Application of Chebyshev approximation techniques applied to banking risk calculations

Grant Mashile
Student No.: 21830585

Supervisor: Prof. E. Maré

Dissertation submitted in fulfilment of the requirements for the degree
Magister Scientiae (Financial Engineering)
in the faculty of Natural and Agricultural Sciences.
University of Pretoria

February 2025

Abstract

Risk management in banking necessitates computationally intensive risk metric calculations, particularly through scenario analysis. This process is often time-consuming and costly. Numerical techniques, such as Chebyshev methods, can mitigate these burdens by enhancing calculation efficiency and reducing complexity.

This study evaluates the application of Chebyshev numerical techniques in risk calculations, specifically focusing on counterparty credit risk due to its relevance and increased importance post the 2007-08 financial crisis. Using adaptations from open-source libraries such as MOCAX Intelligence, trials were conducted on representative instruments and portfolios. Key metrics, including credit value adjustment and potential future exposures, were computed.

The findings reveal that Chebyshev techniques significantly reduce computation costs (1%-10% of current costs) and enhance calculation speeds while maintaining acceptable accuracy for risk management. Thus, Chebyshev numerical methods substantially improve the efficiency of risk metric calculations within the banking sector.

Declaration

I, the undersigned, declare that the dissertation, that I hereby submit for the degree *Magister Scientiae* (Financial Engineering) at the University of Pretoria, is my own work and has not previously been submitted by me fora degree at this or any other tertiary institution

_____

Name: Grant Mashile
Student No.: 21830585
Date: November 2024

Acknowledgement

I would like to extend my sincere gratitude to the Mocax Intelligence team Ignacio Ruiz and Mariano Zeron. I am humbled and honoured for the countless hours, correspondence and meetings we had over the last two years. Thank you for sharing your passion with me.

To my supervisor Prof. Eben Maré, your guidance, patience and motivation was what kept me believing. Thank you.

To my co-supervisor Dr V. van Appel would like to thank you very much.

To my personal support team of my colleagues, friends, and loved ones, in no particular order: Rajesh, Vasili, Andries, Kamo, Lebo, Toni and Mike. Thank you.

I dedicate this work to Kamohelo, Tshegofatjo, Katlego and Toro.

Contents

List of Figures

List of Tables

Index List

List of Terms

credit risk – refers to the possibility that a debtor may default on payments or contractual obligations. Often one thinks of lending risk where the amount owed is known in advance.

market risk – results from the short-term movement of market prices . It can be linear, stemming from exposure to changes in variables such as stock prices, interest rates, foreign exchange rates, commodity prices, and credit spreads.

counterparty credit risk – represents a combination of market risk, which defines the amount being owed, called exposure (this amount is unknown in advance is dependent on market factors) and credit risk, which defines the credit quality of the counterparty in transaction.

value at risk – is denoted by $\text{VaR}_{1-\alpha}(\tau)$, and when $\text{VaR}_{1-\alpha}(\tau) = X$ it says "we are $(1-\alpha)$ percent confident that we will not loose more than X units of currency in the next τ days", see *Section 3.1.2*.

derivatives (finance) – in finance, a derivative is a financial contract whose value is derived from the performance of an underlying asset, group of assets, or benchmark. These underlying assets can include: stocks, bonds, commodities, currencies, interest rates and market indices.

derivatives (maths) – in mathematics, a derivative is a fundamental concept that represents the rate at which a function changes at any given point as its input changes. The derivative of a function at a particular point is the slope of the line tangent to the graph of the function at that point.

over-the-counter – are financial contracts that are traded directly between two parties, outside formal exchanges.

expected tail loss – or expected shortfall (ES) measures the average loss that occurs in beyond a certain confidence level. This provides a more comprehensive view of risk by considering the severity of losses beyond the Value at Risk (VaR) threshold.

LIBOR and JIBAR – LIBOR is a benchmark interest rate at which major global banks lend to one another in the international interbank market for short-term loans. The LIBOR is used as a reference rate for various financial instruments, including loans, mortgages, and derivatives. The benchmark interest rate used in South Africa is called the JIBAR. Global efforts are underway to address weaknesses in current benchmark rates and transition to more robust alternatives, often referred to as near risk-free rates.

prime lending rate – is the interest rate at which commercial banks charge their most creditworthy customers. This rate is often used as a benchmark for various loan

products, including mortgages, personal loans, and credit cards. The prime rate is influenced by the central bank's monetary policy and is typically set a few percentage points above the central bank's repo rate (the rate charged to commercial banks).

real world and risk neutral – In the real world measures, the question is what is the actual probability of an event, and this is often estimated using historical data. In the risk neutral measure, we calculate the risk-neutral (or market-implied) probability from market observables (market prices or rates that imply risk factor levels).

Chapter 1

Introduction

1.1 Background

1.1.1 Risk Management in Banking

Traditionally, risk management in the banking sector has focused on *credit risk* as the main function of lending. Credit risk refers to the possibility that a debtor defaulting on payments or contractual obligations. This default risk varies by jurisdiction and should be evaluated throughout the lifetime of exposure. For example, with bonds, the bond holder must examine the maturity of the bond and the credit risk profile of the bond issuer. Consequently, this process involves assessing the probability of default (PD), expected exposure (EE) and the recovery value or loss given the default (LGD).

Banking services evolved from merely lending to include trading more sophisticated financial instruments, mainly found in merchant banking, such as trading currencies, stocks, bonds (corporate and sovereign debt), commodities, futures, and *derivatives*, which led to the advent of market risk.

Market risk results from short-term of market prices movement. It can be linear, stemming from exposure to changes in variables such as stock prices, interest rates, foreign exchange rates, commodity prices, and credit spreads. Alternatively, it may be non-linear, arising from exposure to market volatility, such as in a hedged position or derivatives. Over the past two decades, quantitative risk management techniques have been extensively applied to measure and manage market risk.

Market risk gained prominence with J.P. Morgan's introduction of RiskMetrics^{TM} and the *value at risk* (VaR) concept in 1992, published by Zangari [44].

This focus was prompted by significant market risk-related losses in the 1990s (e.g., Barings Bank) and subsequent amendments to the Basel I capital accord in 1995 [40], which permitted financial institutions to use proprietary mathematical models to calculate their market risk capital requirements. Market risk considerations have significantly driven the development of the value-at-risk approach to risk quantification.

VaR is defined as a probabilistic measure of the market risk for a portfolio with fixed holdings, considering its future market value as a random variable.

VaR metrics depend on:

- the distribution of the random variable, and
- the portfolio's current market value.

Various metrics such as the standard deviation of profit and loss (P&L) and the 95th-percentile of loss fall under VaR. A VaR measure assigns these metrics to portfolios based on models that link computation to interpretations. Early VaR measures evolved from portfolio theory and capital adequacy computations as documented by [16] and [17], respectively. Key debates about VaR include the assumptions of distribution, volatility estimation, and correlation measurement. [3] highlight that VaR, understood as a percentile, is not a coherent risk measure. They introduced requirements for coherent risk measures, leading to *expected tail loss* (ETL), also known as conditional value-at-risk (CVaR) or expected shortfall (ES), which satisfies properties such as monotonicity, sub-additivity, homogeneity, and translational invariance. For counterparty credit risk, we have a VaR like risk metric called potential future exposure (PFE).

Counterparty Credit Risk (CCR)[1] represents a combination of market risk, which defines the exposure, and credit risk, which defines the credit quality of the counterparty in transactions (often derivatives transactions)[19]. Market risk can be mitigated by entering into an offsetting contract. However, unless this is done with the same counterparty as the original position(s), counterparty risk is generated. Beyond default, deterioration in credit quality can cause mark-to-market losses due to increased future default probabilities, which generates counterparty risk.

Counterparty risk, traditionally seen as credit risk between *over the counter* (OTC) derivative counterparties, has become crucial since the global credit crisis[2] (GCC) highlighted its significance. The main metric of counterparty risk is credit valuation adjustment (CVA), which can be defined as the difference between the fair value of a derivative without counterparty default risk and the fair value of the same derivative considering the potential default of the counterparty. There are various other valuation adjustments collectively known as X-valuation adjustments (XVAs), where the "X" is place holder[3].

Historically, institutions have limited this risk by dealing only with sound counterparties, but the crisis revealed that even highly rated entities could pose substantial risks. Many banks accounted for counterparty credit risk well before the global financial crisis by taking "reserves" against potential future losses. Based on historical data, these reserves remained relatively stable . This method treats CVA as a statistical estimate of expected counterparty risk losses, similar to a loan loss reserve, and is not marked-to-market but estimated actuarially.

Regulatory pressures have since attracted increased attention to counterparty risk across all financial institutions. In particular, since the 2008 financial crisis , capitalization for CVA has become increasingly important as a risk management tool for financial institutions. This is because most losses emanate from CVA (deterioration in credit quality or widening of spreads) rather than from actual defaults.

[1] Or simply *counterparty risk*
[2] also refereed to as the global financial crisis
[3] The 'X' can represent various terms such as debit, collateral, funding, margin, capital, among others. See *Section 3.3*

CVA is generally charged with uncollateralized or partially collateralized trades with a counterparty at the trade or portfolio level to mitigate the risk of default and is marked-to-market regulatory. However, banks are also required to hold capital to cater to CVA losses .

The Basel Committee on Banking Supervision (BCBS) defines CVA risk as the risk of losses arising from changing CVA values in response to changes in counterparty credit spreads and market risk factors that drive the prices of derivative transactions and securities financing transactions (SFTs) [41]. It is a requirement that all banks involved in covered transactions calculate their capital requirements for CVA risk in both the banking and trading books.

1.1.2 Generic risk engine process

Figure *1.1* the typical process involved in modern risk calculations in which the Chebyshev interpolants can fit. The steps are:

1. *Input collection*: This entails the collection of trade and portfolio information: specific trade information, trades, positions or units, counterparty details, valuation models model selection.
2. *Engine preparation*: Curve mapping and selection of stochastic models for underlying risk factors. These could include stock prices, commodity prices, currency exchange rates, credit quality, interest rates and various related sensitivities and parameters.
3. *Scenario Generation*: Risk factors scenarios are generated or loaded for various potential outcomes based on the driving stochastic processes or appropriately representative time-series.
4. *Pricing*: For each node of the scenario matrix, the portfolio or trades within must be priced given the prevailing inputs and parameters at that node. This implies that the average pricing time is multiplied by the number of scenarios considered.
5. *Special modelling* : Collation of valuations or exposures such as positive, negative, expectations, discounting, etc.
6. *Risk metric computation*: Various risk metrics can then be computed, such as CVA, XVA , PFE, VaR, ES (simulation based approaches), initial margin models (IMM)[4] and capital charge .

In general the pricing step tends to be a bottleneck in the process flow. This is because pricing models may take a long time to run or the models my reside elsewhere, say the front office system, or may not be readily accessible to the middle office or risk system owner. Addressing this bottleneck presents an opportunity to improve the overall efficiency of risk engines.

[4]such as ISDA's Standardised IMM

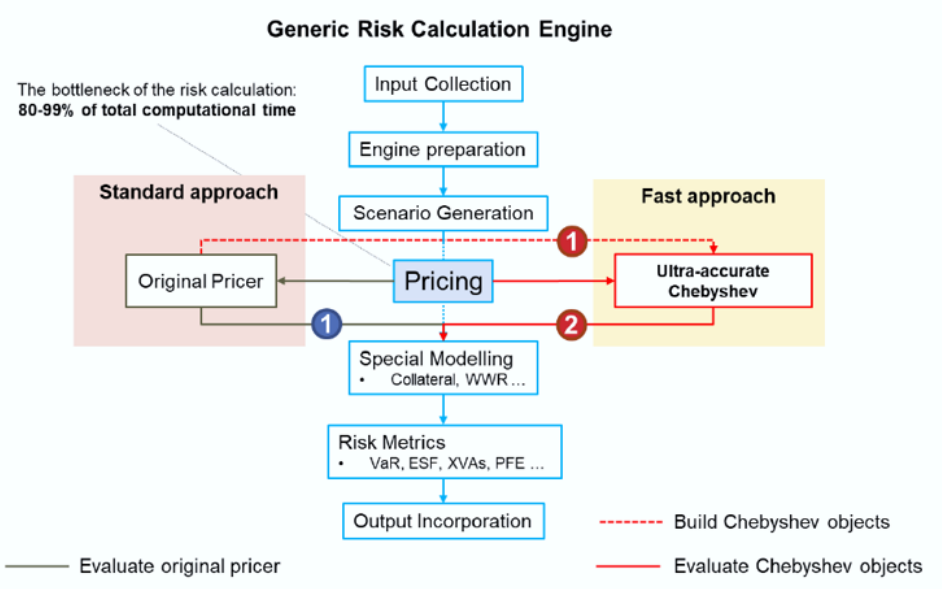


Figure 1.1: The figure display the process flow of generic risk calculation engine [35]

1.2 Objectives

The main idea or appeal of Chebyshev techniques is to use them to build proxy functions to be deployed in computationally intensive risk calculations to exploit their exceptional run times and therefore reduce the total run time without compromising too much on the accuracy of the risk metric being calculated.

This optimization scheme which is the object of our study is underpinned by the following principles:

1. the original pricing function in the building phase should be called considerably fewer times than that in the first principles approach;
2. the precision of the interpolating object should be very good; and
3. the computational cost of evaluating the interpolating object should be low.

The objective of this study is to understand and verify the use of Chebyshev techniques in fulfilling the requirements of the optimization scheme as described above.

1.3 Document structure

The rest of the dissertation is structured as follows:

Chapter 2 Literature Review: This chapter reviews the existing literature on banking risk calculations and the application of Chebyshev techniques. It covers various risk calculation methods, including full valuation VaRs, historical simulation methods, Monte Carlo (MC) simulations, and CVA theory and quantification. This chapter also discusses the motivation for focusing on Chebyshev techniques.

Chapter 4 Methodology : This chapter outlines the theoretical framework of Chebyshev techniques and their applications. It covers approximation theory, multivariate extension, and the use of Chebyshev techniques in composition techniques and tensors in the tensor-train format for high dimensional applications.

Chapter 5 Results: This chapter presents the results of the study, including proxy function tests in the low and high dimensions, pricing models, and counterparty credit risk trials. This demonstrates the effectiveness of Chebyshev techniques in improving the inefficiencies of risk metric calculations.

Chapter 6 Discussion: This chapter provides an analysis and interpretation of the results of this study. It discusses the implications of the findings and how they contribute to the banking risk management field.

Chapter 7 Bibliography: This chapter provides a comprehensive list of all references and sources cited throughout the document.

Appendix A Additional Results : This section presents supplementary data and graphs that support the findings discussed in the Results section. This includes detailed visual representations of proxy function tests, pricing models, and counterparty credit risk trials.

Chapter 2

Literature Review

As a result of the evolving and intricate world of trading, evaluation of derivatives and burgeoning regulatory requirements for prudent risk management, financial institutions have had to adapt to these changing outcomes in financial markets, which require continuous development of their risk management processes. Consequently, the development of efficient and precise computational methods for parametric models is necessary. Daily operations, ranging from pricing to risk assessment , require real-time calculations of numerous financial quantities. Since the advent of the pioneering models by Black and Scholes [11] and Merton [5], the sophistication and complexity of these models has increased significantly, incorporating stochastic volatility such as Levy models, stochastic interest rate modelling such as those covered by short-rate models and extensions thereof anticipated by the Heath-Jarrow and Morton (HJM) *[6]*, and other stochastic processes to reflect market data more accurately.

Beginning in the 1990s, asset models evolved significantly, with notable contributions from [25], [9], and others. The 2007–2009 financial crisis further spurred the development of complex models that integrate additional risk factors to capture market dynamics more effectively. The effectiveness of any pricing model depends on its numerical implementation and ability to capture relevant market aspects. This has also inevitably led practitioners to move away from the analytic (closed-form solution) type to more valuation models and parametric numeric models that require some form of simulation based implementation, such as Monte Carlo, finite differences and lattice models (binomial, tri-nomial trees, etc).

In the literature, complexity reduction for parametric problems often leverages Fourier techniques, as evidenced by influential works from [42], [46], and others. These methods, particularly the fast Fourier transform (FFT), have proven highly effective for option pricing. Additionally, reduced basis methods for solving parametrized partial differential equations have shown promise, as demonstrated by [48] and others. These techniques offer a substantial complexity reduction by focusing on parameter dependence and exploiting the functional structure of the pricing method. Given the diverse array of models and option types that financial institutions handle, exploring a generic complexity reduction method independent of specific pricing techniques is advantageous. Chebyshev interpolation, widely used in physics, engineering, and other applied maths fields, was largely untapped in quantitative finance until recently (approximately 10 years ago) and has since shown great potential. Early researchers in this area include Pistorious and Stolte [33] and Pachon [45] who aim to enhance computational efficiency and accuracy in option pricing. While Pistorious and Stolte focus on rational approximations and time-changed models, Pachon emphasizes the Chebyshev series and its applications to arbitrary option payoffs.

This was taken forward and in the direction explored in detail in this study by *Gaß et al [31]*. The authors observed that for a large set of applications in which most pricing functions in finance are highly regular, admitting sensitivities of high order or even analytic, and that the domain of interest can be restricted to a hyper-rectangular, Chebyshev interpolation is a promising choice, its convergence is sub-exponential for multivariate analytic functions, its implementation is numerically stable, and the coefficients are simply given by a linear transformation of the function values at the nodal points. In *Section 4* we discuss the theory and methodology of applying the Chebyshev techniques.

The Basel Committee's mandates on credit exposure calculations have necessitated overhauls in bank pricing systems. This happened in earnest following the financial crises highlighing the importance of counterparty credit risk in global markets. Calculating risk indicators such as expected exposure (EE), general valuation adjustments (or XVAs), or potential future exposures, requires significant computational power, often requiring large-scale hypothetical scenarios. This makes the valuation of extensive portfolios labour-intensive and time-consuming, especially for portfolios with numerous trades influenced by various risk factors.

For large financial institutionssuch as banks and collective investment funds (such as pension funds), most of the products are linear and non-exotic. This is particularly true because illiquid derivatives are either penalized or outright forbidden by regulators and thus become capital intensive. That said, a portfolio of linear products maybe straightforward to value, but because of the large volumes of trades (often exceeding hundreds of thousands), the complete risk metric computation may take multiple hours to run. In particular a portfolio involves many long-dated derivatives (often swaps) with daily compounding or average rates.

Despite significant advancements, XVA evaluation remains a computational bottleneck. Traditional methods, such as Monte Carlo simulations offer avenues for speed enhancement, yet they often compromise the quality and stability of the result. Moreover, heavy hardware investments are frequently required to meet the computational demands.

The structure of the techniques presented in [31] to [23] and are the subject of our study, can be extended to other tensorized interpolation techniques that are similar to the collocation [1] method with Lagrange interpolations for arbitrage-free option pricing discussed in [26]. The latter authors developed a numerical scheme by employing Stochastic Collocation (SC) in [15]. The scheme starts with a distribution function of interest (a pricing function or risk metric), which is then expanded as a polynomial in terms of a random variable that is inexpensive to sample from at given collocation points, and interpolation occurs between these points. Stochastic collocation points have a specific meaning; that is, they represent critical features of the probability distribution of interest. The SC method enables us to efficiently generate samples from a complex distribution using interpolation .

Later, the author applied SC methods to approximating distribution functions in [1] and showed an alternative to Chebyshev interpolation, although in a similar numerical scheme considered by *Gaß* [31]. In that study, the authors reviewed a novel method that emphasizes efficient exposure computation and can be implemented via parallelization for GPU computation. Using the stochastic collocation method, they developed a fast and accurate numerical scheme for approximating distribution functions. While one dimensional (1D) risk factor cases require minimal collocation points, high-dimensional systems face the curse of dimensionality. The Smolyak sparse grid approach of [47] mitigates this issue, enabling high-dimensional integration and interpolation without succumbing to dimensionality curses. The complexity of financial issues stems from a multitude of risk factors that

[1] Generally, the term collocation denotes techniques that estimate deterministic or stochastic variables by finding a linear predictor from a finite set of observations [1]

influence asset prices. Consequently, the development of efficient computational methods to solve high-dimensional financial problems is crucial. Techniques such as quasi-Monte Carlo and multilevel Monte Carlo methods have shown promise, as have sparse grid techniques and partial differential equation (PDE) methods for multivariate problems.

Practical aspects of fast portfolio evaluation using interpolation for risk calculations, such as value at risk, credit valuation adjustment and various other XVAs, are covered in [35], and in [24] the application of Chebyshev interpolation for exposure calculation for the 1D case of Bermudan interest rate swaptions has been presented. Moreover, alternatives to the sparse grid approach exist for dealing with multi-dimensionality problems in the context of derivatives pricing . Recently, low-rank tensor approximation has gained increasing interest in applications to derivative pricing, see [36] and [23].

2.1 Motivation for focusing on Chebyshev techniques

In this study, we investigate the interpolation of financial quantities in the parameter space using Chebyshev polynomials. [31] noted that their empirical observation is that parameters in quantitative finance and risk management often range within bounded intervals, and Chebyshev polynomial interpolation is well known for its excellent numerical properties in approximating analytic functions on bounded intervals. In the following excerpt the favourable properties of Chebyshev interpolation are summarized understanding and verifying these properties will be one of the objectives of this study. We will unpack them in the theory section and verify them in the experimental trials.

Note 2.1.1. *The following key properties of Chebyshev interpolation are of particular interest [31]:*

- *For univariate functions that are several times differentiable, the method converges polynomially, and converges for univariate analytic functions.*
- *The method can be implemented in a numerically stable way.*
- *The interpolation nodes are explicitly available; thus, the coefficients are explicitly given as a linear transformation of the function values at the nodal points. This makes the implementation straightforward, a feature that is valuable both for the application of the method in the complex information technology (IT) infrastructures of the financial industry and for further development of the method.*
- *The* derivatives *are trivial for interpolation and are known to converge at a rate that is determined by the regularity of the function that is interpolated.*
- *Chebyshev interpolation can be easily concatenated, yielding the Chebyshev-spline approximation, which is extremely appealing when the function, for instance, exhibits discontinuities.*
- *Chebyshev interpolation can be highly efficiently extended to higher dimensionality, for instance, by low-rank tensor and sparse grid techniques (as shown in [23]).*

Chapter 3

Banking Risk Calculations

In this subsection we provide a background on the main banking risk tools that are computationally intensive and the staple of market risk and counterparty credit risk.

We will discuss the following risk metrics:

- Full valuation Value at Risk
- Credit Valuation Adjustment
- General Valuation Adjustment and Potential Future Exposure

We will then end the chapter by reviewing the Hull-White 1-factor interest rate model that we employ in our trials.

3.1 Full Valuation VaRs

3.1.1 The VaR Measure

The VaR is a statistical measure of the downside risk to which a portfolio is exposed. It measures total portfolio risk by, considering portfolio diversification and leverage.

