reshape)
    price_list[0] = data.iloc[-1, s] #iloc = Purely integer-location based indexing for selection by position
    for t in range(1, simulated_days):
        price_list[t] = price_list[t-1]*ret_reshape[t]
```

Appendix 6: Cholesky decomposition in Python

```python
def get_vola(portfolio, volatility_file, stock_file):
    for a in range(2):
        for i in portfolio.keys():
            ticks = portfolio[i]['underlyings']
            today = "30-06-2022"
            today = pd.to_datetime(today)
            end = today
            #end = portfolio[i]['effective_date']
            vola = "VOLATILITY_30D"
            stock_vola = []

            if portfolio[i]["vol_type"] == "I": #calculation of implied volatility
                ids = portfolio[i]["underlyings"]
                for underlying in ids:
                    underlying_vola = volatility_file._get_value(vola, underlying)
                    stock_vola.append(underlying_vola)
                vol = (sum(stock_vola)/len(stock_vola))
                portfolio[i]["vol"] = vol
                #No Impl. vol.?, change vol_type to historical for second loop
                if  math.isnan(vol) == True:
                    portfolio[i]["vol_type"] = "H"
```

Appendix 7: Volatility calculation in Python (1/2)

```python
            else:
                portfolio[i]["vol_type"] = "H"
                def vola_data(tickers): #basically same funcion as used in cholesky
                    vol_data = pd.DataFrame() #create new data frame
                    for t in tickers: #loop through underlying tickers
                        vol_data[t] = stock_file[t].iloc[1:]
                    return(vol_data)

                data_ticks = vola_data(ticks)
                end_date = len(data_ticks.loc[:end]) #determine lenght of data frame for vola
                start_date = end_date - 30

                used_data = data_ticks.iloc[start_date:end_date]
                #print(used_data)

                def log_returns(data):
                    return (np.log(1+data.pct_change()))
                stdev = log_returns(used_data).std().values
                #monthly vola of the option = equal weighting vola of underlyings
                monthly_vol = sum(stdev)/len(stdev)
                vol = monthly_vol * sqrt(12) #annual vola
                portfolio[i]["vol"] = vol #append dictionary


get_vola(options_portfolio, file_vola, file_stocks)
```

Appendix 8: Volatility calculation in Python (2/2)

```python
def interpolation(rates, maturity_date):
    today = "30-06-2022"
    today = pd.to_datetime(today)
    today = today.to_pydatetime().date()

    maturity_day = maturity_date.day
    maturity_month = maturity_date.month
    maturity_year = maturity_date.year
    name = rates.columns[0]

    for a in range(len(rates)-1):
        rate_date = rates.index[a]
        prev_date = rates.index[a-1]
        next_date = rates.index[a+1]
        rate_day = rates.index[a].day
        rate_month = rates.index[a].month
        rate_year = rates.index[a].year

        if rate_date == maturity_date:
            r = rates._get_value(rate_date, name)
```

Appendix 9: Linear interpolation in Python to get discount rates for Option payoffs (1/2)

```python
        elif rate_month == maturity_month and rate_year == maturity_year:
            #structure: rate_day - maturity_date - next_rate
            if rate_day < maturity_day:
                #todays rate, shorter maturity
                r1 = rates._get_value(rate_date, name)
                #next rate, longer maturity
                r2 = rates._get_value(next_date, name)
                #day difference between rate_date and maturity_date
                t1 = abs(rate_date.to_pydatetime().date() - today)
                #day difference bewween next_date and maturity_date
                t2 = abs(next_date.to_pydatetime().date() - today)
                #day difference between rate_date and maturity_date
                tn = abs(today - maturity_date.to_pydatetime().date())



                r = r1 + (r2-r1)/((t2-t1).days)*((tn-t1).days)

            else:
                r1 = rates._get_value(prev_date, name)
                r2 = rates._get_value(rate_date, name)