In the following sections, we formally define the VaR measure and provide a detailed description of VaR methodology.

3.1.2 Definition of VaR

We first consider the intuitive definition given by [21]. When applying the VaR measure, we are primarily interested in making the following statement.

We are $(1-\alpha)$ *percent confident that we will not loose more than* VaR *units of currency in the next* τ *day.*

The variable X represents the portfolio's VaR. This is a function of the following two parameters:

1. the holding period or time horizon (τ days) and

2. the confidence level $(1-\alpha)\%$.

For clarity, we use the symbol $VaR_{1-\alpha}(\tau)$ instead of X. For the purposes of our VaR methodology, we adopt the following definition given by [12].

Definition 3.1.1. *Let $V(t)$ represent the value of the portfolio at time t. By determining VaR, we say that the probability that the loss of the portfolio over a τ-day holding period exceeds $VaR_{1-\alpha}(\tau)$ is $\alpha\%$*

$$P(\Delta V < VaR_{1-\alpha}(\tau)) = \alpha \tag{3.1}$$

where, $\Delta V = V(t+\tau) - V(t)$ represents the change in the value of the portfolio.

In fact, the VaR corresponds to the α percentile of the distribution of changes in the value of the portfolio over a holding period of τ days; the VaR concept is illustrated in Figure *3.1*.

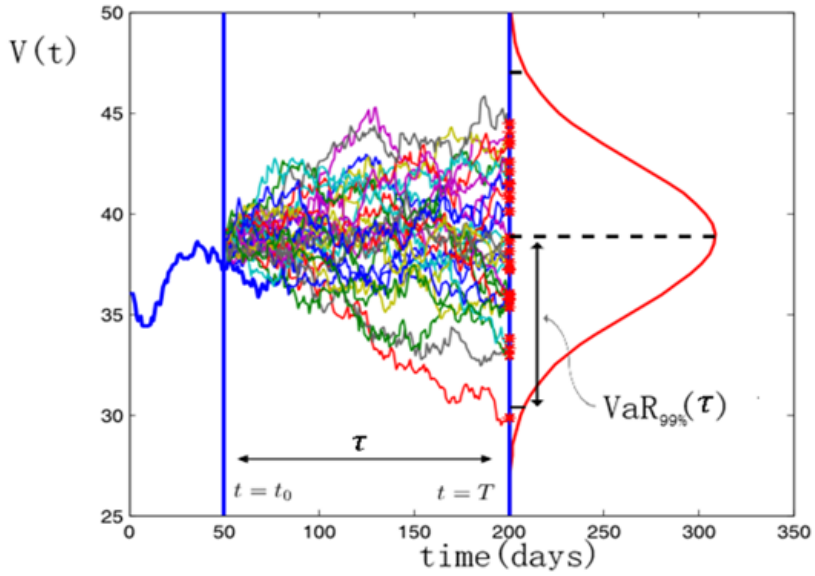


Figure 3.1: Illustration of VaR

3.1.3 VaR Methods

The family of VaR methods can be classified into local valuation and full valuation methods, as shown in Figure *3.2*. Local valuation methods (also known as analytical or parametric methods) are based on the valuation of instruments at a particular point, along with first- and second-order partial derivatives. Full valuation methods are based on repricing instruments over a broad range of values for the associated risk factors. The methodology for the historical simulation (HS) and Monte Carlo methods are similar. The former relies on the time series of the associated risk factors to create scenarios for repricing the instruments, while the latter, uses stochastic simulation.

Financial institutions use simulation methods, which can be particularly useful when the relationship between the value of the portfolio and market risk factors is non-linear and non-monotonic, such as in portfolios that include options. Simulation methods calculate VaR by making assumptions about the (joint) distribution of underlying risk factors or benchmark asset returns, extracting a sample from the joint distribution, and re-evaluating the portfolio of assets, whose VaR should be measured according to each set of risk factor values. By sorting the simulated portfolio values in descending order, it becomes possible to identify the desired percentile of the distribution of portfolio values and thereby determine the VaR.

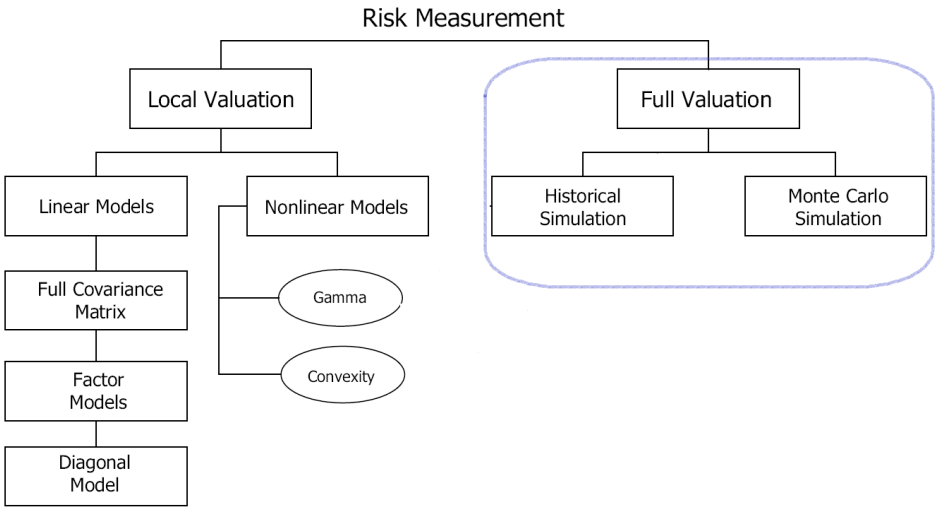


Figure 3.2: VaR Methods - adapted from [43]

3.1.4 Historical Simulation Method

The basic assumption of the HS method is that the joint return distribution of asset returns can be reasonably approximated by the past joint return distribution. Consequently, is method has clear advantages and disadvantages. Advantages include:

1. its ease of implementation and communication,
2. the absence of explicit assumptions on the joint distribution of benchmark asset/risk factor returns, and
3. the ability to capture the risk of portfolios whose value is non-monotonic relative to the risk factors.

HS may consider effects such as correlation breakdown in cases of market shocks that cannot be considered in a variance–covariance approach (which could be difficult and time consuming to model in a Monte Carlo simulation).

Methodology description

Consider a general portfolio $\mathcal{P}$ that comprises of l positions $f_1(t), f_1(t), \ldots, f_l(t)$. In the simplest case, these positions can be directly associated with a fundamental risk factor, or

they can be associated with a financial derivative which is a complex function of m risk factors $x_j(t)$:

$$f_i(t) = \begin{cases} x_j(t) & 1 \leq i \leq l \\ f_i(x_1(t), x_2(t), \ldots x_m(t)) & 1 \leq j \leq m \end{cases} \tag{3.2}$$

The value of portfolio $V(t)$ is determined by holding α_i in each position $f_i(t)$, $1 \leq i \leq l$

$$V(t) = \alpha_1 f_1(t) + \alpha_2 f_1(t) + \ldots + \alpha_l f_l(t) \tag{3.3}$$

Assume that the time series information associated with each risk factor $x_j(t)$, $1 \leq j \leq m$ has the following chronological order

$$\begin{bmatrix} x_j(t_1) \\ x_j(t_2) \\ \vdots \\ x_j(t_n) \end{bmatrix}^{historic}_{current} \quad \ldots \quad 1 \leq j \leq m \tag{3.4}$$

where n represents the number of elements in the time-series. This information is used to generate $n-1$ historical scenarios using either the ratio method or the difference method, depending on the configuration of that risk factor.

$$dr_j(t_k) = \begin{cases} \left(\frac{x_j(t_k)}{x_j(t_{k-1})}\right) & 1 \leq j \leq m \\ x_j(t_k) - x_j(t_{k-1}) & 2 \leq k \leq n. \end{cases} \tag{3.5}$$

The base scenarios for these methods are defined as follows

$$dr_j(t_1) = \begin{cases} 1 \\ 0 \end{cases} \quad 1 \leq j \leq m. \tag{3.6}$$

The $n-1$ scenarios can now be used to determine possible movements in the risk factors given that their current values $x_j(t_n)$ are known

$$r_{j,k}(t_{n+1}) = \begin{cases} x_j(t_n) dx_j(t_k) & 1 \leq j \leq m \\ x_j(t_n) + dx_j(t_k) & 2 \leq k \leq n. \end{cases} \tag{3.7}$$

These movements are then used to recalculate the positions

$$f_{i,k}(t_{n+1}) = \begin{cases} x_{j,k}(t_{n+1}) & 1 \leq i \leq m, \\ & 1 \leq j \leq m, \\ f_i(x_{1,k}(t_{n+1}), x_{2,k}(t_{n+1}), \ldots, x_{m,k}(t_{n+1})) & 2 \leq k \leq n, \end{cases} \tag{3.8}$$

which are used to revalue the portfolio

$$V_k(t_{n+1}) = \alpha_1 f_{1,k}(t_{n+1}) + \alpha_2 f_{2,k}(t_{n+1}) + \ldots + \alpha_l f_{l,k}(t_{n+1}) \qquad 2 \leq k \leq n. \tag{3.9}$$

The change in the portfolio value is the difference between these valuations and the current value

$$\triangle V_k = V_k(t_{n+1}) - V(t_n) \quad \dots \quad 2 \leq k \leq n. \tag{3.10}$$

Note that in the case of the base scenarios we have

$$\triangle V_1 = 0. \tag{3.11}$$

The next step involves formulating the vector $\triangle V$

$$\triangle V = \begin{bmatrix} \triangle V_2 \\ \triangle V_3 \\ \vdots \\ \triangle V_n \end{bmatrix}. \tag{3.12}$$

These $n-1$ deviations represent either a positive or a negative movement in the value of the portfolio, which must be sorted in ascending order:

$$\triangle V \uparrow = sort(\triangle V, ascending). \tag{3.13}$$

In the final step, integer I is determined, which indexes the 1% percentile of $\triangle V \uparrow$. The 1-day VaR is simply

$$I = round\left(\frac{n-1}{100}\right) \tag{3.14}$$

$$VaR_{1-\alpha}(1) = \triangle V \uparrow (I). \tag{3.15}$$

3.1.5 Monte Carlo Simulations

While Historical Simulation assumes that historical return distributions can represent future return distributions, Monte Carlo simulations require that the risk manager formally models both the marginal distributions of each asset's returns and the dependence structure among the returns of different assets. After defining the shape and parameters of the joint distribution of benchmark assets, a random vector of values is extracted from the joint distribution and the portfolio value is re-evaluated accordingly. This approach derives a simulated return distribution for the entire portfolio, enabling the calculation of the quantile (e.g., 1%) to identify the percentage VaR at the desired confidence level.

Simulation Value at Risk (VaR) is a widely employed method for quantifying the potential loss in value of a portfolio over a specified time horizon, given normal market conditions, at a certain confidence level. Unlike parametric VaR, which relies on the assumption of a normal distribution of returns, simulation VaR, particularly the Monte Carlo simulation, does not require such assumptions and can accommodate a wide range of distributions and risk factors. This flexibility renders simulation VaR an invaluable tool for financial institutions to assess and manage market risk.

The process of estimating VaR using Monte Carlo simulation involves several steps. First, we identify and model the risk factors affecting the portfolio . Subsequently, a large number

of random scenarios was generated based on the statistical properties of these risk factors . For each scenario, the portfolio was revalued at the end of the specified time horizon to compute the potential loss or gain. Figure *3.3* provides a simple conceptual summary of the full valuation of VaR methodologies.

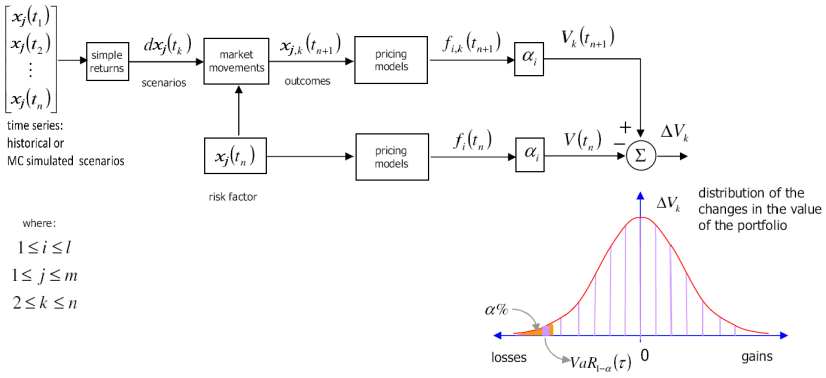


Figure 3.3: Full valuation VaR methodology process

Remark 3.1.1. Draw back of full valuation VaR
Computational costs in Monte Carlo and historical simulations arise primarily from the necessity to re-evaluate the portfolio multiple times for each simulated scenario. For portfolios with a large number of risk factors and instruments, this can entail evaluating the portfolio hundreds of thousands to millions of times. This extensive revaluation process demands significant computational resources and time, presenting a major challenge for real-time risk management applications.

3.2 CVA Primer

3.2.1 CVA Theory

The starting point for understanding CVA is to understand the notion of counterparty credit risk (or simply counterparty risk). In this section, we will closely follow [19] and [50]. Counterparty risk is the risk that the entity with whom one has entered into a financial contract will fail to fulfil their side of the contract (i.e., pay when they owe). Unlike traditional credit risk that arises from lending (so called lending risk), where the notional (or lent) amount is known upfront and the only one party takes the lending risk (i.e. the lender). Two important differences from counterparty credit risk are that the value of the contract is uncertain (and can be significantly large) and may be positive or negative for either counterparty (the risk is bilateral).

The CVA "charge" should be calculated in a sophisticated way to account for all aspects that define the CVA:

- the default probability of the counterparty,
- the default probability of the institution,
- the transaction in question,

- netting of existing transactions with the same counterparty,
- collateralization, and
- hedging aspects.

In pricing derivatives, after the 2008 global credit crisis, a derivative valuation has two components: the risky price of a derivative can be expressed as the risk-free price (the price assuming no counterparty risk) less a component to correct for counterparty risk[1].

Therefore, we have

Proposition 3.2.1. Derivative valuation principle

Derivative valuation with risk = Risk-free valuation − Risk Adjustments.

In the following, we only consider unilateral-CVA (UCVA), which means we only consider the default risk of the counterparty and not that of the issuer bank.

3.2.2 CVA Quantification

We consider a contract entered between a bank and a counterparty (C) that starts at time t and matures at time T. Let R_t denote the recovery at time t, faced by the bank in the contract; $\mathbb{1}_\tau$ is an indicator function representing the outcome of a default by C occurring at time τ and let DF_t denote the stochastic discount term defined in terms of the money market account $M(t)$

$$DF(0,t) = \frac{1}{M(t)}.$$

Denote by $V^D(t,s)$ be the discounted payoff of the contract between time t and s (where s is some general time $s \in (\, t, T\,]$), and let $V(t,s)$ be the discounted payoff of the contract without counterparty risk (risk-free).

Our risky valuation needs to capture all possible default outcomes and consider them .

1. If the default of a counterparty occurs after the final payment of derivative V_T, the value at time t is simply
$$\mathbb{1}_{\tau>T} \cdot V(t;T).$$
2. If the default occurs before the maturity time $\tau < T$:
   (a) we receive/pay all payments until the default time:
   $$\mathbb{1}_{\tau\leq T} \cdot V(t;T).$$
   (b) Depending on the counterparty, we may be able to recover some of the future payments. Assuming the recovery fraction to be R, the value yields:
   $$\mathbb{1}_{\tau\leq T} \cdot R \cdot \max(V(t;T),0),$$

[1] which we later define as CVA

3. Alternatively, if we owe the money to the counterparty that has defaulted, we cannot keep it, but we need to pay it back:

$$\mathbb{1}_{\tau \leq T} \cdot \min(V(t;T), 0).$$

When we combine all components, we get the following expression for the "risky" derivative valuation

$$\begin{aligned} V^D(t,T) &= \mathbb{E}^{\mathbb{Q}}\left[\, \mathbb{1}_{\tau>T} \cdot V(t,T) + \mathbb{1}_{\tau \leq T} \cdot V(t,\tau) + DF(t,\tau) \cdot \mathbb{1}_{\tau \leq T} \cdot R \cdot \max(V(t;T),0) + \right. \\ &\quad \left. DF(t,\tau) \cdot \mathbb{1}_{\tau \leq T} \cdot R \cdot \min(V(t;T),0) | \mathcal{F}(t) \,\right]. \end{aligned}$$

Using the equality $x = \max(x,0) + \min(x,0)$, we can simplify the expression as follows:

$$\begin{aligned} V^D(t,T) &= \mathbb{E}^{\mathbb{Q}}\left[\, \mathbb{1}_{\tau>T} \cdot V(t,T) + \mathbb{1}_{\tau \leq T} \cdot V(t,\tau) + DF(t,\tau) \cdot \mathbb{1}_{\tau \leq T} \cdot V(\tau,T) \right. \\ &\quad \left. + DF(t,\tau) \cdot \mathbb{1}_{\tau \leq T} \cdot (R-1) \cdot \max(V(t;T),0) | \mathcal{F}(t) \,\right]. \end{aligned}$$

We can further simplify the first three terms and reason that they combine to give the value of the risk-free/default free part of the equation as follows:

$$\begin{aligned} \mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{\tau>T} \cdot V(t,T)\right. &+ \left. \mathbb{1}_{\tau \leq T} \cdot V(t,\tau) + DF(t,\tau) \cdot \mathbb{1}_{\tau \leq T} \cdot V(\tau,T) | \mathcal{F}(t)\right] \\ &= \mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{\tau>T} \cdot V(t,T) + \mathbb{1}_{\tau \leq T} \cdot V(t,T) | \mathcal{F}(t)\right] \\ &= V(t,T) \cdot \mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{\tau>T} + \mathbb{1}_{\tau \leq T} | \mathcal{F}(t)\right] \\ &= V(t,T). \end{aligned}$$

We make the following assumptions to make the modelling manageable:

- all analysis is done from the perspective of the institution B,
- the bank is default free,
- $EE = \mathbb{E}^{\mathbb{Q}}\left[DF(t,\tau) \cdot \max\left(V(\tau,T)\right)\right]$ and $\mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{\tau>T}\right]$ are independent of each other (the probability of default of counterparty C is independent of the value of the derivative); that is there is no Wrong Way Risk.
- $LGD = (1-R)$ is constant for counterparty C, and
- closeout value of transactions will be based on risk-free valuation.

Bringing all of this together we get

$$\begin{aligned} V^D(t,T) &= \mathbb{E}^{\mathbb{Q}}\left[V(t,T) + D(t,\tau) \cdot \mathbb{1}_{\tau \leq T} \cdot (R-1) \cdot \max\left(V(\tau,T),0\right) | \mathcal{F}(t)\right] \\ &= V(t,T) + \mathbb{E}^{\mathbb{Q}}\left[D(t,\tau) \cdot \mathbb{1}_{\tau \leq T} \cdot (1-R) \cdot \max\left(V(\tau,T),0\right) | \mathcal{F}(t)\right] \\ &= V(t,T) - (R-1) \cdot \mathbb{E}^{\mathbb{Q}}\left[D(t,\tau) \cdot \mathbb{1}_{\tau \leq T} \cdot \max\left(V(\tau,T),0\right) | \mathcal{F}(t)\right] \quad (3.16) \\ &= V(t,T) - \mathrm{CVA}(t,T). \quad (3.17) \end{aligned}$$

This takes us back to our starting preposition *Prop. 3.2.1*, and gives us a mathematical definition of the risky adjustment we referred to earlier; from here on, we will define the risky adjustment of counterparty credit risk as CVA.

Derivative valuation with risk=risk-free valuation−risk adjustments
which implies that
Derivative valuation with risk=risk-free valuation−credit value adjustment.

We have now defined the CVA formula:

$$\begin{aligned} \text{CVA}(t,T) &= (1-R)\cdot\mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_(\tau\leq T)\cdot DF(t,\tau)\cdot\max\left(V(\tau,T),0\right)|\mathcal{F}(t)\right] \quad (3.18)\\ &\approx (1-R)\cdot\mathbb{E}_t^{\mathbb{Q}}[\mathbb{1}_{\tau\leq T}]\cdot\mathbb{E}_t^{\mathbb{Q}}[DF(t,\tau)\cdot\max\left(V(\tau,T),0\right)] \\ &= \text{LGD}\times\text{PD}\times\text{EE}. \quad (3.19)\end{aligned}$$

The last two steps use our assumptions and we obtain popular results akin to the traditional credit risk.

3.2.3 Interest Rate Swap

An interest rate swap (IRS) is a financial contract between two counterparties that obligates one counterparty to make fixed payments based on a pre-specified fixed interest rate on a notional amount for a certain period to the opposing counterparty. In return, the opposing counterparty will make floating payments based on a specific floating interest rate on the same notional amount for the same period.

Common uses

Interest rate swaps are one of the most important instruments for the interest rate and credit risk markets. It is a building block for many popular credit structures, including asset swaps. It is also the primary interest rate hedging instrument for credit derivatives. The IRS is a bilateral over the counter contract. Standard swap contracts are transacted within the legal framework of the International Swap Master Agreement produced by the swap market's International Swaps and Derivatives Association (ISDA) [7].

The fundamental reason for this is to transform assets or liabilities from one type into another. If a company has assets of one type and liabilities of the other, it is are severely exposed to possible changes in the yield curve [14]. Banks traditionally fund themselves by issuing floating-rate liabilities, such as deposits/liabilities, but may hold some fixed-rate assets on their balance sheets, such as fixed-rate bonds and loans. As a result, banks and other investors with potentially sizable mismatches between the duration of their liabilities and assets may benefit from buying asset swaps. For instance, a floater-funded investor who invests in fixed-rate bonds could be substantially adversely affected by a sudden rise in market interest rates. Although the value of its liabilities would be relatively unaffected by the rise in rates, that of its assets could potentially plunge [7].

Valuation

Market terminology is used to distinguish between the two parties to an interest rate swap, based on whether they pay or receive a fixed rate. The party that pays the fixed rate is known as the payer, and the party that receives the fixed rate is known as the receiver.

The valuation of a swap is divided into two separate valuations of the fixed and floating legs. These separate valuations are then netted to obtain the value of the swap.

To simplify the model (and to align with general practice in South African markets), we will assume that the payments of both legs will take place on the same dates and that the notional is fixed/constant for all swap periods. Furthermore, we adopt the following notation:

- t_0 is today.

- T_0 is swap inception date.
- T_i is i-th swap inception payment date (start of swap period i).
- T_M is swap maturity.
- Not is notional of the swap.
- $\pi_i = Coverage(T_i, T_{i-1}, DCB)$ is coverage[2] between T_i and T_{i-1} using the day count basis DCB.
- K is swap rate (simple).
- $l_i(T_{i-1})$ is forward simple rate (LIBOR rate) that applies between T_i and T_{i-1}.
- $DF(t; T_i, T_{i-1})$ is discount factor between T_i and T_{i-1} using a zero curve at time t.

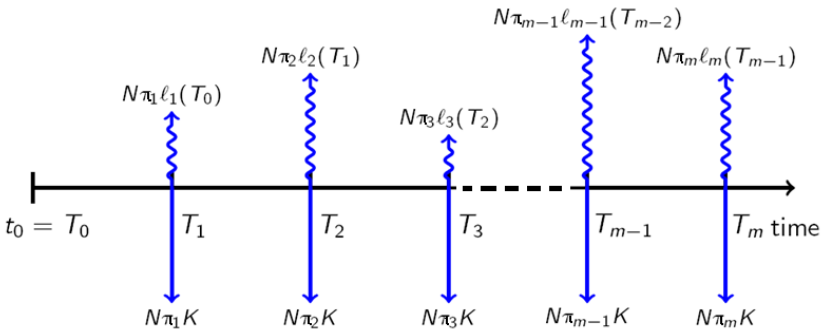


Figure 3.4: Interest Rate Swap Cashflows

Consider valuation date t, such that $t \leq T_M$, then the value at time t of the fixed leg of the swap is given by [50]:

$$V_{\text{fix}}(t) = \mathbb{E}^{\mathbb{Q}}\left[\sum_{t_i > t}^{m} Not \cdot \pi_i \cdot K \cdot DF(t; t, T_i) | \mathcal{F}(t)\right]$$

For the floating leg, there are two cases to consider:

1. where the $t < T_0$, future starting swap,
2. where $T_0 < t \leq T_M$, on the run swap.

Case 1 - future starting:

$$\begin{aligned} V^1_{\text{flt}}(t) &= \mathbb{E}^{\mathbb{Q}}\left[\sum_{T_i > t}^{m} Not \cdot \pi_i \cdot l_i(T_{i-1}) \cdot DF(t; t, T_i) | F(t)\right] \\ &= \sum_{t_i > t}^{m} Not \cdot \pi_i \cdot \mathbb{E}^{\mathbb{Q}}[l_i(T_{i-1})] \cdot DF(t; t, T_i) | \mathcal{F}(t). \end{aligned}$$

[2] $Coverage(T_i, T_{i-1})$ means the time between T_i and T expressed in years.

Case 2 - on the run:
There exists k, such that $T_0 < T_k < t < T_{k+1} < T_M$

$$\begin{aligned} V^2_{\text{flt}}(t) &= Not \cdot \pi(t, T_{k+1}) \cdot l_i(T_{i-1}) \cdot DF(t; t, T_i) \\ &\quad + \sum_{t_i > t}^{m} Not \cdot \pi_i \cdot \mathbb{E}^{\mathbb{Q}}_t[l_i(T_{i-1})] \cdot DF(t; t, T_i). \end{aligned}$$

The arbitrage free condition for fair forward LIBOR rate must satisfy the condition

$$\mathbb{E}^{\mathbb{Q}}[l_k(t)] = \frac{1}{\pi_k}\left(\frac{DF(t, T_{k-1})}{DF(t, T_k)} - 1\right). \tag{3.20}$$

The net value of the swap is then:

$$V_{\text{swap}}(t) = V_{\text{fixed}}(t) - V_{\text{floating}}(t). \tag{3.21}$$

Dual curve valuation

In our numerical trials presented in the results section, we use the dual curve valuation methodology, with overnight index swap (OIS) curve for discounting and the *Johannesburg Interbank Average Rate*(JIBAR)[3] for forecasting cash-flows. The background of this method is as follows:

Before the onset of the credit crisis in 2007, derivatives dealers commonly used LIBOR the short-term borrowing rate of AA-rated financial institutions as a proxy for the risk-free rate [22]. However, the global credit crisis that unfolded between 2007-2010 period raised significant concerns about the liquidity and creditworthiness of major banks, thereby influencing the valuation and structuring of instruments such as interest rate swaps [13].

During this period, there was mounting pressure on dealers to mitigate counterparty risk associated with over the counter (OTC) derivative transactions, particularly following the collapse of Lehman Brothers, a commercial bank that was considered AA credit. This necessitates the implementation of more stringent measures, including daily margins collateral calculations and maintenance. Moreover, in years subsequent to the GCC, regulatory investigations into the computation of LIBOR by large banks that were members of the LIBOR fixing panels led to a loss of confidence in LIBOR as an accurate and fair market rate [34]. Consequently, the market began to favour the OIS [4] curve as a new risk-free swap curvewhich led to the dual curve pricing regime [4]. Dual-curve discounting refers to the practice of using one interest rate curve to project a swap's cash flows while employing another curve for discounting those cash flows [32].

A discount curve is constructed at the valuation point to measure the fair value of a financial instrument. This curve was used to calculate the present value of the instrument's cash flows. Within the risk- neutral framework, a unique risk-free rate is used to discounting future cash flow. OIS discounting involves using the OIS curve, which is considered nearly risk-free, to discount the expected cash flow of a derivative (*Bianchetti [29]*).

[3]Global efforts are underway to address weaknesses in current benchmark rates and transition to more robust alternatives. In South Africa the transition is to South African Rand (ZAR) Overnight Index Average (ZARONIA).

[4]An Overnight Indexed Swap (OIS) is a fixed/floating interest rate swap where the floating rate is based on the published effective federal funds rate (for USD) or equivalent benchmarks for other currencies, such as the Euro overnight index average (EONIA) or the sterling overnight index average (SONIA)[39] .

A forecast curve was applied to determine the expected value of the reference rate for future dates. In the South African market, examples of such curves include those for forecasting JIBAR (with separate 3-month, 6-month, and 1-year curves), the Consumer Price Index (CPI), and *prime*. While it may be feasible to use a forecasting curve for discounting, the ZAR swap curve (referencing to the 3-month JIBAR curve) is often employed as a risk-free curve. Nevertheless, the forecast and discount curves should generally be regarded as distinct entities. The forecast curves utilized in valuations should align with the contractually specified reference rate.

3.3 XVAs Primer

The common element in the calculation of many counter party credit risk metrics is the computation of exposure. Herein, we discuss the computation of exposures and how they are generally used in XVA computations and in PFE.

3.3.1 Generic XVAs

The adjustments referred to in XVAs are an essential element in the valuation of financial instruments, encompassing a range of adjustments that address various risks and costs linked to trading derivatives.

The primary components of XVAs include CVA, Debit Valuation Adjustment (DVA), Funding Valuation Adjustment (FVA), Collateral Valuation Adjustment (ColVA), Capital Valuation Adjustment (KVA) and Margin Valuation Adjustment (MVA). The CVA reflects counterparty credit risk, accounting for the possibility of a counterparty's default. Conversely, the DVA considers the entity's own credit risk, encompassing the risk that the firm itself might default. FVA represents the funding costs associated with maintaining a derivative position, including the cost of collateral and borrowing. ColVA represents costs and benefits, such as the ability to choose the currency or type of collateral to post and any other non-standard collateral terms. On the other hand, KVA accounts for regulatory capital requirements, reflecting the cost of holding capital against potential future losses. Each adjustment addresses a specific aspect of risk and cost, contributing to a more accurate and holistic valuation of derivatives (*Gregory [19]*). See Figure *3.5* for a diagrammatic representation of the various valuation adjustments.

These adjustments ensure that derivative pricing reflects the true economic costs and risks, thereby enhancing the accuracy and reliability of valuations. By incorporating XVAs, financial institutions can better manage their risk exposure, allocate capital more efficiently, and comply with regulatory requirements. Furthermore, XVAs play a crucial role in risk management strategies, enabling firms to assess the full spectrum of risks and costs associated with their derivative portfolios.

3.3.2 XVA - quantification considerations

This presentation closely follows the discussion on expected exposures for XVAs in [1].

Given a financial instrument with a value denoted by $V(t, \mathbf{X})$ at time t and is a function of risk factor $\mathbf{X}(t)$ [5] traded between a bank and client or counterparty. The positive and

[5]Note: below we will focus on the unknown/modelled risk factors and we will refer to known/deterministic or fixed inputs as parameters

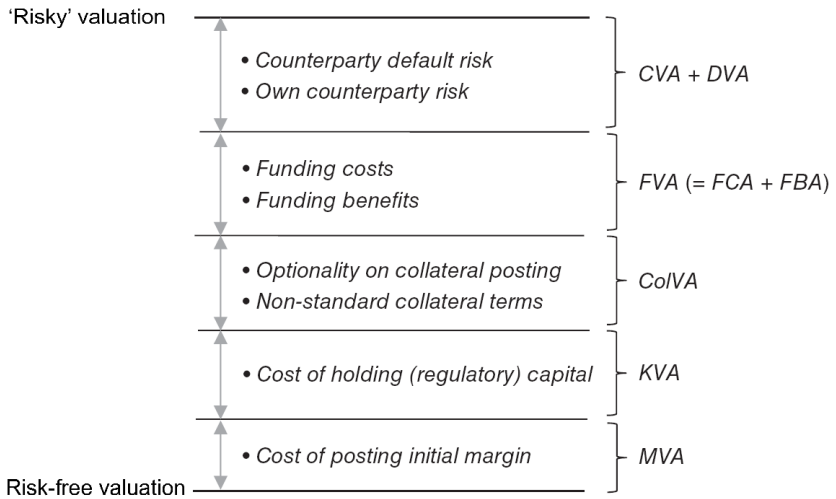


Figure 3.5: Illustration of the role of valuation adjustments (XVAs), adapted from [19]

negative exposures, $E^+(t, \mathbf{X}(t))$, $E^-(t, \mathbf{X}(t))$, – derivative is asset or liability from the perspective of the bank – are defined as:

$$E^+(t, \mathbf{X}(t)) := \max(V(t, \mathbf{X}(t)), 0), \quad \text{positive exposure, and} \tag{3.22}$$

$$E^-(t, \mathbf{X}(t)) := \min(V(t, \mathbf{X}(t)), 0), \quad \text{negative exposure} \tag{3.23}$$

with

$$V(t, \mathbf{X}(t)) \quad = \quad \mathbb{E}^{\mathbb{Q}} \left[\sum_{j=1}^{M} DF(t, T_j) H(T_j, \mathbf{X}(T_j)) \middle| \mathcal{F}(t) \right], \quad T_j > t, \tag{3.24}$$

where $V(t, \mathbf{X}(t))$ represents the discounted value of payoff $H(T_j, \mathbf{X}(T_j))$ at time t. The payments occur at time step T_j, $j = 1, \ldots, M$, with $T_j > t$, thus only outstanding payments are considered in the exposure computation. $DF(t, T) = \frac{M(t)}{M(T_j)}$ is the stochastic discount factor defined in terms of the money market account $M(t)$.

Definition 3.3.1. *The money market account is function of the short-rate $r(t)$*

$$M(t) = \mathrm{e}^{\int_0^t r(s)\mathrm{d}s},$$

repwhichthi represents the future value of one unit of the base currency invested today at the prevailing interest rate for maturity t

In practice, the exposures are computed per netting set, often involving hundreds of trades ; thus with a netted portfolio involving L trades and d different risk factors, the value $V(\cdot)$ is given by:

$$V(t, \mathbf{X}(t)) := \sum_{i=1}^{L} V_i(t, \mathbf{X}(t)), \quad \mathbf{X}(t) = [X_1(t), \ldots, X_d(t)]^{\mathrm{T}}. \tag{3.25}$$

Each risk factor in $\mathbf{X}(t)$ represents a stochastic process that influences of the portfolio value. These can be interest rates for different currencies, stocks, inflation, foreign exchange, and

commodities. The number of risk factors typically varies with time and depends on portfolio composition.

Every component of XVA depends on the exposure computation for any exposure date, T_i. In general, this can be represented as follows.

$$\begin{aligned} \text{XVA}(t_0) &= \int_{t_0}^{T_M} \mathbb{E}^{\mathbb{Q}} \left[DF(t_0, T_M)\chi(t, V(t, \mathbf{X}(t))) \middle| \mathcal{F}(t_0) \right] \mathrm{d}t, \\ &\approx \sum_{k=1}^{M} \mathbb{E}^{\mathbb{Q}} \left[DF(t_0, T_k)\chi(T_k, V(T_k, \mathbf{X}(T_k))) \middle| \mathcal{F}(t_0) \right] \Delta t, \end{aligned} \tag{3.26}$$

with a generic function of exposures $\chi(t, x)$ and a discretization grid, $T_1, \ldots, T_M$.

For example in the case of CVA, $\chi(t, x)$ is

$$\chi(t, x) = (1 - R_c) \cdot E^+(t, x) \cdot Q_D(t),$$

with $E^+(t, x)$ is defined in Equation *(3.22)*, $Q_D(t)$ being the default probability (we also assume no wrong-way-risk, probability of default is independent of value of derivative [6]) and R_c is the recovery rate.

To estimate the value of portfolio $V(t, \mathbf{X}(t))$ using N simulated Monte Carlo paths, the portfolio must be evaluated N times at each exposure date T_k, resulting in a total of $N \times M$ evaluations. For portfolios that involve multiple risk factors and instruments, the number of paths can range from hundreds to millions. From a computational standpoint, it is desirable to minimize the number of portfolio evaluations.

Remark 3.3.1. *The Chebyshev technique aims to considerably reduce the number of portfolio evaluations without compromising computation accuracy. This can be achieved by employing Chebyshev tensors as proxy functions instead of the original pricer functions, as (it will be demonstrated later that) they are very accurate and are considerably faster to execute.*

Therefore, the portfolio value $V(t, \mathbf{X}(t))$ is replaced by the approximating function $\widetilde{p}_n(\{V\}_{i_1,\ldots,i_d}, \mathbf{X}(t))$, where $\{V\}_{i_1,\ldots,i_d} := V(t, \{\mathbf{X}\}_{i_1,\ldots,i_d})$ and n is the degree of the Chebyshev interpolant (determined by the number of Chebyshev anchor points).

The proxy function $\widetilde{p}(\cdot)$ is built using only a few evaluations of the portfolio $V(t, \{\mathbf{X}\}_{i_1,\ldots,i_d})$ on the set of Chebyshev anchor points $\{\mathbf{X}\}_{i_1,\ldots,i_d}$.

Once the approximating function $\widetilde{p}(\cdot)$ is determined, XVA is computed as follows.

$$\text{XVA}(t_0) \approx \sum_{k=1}^{M_T} \mathbb{E}^{\mathbb{Q}} \left[DF(t_0, T_k)\chi(T_k, \widetilde{p}(\{V\}_{i_1,\ldots,i_d}, \mathbf{X}(T_k))) \middle| \mathcal{F}(t_0) \right] \Delta t, \tag{3.27}$$

with $\{V\}_{i_1,\ldots,i_d} := V(T_k, \{\mathbf{X}\}_{i_1,\ldots,i_d})$. Function $\widetilde{p}(\cdot)$ does not require portfolio evaluations for every Monte Carlo path. Once the approximating function $\widetilde{p}(\cdot, \mathbf{X}(t))$ is established (this is done offline), it can be evaluated for all Monte Carlo paths.

We now return to the exposure Equation *(3.26)* and plug in our proxy function $\widetilde{p}(\cdot; V(T_k, \{\mathbf{X}\}_{i_1,\ldots,i_d})$ which we shall shorten to $\widetilde{p}_k(\cdot)$ in place of the original valuation model V at each time step.

$$E^+(T_k, \mathbf{X}(T_k)) := \max(V(T_k, \mathbf{X}(T_k)), 0) \quad \approx \quad \max\left(\widetilde{p}_k(\mathbf{X}(T_k)), 0\right).$$

[6] However, the methodology would stay intact even when these assumptions were relaxed.

We can then plug this back into the expression for the expected exposure (EE)

$$\begin{aligned} \mathrm{EE}(t_0, T_k) &= \mathbb{E}^{\mathbb{Q}}\left[DF(t_0, T_k) \cdot E^+(T_k, \mathbf{X}(T_k)) \middle| \mathcal{F}(t_0)\right] \\ &\approx \frac{1}{N}\sum_{i=1}^{N} DF_i(t_0, T_k) \cdot E^+\left(T_k, \mathbf{X}(T_k)\right)^i \qquad (3.28) \\ &= \frac{1}{N}\sum_{i=1}^{N} \exp\left\{\sum_{j=1}^{k} r^i(t_j) \cdot (t_j - t_{j-1})\right\} \cdot E^+\left(T_k, \mathbf{X}(T_k)\right)^i . \qquad (3.29) \end{aligned}$$

3.3.3 PFE - quantification

Another metric to measure quality is the so-called *PFE* which measures the maximum credit exposure calculated at a confidence level α. This measure can be associated with measuring the value function in the tails of the distribution of $V(t, \mathbf{X}(t))$, at $t = T_k$. PFE today for time T, that is, $\mathrm{PFE}(t_0, T)$, is defined as the α-quantile of the positive exposure $E^+(t_0; T, \mathbf{X}(T))$.

The PFE describes a curve from now (t) until of the transaction maturity (T), which outlines the α quantile of the positive valuation of the transaction.

$$\begin{aligned} \mathrm{PFE}_\alpha(t_0, T) &= \inf\left\{x_t \in \mathbb{R} : \alpha \leq F_{E^+(t,\mathbf{X}(t))}(x_t)\right\}, \quad \forall t \leq T \qquad (3.30) \\ &\approx \min\left\{x_k \middle| \mathbb{P}(E^+(t_0; t_k, \mathbf{X}(t_k)) > x_k) \geq \alpha\%\right\}, \quad \forall t_k \leq T_M \qquad (3.31) \end{aligned}$$

where $F_{E^+(t,\mathbf{X}(t))}(x)$ is the cumulative density function (CDF) of positive exposures observed at time t. Coefficient α represents the certainty level, that is, the quantile level. Notice that the probability measure used for PFE is the *real world measure* $\mathbb{P}$ and not the *risk neutral measure* $\mathbb{Q}$ that we used before, as discussed below in *Note 3.3.1* [7].

The PFE concept is illustrated in Figure *3.6*

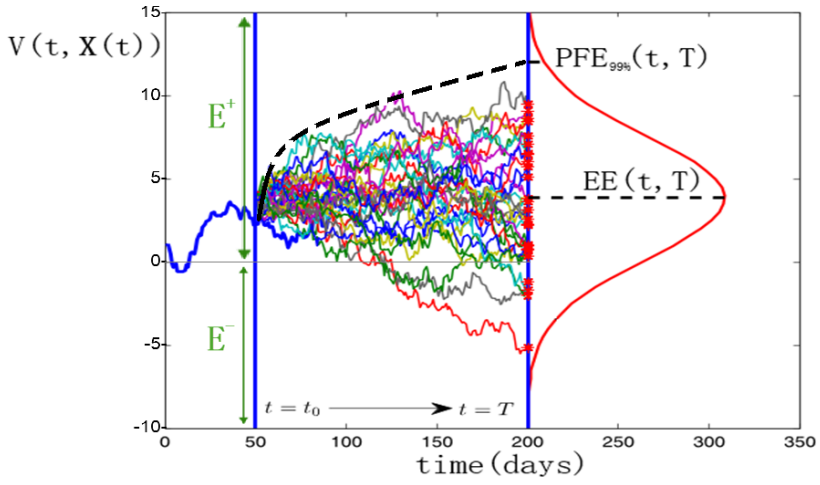


Figure 3.6: An illustration of PFE and EE

[7] In the results section, *Section 5*, we ignore the distinction between $\mathbb{Q}$ and $\mathbb{P}$ and all computations are done using one measure $\mathbb{Q}$

Note 3.3.1. Risk Neutral $\mathbb{Q}$ vs Real World Measure $\mathbb{P}$
The general rule is that risk-neutral parameters are typically used in pricing applications (such as the calculation of CVA), while real-world parameters generally form the basis of risk management models (such as PFE, which is often used for limit setting). CVA is a valuation adjustment; hence, it needs to be hedged in the market; hence, a risk-neutral measure is more appropriate.

*However, there is more to consider than this general rule, such as the three key parameters (*Gregory [19]*):*

- Drift *: in a pricing or market dynamic model, uncertainty or volatility is important in the short term but in the long term the drift/mean rate dominates. CVA type calculations are often long term (+ 5 years). Therefore, drifts were more influential in the calculations. The factors that impact drift vary from one asset class to another and need to be considered; thus, one cannot simply rely solely on current futures prices.*
- Volatility *: implied volatility is often the best forward looking volatility estimate. However, risk premiums embedded in market implied volatilities lead to a systematic overestimation of the overall risk.*
- Correlation *: is tricky in that a high or low correlation is not always necessarily conservative, as it depends on the structure of the trades and portfolio being considered. In some instances, market estimates are available for correlation; however, this is often not the case. Therefore, in most cases, market related data on correlation are not available, and one has to rely on a real world measure to estimate this parameter.*
- Mean reversion *: Many market variables (e.g., commodities and interest rates) tend to mean-revert over time, which pulls long term rates back to an average level. Mean reversion has an impact on future spot prices and volatilities. Risk neutral mean reversions, which are often hard to calibrate, tend to be smaller than mean-reversions estimated from historical data.*

Figure 3.7 *illustrates the of the empirical difference between real-world and risk-neutral default probabilities derived from economic studies, see for details and discussion [19] with respect to credit risk and CCR.*

3.4 Hull-White 1-factor model

In this section, we describe the 1-factor Hull-White (HW), that we use in the numerical trials to model stochastic interest rates.

In 1990, Hull and White *[20]* introduced an extension to the single factor Vasicek model [38] that allowed the short-rate process to fit the observed bond prices. This was achieved using a time-dependent risk-neutral drift parameter that matched the prevailing market curve.

Consider short rate process $r(t)$, with the dynamics given by [20]:

$$\mathrm{d}r(t) = (\theta(t) - a \cdot r(t))\mathrm{d}t + \sigma \mathrm{d}W^{\mathbb{Q}}(t), \tag{3.32}$$

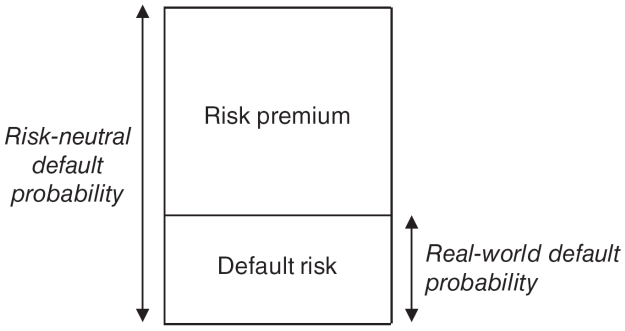


Figure 3.7: Empirical difference between real-world and risk-neutral default probabilities. Source (*Gregory [19]*)

where:

1. a, represents the speed of mean reversion
2. σ, represents the local volatility, and
3. $\theta(t)$, represents the time-dependent long term mean, which satisfies the following relationship:

$$\theta(t) = \frac{\partial}{\partial t} f^{mkt}(0,t) + a \cdot f^{mkt}(0,t) + \frac{\sigma^2}{2a}\left(1 - \mathrm{e}^{-2a\cdot t}\right), \tag{3.33}$$

f^{mkt} represents the observed market forward-rate,

$$f^{mkt}(0,t) = -\frac{\partial}{\partial t}\log(P_{mkt}(0,t)),$$

where $P_{mkt}(0;t)$ is a zero coupon bond (ZCB) from the financial market.