                t1 = abs(prev_date.to_pydatetime().date() - today)
                t2 = abs(rate_date.to_pydatetime().date() - today)
                tn = abs(today - maturity_date.to_pydatetime().date())


                r = r1 + (r2-r1)/((t2-t1).days)*((tn-t1).days)
    return r
```

Appendix 10: Linear interpolation in Python to get discount rates for Option payoffs (2/2)

$$\Delta = \frac{S_t(\varepsilon) - S_t}{\varepsilon} \tag{18}$$

$$\Gamma = \frac{\Delta_t(\varepsilon) - \Delta_t}{\varepsilon} \tag{19}$$

Appendix 11: Equations used for Delta/Gamma calculation

```python
# Calculating per Option:
for i in options_portfolio.keys():
    print(i)
    # Determining the risk free rate by maturity matching the Swap Rate with the maturity of the option
    r = interpolation(swaps, options_portfolio[i]['maturity'])

    # Get infos
    ticks = options_portfolio[i]['underlyings']
    start = options_portfolio[i]['effective_date']
    S = options_portfolio[i]['spot']
    K = options_portfolio[i]['strike']

    today = "30-06-2022"
    today = datetime.strptime(today, '%d-%m-%Y').date()

    T = options_portfolio[i]['maturity'].to_pydatetime().date() - today
    T = T.days/365
    div = options_portfolio[i]['div']
    vol = options_portfolio[i]["vol"]

    price=0
    delta=0
    sym_delta = 0
    delta_2=0
    delta_3=0
    g=0
    var = 0
    z = 3.291
```

Appendix 12: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (1/15)

```python
# Call different classes for payoffs and delta
    if options_portfolio[i]["payoff_id"] == 2:

        altiplano = Altiplano(options_portfolio,i)
        payoffs = altiplano.payoff()
        # Discounting the payoff with the maturity matched risk free rate
        price = payoffs[0] * math.exp(-r*T)

        #Deltas
        pos_price = payoffs[1] * math.exp(-r*T)
        neg_price = payoffs[2] * math.exp(-r*T)
        delta = (pos_price - price) / (percentage_change)
        sym_delta = (pos_price - neg_price)/(2*percentage_change)

        #Second & Third Deltas
        pos_price2 = payoffs[3] * math.exp(-r*T)
        neg_price2 = payoffs[4] * math.exp(-r*T)
        delta_2 = (pos_price2 - pos_price) / (percentage_change)
        delta_3 = (neg_price2 - neg_price) / (percentage_change)

        #Gamma
        delta_dif = delta_2 - delta
        g = abs(delta_dif)/percentage_change
        #sym_g = ()

        #print(pos_price)
        #print(neg_price)

        # 1d VaR and 99,9% interval (Z-score=3.291), calculated on 1y
        var = (delta*3.291*np.sqrt(1/252)*vol - g/2*(3.291*np.sqrt(1/252)*vol)**2)*np.sqrt(252)
```

Appendix 13: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (2/15)

```python
    elif options_portfolio[i]["payoff_id"] == 4:
        auto = Autocall(options_portfolio, i)
        payoffs = auto.payoff()
        # Discounting the payoff with the maturity matched risk free rate
        price = payoffs[0] * math.exp(-r*T)

        #Deltas
        pos_price = payoffs[1] * math.exp(-r*T)
        neg_price = payoffs[2] * math.exp(-r*T)
        delta = (pos_price - price) / (percentage_change)
        sym_delta = (pos_price-neg_price)/(2*percentage_change)

        #Second & Third Deltas
        pos_price2 = payoffs[3] * math.exp(-r*T)
        neg_price2 = payoffs[4] * math.exp(-r*T)
        delta_2 = (pos_price2 - pos_price) / (percentage_change)
        delta_3 = (neg_price2 - neg_price2) / (percentage_change)

        #Gamma
        delta_dif = delta_2 - delta
        g = abs(delta_dif)/percentage_change
        #sym_g = ()