The Hull-White process in Equation *(3.32)* has a solution that allows for a large time step Monte Carlo simulation with the following *exact* formula*[20]*:

$$r(t) = r(0)\mathrm{e}^{-a\cdot t} + \gamma(t) - \gamma_0 \mathrm{e}^{-a\cdot t} + \sigma \int_0^t \mathrm{e}^{-a\cdot(t-s)} \mathrm{d}W_s, \tag{3.34}$$

where

$$\begin{aligned} \gamma(t) &= f^{mkt}(0,t) + \frac{\sigma^2}{2a^2}\left(1 - \mathrm{e}^{-a\cdot t}\right)^2 \quad \text{or alternative} \\ &= \int_0^t \theta(s)\mathrm{e}^{-a\cdot(t-s)}\mathrm{d}s + f^{mkt}(0,t). \end{aligned}$$

The distribution of the shortrate at time t is given by the following mean and standard deviation.

$$\begin{aligned} \mathbb{E}^{\mathbb{Q}}[r(t)] &= r_0\mathrm{e}^{-a\cdot t} + \gamma(t) - \gamma_0\mathrm{e}^{-a\cdot t}. \\ \mathbb{V}^{\mathbb{Q}}[r(t)] &= \frac{\sigma^2}{2a}\left(1 - \mathrm{e}^{-2a\cdot t}\right). \\ \Rightarrow r(t) &= \mathbb{E}^{\mathbb{Q}}[r(t)] + \sqrt{\mathbb{V}^{\mathbb{Q}}[r(t)]}\cdot \mathrm{d}W_t. \end{aligned} \tag{3.35}$$

The zero coupon bond P, which we describe as the discount factor DF, predicted by the Hull-White model, is given by *[20]*

$$\begin{aligned} DF(t,T) &= P(t,T) \\ &= \mathbb{E}^{\mathbb{Q}}\left[\frac{M(t)}{M(T)}|\mathcal{F}(t)\right] \\ &= \mathbb{E}^{\mathbb{Q}}\left[\mathrm{e}^{-\int_t^T r(s)\mathrm{d}s}|\mathcal{F}(t)\right] \\ &= \mathrm{e}^{A(t,T)+B(t,T)r(t)}, \end{aligned}$$

where:

1. $B(t,T)$ is a function that is directly coupled to the short rate, allow ing the bond price to be expressed as an exponentially linear function of the short rate.

$$B(t,T) = \frac{1}{a}\left(1-\mathrm{e}^{-a(T-t)}\right).$$

2. $A(t,T)$ is a function that is not directly coupled to the short rate, allowing the bond price to be expressed as an exponentially linear function of the short rate.

$$\begin{aligned} A(t,T) &= -\int_t^T \theta(s)B(s,T)\mathrm{d}s + \frac{1}{2}a\int_t^T B(s,T)^2\mathrm{d}s \\ &= \ln\left(\frac{P_{mkt}(0;t)}{P_{mkt}(0;T)}\right) + B(t,T)\cdot f^{mkt}(0,t) - \frac{\sigma^2}{4a}B^2(t,T)\left(1-\mathrm{e}^{-2at}\right). \end{aligned}$$

Chapter 4

Methodology

In this chapter we will go into the details of the following:

- mathematical theory underpinning the Chebyshev techniques, and
- the applications of the techniques.

4.1 Theoretical framework - Chebyshev techniques

In this section, we review the key mathematical results underpinning what we shall refer to as "Chebyshev numerical techniques," which we then apply to problems in quantitative finance, with a focus on risk management calculations. We start our journey from the basic and key results from approximation theory and then build a case for paying attention to Chebyshev interpolants above all other approximation methods. We follow closely the presentations of "*Approximation Theory and Approximation Practice*" by Lloyd N. Trefethen (2018) [28] and "*Machine Learning for Risk Calculations*" by Zeron and Ruiz (2022)[18].

4.1.1 Approximation Theory

It is well understood that well-behaved functions (e.g. analytical, smooth/continuous and many-times differentiable) can be sufficiently approximated by polynomials. This intuitive interpretation comes from the visualization of how any function appears to be made up of a superposition of polynomials.

The above intuition is captured succinctly by one of the most important results of approximation theory, "*that every continuous function on a bounded interval can be approximated to an arbitrary accuracy by polynomials*." This is known as the Weierstrass approximation theorem, which was credited to Karl Weierstrass in 1885[1] [28].

4.1.2 First some preliminaries

Any interval $[a, b]$ can be scaled to $[-1, 1]$ by means of a linear map, and for the remainder of this chapter, we refer to the domain $[-1, 1]$ but the result generalize to any bounded

[1] Carl Runge also independently discovered the result in 1886 [28]

domain. Also in what follows $\|\cdot\|_\infty$ denotes the supremum norm on $[-1, 1]$.

Consider the canonical filtration $\mathcal{P} = \bigcup_{n\in N} \mathcal{P}_n$, where $\mathcal{P}_n$ is the space of polynomials of degree at most n. The higher the n, the more polynomials available to choose.

Theorem 4.1.1. *Let f be a continuous function on [-1,1], and let $\epsilon > 0$ be arbitrary. Then a polynomial exists p such that*

$$\|f - p\|_\infty < \epsilon$$

.

However, this result does not provide a recipe to find a suitable approximating polynomial p, for a given continuous function f. Set $\mathcal{P}$, could be and is highly likely to massive . How do we find the right p? A systemic method is needed for finding or constructing p efficiently.

4.1.3 Preliminaries on well-behaved functions:

Herein, we review various definitions of continuity[2] and differentiability [3] for functions that will be useful later when we discuss Chebyshev interpolation. A good starting point, is the helicopter view of the hierarchy of continuity, which is captured in this result from a real analysis:

Remark 4.1.1. *The following chains of inclusion apply for functions over a compact interval of the real line:*

continuously differentiable $\subset$ Lipschitz continuous $\subset$ absolutely continuous $\subset$ bounded variation $\subset$ differentiable almost everywhere
source: [8]

We consider the Lipschitz continuity [4].

Definition 4.1.1. *A real-valued function $f : \mathbb{R} \to \mathbb{R}$ is called Lipschitz continuous if there exists a positive real constant K such that for all real values x_1 and x_2 in the domain $[-1, 1]$.*

$$|f(x_1) - f(x_2)| < K\,|x_1 - x_2|$$

By looking at the definition of Lipschitz continuity, one observation comes to mind, which resembles its condition on the first derivative. Because there is no restriction on the closeness of points x_1 and x_2, the condition should hold for when they are arbitrarily close ($\lim_{\Delta x\to 0}$), where we obtain a derivative or approximation of an absolute derivative. We see this by

[2] think of the concept smoothness and no gaps/holes
[3] think no sharp corners or excessive steepness/near vertical curves
[4] intuitively, one can think of it as a strong form of uniform continuity for functions, in that it ensures that a function does not change too rapidly

rearranging the terms as follows

$$\begin{aligned} & |f(x_1) - f(x_2)| && < K\,|x_1 - x_2| \\ \Rightarrow \quad & \tfrac{|f(x_1)-f(x_2)|}{|x_1-x_2|} && \leq K \\ \Rightarrow \quad & \tfrac{|\Delta f|}{|\Delta x|} && \leq K \\ \Rightarrow \quad & \tfrac{|\partial f|}{|\partial x|} && \leq K \text{ after taking the limit } \lim_{\Delta x \to 0} \end{aligned}$$

Therefore, we can say that a function is Lipschitz continuous if it has a bounded first derivative. In fact, we have that *any function defined on an interval and having a bounded first derivative is Lipschitz continuous.*

Another notion of a well-behaved function that we will encounter is the so-called analytic function. This concept is very different from continuity, although the functions we often encounter such polynomials, exponential functions, and trigonometric functions are by and large analytic and continuous (smooth) functions.

Definition 4.1.2. *Function f is analytic if for every point x in the domain, the Taylor Series of f at x exists and converges to the value of $f(x)$.*

Note that the analytic implies that a *function is infinitely differentiable and can be represented by its Taylor series.* In addition, relative to Lipschitz continuity, the *analytical* condition is much stricter and therefore harder to achieve (in general). The general rule is that *any polynomial function, which is analytic, is also Lipschitz continuous on any bounded interval.* Although analytic functions are continuous, they are not necessarily Lipschitz continuous.

4.1.4 Chebyshev Polynomials

We begin by building some intuition by deriving the (form of) Chebyshev polynomials by considering some examples first.

Example 4.1.4.1. *Recall the following trigonometric identities (known from high school):*

$$\cos(A) + \cos(B) = 2\cos(\frac{A+B}{2})\cos(\frac{A-B}{2}) \tag{4.1}$$

Now, consider the following sequence of functions, denoted by $\{t_k\}$ for $k \in \mathbb{N}$,

$$t_k = \cos(k \cdot \theta) \tag{4.2}$$

Now, letus expand the sequence by recursively using Equation (4.1) and simplifying, and see what it looks like.

$$
\begin{aligned}
\textit{When} \quad k = 0 \Rightarrow t_0 = \cos(0) &= 1 \\
k = 1 \Rightarrow t_1 = \cos(\theta) & \\
k = 2 \Rightarrow t_2 = \cos(2\theta) &= 2\cos^2(\theta) - \cos(0) \\
&= 2(t_1)^2 - 1 \\
k = 3 \Rightarrow t_3 = \cos(3\theta) &= 2\cos(2\theta)\cos(\theta) - \cos(\theta) \\
&= 2(2\cos^2(\theta) - \cos(0))\cos(\theta) - \cos(\theta) \\
&= 4\cos^3(\theta) - 3\cos(\theta) \\
&= 4(t_1)^3 - 3(t_1)
\end{aligned}
$$

Therefore, we observe that a polynomial emerges in $t_1 = \cos(\theta)$ emerge. And so we can generalize the recursive relation between the conservative terms by using Equation (4.1) recursively and simplifying. By setting $A = (k+1)\theta$ and $B = (k-1)\theta$, and rearranging the terms, we obtain

$$
\begin{aligned}
& \cos((k+1)\theta) + \cos((k-1)\theta) && = 2\cos(k\theta)\cos(\theta) \\
\Rightarrow \quad & \cos((k+1)\theta) && = 2\cos(k\theta)\cos(\theta) + \cos((k-1)\theta) \\
\Rightarrow \quad & t_{k+1} && = 2t_k \times t_1 + t_{k-1}
\end{aligned}
$$

These are the Chebyshev polynomials, for the formal definition we set the first term to equal some real valued variable x i.e. $T_1 = \cos(\theta) = x$.

Now, for the formal definition of the Chebyshev polynomials of the first kind[5]

Definition 4.1.3. *The $k-th$ Chebyshev polynomial is defined as*

$$T_k = \cos(k \cdot \theta)$$

where $\theta = \cos^{-1}(x) = \arccos(x)$

$$\Rightarrow T_k = cos(k \cdot \arccos(x)).$$

This is known as a Chebyshev polynomial of the first kind.

Definition 4.1.4. *Alternatively, we can use a recursive formula to define the Chebyshev polynomials:*

$$T_0(x) = 1,$$
$$T_1(x) = x,$$
$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x).$$

[5]There are Chebyshev polynomials of second kind , denoted by U_n, are defined by

$$U_n(\cos\theta)\sin\theta = \sin((n+1)\theta)$$

. We will also refer to them when discussing the Chebyshev points.

We show some examples of the first six Chebyshev polynomials in Figure *4.1* on the domain $[-1;1]$. For brevity, we will not discuss all the specific properties. However, theses Chebyshev functions form an orthogonal basis for the space of polynomials $\mathcal{P}$, which means that any polynomial can be expressed as linear combination of Chebyshev functions. This is more formally and thoroughly articulated in *Theorem 4.1.2* [28].

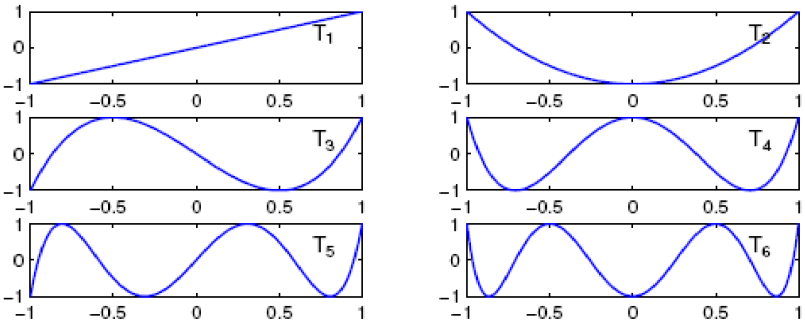


Figure 4.1: This figure displays Chebyshev polynomials T_1 to T_6 (*Trefethen [28]*)

Theorem 4.1.2. *If f is Lipschitz continuous on [-1,1], it has a unique representation as a Chebyshev series:*

$$f(x) = \sum_{k=0}^{\infty} a_k \cdot T_k(x)$$

which are completely and uniformly convergent. The coefficients are given for $k > 1$ using the following formula:

$$a_k = \frac{2}{\pi} \int_{-1}^{1} \frac{f(y) \cdot T_k(y)}{\sqrt{1-y^2}} dy$$

and

$$a_0 = \frac{1}{\pi} \int_{-1}^{1} \frac{f(y) \cdot T_k(y)}{\sqrt{1-y^2}} dy.$$

For a fixed natural number n, we define a Chebyshev projection of degree n, which is truncated Chebyshev series

$$f_n(x) = \sum_{k=0}^{n} a_k \cdot T_k(x).$$

Theorem 4.1.2 tells us that for a Lipschitz continuous function we can always find a unique convergent Chebyshev series, but it does not say anything about the effort required to do so. Moreover, often (in practice), we do not need the entire Chebyshev series but can be content while dealing with a very reasonable estimate or truncated series. Therefore, we need to know more about how to select the truncated Chebyshev series, and the following theorem provides some useful insights. However, before we obtain the theorem, we quickly recap the concept of total variation

Definition 4.1.5. *A function, whether continuous or not, has bounded variation if its total variation is finite. The total variation is the $1-norm/\|\cdot\|_1$ of the derivative [28]*

Now we are ready to review the theorem [28]:

Theorem 4.1.3. *For an integer $m > 0$, let f and its derivatives up to and including f^{m-1} be absolutely continuous on $[-1, 1]$. Furthermore, suppose the m-th derivative f^m is of bounded variation V. Then for any $n > m$ its Chebyshev projection satisfies*

$$|f - f_n|_\infty \leq \frac{2V}{\pi \cdot m \cdot (n-m)^m}$$

.

For a particular class of functions, the convergence rate improves considerably. This is a class of analytical functions and the convergence rate is geometric. This was first announced by Bernstein in 1911 [37]. We also note similar results for slightly weaker conditions, that if f^{m-1} is only Lipschitz continuous, then $\|f - p_n^*\| = \mathcal{O}(n^{-m})$, where p^* is the best polynomial approximator for f. However, as Trefethen notes in [28], in practice, one rarely deals with a function that is Lipschitz continuous while lacking a derivative of bounded variation, and one often works with polynomial projections and Chebyshev interpolants rather than best approximations.

4.1.5 Convergence for analytical functions

Recall that a function f is analytical if, for every point x in its domain $\subset [-1, 1]$, the Taylor Series of f at x exists and converges to the value of $f(x)$. We also refer to the well-known result that if a function f is analytic on $[-1, 1]$, then it can be analytically continued to the neighbourhood of $[-1, 1]$ in the complex plane. In particular, for polynomial approximations, the neighbourhood that matters is the regions in the complex plane bounded by ellipses with focci at -1 and 1, known as the Bernstein ellipse.

Definition 4.1.6. *Bernstein ellipse with radius ρ, $E_\rho^{[-1;1]} \subset C$ with parameter $\rho > 1$ is defined as the open region in the complex plane bounded by the ellipse with foci ± 1 and semiminor and semimajor axis lengths summing up to ρ. See Figure 4.2 for an illustration.*

In general, the larger the neighbourhood, the faster the convergence. See Figure *4.3* for examples of these ellipses.

Theorem 4.1.4. *If f is analytic on $[-1, 1]$. Consider its analytical continuation to the open Bernstein ellipse E_ρ, of radius ρ, where it satisfies $|f(x)| \leq M$ for some M. Then for each $n > 0$*

$$|f - f_n|_\infty \leq \frac{2M\rho^{-n}}{\rho - 1},$$

*where f_n is the Chebyshev projection of degree n (*Trefethen [28]*).*

At this point, one concern is determining the Chebyshev series (or truncated series) particularly the calculation of the coefficients

$$a_k = \frac{2}{\pi} \int_{-1}^{1} \frac{f(s) \cdot T_k(s)}{\sqrt{1-s}} ds.$$

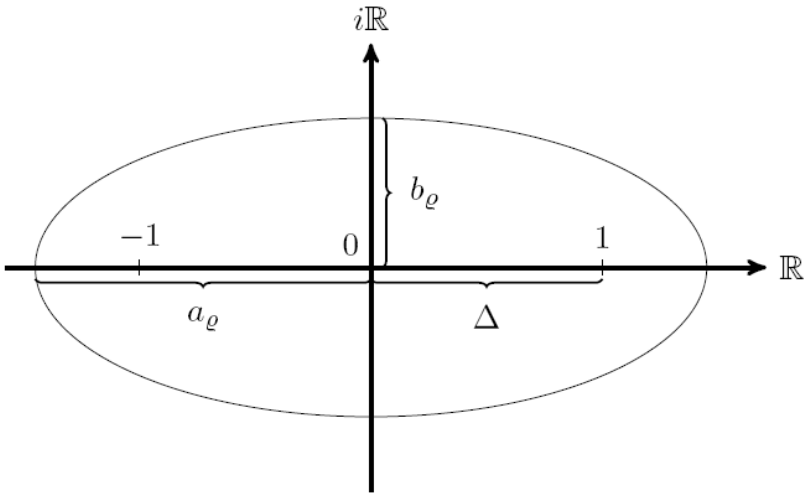


Figure 4.2: A Bernstein ellipse $E([-1;1];\rho)$ with foci -1, 1 and ellipse parameter $\rho > 1$. The semimajor a_ρ is part of the real line, the semiminor b_ρ is part of the complex line and satisfy: $b_\rho + b_\rho = 1$ and $\pm 1 = \sqrt{a_\rho^2 - b_\rho^2}$ *Gaß [30]*

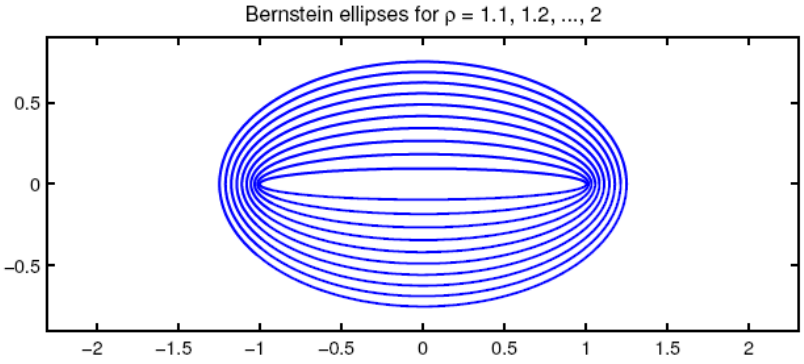


Figure 4.3: Bernstein ellipses for $\rho = 1.1, 1.2, \ldots, 2$ (*Trefethen [28]*)

Note that this integral entails evaluating $f(x)$, the subject function that we are trying to estimate many times. Therefore, the Chebyshev series implies that we should already know the function in the domain we where we are trying to estimate in detail. It is circular logic.

This is where we introduce Chebyshev interpolants, which we denote by p_n, and there are polynomial functions with similar properties to the Chebyshev series (that is, they satisfy similar theorems) we have seen so far but are also much easier to evaluate.

4.1.6 Chebyshev points

Recall that any interval $[a, b]$ can be scaled to $[-1, 1]$. Now, consider a circle on the complex plane with the centre at the origin $(0, 0)$, passing through -1 and 1 on the real line. This is a unit circle on the complex plane $z, |z| = 1$.

We delwill now consider "n" equally spaced out points on the circle. Consider "$n+1$" equally spaced angles $\{\theta_i\}_{0\leq i\leq n+1}$ from 0 to π. These are the $2n$th roots of unity in the closed upper halfplane.

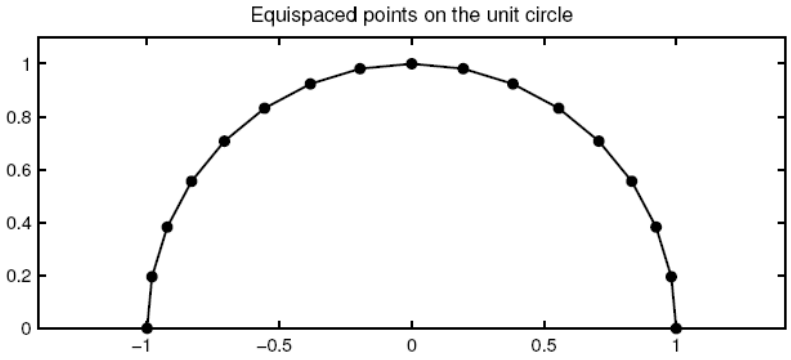


Figure 4.4: Equispaced points on the upper half of the unit circle(*Trefethen [28]*)

Let $\{z_i\}_{j=0}^{2n-1}$ be the $2n$-roots of unit, that is $2n$-equidistant points on the unitary circle. These are points on the complex plane that satisfy the equation $z = z^{2n}$ and are explicitly given by

$$\begin{aligned} z_j &= \exp(\frac{j\pi}{n}\cdot i) \\ &= \cos(\frac{j\pi}{n}) + i\sin(\frac{j\pi}{n}) \qquad \text{where } j = 0, 1, 2, \ldots, 2n-1 \end{aligned}$$

The Chebyshev points associated with the natural number n are the real parts of point z_i

$$\begin{aligned} x_j &= Re(z_j) \\ &= \frac{1}{2}(z_j + z_j^{-1}) \\ &= \cos(\frac{j\pi}{n}), \qquad\qquad 0 \leq j \leq n \end{aligned}$$

Figure *4.5*, shows examples of the Chebyshev points. Note that the Chebyshev points of the second kind always include the bounds of the domain (the end points).