        #print(pos_price)
        #print(neg_price)

        # 1d VaR and 99,9% interval (Z-score=3.291), calculated on 1y
        var = (delta*3.291*np.sqrt(1/252)*vol - g/2*(3.291*np.sqrt(1/252)*vol)**2)*np.sqrt(252)
```

Appendix 14: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (3/15)

```python
elif options_portfolio[i]["payoff_id"] == 6:
    digi = Call_Digital(options_portfolio, i)
    payoffs = digi.payoff()
    # Discounting the payoff with the maturity matched risk free rate
    price = payoffs[0] * math.exp(-r*T)

    #Deltas
    pos_price = payoffs[1] * math.exp(-r*T)
    neg_price = payoffs[2] * math.exp(-r*T)
    delta = (pos_price - price) / (percentage_change)
    sym_delta = (pos_price-neg_price)/(2*percentage_change)

    #Second & Third Deltas
    pos_price2 = payoffs[3] * math.exp(-r*T)
    neg_price2 = payoffs[4] * math.exp(-r*T)
    delta_2 = (pos_price2 - pos_price) / (percentage_change)
    delta_3 = (neg_price2 - neg_price2) / (percentage_change)

    #Gamma
    delta_dif = delta_2 - delta
    g = abs(delta_dif)/percentage_change
    #sym_g = ()

    #print(pos_price)
    #print(neg_price)

    # 1d VaR and 99,9% interval (Z-score=3.291), calculated on 1y
    var = (delta*3.291*np.sqrt(1/252)*vol - g/2*(3.291*np.sqrt(1/252)*vol)**2)*np.sqrt(252)
```

Appendix 15: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (4/15)

```python
elif options_portfolio[i]["payoff_id"] == 11:
    indi = Indicap(options_portfolio,i)
    payoffs = indi.payoff()
    # Discounting the payoff with the maturity matched risk free rate
    price = payoffs[0] * math.exp(-r*T)

    #Deltas
    pos_price = payoffs[1] * math.exp(-r*T)
    neg_price = payoffs[2] * math.exp(-r*T)
    delta = (pos_price - price) / (percentage_change)
    sym_delta = (pos_price-neg_price)/(2*percentage_change)

    #Second & Third Deltas
    pos_price2 = payoffs[3] * math.exp(-r*T)
    neg_price2 = payoffs[4] * math.exp(-r*T)
    delta_2 = (pos_price2 - pos_price) / (percentage_change)
    delta_3 = (neg_price2 - neg_price2) / (percentage_change)

    #Gamma
    delta_dif = delta_2 - delta
    g = abs(delta_dif)/percentage_change
    #sym_g = ()

    #print(pos_price)
    #print(neg_price)

    # 1d VaR and 99,9% interval (Z-score=3.291), calculated on 1y
    var = (delta*3.291*np.sqrt(1/252)*vol - g/2*(3.291*np.sqrt(1/252)*vol)**2)*np.sqrt(252)
```

Appendix 16: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (5/15)

```python
    else:
        capprotect = Capital_Protected(options_portfolio,i)
        payoffs = capprotect.payoff()
        # Discounting the payoff with the maturity matched risk free rate
        price = payoffs[0] * math.exp(-r*T)

        #Deltas
        pos_price = payoffs[1] * math.exp(-r*T)
        neg_price = payoffs[2] * math.exp(-r*T)
        delta = (pos_price - price) / (percentage_change)
        sym_delta = (pos_price-neg_price)/(2*percentage_change)

        #Second & Third Deltas
        pos_price2 = payoffs[3] * math.exp(-r*T)
        neg_price2 = payoffs[4] * math.exp(-r*T)
        delta_2 = (pos_price2 - pos_price) / (percentage_change)
        delta_3 = (neg_price2 - neg_price2) / (percentage_change)

        #Gamma
        delta_dif = delta_2 - delta
        g = abs(delta_dif)/percentage_change
        #sym_g = ()