Observation 4.1.6.1. *It is worth noting that Chebyshev points are the extrema of Chebyshev polynomials. This follows from the definition of recall.*

$$T_k(x) = \cos(k\theta) \Rightarrow -1 \leq T_k(x) \leq 1,$$

where $x \in [-1, 1]$ *and takes values* ± 1 *at the* $k+1$ *Chebyshev points*

Observation 4.1.6.2. *"Polynomial interpolants through equally spaced points have terrible properties. However, polynomial interpolants through Chebyshev pointsare excellent. It is clustering near the ends of the interval that makes the difference" - Lloyd N. Trefethen* [28].

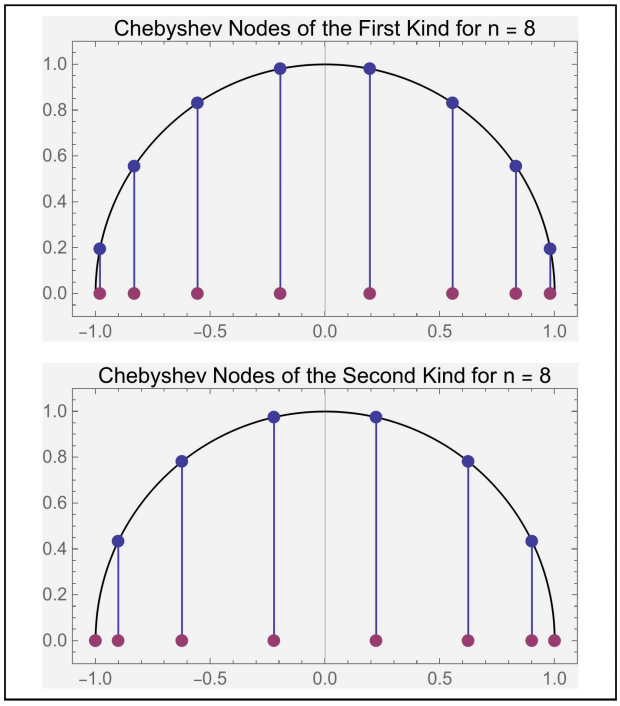


Figure 4.5: Equispaced points on the upper half plane ($n = 8$) for first and second kind, with the real parts projected, these red dots are the Chebyshev points. Source (*A. R. Kaiser* https://commons.wikimedia.org/wiki/File:ChebyshevNodes.jpg#/media/File:ChebyshevNodes.jpg)

4.1.7 Chebyshev interpolant

Let $\{f_i\}_{0\leq j\leq n}$, be a set of numbers that may or may not come from sampling a function $f(x)$ at Chebyshev points. Then there exists a unique polynomial p of degree n that interpolates this data, that is, $p(x_j) = f_j$, for each j. When we say "of degree n," we mean a degree less than or equal to n and we let $\mathcal{P}_n$ denote the set of all such polynomials:

$$\mathcal{P}_n = \{\text{polynomials of degree at most } n\}. \tag{4.3}$$

The existence and uniqueness of polynomial interpolants apply to any distinct set of interpolation points. In the case of Chebyshev points, we call this the polynomial *Chebyshev interpolant*.

Denote the polynomial interpolant to a continuous function f on the first $n+1$ Chebyshev points by p_n. Such a polynomial lies in $\mathcal{P}_n$, as such it may be expressed as a linear combination of the first $n+1$ Cheyshev polynomials

$$p_n(x) = \sum_{k=0}^{n} c_k \cdot T_k(x). \tag{4.4}$$

One of the main advantages of working with Chebyshev points is that polynomial interpolation on Chebyshev points has the same convergence properties as the Chebyshev projections.

4.1.8 Convergence of Chebyshev interpolants

The next theorem shows that the convergence of the Chebyshev interpolant is exponential for theanalytical functions.

Theorem 4.1.5. *Let f be an analytical function of the interval $[-1, 1]$. Considering its analytical continuation to the open Bernstein ρ-ellipse, it satisfies $|f(x)| \leq M$, for some M. Then, for each $n \geq 0$, we have*

$$\|f - p_n\|_\infty \leq \frac{4M\rho^{-n}}{\rho - 1} \tag{4.5}$$

where p_n is the n-th degree Chebyshev interpolant

4.1.9 Clenshaw Algorithm

Next, we consider two ways of evaluating Chebyshev tensors and their corresponding interpolants in a fast and stable manner: the Clenshaw algorithm and barycentric interpolation formula.

One major advantage of Equation *(4.4)* is that the coefficients C_k are easy to obtain. All that is required is to evaluate f at Chebyshev points and then apply the Fast Fourier Transform which reduces the complexity to $\mathcal{O}(n \log n)$ number of operations, which is less than n^2.

Technical Result 4.1.1. ***The Clenshaw Algorithm***
Set $\nu_{n+1} = 0$, $\nu_n = c_n$ and

$$\nu_k = 2x\nu_{k+1} - \nu_{k+2} + C_k,$$

where $k = n - 1, n - 2, \ldots, 0$.
Then $p(x) = \frac{1}{2}(c_0 + \nu_0 - \nu_2)$.

The c_k are the coefficients of the Chebyshev interpolants

$$p_n(x) = \sum_{k=0}^{n} c_k \cdot T_k(x)$$

Observation 4.1.9.1. *Three important aspects to note about the Clenshaw Algorithm:*

1. *the algorithm only uses the coefficients C_k,*
2. *Clenshaw has linear complexity of $\mathcal{O}(n)$, and*
3. *it is numerically stable.*

*Refer to (*Zeron and Ruiz [18]*) for details.*

4.1.10 Barycentric Interpolation Formula

Herein, we consider an alternative way to evaluate a Chebyshev interpolant, other than Fast Fourier Transform (which is of $\mathcal{O}(n \log n)$ effort) or the Clenshaw Algorithm (which is of $\mathcal{O}(n \log n)$ effort). There is a direct method that require only just $\mathcal{O}(n)$ workand is, not based on the series expansion, which is both elegant and numerically stable. It also has the advantage of generalizing sets of points other than Chebyshev. This is called the barycentric interpolation formula, introduced by (*Salzer [10]*).

The deduction of the barycentric interpolation formula begin with the ease of expressing polynomial interpolants in terms of Lagrange polynomials. For a set of points $\{x_0, \ldots, x_n\}$, the jth Lagrange polynomial is defined as the unique polynomial that takes the value 1 on x_j and 0 on x_i, for $i \neq j$, where $0 \leq i \leq n$.

$$l_j(x_i) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}.$$

Its expression is given by

$$l_j(x) = \frac{\prod_{i \neq j} (x - x_i)}{\prod_{i \neq j} (x_j - x_i)}. \tag{4.6}$$

Observation 4.1.10.1. *Some things to point out:*

- *Notice that the denominator is a constant term independent x.*
- *Also, when $x = x_i | i \neq j$, one of the numerator terms becomes 0 and hence $l_j(x_i) = 0$,*
- *and when $x = x_j$, the numerator is the same as the constant denominator term. Thus the product 1*

Regarding polynomial interpolation, one advantage of working with Lagrange polynomials is that an expression for the interpolant is obtained with only the values of the function at the points of interpolation.

Let the points of the interpolation be $\{x_0, x_1, \ldots, x_n\}$ and the values f on then $\{f_0, f_1, \ldots, f_n\}$. Then, p_n, the unique polynomial interpolation to f of degree at most n, on points $\{x_0, x_1, \ldots, x_n\}$ can be expressed as

$$p_n(x) = \sum_{i=0}^{n} f_i \cdot l_i(x). \tag{4.7}$$

In this form Equation *(4.7)* has an evaluation cost of $\mathcal{O}(n^2)$ [6]. However, some of the products present in $\prod_{i \neq j} (x - x_i)$ are repeated several when evaluating Equation *(4.7)*.

[6] It requires $\mathcal{O}(n)$ operations to evaluate $l_j(x)$ for each value of x, and then $\mathcal{O}(n)$ such evaluations must be added up in Equation *(4.7)*, giving a total operation count of $\mathcal{O}(n^2)$

We define the node polynomial $l \in \mathcal{P}_{n+1}$ for the given grid $\{x_0, \ldots, x_n\}$ by

$$l(x) = \prod_{k=0}^{n} (x - x_k) \tag{4.8}$$

so that l_j can be rewritten in terms of l

$$\begin{aligned} l_j(x) &= \frac{\prod_{i \neq j} (x - x_j)}{\prod_{i \neq j} (x_i - x_j)} \\ &= \frac{l(x)}{x - x_j} \cdot \frac{1}{\prod_{i \neq j} (x_i - x_j)} \\ &= \frac{l(x)}{x - x_j} \cdot \lambda_j \qquad \text{where we have defined } \lambda_j, \end{aligned}$$

and we get that

$$p_n(x) = l(x) \sum_{j=0}^{n} f_i \cdot \frac{\lambda_j}{x - x_j}. \tag{4.9}$$

This is the "first form of the barycentric interpolation formula" or "type 1 barycentric formula."

What is valuable here is that the dependence of x the sum on x is simple. If the weights $\{\lambda_j\}$ are known, Equation *(4.9)* produces each value $p(x)$ with only $\mathcal{O}(n)$ operations. Computing the weights requires $\mathcal{O}(n^2)$ operations, but this is done once and for all, independent of x_j. For special grids $\{x_j\}$ such as Chebyshev, the weights are known analytically and do not need to be computed at all. Therefore, we are left with $\mathcal{O}(n)$ operation if we choose a Chebyshev grid. If we use the result that the sum of all the Legrange polynomials $\{l_j\}$ at any point in the domain is one

$$\begin{aligned} \sum_{j=0}^{n} l_j(x) = 1 \quad &\Rightarrow \textstyle\sum_{j=0}^{n} l(x) \cdot \frac{\lambda_j}{x - x_j} = 1, \\ &\Rightarrow l(x) \cdot \textstyle\sum_{j=0}^{n} \frac{\lambda_j}{x - x_j} = 1. \end{aligned}$$

And therefore we have an expression for $l(x)$

$$l(x) = \frac{1}{\sum_{j=0}^{n} \frac{\lambda_j}{x - x_j}}.$$

And finally we have an expression for the interpolant,

$$\begin{aligned} p_n(x) &= l(x) \cdot \sum_{j=0}^{n} f_j \cdot \frac{\lambda_j}{x - x_j} \\ &= \frac{1}{\sum_{j=0}^{n} \frac{\lambda_j}{x - x_j}} \cdot \sum_{j=0}^{n} f_i \cdot \frac{\lambda_j}{x - x_j} \end{aligned}$$

4.1.11 Multivariate Extension and convergence results

The Chebyshev polynomial interpolation in 1-dimension, as seen in *Theorem 4.1.2* - 4.1.5, has a tensor based extension to the multivariate case that we present here, following closely *Gaß et al. [31]*, *Gaß [30]*, and [49].

Consider the interpolation of a valuation function $f(p)$ of D variables (thus we work in D-dimensional space, $p \in \mathcal{P}$, as defined below).

We expand our closed interval to be a hype-rectangular in parameter space

$$\mathcal{P} = [\underline{p}_1, \overline{p}_1] \times \ldots \times [\underline{p}, \overline{p}]$$

, with the appropriate linear transformation we can return to length 2 hypercube, that is,

$$\mathcal{P} = [-1, 1] \times \ldots \times [-1, 1].$$

Let $\overline{N} = (N_1, N_2, \ldots, N_D)$ with $N_i \in \mathbb{N}_0$ for $i \in \{1, \ldots, D\}$ the univariate interpolator extends logically to the multivariate case with $\prod_{i=1}^{D}(N_i + 1)$ summands and is given by

$$f(p) \approx \sum_{j \in J} c_j \cdot T_j(p) \quad , p \in \mathcal{P}, \tag{4.10}$$

where the summation index j is a multi-index with values

$$J = \left\{(j_i, \ldots, j_D) \in \mathbb{N}_0^D, \text{ where } j_i \in \{0, \ldots, N_i\}, \text{ for } i \in \{1, \ldots, D\}\right\}.$$

Therefore we can write Equation *(4.10)* the expanded form

$$f(p) \approx \sum_{j_1=0}^{N_1} \cdots \sum_{j_1=0}^{N_1} c_{(j_1,\ldots,j_D)} \cdot T_{(j_1,\ldots,j_D)} \quad , p \in \mathcal{P} \tag{4.11}$$

In the above, we used the basis function T_j for $j = (j_1, \ldots, j_D) \in J$ which is defined by

$$T_j(p_1, \ldots, p_D) = \prod_{i=1}^{D} T_{j_i}(p_i), \quad p \in \mathcal{P} \tag{4.12}$$

and the associated coefficients c_j with $j = (j_1, \ldots, j_D) \in J$ are given by

$$c_j = \left(\prod_{i=1}^{D} \frac{2^{\mathbb{1}_{\{0<j_i<N_i\}}}}{N_i}\right) \sum_{k_1=0}^{N_1}{}'' \cdots \sum_{k_D=0}^{N_D}{}'' f(p^{(k_1,\ldots,k_D)}) \prod_{i=1}^{D} \cos\left(j_i \pi \frac{k_i}{N_i}\right), \tag{4.13}$$

where $f(p^{(k_1,\ldots,k_D)})$ is a tensor containing the sampled values of f on the tensorized Chebyshev grid, $\sum''$ indicates that the first and last summands are halved, and the Chebyshev nodes p^k for multi-index $k = (k_1, \ldots, k_D) \in J$ are given by:

$$p^k = (p_{k_1}, \ldots, p_{k_D}) \tag{4.14}$$

with univariate Chebyshev nodes $p_{k_i} = \cos\left(\pi \frac{k_i}{N_i}\right)$ for $k_i = 0, \ldots, N_i$ and $i = 1, \ldots, D$. A set of D- variate Chebyshev nodes $p^{(k_1,\ldots,k_D)}$ for $D = 2$ and $N_1 = N_2 = 20$ is shown in Figure 4.6.

Observation 4.1.11.1. *The determination of Equation* (4.26) *increases the cost considerably as d increases. We term the increased computational cost with increasing dimensionality, which we will see later as exponential in d, as the curse of dimensionality. Futher details are covered in* Section 4.2.2.2.

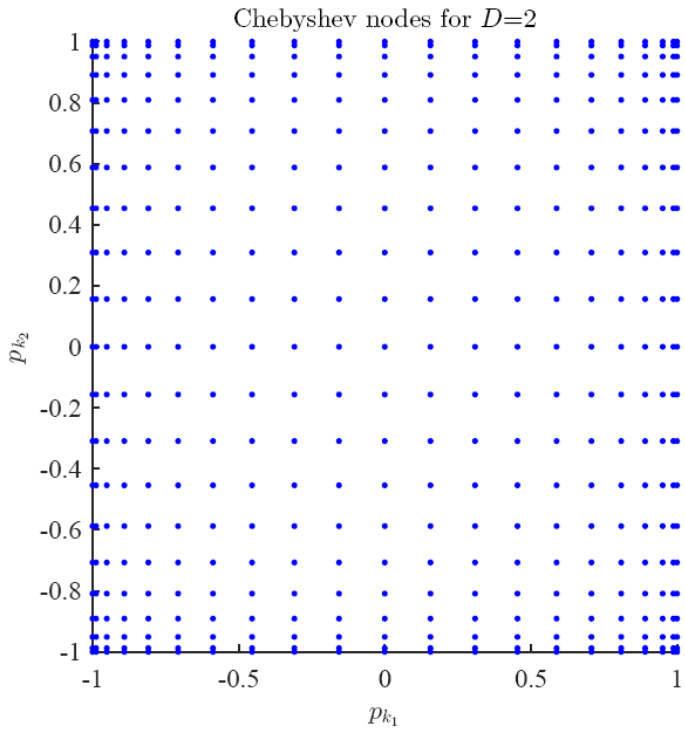


Figure 4.6: A set of D-variate Chebyshev points $p^k \in [-1,1]^D$ for $D = 2$ and $N_1 = N_2 = 20$. *Gaß et al.*[31]

We are building towards extend the convergence properties discussed in the univariate case, recall Equation *(4.1.3)* and Equation *(4.1.5)*, to the multivariate case. In what follows, we quote without proof or detailed discussion of the context the multivariate analogs of the theorems discussed in the onedimensional case that demonstrate the convergence properties of our Chebyshev interpolators in higher dimensions. For details of the definitions, derivations and formal details of the theory, especially our requirements, the reader is referred to *Gaß*[30] as a good introductory reference.

Theorem 4.1.6. *Let f be a D-dimensional analytic function defined on the hypercube $[-1,1]^D$. Consider its analytic continuation to a generalized Bernstein ellipse E_ρ with radius $\rho \in (1,\infty)^D$, where it satisfies $\|f\|_\infty \leq M$, for some M. Then, there exists a constant $C > 0$, such that*

$$\|f - p_n\|_\infty \leq 2^{\frac{d}{2}+1} \cdot M \cdot \left(\sum_{i=1}^{D} \rho_i^{-2N_i} \prod_{j=1}^{D} \frac{1}{1-\rho_j^{-2}} \right)^{\frac{1}{2}} \tag{4.15}$$

and it follows that

$$\|f - p_n\|_\infty \leq C\rho^{-\underline{N}} \tag{4.16}$$

where $\rho = \min_{1\leq i\leq D} \rho_i$ *and* $\underline{N} = \min_{1\leq i\leq D} N_i$ *The collection of values* ρ_i *define the radius of the generalised Bernstein ellipse* E_p*, and the values* N_i *the size of he Chebyshev mesh in each dimension*

Therefore, this theorem states that we obtain an exponential error convergence (in N) when the same number of Chebyshev nodes N is chosen in each dimension of the parameter space.

4.1.12 Derivative approximation

Suppose that we have a process or technique that yields an approximation $\tilde{f}$ for any given function f. Regardless of the level of accuracy, it does not generally follow that *derivatives* (math) of $\tilde{f}$ will provide good proxies for the derivative of f.

Remark 4.1.2. *Spline interpolation can be highly accurate, especially when compared to higher-order polynomial interpolation, which can suffer from oscillatory behaviour. However, while spline interpolation provides a good approximation of the function itself, the derivatives of the spline can be less accurate because of the piecewise nature of splines, which can introduce discontinuities or inaccuracies in higher-order derivatives.*

Deep Neural Networks (DNNs) can approximate functions with high accuracy, particularly when trained on large data sets. However, similar to splines, DNNs cannot accurately approximate the derivatives of the functions they model. This is due to the complex, non-linear nature of DNNs, which can lead to issues such as vanishing or exploding gradients during training.

More on this is covered in [18].

repIt is fFortunately, another property of the Chebyshev interpolant's derivative converges to the derivative of the subject function in a manner similar to that observed in *Theorem 4.1.5*. This is described in *Theorem 4.1.7* below (*Zeron and Ruiz [18]*). Therefore, if we can efficiently compute the derivative of the Chebyshev interpolant, we have also found a very good approximation for the derivative of our subject function without having to repeat the Chebyshev procedure or require more Chebyshev points (this will become more apparent in what follows).

Theorem 4.1.7. *Let* f *be an analytical function of the interval* $[-1, 1]$*. Then for each* $k \geq 0$*, the Chebyshev interpolants* p_n *satisfy*

$$\left\| f^k - p_n^k \right\|_\infty = \mathcal{O}(\rho^{-n}) \tag{4.17}$$

as $n \to \infty$*, where* f^k *and* p^k *are the* k*-th derivatives of* f *and* p_n*, respectively and* ρ *is the radius of the open Bernstein ellipse* E_ρ *onto which* f *can be analytically extended.*

We now consider the work required to compute the expressions for the derivatives of Chebyshev interpolants. We shall see that the generation and evaluation of the derivatives of Chebyshev interpolants elegantly decompose into the multiplication of a matrix known as the Chebyshev differentiation matrix with vectors of f on Chebyshev points. Consider the

kth derivative p_n^k. Because p_n is a polynomial, then p_n^k is also a polynomial. Moreover, the polynomial degree of p_n^k is less than that of p_n. This implies that the unique polynomial that interpolates p_n^k on the interpolating point $\{x_0, x_1, \ldots, x_n\}$ is p_n^k itself. To find a method of computing $p_n^k(x_i)$, consider the interpolant p_n to f expressed in terms of the Lagrange polynomials

$$p_n(x) = \sum_{j=0}^{n} f_i \cdot l_j(x) \quad \text{where } f_i = f(x_i). \tag{4.18}$$

Then the derivative of p_n' at x_i is given by

$$p_n'(x_i) = \sum_{j=0}^{n} f_i \cdot l_j'(x_i). \tag{4.19}$$

Using the values of $l_j'(x_i)$, for all i, j such that $0 \leq i, j \leq n$, the vector $\{p_n'(x_0), p_n'(x_1), \ldots, p_n'(x_n)\}$ can be represented as the product of matrix $\mathcal{D}$ by $\vec{f} = \{f_0, \ldots, f_n\}$ where $(\mathcal{D})_{ij} = l_j'(x_i)$, that is

$$p_n'(x_i) = \mathcal{D}\vec{f}$$

.

Note that this matrix is completely determined by the points $\{x_0, x_1, \ldots, x_n\}$ and nothing else. This implies that $\mathcal{D}_n$ can be computed with no reference to any particular function f', before any approximation calculation is performed. The information of the function f being approximated is in the form of its value $\{f_0, f_1, \ldots, f_n\}$ on $\{x_0, x_1, \ldots, x_n\}$. The latter is needed only when the values of p_n' on the grid points are required to compute or generate an interpolant (in this case the interpolant itself).

We now consider the tasks of computing the interpolating grid $\{x_0, x_1, \ldots, x_n\}$ and the corresponding $l_j'(x_i)$. In the case where $\{x_0, x_1, \ldots, x_n\}$ are Chebyshev points, matrix $\mathcal{D}$ reduces to a very simple expression, given in the following theorem:

Theorem 4.1.8. ***Chebyshev differentiation matrix***

For $N \geq 1$, the entries of the Chebyshev differentiation matrix $\mathcal{D}_N$ are given by

$$(\mathcal{D}_N)_{00} = \frac{2N^2+1}{6} \qquad (\mathcal{D}_N)_{NN} = \tfrac{2N^2+1}{6}$$

$$(\mathcal{D}_N)_{jj} = \frac{-x_j}{2(1-x_j^2)}, \qquad j = 1, \ldots, N-1$$

$$(\mathcal{D}_N)_{ij} = \frac{c_i}{c_j}\frac{(-1)^{i+j}}{x_i - x_j}, \qquad j \neq i, \text{ and } i, j = 0, 1, \ldots, N$$

where:

$$c_i = \begin{cases} 2, & i = 0 \text{ or } N \\ 1, & \text{Otherwise} \end{cases}$$

This is better represented visually, as shown below. In the diagram in Figure *4.7*, the jth column of $\mathcal{D}_N$ contains the derivative of the degree N polynomial interpolant $p_j(x)$ to the delta function supported at x_j, sampled at grid points x_i. For more details and examples on the Chebyshev derivatives, refer to [27].

$$D_N = \begin{array}{|c|c|c|} \hline \frac{2N^2+1}{6} & 2\frac{(-1)^j}{1-x_j} & \frac{1}{2}(-1)^N \\ \hline -\frac{1}{2}\frac{(-1)^i}{1-x_i} & \frac{(-1)^{i+j}}{x_i-x_j} \quad \frac{-x_j}{2(1-x_j^2)} \quad \frac{(-1)^{i+j}}{x_i-x_j} & \frac{1}{2}\frac{(-1)^{N+i}}{1+x_i} \\ \hline -\frac{1}{2}(-1)^N & -2\frac{(-1)^{N+j}}{1+x_j} & -\frac{2N^2+1}{6} \\ \hline \end{array}$$

Figure 4.7: Visual representation of Chebyshev differentiation matrix (*Trefethen [27]*)

Once the matrix $\mathcal{D}_n$ and vector $\mathcal{D}_n\vec{f}$ have been computed we have an expression for the derivative p'_n. if the grid of points are Chebyshev points, then p'_n is a good approximator to f' because of *Theorem 4.1.7*. Given that we now have have approximation points of $\{p'_n(x_0), p'_n(x_1), \ldots, p'_n(x_n)\}$ at the Chebyshev points the Barycentric interpolation formula can be employed (the procedure starts over again) to give us a very efficient and stable method of evaluating p'_n.