        #print(pos_price)
        #print(neg_price)

        # 1d VaR and 99,9% interval (Z-score=3.291), calculated on 1y
        var = (delta*3.291*np.sqrt(1/252)*vol - g/2*(3.291*np.sqrt(1/252)*vol)**2)*np.sqrt(252)
```

Appendix 17: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (6/15)

```python
    options_portfolio[i]['option_price'] = price
    options_portfolio[i]['delta'] = delta
    options_portfolio[i]["symmetric_delta"] = sym_delta
    options_portfolio[i]['delta2'] = delta_2
    options_portfolio[i]['delta3'] = delta_3
    options_portfolio[i]['g'] = g
    options_portfolio[i]["var"] = var
```

Appendix 18: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (7/15)

Per Option Type: Delta, Second Delta, Gamma and VaR

Weights

```
alti_notional = 0
auto_notional = 0
digi_notional = 0
indi_notional = 0
capprotect_notional = 0
sum_notional = 0
for key in options_portfolio.keys():

    if options_portfolio[key]["payoff_id"] == 4:
        notional = options_portfolio[key]["notional"]
        auto_notional += notional
        sum_notional += notional

    elif options_portfolio[key]["payoff_id"] == 4:
        notional = options_portfolio[key]["notional"]
        options_portfolio[key]["weight_type"] = notional/auto_notional
        options_portfolio[key]["weight_total"] = notional/sum_notional

        options_portfolio[key]["weighted_delta_type"] = options_portfolio[key]["weight_type"]*options_portfolio[key]["delta"]
        options_portfolio[key]["weighted_gamma_type"] = options_portfolio[key]["weight_type"]*options_portfolio[key]["g"]

        options_portfolio[key]["weighted_delta_total"] = options_portfolio[key]["weight_total"]*options_portfolio[key]["delta"]
        options_portfolio[key]["weighted_gamma_total"] = options_portfolio[key]["weight_total"]*options_portfolio[key]["g"]

#options_portfolio
options_type = {"Altiplano": {}, "Autocall": {}, "Call_Digital": {}, "Indicap": {}, "Capital_Protect": {}}
options_type["Altiplano"]["notional"] = alti_notional
options_type["Autocall"]["notional"] = auto_notional
options_type["Call_Digital"]["notional"] = digi_notional
options_type["Indicap"]["notional"] = indi_notional
options_type["Capital_Protect"]["notional"] = capprotect_notional
```

Appendix 19: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (8/15)

```
alti_delta = 0
auto_delta = 0
digi_delta = 0
indi_delta = 0
capprotect_delta = 0

alti_gamma = 0
auto_gamma = 0
digi_gamma = 0
indi_gamma = 0
capprotect_gamma = 0

for key in options_portfolio.keys():

    if options_portfolio[key]["payoff_id"] == 4:
        auto_delta += options_portfolio[key]["weighted_delta_type"]
        auto_gamma += options_portfolio[key]["weighted_gamma_type"]

options_type["Altiplano"]["delta"] = alti_delta
options_type["Autocall"]["delta"] = auto_delta
options_type["Call_Digital"]["delta"] = digi_delta
options_type["Indicap"]["delta"] = indi_delta
options_type["Capital_Protect"]["delta"] = capprotect_delta

options_type["Altiplano"]["gamma"] = alti_gamma
options_type["Autocall"]["gamma"] = auto_gamma
options_type["Call_Digital"]["gamma"] = digi_gamma
options_type["Indicap"]["gamma"] = indi_gamma
options_type["Capital_Protect"]["gamma"] = capprotect_gamma
```

Appendix 20: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (9/15)

```python
alti_underlyings = []
auto_underlyings = []
digi_underlyings = []
indi_underlyings = []
capprotect_underlyings = []

alti_underlyings_weights = []
auto_underlyings_weights = []
digi_underlyings_weights = []
indi_underlyings_weights = []
capprotect_underlyings_weights = []

today = "30-06-2022"
for key in options_portfolio.keys():

    if options_portfolio[key]["payoff_id"] == 4:
        underlyings_l = options_portfolio[key]["underlyings"]
        for underlyings in underlyings_l:
            single_weight = (1/5) * options_portfolio[key]["weight_type"] #stock weight in option * total option-type weight
            auto_underlyings_weights.append(single_weight)
            auto_underlyings.append(underlyings)

alti_underlyings_weights = np.array(alti_underlyings_weights)
auto_underlyings_weights = np.array(auto_underlyings_weights)
digi_underlyings_weights = np.array(digi_underlyings_weights)
indi_underlyings_weights = np.array(indi_underlyings_weights)
capprotect_underlyings_weights = np.array(capprotect_underlyings_weights)

options_type["Autocall"]["underlyings"] = auto_underlyings
options_type["Autocall"]["weights"] = auto_underlyings_weights

#options_type
```

Appendix 21: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (10/15)

For the whole Portfolio: Delta, Second Delta, Gamma and VaR

```python
total_delta = 0
total_gamma = 0

for key in options_portfolio.keys():

    total_delta += options_portfolio[key]["weighted_delta_total"]
    total_gamma += options_portfolio[key]["weighted_gamma_total"]

total_portfolio = {}
total_portfolio["notional"] = sum_notional
total_portfolio["delta"] = total_delta
total_portfolio["gamma"] = total_gamma
```

Appendix 22: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (11/15)

Weights of the underlyings

```python
list_underlyings = [] #== tickers
weights = []
today = "30-06-2022"
for key in options_portfolio.keys():
    underlyings_l = options_portfolio[key]["underlyings"]
    for underlyings in underlyings_l:
        single_weight = (1/5) * options_portfolio[key]["weight_total"] #stock weight in option * total option weight
        weights.append(single_weight)
        list_underlyings.append(underlyings)

weights = np.array(weights)
total_portfolio["underlyings"] = list_underlyings
total_portfolio["weights"] = weights
```

Appendix 23: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (12/15)

```python
weights = np.array(weights)

def vola_data(tickers): #basically same funcion as used in cholesky
    vol_data = pd.DataFrame(columns=tickers) #create new data frame
    for t in tickers: #loop through underlying tickers
        vol_data[t] = file_stocks[t].iloc[1:]
    return(vol_data)

tickerrs = list_underlyings


data_ticks = vola_data(tickerrs)

end_date = len(data_ticks.loc[:today]) #determine lenght of data frame for vola --> 30d volatility since effective date or today?
start_date = end_date - 30

used_data = data_ticks.iloc[start_date:end_date]


def log_returns(data):
    return (np.log(1+data.pct_change()))

returns = log_returns(used_data)
covar = returns.cov() * 12 #annualize monthly covariance

vol = np.sqrt(np.dot(weights.T, np.dot(covar, weights)))

total_portfolio["vol"] = vol
```

Appendix 24: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (13/15)

```python
# 1d VaR and 99,9% interval (Z-score=3.291), calculated on 1y
var = (total_delta*3.291*np.sqrt(1/252)*vol - total_gamma/2*(3.291*np.sqrt(1/252)*vol)**2)*np.sqrt(252)
total_portfolio["var"] = var
```

Appendix 25: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (14/15)

```python
elif key == "Autocall":
    notional = options_type[key]["notional"]
    tickerrs = options_type[key]["underlyings"]
    weights = options_type[key]["weights"]
    data_ticks = vola_data(tickerrs)
    end_date = len(data_ticks.loc[:today]) #determine lenght of data frame for vola --> 30d volatility since effective date or today?
    start_date = end_date - 30
    used_data = data_ticks.iloc[start_date:end_date]

    returns = log_returns(used_data)
    covar = returns.cov() * 12 #annualize monthly covariance

    vol = np.sqrt(np.dot(weights.T, np.dot(covar, weights)))
    options_type[key]["vol"] = vol

    delta = options_type[key]["delta"]
    g = options_type[key]["gamma"]
    # 1d VaR and 99,9% interval (Z-score=3.291), calculated on 1y
    var = (delta*3.291*np.sqrt(1/252)*vol - g/2*(3.291*np.sqrt(1/252)*vol)**2)*np.sqrt(252)
    options_type[key]["var"] = var
```

Appendix 26: Delta/ gamma & VaR Calculation for each option & overall portfolio in Python (15/15)