We iterated the process described for derivatives of the higher order. Obtaining the values $\{p^k_n(x_0), p^k_n(x_1), \ldots, p^k_n(x_n)\}$ is reduced by multiplying $\mathcal{D}^k_n$ by $\{f_0, f_1, \ldots, f_n\}$.

Remark 4.1.3. *Once again matrix $\mathcal{D}^k$ can be obtained before any function f is considered or computed.*

Furthermore, obtaining an expression for p^k_n involves only linear algebra operations and there are many efficient implementations of these operations in most computational software packages. Therefore, once an interpolant or tensor is obtained for $f$the corresponding proxies for its derivatives can be determined very efficiently and rapidly.

4.1.13 Derivatives in high dimension

In this section, we present the main results derived by *Gaß et al*[31] and describe its practical application to computations.

Theorem 4.1.9. *Let f be a D-dimensional analytic function defined as $\mathcal{A} = [-1, 1]^D$. Consider its analytical continuation to the generalized Bernstein ellipse E. Then for every $m \in \mathbb{N}$, there exists a constant K_m such that*

$$\|f - p_n\|_{C^l(\mathcal{A})} \leq K_m \cdot n^{-m} \|f\|_{C^{2(l+1)+d+m}(\mathcal{A})},$$

where, the weighted Sobolev norm is,

$$\|u\|_{C^l(\mathcal{A})} = \max_{|\alpha| \leq l} \max_{x \in \mathcal{A}} |\partial^\alpha u(x)|.$$

The theorem statesthat the partial derivatives of the Chebyshev interpolants p_n to an analytic function f of dimension greater than one, converge polynomially to the partials of f.

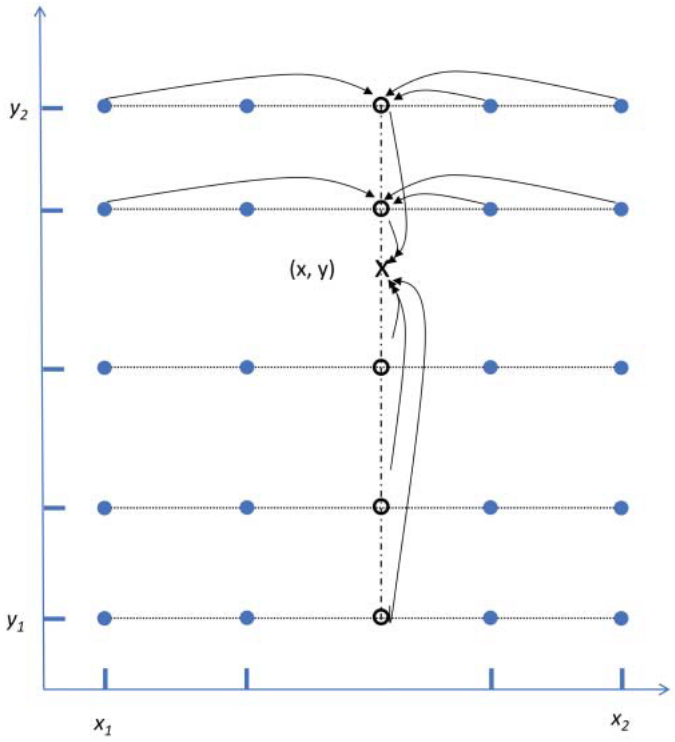


Figure 4.8: Computation on 2dimensional Chebyshev grid (*Zeron and Ruiz [18]*)

4.2 Applications of Chebyshev techniques

Applications of Chebyshev techniques will be discussed as follows:

- Low dimensional techniques

1. Composition techniques

- High dimensional techniques– combating the curse of dimensionality
  1. Tensors in Tensor-Train (TT) format

4.2.1 Composition Techniques

Consider the process diagram in Figure *4.9*, which maps the high level key steps of a simulation based risk engine. The modelling starts in the state space, where it is assumed that a factor, say n, is the driver of uncertainty or stochastic behaviour of (observable) market factors,say N in total, which are then inputs to calculate prices and/or univariate risk metrics. Examples include the state space, which could be a number of factors in, for example a Hull-White short rate model (in general, the parameters of the model are known and fixed at this stage of the computation), and the market space may comprise the resulting yield curves, zero coupon bonds, volatility surfaces, and the output space may comprise valuations that are used to determine risk metrics.

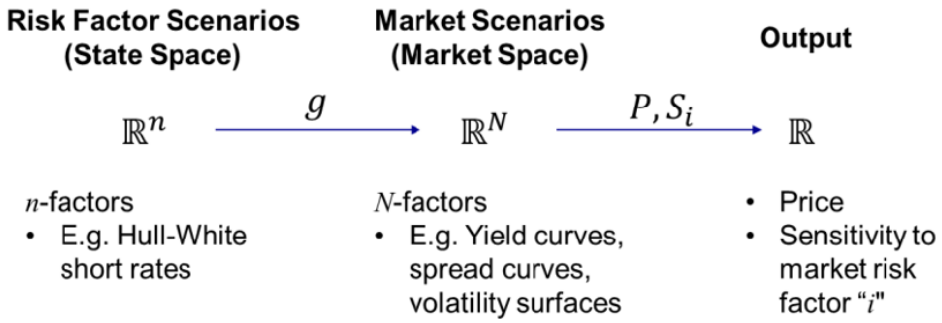


Figure 4.9: The figure displays the mathematical maps involve in the scenario generation, valuation and metric calculation parts of the generic risk engine (*Zeron and Ruiz [35]*)

. In this section we consider in the main functions P, usually a pricing function that needs to called many times in the risk management processes and also g, usually the market factor generating function such yield curves or zero coupon bonds. The idea is to approximate these functions using one of the techniques, such as composition and proxy functions, and also reduce the dimension from n to a much lower value k. We assume $P : \mathbb{R}^n \to \mathbb{R}$ and so we want a way to parametrise X in terms of Y which is of dimension k.

Let P be a function of domain $\mathbb{R}^n$. Assume there is a set $X \subset \mathbb{R}^n$ in the domain of P. Suppose there is a parametrisation $g : Y \subset \mathbb{R}^k \to \tilde{X}$, where $\tilde{X}$ is either X or closely approximates X and $k < n$. Then

$$Y \subset \mathbb{R}^k \xrightarrow{g} \tilde{X} \subset \mathbb{R}^n \xrightarrow{P} \mathbb{R}$$

We can construct an f composition function g and P (and use f to build a proxy function using Chebyshev interpolants). The composition function f will be in the form

$$f(y) = P(g(y)) \tag{4.20}$$

The map of f is given by

$$Y \subset \mathbb{R}^k \xrightarrow{g} \tilde{X} \subset \mathbb{R}^n \xrightarrow{P} \mathbb{R} \qquad f \tag{4.21}$$

The resulting function f has the following properties:

1. every point in $\tilde{X}$ has at least on in Y s.t. $g(Y)$, that is $\forall x \in \tilde{X} P(x) = f(y)$
2. If k is sufficiently low, then the approximation in part 1 will not be affected by the curse of dimensionality when applied to f

4.2.2 Tensors in tensor-train (TT) format

One of the main problems affecting tensors is the so-called *curse of dimensionality*. This is because as the dimension of the function being approximated increases, the number of Chebyshev grid points increases exponentially.

In this section, we introduce the basics of tensors and the tensor train format. We do not cover the full set of techniques required to deploy high-dimensional tensorized Chebyshev interpolations. The reader is referred to Completion Algorithms in (*Glau et al [23]*) and Tensor Extension Algorithms in (*Zeron and Ruiz [18]*) for detailed discussions on the omitted techniques.

4.2.2.1 Tensors

A tensor is a very basic mathematical object and we will see herein it is closely related to polynomial interpolants, and we will often refer to them interchangeably.

Definition 4.2.1. *A tensor $\mathcal{A}$, consists of a grid of points in the Euclidean space along with a collection of real values, one per point in the grid.*

A d-dimensional tensor, with dimension size $n_i | i = 1, \ldots, d$, is a multidimensional array

$$\mathcal{A} = \left[\mathcal{A}_{(i_1,\ldots,i_d)}\right] \qquad i_k \in (1, 2, \ldots, n_K)$$

which has d indices that represent the position of each element in the tensor. Therefore the indices of the tensor represent the a Cartesian product of the elements in each dimension.

Example 4.2.1. *So for the case where:*

- $d = 0$ *we have a scalar (a number)*
- $d = 1$ *we have a vector (visually a column vector),*
- $d = 2$ *we have a matrix (visually an object with rows and columns),*

- *$d = 3$ we have a tensor or vector of matrices (visually a cube or rectangular prism)*
- *$d > 4$ we have a tensor (visually a hyper-rectangle in d-dimensions)*

Note 4.2.1. ***The curse of dimensionality for general tensors***
For general tensors, the curse of dimensionality emanates from the fact that its size (total number of elements or nodes) grows exponentially with the number of dimensions. To see this, consider again the general d-dimensional tensor (where n_i is the maximum number of indices in each axis) . The size is given by

$$size(\mathcal{A}) = n_1 \times n_2 \times \ldots \times n_d \tag{4.22}$$

$$\implies size(\mathcal{A}) \geq n_{\min}^d, \qquad *n_{\min} = \min\{n_1, n_2, \ldots, n_d\}* \tag{4.23}$$

Therefore, the size of the tensor is bounded below, and similarly bounded above, by an exponential function in d. We say that the size of the tensor is of order n^d, where $n = \max\{n_1, n_2, \ldots, n_d\}$.

For example, a set of points $x_0, x_1, \ldots, x_n$ on the real line and a set of real values $f(x_0), f(x_1), \ldots, f(x_n)$ constitutes a tensor $\mathcal{T}$ of dimension 1. Polynomial interpolants are closely related to tensors. This stems from the well known result of numerical analysis called polynomial interpolation, as outlined below.

Definition 4.2.2. *Let $\mathcal{T}$ be a tensor, with a set of points $x_0, x_1, \ldots, x_n$ in Euclidean space and the associated real values $f_0, \ldots, f_n$. A polynomial interpolant to $\mathcal{T}$ is a polynomial $p \in \mathcal{P}$ such that $p(x_i) = f_i, \forall i, \quad 0 \leq i \leq n$.*

Polynomial interpolation is the interpolation of a given bivariate data set by the polynomial of the lowest possible degree that passes through the points. Note that there is always a unique polynomial that passes through the given points, commonly presented by Lagrange or Newton interpolating polynomials.

Obtaining a tensor given a function f is performed as follows: *the set of points $x_0, x_1, \ldots, x_n$ on its domain and the values $f(x_0), f(x_1), \ldots, f(x_n)$ then there exists a unique polynomial of degree at most n and we associate this polynomial with the tensor. Therefore, given a function f and a grid of Chebyshev points, there is a tensor, denoted by CT, and a corresponding polynomial interpolant p_n that can be used as proxy for f.*

4.2.2.2 Curse of dimensionality for Chebyshev Tensors

recall Equation *(4.10)* which has the expanded form

$$f(p) \approx \sum_{j_1=0}^{N_1} \cdots \sum_{j_1=0}^{N_1} c_{(j_1,\ldots,j_D)} \cdot T_{(j_1,\ldots,j_D)} \quad , p \in \mathcal{P} \tag{4.24}$$

In the above, we used the basis function T_j for $j = (j_1, \ldots, j_D) \in J$ which is defined by

$$T_j(p_1, \ldots, p_D) = \prod_{i=1}^{D} T_{j_i}(p_i), \quad p \in \mathcal{P} \tag{4.25}$$

and the associated coefficients c_j with $j = (j_1, \ldots, j_D) \in J$ are given by

$$c_j = \left(\prod_{i=1}^{D} \frac{2^{\mathbb{1}_{\{0<j_i<N_i\}}}}{N_i} \right) \sum_{k_1=0}^{N_1}{}'' \cdots \sum_{k_D=0}^{N_D}{}'' f(p^{(k_1,\ldots,k_D)}) \prod_{i=1}^{D} \cos\left(j_i \pi \frac{k_i}{N_i} \right), \tag{4.26}$$

where $f(p^{(k_1,\ldots,k_D)})$ is a tensor containing the sampled values of f on the tensorized Chebyshev grid, with univariate Chebyshev nodes $p_{k_i} = \cos\left(\pi \frac{k_i}{N_i}\right)$ for $k_i = 0, \ldots, N_i$ and $i = 1, \ldots, D$.

This tensor $f(p^{(k_1,\ldots,k_D)})$ in Equation *(4.26)* is of order D and size $(n_1+1) \times \cdots \times (n_D+1)$. The interpolation procedure first requires computing each entry of this tensor using the reference method. This becomes expensive when the interpolation order (Chebyshev anchor points) and the dimension D increases.

Example 4.2.2. *Suppose we have a function $f : \mathbb{R}^D \mapsto \mathbb{R}$, and that we would like to use a Chebyshev interpolant to build a proxy function (approximator). We choose to have five Chebyshev points evaluated in each dimension; the size of the Chebyshev grid or tensor is shown in the table below:*

Chebyshev points/dimension = 5					
Dimension	*1*	*3*	*7*		*12*
Grid size	*5*	*125*	*78125*	*244 140*	*625*

4.2.2.3 Tensor rank decomposition

These methods are an extension of rank decomposition techniques for matrices.

Remark 4.2.1. ***Rank decomposition***
Recall for a given matrix $A \in \mathbb{R}^{m \times n}$, it can be shown that this matrix can be decomposed into the product of two matrices CF, where $C \in \mathbb{R}^{m \times r}$ and $F \in \mathbb{R}^{r \times n}$, and $r = rank(A)$ the rank of A.

We write matrix multiplication simply as

$$A = CF$$

or in expanded summation (element-wise) form

$$A(i_1, i_2) = \sum_{\alpha=1}^{r} C(i_1, \alpha) F(\alpha, i_2)$$

Therefore, all the information stored in the full matrix (A), which is $m \times n$ elements can be recovered by the operation of simple matrix multiplication (which the computers

are very efficient performing) of two matrices containing $m \times r$ and $r \times m$, so the total storage costs goes down from $m \times n$ to $m \times r + r \times n = (m + n) \times r$.

To obtain a better sense of the reduction in information requirements, we set $m = n$, so that A is square and has size n^2. Thus, the reduced rank will have a size $2nr$, a linear function in n as opposed to a quadratic or from the perspective of variable dimensions, where matrix A has a size in which the row length increase to to the power of the dimension whereas the reduced form matrix sizes are linear in dimension size and a new variable called the rank of the matrix.

Therefore, if we ensure that the rank is low, we are likely to achieve an effective reduction in the tensor storage issue.

The grid of points of the d-dimensional tensor $\mathcal{X}$ is defined as follows:

Note 4.2.2. *Let X_i be a set of points of dimension one, where $1 \leq i \leq d$. Then the Cartesian product[7] of the one-dimensional grid point X_i,*

$$M = X_1 \times X_2 \times \cdots \times X_d,$$

defines a mesh of points in $\mathbb{R}^d$ and represents all possible labels for each element of our tensor $\mathcal{X} = [\mathcal{X}_m]|m \in M$.

If X_i has n_i points, then tensor $\mathcal{X}$ has a total of $n_1 \times n_2 \times \ldots n_d$ points. We denote the space of tensors on M by $\mathbb{R}^{n_1 \times n_2 \times \ldots \times n_d}$ so $\mathcal{X}$ lives in an $n_1 \times n_2 \times \ldots n_d$-dimensional space.

With this basic idea in mind we can now define class tensors of interest to us. These are tensors that conform to rank decomposition, more on this little later, for now we start with a proposition.

Proposition 4.2.1. *Given a grid of points in d-dimensions $(n_1, n_2, \ldots, n_d)$. We define tensor $\mathcal{A}$, at the grid point $(j_1, \ldots, j_d)$. Let U_i be a vector of real values with length n_i, where $1 \leq i \leq d$, then $\mathcal{A}$ can be written as follows:*

$$\mathcal{A}_{(j_1,\ldots,j_d)} = U_1(j_1) \times U_2(j_2) \times \ldots \times U_d(j_d) \tag{4.27}$$

where $\times$ is the normal multiplication of the number.

Therefore, we propose that it represents the tensor value $\mathcal{A}_{(j_1,\ldots,j_d)}$ at each grid point as product of numbers that come from vectors U_i that are yet to be determined vectors .

This is a subclass of the larger class of general tensors. Normally a tensor with dimension d and $n_1 \times n_2 \times \ldots n_d$ grid points requires the storage $\mathcal{O}(n^d)$ elements, where $n = \max\{n_1, n_2, \ldots, n_d\}$. A tensor such as $\mathcal{A}$, which is fully defined by vectors U_i, requires only $\mathcal{O}(dn)$. The the fundamental difference is that an exponential growth object is reduced to a linear growth object.

Now, we generalize even further, the entries of the vectors U_i can be thought of as matrices of rank [8] 0. We generalize this by making $\{U_i\}_{i=1}^{d}$ – also known as the *cores* of $\mathcal{A}$– vectors of two dimension (or matrices). A core U, then has the property that $U(i)$ is a matrix of rank $r_1 \times r_2$, for some values r_1 and r_2.

To keep the elements of $\mathcal{A}$ as scalars (inline with Equation *(4.27)*) we must ensure that the results of the matrix multiplications are scalar. For matrix multiplication , we need the columns dimensions/rank of subsequent cores to match. Therefore, each U_i must take the form of a $r_{i-1} \times r_i$ matrix and U_{i+1} will be $r_i \times r_{i+1}$. For the complete product of the U_i ($\prod U_i$) to yield a scalar (rank 1 tensor) we must have that $r_1 = r_d = 1$ i.e. U_0 and U_d must be vectors.

Bringing it altogether we have:

Definition 4.2.3. ***Tensor train (TT) format***
A tensor $\mathcal{A}$ in $\mathbb{R}^{n_1 \times \cdots \times n_d}$ expressed in TT format with ranks $(r_0, \ldots, r_d)$ consists of the following:

1. *a set of vectors of matrices $\mathbf{U^{(i)}}$, $1 \leq i \leq d$, these are the TT cores, (this is 3-d tensor, currently all three indices are suppressed)*
2. *$\mathbf{U^{(i)}}_{(j)}$ is a matrix of rank $r_{i-1} \times r_i$ for $1 \leq j \leq n_i$ (index j represents a row index of the vector and we suppress the two matrix indices) and*
3. *the first and last rank are one $r_0 = r_d = 1$.*

This is sufficient information to recover the whole tensor in the following way

$$\mathcal{A}_{(j_1,\ldots,j_d)} = \mathbf{U^{(1)}}_{(j_1)} \cdots \mathbf{U^{(d)}}_{(j_d)}. \tag{4.28}$$

Or in expanded matrix-multiplication notation or canonical decomposition form

$$\mathcal{A}_{(j_1,\ldots,j_d)} = \sum_{\alpha}^{R} \mathbf{U^{(1)}}_{(j_1,\alpha)} \cdots \mathbf{U^{(d)}}_{(j_d,\alpha)}. \tag{4.29}$$

The minimal possible R is the canonical rank of the tensor (this is the largest rank of the core matrices).

Note that TT-format is a form of data compression.

Network diagrams Tensors in TT-format are often presented as tensor network diagrams. Figure *4.10* shows an example in which the dimension of the tensor is five. The core corresponding to the i-th dimension is denoted by $U^{(i)}$.

$$\mathcal{A}_{(i_1,i_2,\cdots,i_5)} = \sum_{k_1=1}^{r_1} \cdots \sum_{k_4=1}^{r_4} \mathbf{U^{(1)}}_{(i_1,1,k_1)} \mathbf{U^{(2)}}_{(i_2,k_1,k_2)} \cdots \mathbf{U^{(5)}}_{(i_5,k_4,1)}. \tag{4.30}$$

[8] *Rank* – heuristically its the smallest number rows and columns that are linearly independent. Essentially, the rank tells us the dimension of the vector space spanned by its rows or columns. It sets a limit on how much the matrix can be simplified without loss of information

The $\mathbf{U}^{(i)}$ is defined by n_i elements and each is a core matrix with rank $r_{i-1} \times r_i$, except for the first and last vectors ,that is, with ranks $1 \times r_1$ and $r_4 \times 1$ respectively. Third order tensors $\mathbf{U}$

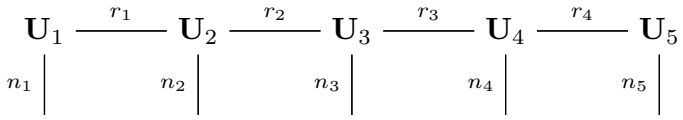


Figure 4.10: Tensor network diagram of TT decomposition for a tensor of order $d = 5$. Excerpt from (*Glau et al [23]*)

Therefore, we can generalize that every entry $\mathcal{A}(i_1, i_2, \cdots, i_d)$ of a tensor can be expressed as a product of d matrices as follows:

$$\mathcal{A}_{(i_1,i_2,\cdots,i_d)} = U^{(1)}_{(i_1)} U^{(2)}_{(i_2)} \cdots U^{(d)}_{(i_d)},$$

where $U^{\mu}_{(i_\mu)}$ is an $r_{\mu-1} \times r_\mu$. For each $\mu = 1, \cdots, d$, one can then collect the n_μ matrices $U^{(\mu)}_{(i_\mu)}$, $i_\mu = 1, 2, \cdots, n_\mu$ into a third order tensor $\mathbf{U}^{(\mu)}$ of size $r_{\mu-1} \times n_\mu \times r_\mu$. These tensors are called *TT cores* and, by construction, we have

$$\mathcal{A}_{(i_1,i_2,\cdots,i_d)} = \sum_{k_1=1}^{r_1} \cdots \sum_{k_{d-1}=1}^{r_{d-1}} \mathbf{U^{(1)}}_{(i_1,1,k_1)} \mathbf{U^{(2)}}_{(i_2,k_1,k_2)} \cdots \mathbf{U^{(d)}}_{(i_d,k_{d-1},1)}. \tag{4.31}$$

Note 4.2.3. *The cost of $\mathcal{O}(n^d)$ associated with the ordinary tensor is reduced to $\mathcal{O}(dnr^2)$, where r is $\max\{r_1, r_2, \ldots, r_d\}$ and n are $\max\{n_1, n_2, \ldots, n_d\}$.*

Worked example. To ensure that we have an intuitive understanding of what we have considered in this section, let us consider a worked example from [2].