```python
class Call_Digital():

    def __init__(self,portfolio,id):
        self.id = id
        self.portfolio = portfolio


    def cholesky(self):

        def import_stock_data(tickers):
            data = pd.DataFrame()
            for t in tickers:
                data[t] = file_stocks[t].iloc[1:]
            return(data)

        def get_timeseries(data_frame):
            data = data_frame
            date = self.portfolio[self.id]["effective_date"]
            row_number = len(data.loc[:date])
            maturity_date = self.portfolio[self.id]["maturity"]

            date = date.strftime("%Y-%m-%d")
            maturity_date = maturity_date.strftime("%Y-%m-%d")

            delta = np.busday_count(date, maturity_date)

            #today = datetime.today()
            #today = today.strftime("%Y-%m-%d")
            today = '2022-06-12'

            #print(delta)
            if delta >= 504:
                calc = 504
```
Appendix 55: Call digital class in Python (1/8)

```
        else:
            delta_today = np.busday_count(date, today)
            #print(delta_today)
            days_used = 504 - delta_today
            minimum = 30
            calc = max(minimum, days_used)

        d1 = row_number - calc
        data = data.iloc[d1:row_number]
        return data


    ticks = self.portfolio[self.id]["underlyings"]
    num_stocks = len(ticks)
    data = import_stock_data(ticks)
    #print(f"full data: {data}")
    modified_data = get_timeseries(data)
    #print(f"modified data: {modified_data}")
    #print()

    def log_returns(data):
        return (np.log(1+data.pct_change()))

    log_return = log_returns(data)
```

Appendix 56: Call digital class in Python (2/8)

```
    def drift_calc(data, return_type='log'):
        if return_type == 'log':
            lr = log_returns(data)
        elif reutrn_type == 'simple':
            lr = simple_returns(data)
        u = lr.mean()
        var = lr.var()
        drift = u-(0.5*var)
        try:
            return drift.values
        except:
            return drift

    drift = drift_calc(modified_data)

    div = portfolio[self.id]['div']
    # Drift adjusting if dividend paying (for Brownsche Motion)
    if div > 0:
        drift = drift - div
    #print(drift)
    #print()

    log_ret = log_returns(modified_data)

    stdev = log_returns(modified_data).std().values
    covar = log_ret.cov() #covariance matrix.
    #print(covar)
    #print()
```

Appendix 57: Call digital class in Python (3/8)

```
    log_ret = log_returns(modified_data)

    stdev = log_returns(modified_data).std().values
    covar = log_ret.cov() #covariance matrix.
    #print(covar)
    #print()

    chol = np.linalg.cholesky(covar)
    #print(chol)
    #print()

    uncorr_x = norm.ppf(np.random.rand(num_stocks, simulated_days)) #stocks, days
    #print(uncorr_x)
    #print()

    corr_x = np.dot(chol, uncorr_x)
    #print(corr_x)
    #print()

    corr_2 = np.zeros_like(corr_x)
    for i in range(num_stocks):
        corr_2[i] = np.exp(drift[i] + corr_x[i])
    corr_2[0]

    stock0 = pd.DataFrame()
    for s in range(len(ticks)):
        ret_reshape = corr_2[s]
        ret_reshape = ret_reshape.reshape(simulated_days)
        price_list = np.zeros_like(ret_reshape)
        price_list[0] = data.iloc[-1, s]
        for t in range(1, simulated_days):
            price_list[t] = price_list[t-1]*ret_reshape[t]
```

Appendix 58: Call digital class in Python (4/8)