Example 4.2.3. *Consider a 3-dimensional tensor with elements representing the sum of the indices, as follows*

$$\mathcal{A} = \left[\mathcal{A}_{(i_1,i_2,i_3)} = i_1 + i_2 + i_3\right]$$

where

- $i_1 \in \{1, 2, 3\}$ *and hence* $n_1 = 3$,
- $i_2 \in \{1, 2, 3, 4\}$ *and hence* $n_2 = 4$, *and*
- $i_3 \in \{1, 2, 3, 4, 5\}$ *and hence* $n_3 = 5$.

The size of the tensor $\mathcal{A}$, is $n_1 \times n_2 \times n_3 = 60$. Consider the TT-format decomposition of rank 2, given by

$$\mathcal{A}(i_1, \ldots, i_d) = U_1(j_1) U_2(j_2) U_3(j_3)$$

where the U matrices have the form

$$U_1(i_1) = \begin{bmatrix} i_1 & 1 \end{bmatrix}, \quad U_2(i_2) = \begin{bmatrix} 1 & 0 \\ i_2 & 1 \end{bmatrix}, \quad U_3(i_3) = \begin{bmatrix} 1 \\ i_1 \end{bmatrix}$$

Let's check the matrix multiplication:

$$\begin{aligned} U_1(i_1)U_2(i_2)U_3(i_3) &= \begin{bmatrix} i_1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ i_2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ i_1 \end{bmatrix} && (4.32) \\ &= \begin{bmatrix} i_1 + i_2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ i_3 \end{bmatrix} && (4.33) \\ &= i_1 + i_2 + i_3 && (4.34) \\ &= A_{(i_1,i_2,i_3)} && (4.35) \end{aligned}$$

Therefore, we determined how each element of the tensor is recovered. Thus if we expand the matrices in full detail we have

$$\begin{aligned} U_1 &= \left(\begin{bmatrix} 1 & 1 \end{bmatrix}, \begin{bmatrix} 2 & 1 \end{bmatrix}, \begin{bmatrix} 3 & 1 \end{bmatrix} \right) && (4.36) \\ U_2 &= \left(\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 3 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 4 & 1 \end{bmatrix} \right) && (4.37) \\ U_3 &= \left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \begin{bmatrix} 1 \\ 4 \end{bmatrix}, \begin{bmatrix} 1 \\ 5 \end{bmatrix} \right) && (4.38) \end{aligned}$$

As we can see above, the TT-format uses 32 elements ($2 \times n_1 + 4 \times n_2 + 2 \times n_3$) to fully describe tensor $\mathcal{A}$ which has 60 elements.

4.2.2.4 Basic tensor operations

Some operations can be performed quite cheaply in the TT format for tensors with low TT ranks.

Let us first consider the inner product of two tensors $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ defined as

$$\langle \mathcal{A}, \mathcal{B} \rangle = \langle \mathrm{vec}(\mathcal{A}), \mathrm{vec}(\mathcal{B}) \rangle = \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} \mathcal{A}_{(i_1,\ldots,i_d)} \mathcal{B}_{(i_1,\ldots,i_d)}, \tag{4.39}$$

where $\mathrm{vec}(\cdot)$ stacks the entries of the tensor into a long vector. The corresponding tensor network diagram when $\mathcal{X}$ and $\mathcal{Y}$ are both in TT decomposition is shown in Figure *4.11*. It can be seen that the summations in Figure *4.11* become contractions between the TT cores of $\mathcal{X}$ and $\mathcal{Y}$. By carrying out these contractions of cores from left to right, the cost of evaluating the inner product is reduce from $O(n^d)$ to $O(dnr^3)$, where r denotes the maximum of all the involved TT ranks [23].

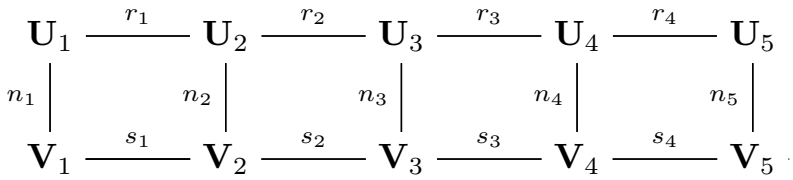


Figure 4.11: Tensor network diagram of inner product of two tensors of order $d = 5$ in TT decomposition. Excerpt from (*Glau et al [23]*)

Chapter 5

Results

In this section, we review the results we obtained from running various tests that employ the techniques discussed in the theoretical framework sections. Our results are as follows:

- Proxy function tests in low dimensions
  - pricing models
  - counter-party credit risk models I
  - counter-party credit risk models II
- Proxy function tests in high dimensions
  - pricing models
- Sensitivity approximation
  - pricing models
- Summary and discussion of results

Note 5.0.4. *All algorithms were implemented in Python and run on a laptop with the following specifications.*

Processor	:	*Intel(R) Core(TM) i7-8665U CPU 1.90GHz, 2112 Mhz, 4 Core(s), 8 Logical Processor(s).*
Operating system	:	*Microsoft Windows 11 Enterprise*
Physical Memory (RAM)	:	*16.0 GB*

The MoCaX Suite software package was downloaded from www.mocaxintelligence.com and includes:

- *The MoCaX library – which has functionality for the full Chebyshev Tensors, is used in low dimensional problems.*
- *MoCaX Extend library – functionality for Chebyshev Tensors in TT-format and includes supporting Rank Adaptive Algorithms, which are used for high dimensional problems.*

Further details on the software and use guide can be found in [18].

5.1 Proxy function tests in low dimensions

We start by looking at the composition method that we apply to replicating pricing functions and then on components of complex computation such as CVA or PFE calculation.

5.1.1 Composition method: pricing models

In [35] the authors demonstrated the power of Chebyshev interpolants as proxy functions against a sample of analytical and numerical pricing models that are common in quantitative finance. In Table *5.1*, we have an excerpt of the table of results that shows the standard run times of the pricing models using *QunatLib*[1] and contrasts this against Chebyshev build and run times, and accuracy measured by maximum absolute error. The test results in the excerpts were based on building Chebyshev proxy functions for 1-dimensional pricing functions and approximately 11 anchor points[2]. We note that the key take away is that the Chebyshev runs times are considerably faster than the subject pricing models – run speed multipliers ranging from 100 to 6 million times faster– and have a very high level of accuracy, ranging from 3 to 10 decimal places (10^{-15}).

Pricing function	Pricing Method	QuantLib Run Time (ms)	Chebyshev Build Time (ms)	Chebyshev Run Time (ms)	Accuracy	Chebyshev Run Speed Multiplier
IRS	Analytic	0.214	2.675	0.000103	10^{-15}	2,088
European Option	Analytic (BS)	0.013	0.209	0.000127	10^{-6}	110
American Option	Monte Carlo	23.103	247.117	0.000096	10^{-6}	239,668
Bermudan Swaption	Tree	318.99	3642.12	0.000127	10^{-5}	2,511,737
Barrier Option	Analytic	0.024	0.3	0.000125	10^{-4}	192
Barrier Option	Monte Carlo	601.919	6590.522	0.000103	10^{-3}	5,843,883

Table 5.1: The figure displays a table excerpt from (*Zeron and Ruiz [35]*) displaying the results runtime and accuracy results of Chebyshev interpolants relative to original pricer functions from QuantLib

In our study, we conducted tests similar to those described below. In our study we consider a small sample of pricing functions, namely, the standard Black-Scholes option formula for call/put, an American put option using binomial trees and an America put option using Monte Carlo (with Laguerre and Monomial methods) with and without antithetic sampling. For our test we used 2-dimensional functions, variable stock price $[5, 150]$ and volatility $[5\%, 70\%]$, with 42 Chebyshev anchor points (6×7). The fixed parameters were: strike $S = 100$, interest rate $r = 0.05$, maturity $T = 1$, time steps $T_{steps} = 52$ (applicable for binomial trees and MC), and number of paths $M = 100,000$ (applicable to MC only). The time trials were determined by running the subject model and Chebyshev interpolants on a 2500 point (500×500) test grid, which was done to average run time anomalies and to make the run-times large enough to display sensibly on one table. Table *5.2* displays the results obtained from the performance trials.

[1] QuantLib is a free, open source library designed for quantitative finance. It provides a comprehensive framework for modelling, trading, and risk management of financial instruments. See https://www.quantlib.org/

[2] it is not explicitly mentioned in the source but one can infer from the ratio of build and run times of the subject function

Pricing Function	**Test model runtime (s)**	**Chebyshev build time (s)**	**Chebyshev run time**	**Max Error (abs diff)**	**Run time multiplier**
BS call/put (Analytical)	0.41	0.02	0.0149	0.854	28
American put (binomial_tree1)	3.26	0.06	0.010	0.937	319
American put (binomial_tree2)	2.95	0.05	0.009	0.935	333
American put-antithetic (MC_Laguerre)	41.41	58.33	0.002	0.558	24 926
American put - antithetic (MC_Monomial)	50.57	21.78	0.001	0.556	35 587
American put (MC_Laguerre)	130.93	44.82	0.002	0.555	87 199
American put (MC_Monomial)	56.64	20.72	0.001	0.561	56 793

Table 5.2: Low dimensional (2D) performance results, with each trial using 42 (6×7) Chebyshev anchor points.

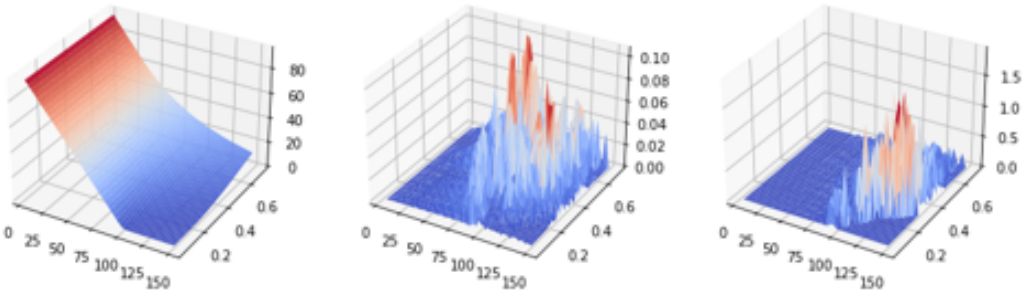


Figure 5.1: The graph displays, from left to right: [*left*] a valuation surface of an american put, [*middle*] the nominal difference in original pricer against Chebyshev interpolant, and [*right*] the percentage difference of original pricer against Chebyshev interpolant.

Remark 5.1.1. Key take-aways in this subsection:

- *We note from Table* 5.2 *that as expected the Chebyshev interpolants are significantly faster than the subject pricing functions, as can be seen from the run multipliers being 28 for the analytical Black Scholes option pricing formula to an average of 300 times for the binomial trees and for Monte Carlo based models the speed multiplier is as high as 80 000 times.*
- *We also observe that at 6× 7 anchor points, we obtain absolute errors that are less than one unit of currency.*
- *Furthermore, we observe that the errors in the Chebyshev interpolants tend to be*

pronounced when the resulting valuation is very low. This is clear when considering the error plots shown in Figure 5.1.

5.1.2 Composition method: CCR trials I

We begin by showing the replication of the entire CVA calculation via Chebyshev interpolants. This may be useful for traders or front office sales teams that regularly trade a specific instrument type with a specific counterparty and may need the sight of an approximation of the CVA impact of a new position.

For this trial, we explored the CVA calculation for interest rate swap under a 1-Factor Hull-White (1FHW) regime, see Table *5.3*. As expected, we observed that the Chebyshev interpolants markedly reduced the run time of the calculation and allowed one to run the CVA calculation for any scenario where the interest rate model was static, but the strike and maturity of the IRS was variable. Figure *5.3* and Figure *5.2* shows the CVA surface and corresponding Chebyshev errors for the IRS under the 1FHW model strike and maturity of the IRS were variable, with plot (a) showing the results for using 36 anchor points and sub-plot (b) 63 anchor points.

Test model runtime	**Chebyshev anchor points**	**Chebyshev build time (s)**	**Chebyshev run time (s)**	**Max. Error (%)**	**Run speed multiplier**	Figure
5 252.64	36	2 261.88	0.002	8.00%	2 679 863.13	Figure *5.2*
11 242.22	63	3 401.80	0.0006	12.10%	20 133 765.05	Figure *5.3*

Table 5.3: Low dimensional performance results, here we consider a proxy for full CVA for 1FHW model IRS on test grid with 100-points.

Remark 5.1.2. Key take-aways in this subsection:

- *Table* 5.3 *table displays maximum errors over the entire surface in nominal values and not as percentages. However, as can be seen from Figure* 5.2 *and Figure* 5.3 *the errors are high when the value of the CVA is very small or equivalents when the strike is very small.*
- *The CVA surface increases with strike and tenure of the IRS and and the error surface is lower than the maximum values attained along the low strike edge therefore, it can be inferred that the Chebyshev interpolants accuracy is higher everywhere outside of the low strike edge.*

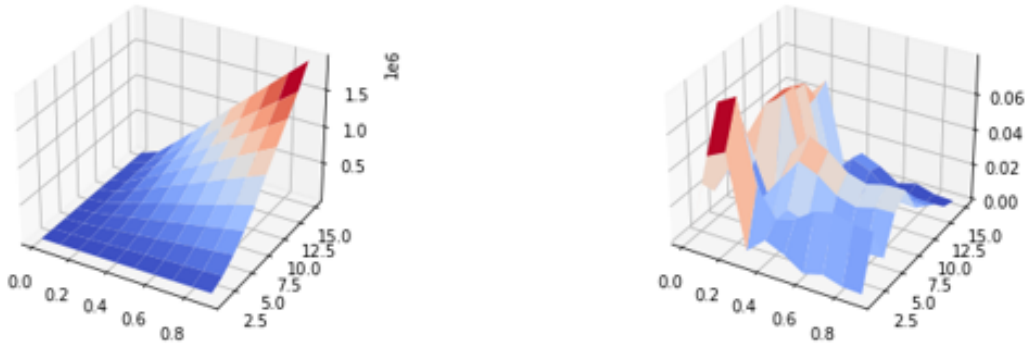


Figure 5.2: Composition methods: Chebyshev methods on CCR type models, with 36 Chebyshev anchor points. We calculate the full CVA of IRS in 1 factor HW model surface against variable strike [0.1,0.9] and maturity [1,15]yrs . The graph on the left depicts CVA value sureface and on the right is the percentage error surface.

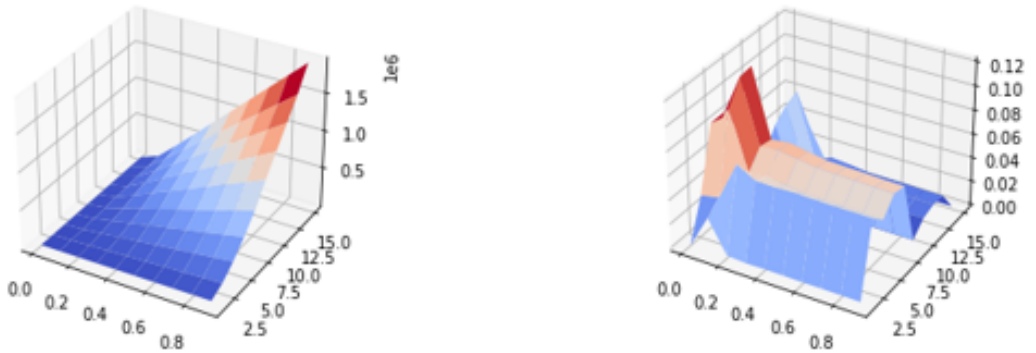


Figure 5.3: Composition methods: Chebyshev methods on CCR type models, with 63 Chebyshev anchor points. We calculate the full CVA of IRS in 1 factor HW model surface against variable strike [0.1,0.9] and maturity [1,15]yrs. The graph on the left depicts the CVA valuation surface and on the right is the percentage error surface.

5.1.3 Composition method: CCR trial II

In this subsection, we demonstrate the implementation of Chebyshev techniques in common CCR risk metric calculations. We will consider a single transaction and netting set (portfolio) example.

5.1.3.1 CVA for single IRS

In this paragraph, we consider the most common applications of the Chebyshev method for CCR calculations. This is done by employing the Chebyshev interpolant when the pricer or valuation of the instrument is called. Therefore, this is done at each time step of the time grid. Depending on the form of the stochastic process chosen for the key risk factors, we have in general that the price or valuation function at each time step is only dependent on the filtration (information available) at the time step. Therefore, the valuation function is the same across all Monte Carlo simulations at each time step, that is the symmetry (invariance) that we take advantage of in the problem, and we only have to build one proxy function at each time step.

We used the same 1FHW model with a dual curve for IRS. The details of the model are as follows:

Assumption 5.1.1. *Trial parameters*
1FHW IRS CVA model parameters

1. *Simulation parameters*
   - *time step = 0.25*
   - *MC simulations = 10,000*
2. *Swap parameters*
   - *t0 = 0*
   - *Notional = 1000000*
   - *direction = -1*
   - *strike=0.05*
   - *Maturity=10*
3. *Market parameters - Hull-White OIS and JIBAR Parameters*
   - $r0_{OIS} = 0.05$
   - $a_{OIS} = 0.1$
   - $vol_{OIS} = 0.01$
   - $r0_{JIB} = 0.06$
   - $a_{JIB} = 0.15$
   - $vol_{JIB} = 0.05$
   - $\rho = -0.9$
4. *CVA parameters - Counterparty parameters*
   - *Recovery rate = 0.4*
   - *Credit spread = 500*

Table *5.4* shows the trials that were run on a single IRS. For the test trials we varied the number of Chebyshev anchor points from $2x2$, $2x3$, $3x3$ for 10k and 100k MC runs. As expected, we observed that as the number of Chebyshev anchor points increased the CVA errors (which represent the sum of errors across the time steps) decreased from $\sim 10\%$ to below 0.1% and the extra computation effort increased, albeit it was never greater than 0.1% at 10K MC runs or 0.01% at 100K MC runs. We also observe that the run and total multipliers are on average ~ 10 and ~ 8times respectively at 10K MC simulations.

Anchor pts/variable	MC sims & time steps	Chebyshev anchor points/ time step	Build evaluations/Total valuation (extra effort)	CVA Error (%)	Run speed multiplier	Run + build speed multiplier	Total speed multiplier
2×2	10,000; 41	4	0.040%	11.897%	10.27	10.19	7.82
2×3	10,000; 41	6	0.060%	-0.704%	9.84	9.74	7.79
3×3	10,000; 41	9	0.090%	-0.054%	9.98	9.85	7.95
2×2	100,000; 41	4	0.004%	16.606%	15.60	15.59	12.26
2×3	100,000; 41	6	0.006%	-0.994%	10.62	10.62	8.48
3×3	100,000; 41	9	0.009%	-0.066%	10.94	10.91	8.56

Table 5.4: Low dimensional performance results: CVA for IRS

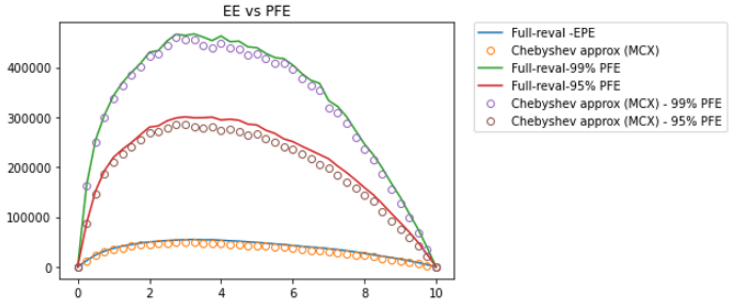
Figures Figure *5.4a* to Figure *5.6b* show the graphs of the key CCR risk metrics, such as PFE, EE and CVA of the trials run on a single IRS with 10k MC runs.

Figures Figure *5.7a* to Figure *5.9b* show the graphs of the key CCR risk metrics, such as PFE, EE and CVA of the trials run on single IRS with 100k MC runs.

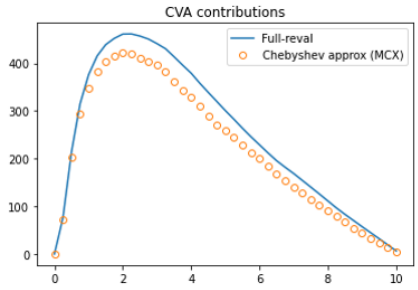
Remark 5.1.3. Key take-aways in this subsection:

- *We notice that the Chebyshev proxies work very well for 2×3 anchor points, as this yields a very good balance of minimum computational effort (six anchor points per time step), superior processing speed (as they improve run times by a factor of approximately eight times) and fair errors (below 1%).*
- *The computational benefit at the 2×3 anchor points, is 99.994% (1-0.006%) for 100,000 MC simulations. This means that we only require the original pricer to perform on 0.006% of the computations it would have done with the original pricer function.*
- *Increasing the number of Chebyshev anchor points yields even higher accuracy although at a slight cost in terms of build time and run speed time.*

figures Figure *5.10a* and Figure *5.10b* show the graphs of the speed multiplier against anchor points for 10k and 100k MC runs. As expected, the multiplier decreases linearly with increasing anchor points.