```
        x = pd.DataFrame(price_list).iloc[-1]
        x = pd.DataFrame(price_list)
        stock0[ticks[s]]=x.loc[:,0]


    last_date = data.last_valid_index()
    last_date = pd.Timestamp(last_date)
    dates = [last_date]
    a = 1
    while len(stock0) != len(dates):
        next_date = last_date + timedelta(days = a)
        if next_date.weekday() < 5:
            dates.append(next_date)
            a += 1
        else:
            a+=1

    stock0["Date"] = dates
    stock0 = stock0.set_index("Date")

    output_stocks_combined = data.append(stock0[1:])

    return output_stocks_combined
```

Appendix 59: Call digital class in Python (5/8)

```python
def call_digital(self,stock_prices,strikes):

    #notional = self.portfolio[self.id]["notional"]
    notional = 1
    cap = self.portfolio[self.id]["cap_cd"]
    floor = self.portfolio[self.id]["floor_cd"]

    j = 0
    for u in range(len(stock_prices)):
        if stock_prices[u] >= strikes[u]:
            j += 1

    if j == 5:
        payoff = notional * (1 + cap)
        return payoff
    else:
        payoff = notional * (1 + floor)
        return payoff

def payoff(self):
    ticks = self.portfolio[self.id]["underlyings"]
    maturity_date = self.portfolio[self.id]['maturity']
    payoffs = []
    payoffs_pos = [] #for Delta
    payoffs_pos2 = [] #for second Delta

    payoffs_neg = []
    payoffs_neg2 = []
```

Appendix 60: Call digital class in Python (6/8)

```python
    for i in range(number_simulations):
        output = self.cholesky()

        stock_prices = []

        stocks_pos = []
        stocks_pos2 = []

        stocks_neg = []
        stocks_neg2 = []

        for k in ticks:
            maturity_value = output._get_value(maturity_date, k)
            stock_prices.append(maturity_value)

        for k in range(len(stock_prices)):
            pos_change = stock_prices[k] * (1 + percentage_change)
            pos_change2 = pos_change * (1 + percentage_change)
            stocks_pos.append(pos_change)
            stocks_pos2.append(pos_change2)


            neg_change = stock_prices[k] * (1 - percentage_change)
            neg_change2 = neg_change * (1 - percentage_change)
            stocks_neg.append(neg_change)
            stocks_neg2.append(neg_change2)

        normal_payoff = self.call_digital(stock_prices, self.portfolio[self.id]["strikes"])
        payoffs.append(normal_payoff)
```

Appendix 61: Call digital class in Python (7/8)

```python
        #For Delta
        pos_payoff = self.call_digital(stocks_pos, self.portfolio[self.id]["strikes"])
        payoffs_pos.append(pos_payoff)

        neg_payoff = self.call_digital(stocks_neg, self.portfolio[self.id]["strikes"])
        payoffs_neg.append(neg_payoff)


        #Gamma-Calculation-Payoffs
        pos_payoff2 = self.call_digital(stocks_pos2, self.portfolio[self.id]["strikes"])
        payoffs_pos2.append(pos_payoff2)

        neg_payoff2 = self.call_digital(stocks_neg2, self.portfolio[self.id]["strikes"])
        payoffs_neg2.append(neg_payoff2)


    average_payoff = sum(payoffs)/len(payoffs)
    average_pos = sum(payoffs_pos)/len(payoffs_pos)
    average_neg = sum(payoffs_neg)/len(payoffs_neg)
    average_pos2 = sum(payoffs_pos2)/len(payoffs_pos2)
    average_neg2 = sum(payoffs_neg2)/len(payoffs_neg2)

    return [average_payoff, average_pos, average_neg, average_pos2, average_neg2]
```
Appendix 62: Call digital class in Python (8/8)