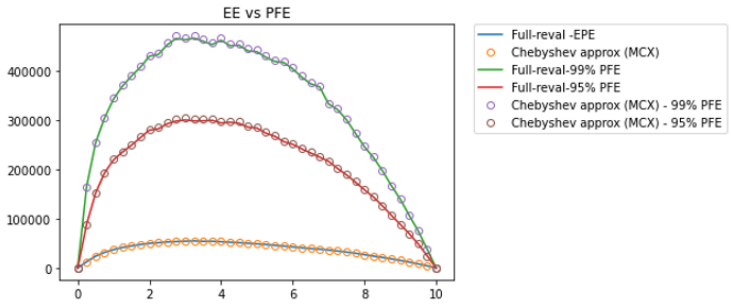


(a)

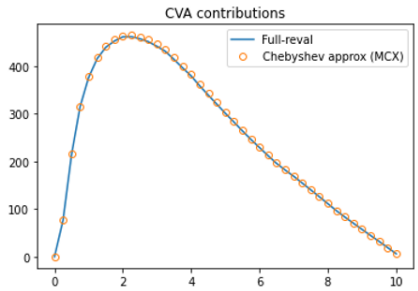


(b)

Figure 5.4: Composition methods at each time step: Example 3 with 10k monte carlo runs for a CVA calculation and 2x2 Chebyshev anchor points per time step

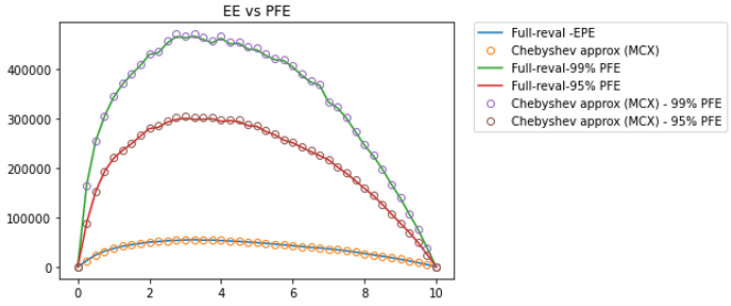


(a)

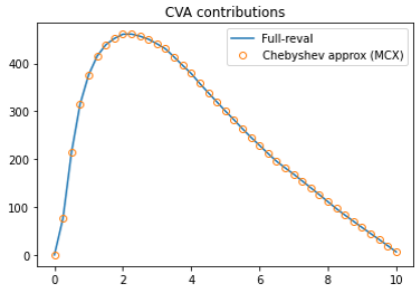


(b)

Figure 5.5: Composition methods at each time step: Example 2 with 10k monte carlo runs for a CVA calculation and 2x3 Chebyshev anchor points per time step

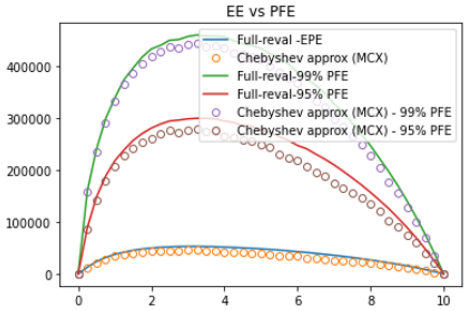


(a)

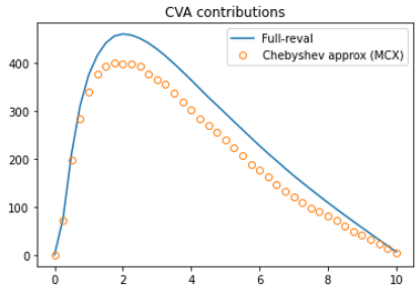


(b)

Figure 5.6: Composition methods at each time step: Example 1 with 10k monte carlo runs for a CVA calculation and 3x3 Chebyshev anchor points per time step

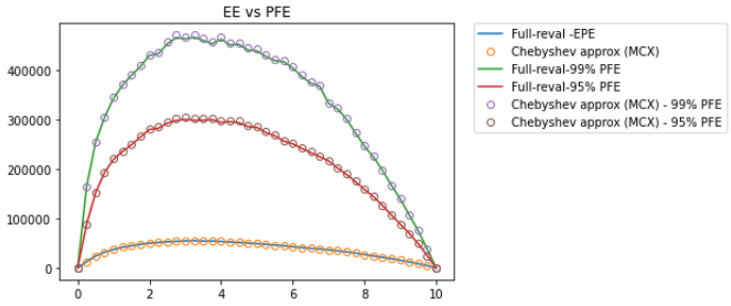


(a)

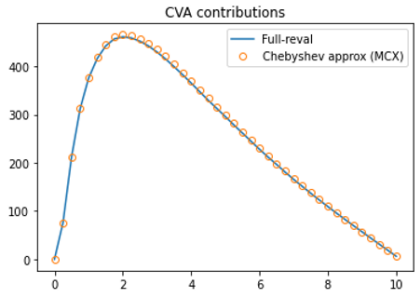


(b)

Figure 5.7: Composition methods at each time step: Example 3b with 100k Monte Carlo runs for a CVA calculation and 2x2 Chebyshev anchor points per time step

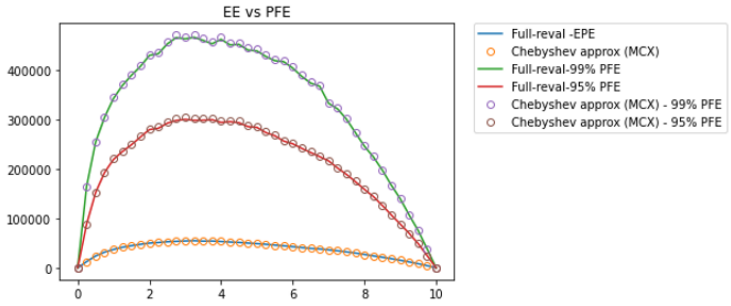


(a)

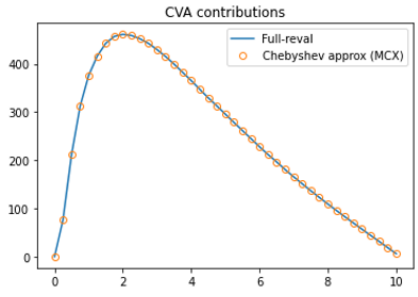


(b)

Figure 5.8: Composition methods at each time step: Example 2b with 100k Monte Carlo runs for a CVA calculation and 2x3 Chebyshev anchor points per time step

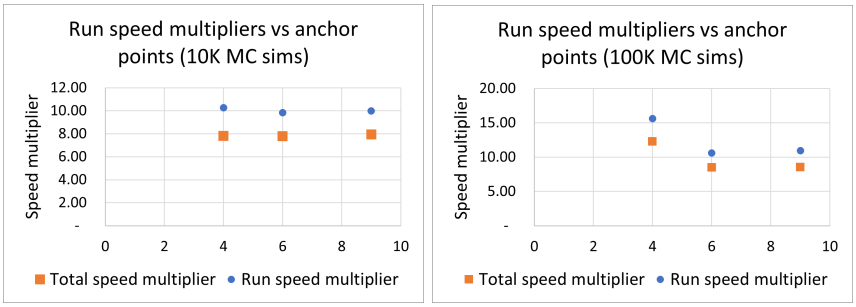


(a)



(b)

Figure 5.9: Composition methods at each time step: Example 1b with 100k Monte Carlo runs for a CVA calculation and 3x3 Chebyshev anchor points per time step



(a) (b)

Figure 5.10: Speed multiplier vs anchor points graphs for 10k and 100k MC runs for a CVA calculation

5.1.3.2 CVA and PFE for IRS netting set

In this paragraph, we consider extending the of the Chebyshev method in CCR calculations to a portfolio, in particular, a netting set in which a number of trades with one counterparty are allowed to be grouped together and have their exposures offset from each other. This is also a very common practise in banking, and allows banks to optimize their CCR risk management. We deploy the Chebyshev interpolant when the pricer or valuation of the netting set or portfolio is called; therefore this is repeated at each time step of the time grid. Because netting sets do not often change from day to day, we use proxy functions for the entire valuation of the netting set. Here, we take full advantage of speed of the Chebyshev interpolant instead of calling a pricer function multiple times for (often) thousands of MC paths.

Therefore, we expect the run time multipliers to increase considerably, as we will replace the valuation of the entire portfolio with one super-fast Chebyshev interpolant.

The rest of the setup is much the same as in the single instrument case, because our netting set contains only one type of instrument, namely the swap. If the netting set contains other types of instruments, the risk practitioner should assess and weigh the benefits of having one proxy price for the entire portfolio, build proxy functions per instrument type or group similar instruments into sub-netting sets with one pricer.

We used the same 1FHW model with a dual curve for IRS. The details of the model are as follows:

Assumption 5.1.2. *Trial parameters*
1FHW IRS CVA model parameters

1. *Simulation parameters*
   - $delt = 0.25$
   - *time step :*
     - *for 40 time steps* $\Rightarrow$ *the* $T_{max} \leq 10yrs$
     - *for 60 time steps* $\Rightarrow$ *the* $T_{max} \leq 15yrs$
     - *for 80 time steps* $\Longrightarrow$ *the* $T_{max} \leq 20yrs$
   - *MC simulations = either 1k, 2k, 5k and 10k*
2. *Swap parameters*
   - $t_0 = 0$
   - *notional = random between* $[500k, 12.5M]$
   - *portfolio_size* $(L) = 25$
   - *direction = random* $\{\pm 1\} \times L$
   - *strike= near ATM between [0.05,0.07] centered around* $r0_{JIB} = 0.06$
   - *Maturity= random [1,15]yrs* $\leq$ *Tmax (tenure must only be multiples of timestep)*
3. *Market parameters - Hull-White OIS and JIBAR Parameters*
   - $r0_{OIS} = 0.05$
   - $a_{OIS} = 0.1$

- $vol_{OIS} = 0.01$
- $r0_{JIB} = 0.06$
- $a_{JIB} = 0.15$
- $vol_{JIB} = 0.05$
- *Correlation* $\rho = 0.5$

4. *CVA parameters - Counterparty parameters*
   - *Recovery rate* $= 0.4$
   - *Credit spread* $= 500bp$

Table *5.5* shows the trials run on a netting set of interest rate swaps. For the test trials we varied the number of Chebyshev anchor points from 2×3, 2×2, 3×3 and 4×4, and for either 1k, 2k, 5k and 10k MC runs. As expected, we observe that as the number of Chebyshev anchor points increases the Chebyshev approximations deviations decrease, this is seen in the CVA error decreasing accordingly. We also observe that the run speed multipliers are considerably higher, as they easily peak above 100 across the board, and we note that the total run speed multipliers [3] get as high as 30-40 times.

Chebyshev anchor pts/ dim	MC sims & time steps	Total Chebyshev anchor points	Build evaluations /Total valuation (extra effort)	CVA value	CVA Error (%)	Run speed multiplier	Run + build speed multiplier	Total speed multiplier
2×2	1k; 55	164	0.40%	288 089.03	9.63%	104	78	24
2×3	5k; 60	360	0.12%	324 792.01	-2.00%	137	123	37
3×3	2k; 55	495	0.45%	53 163.43	-0.05%	133	88	32
4×4	10k; 40	640	0.16%	371 239.90	-0.05%	111	110	29

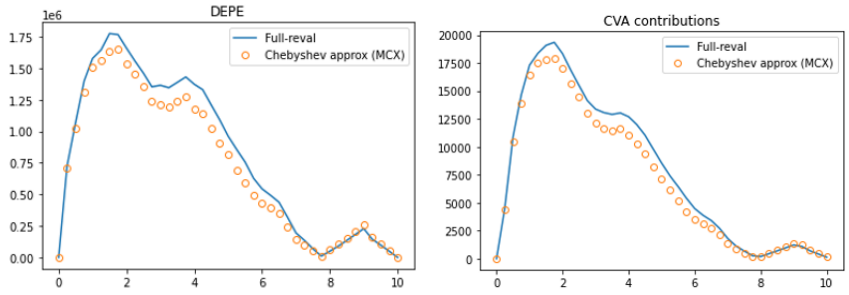
Table 5.5: Low dimensional performance results: CVA for IRS

The figures 5.11a to 5.14b show the graphs of the key CCR risk metrics, such as PFE, EE and CVA of the trials run on single IRS.

Remark 5.1.4. Key take-aways in this subsection:

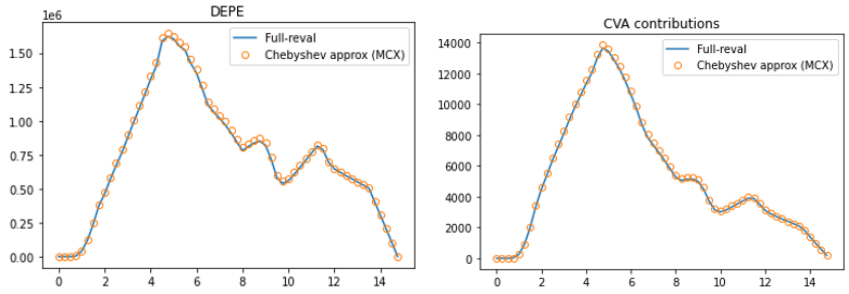
- *Again, we that the Chebyshev proxies work very well for* 3×3 *and* 4×4 *anchor points, which yields a very good balance of low computational effort (an improvement of 99.5% relative to using the original function) and fair errors for practical purposes (less than 1% in the portfolio CVA value).*
- *The run speed multipliers alone are on average well over 100 times faster, which is a remarkable saving in time. When one factors in the building phase, the speed multiplier is still notably high at levels between 80 and 100 times faster.*

[3] running the full risk metric calculation start to end, instead of focusing on the run times of the pricer step



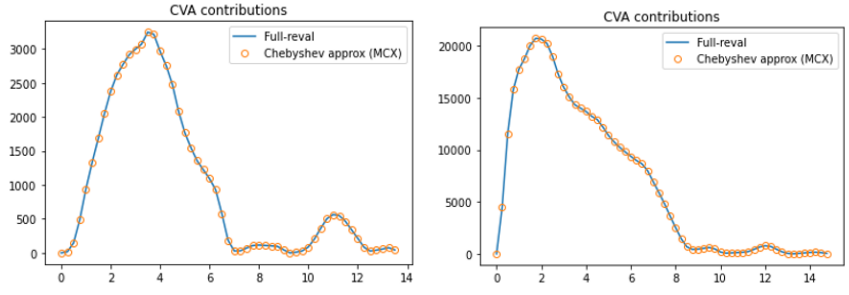
(a) (b)

Figure 5.11: Composition methods at each time step: Netting example 3 with 1k Monte Carlo x 55 time steps runs for CVA calculation and 2x2 Chebyshev anchor points per time step



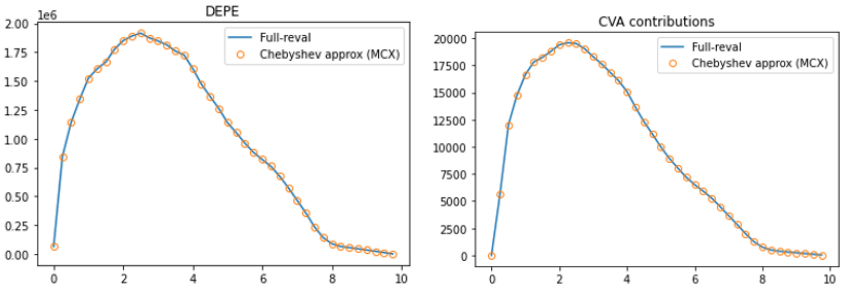
(a) (b)

Figure 5.12: Composition methods at each time step: Netting example 4 with 5k Monte Carlo 55 time steps runs for CVA calculation and 2x3 Chebyshev anchor points per time step



(a) (b)

Figure 5.13: Composition methods at each time step: Netting example 2 with 1k Monte Carlo x 55 time steps runs for CVA calculation and 3x3 Chebyshev anchor points per time step



(a) (b)

Figure 5.14: Composition methods at each time step: Netting example 1 with 10k Monte Carlo x 40 time steps runs for CVA calculation and 4x4 Chebyshev anchor points per time step

5.1.4 Demonstration of curse of dimensionality

In this subsection, we demonstrate the curse of dimensionality by considering two trials.

5.1.4.1 Incrementally increasing dimension

In this trial we consider the standard Black Scholes analytic solution for a put option and incrementally increase the number of dimensions while keeping the Chebyshev anchor points per dimension at five. In our example, we used the base parameters strike = 100, rate = 0.05, tenor = 1 and volatility = 0.2 unless otherwise specified in the specific trial. The results of the trials are displayed in Table *5.6*. As expected, the number of anchor points grows exponentially with dimensions and correspondingly the run multipliers decay.

Figure *5.15* shows a logarithmic plot of the Chebyshev build time against the number of anchor points as the dimensions increases, we note here that the build time grows linearly with the number of dimension (given that the number of Chebyshev anchor points are fixed per dimension).

Remark 5.1.5. Key take-aways in this subsection:

- *the curse of dimensionality affects the number of anchor points required to build our Chebyshev interpolant, and*
- *also affects the run time of the Chebyshev interpolant.*

Dimensions	Num. points/ dim.	Test model runtime	Chebyshev anchor points	Chebyshev run time (s)	Run speed multiplier	
$S[5, 150]$	2 500	0.469598	6	0.00930	51	
$S[5, 150]$, $\sigma[0.05,0.7]$	2 500	0.382106		36	0.01118	34
$S[5, 150]$, $\sigma[0.05,0.7]$, $r[0.005, 01]$	2 500	0.387789	216	0.01240	31	
$S[5, 150]$, $\sigma[0.05,0.7]$, $r[0.005, 01]$, $T[1, 15]$	15 000	4.427177	1296	0.22960	19	
$S[5, 150]$, $\sigma[0.05,0.7]$, $r[0.005, 01]$, $T[1, 15]$, $K[70, 120]$	30 000	4.614051	7776	0.94605	5	

Table 5.6: Curse of dimensionality: Demo 1, where B/S put formula with base line parameters: strike=100, rate=0.05, maturity= T=1, volatility.=σ=0.2

5.1.4.2 Composition method done inefficiently

We also considered deploying the Chebyshev interpolants on the scenario generating component of the risk engine, as this is sometimes the most time consuming part, if the pricing function is simple or has an analytic solution (and hence is already very quick to compute).

Therefore, for each time step ($k \in \{1, 2, \ldots, 41\}$) we build a Chebyshev interpolant to compute the risk factor (interest rate) and another interpolant to value the swap. In this example we select to have 4 Chebyshev points to approximate each interest rate dimension[4]

[4] we have two dimensions, the initial or time-0 rate $(r_0)_{i,j}$ and the Weiner process generated outcome $W_{i,j}$ at the node and time step.

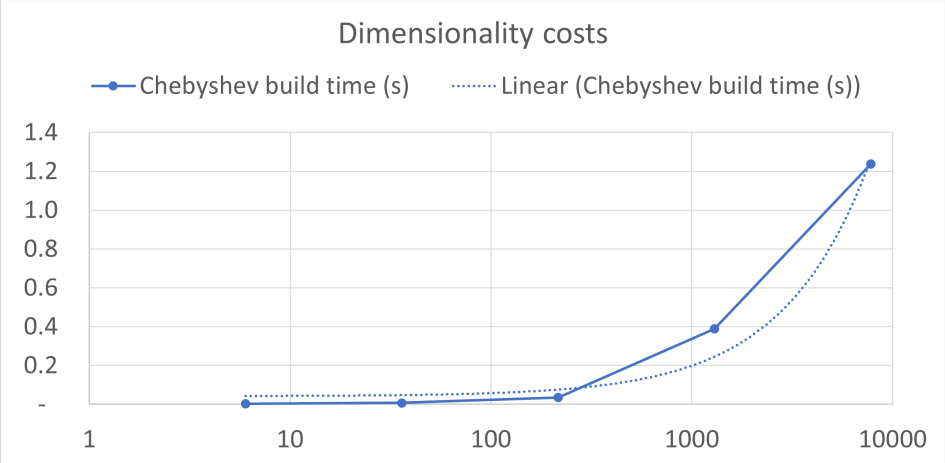


Figure 5.15: Chebyshev build time against the logarithm of number of anchor points required as dimension grows from 1 -5.

and swap valuation function which took in four variables. This will result in 288^5 Chebyshev anchor points in total at each time step for the eight dimensional case, instead of say 16 of done efficiently in two dimensions.

If we apply this to the example observed in *Section 5.1.3*, we effectively increase the number of dimensions of the problem from two to eight, as follows:

- two dimensional – because there is one swap pricer that takes in two generated rates/curve r_OIS and r_JIB at each time step k, we have

$$Swapfn(r^k_{OIS}, r^k_{OIS}).$$

- Eight dimensional – this is where the inefficiency gets built in, this deliberately done to increase the number of dimension [6], we have three functions one for each interest rate and one for the swap pricer:
  - OIS curve generator

$$r^k_{OIS}(r0^{k-1}_{OIS}, W^k_{OIS}; \text{other fixed parameters}) : \mathbb{R}^2 \mapsto \mathbb{R}$$

  - JIBAR curve generator

$$r^k_{JIB}(r0^{k-1}_{JIB}, W^k_{JIB}); \text{other fixed parameters} : \mathbb{R}^2 \mapsto \mathbb{R}$$

  - Swap pricer

$$Swapfn(r0^{k-1}_{JIB}, W^k_{JIB}, r0^{k-1}_{OIS}, W^k_{OIS}; \text{other fixed parameters}) : \mathbb{R}^4 \mapsto \mathbb{R}$$

Remark 5.1.6. Key take-aways in this subsection:

[5] Chebyshev anchor points= $(4)^4 + (4)^2 + (4)^2 = 288$

[6] It is also how I initially implemented the CVA for IRS before I realized the silliness of my implementation, so humour me and just follow through the example and ignore inefficiency.

- *Table* 5.7 *shows the results of the inefficient but high dimensional implementation for CVA calculation seen in* Section 5.1.3. *We note that the main take away that the multipliers have dropped considerably from between 8 and 10 (observed before) to between 2 and 3.*
- *Furthermore, a good understanding of the portfolio and risk metric being calculated is critical in obtaining the most benefit from applying Chebyshev techniques. The application has to be done intelligently, that is with some foresight from experience or careful considerations of the structure of the problem or through incremental trials to establish the best fit for the Chebyshev interploants or tensors.*

MC sims & time steps	Test model runtime	Build evaluations/ Total valuation (extra effort)	Build time (s)	Chebyshev run time (s)	Total Chebyshev model run time (s)	Accuracy/Max Error (%)	Run speed multiplier	Run + build speed multiplier	Total speed multiplier
10^4; 41	22.098	14%	2.7499	6.4385	11.965	3.99E-05	3.43	2.4	1.85
10^5; 41	284.634	1.40%	2.6509	86.9142	127.7919	3.96E-05	3.27	3.18	2.23
10^6; 41	284.634	0.14%	2.6264	941.72	1 349.23	3.95E-05	3.03	3.02	2.12

Table 5.7: Demonstration of curse of dimensionality - here we consider an implementation of 1FHW IRS CVA calculation using an eight dimensions (instead of two) to demonstrate the drop in multipliers.

Now that we have demonstrated the curse of dimensionality in two different setting, we can move on to consider high dimensional implementation of Chebyshev techniques.

5.2 Proxy function tests in high dimensions

Here we examine high dimensional trials using the Black Scholes European option model and a lattice based American option

5.2.1 European option

Here we conduct some basic trials to familiarize ourselves with the method and set a benchmark that we can reference when we consider more complicated experiments.

Table *5.8* and Figure *5.16a* to Figure *5.17b* show the results of the Mocax extended library which implements the TT-format low rank tensor approximations.

For these trials we use the analytic Black Scholes option model with the following parameters

Assumption 5.2.1. *Trial parameters*
European option parameters

- *spot* $= [50, 150]$
- *strike* $= [50, 150]$
- *interest rate* $= [0.01, 0.1]$
- *volatility* $= [0.2, 0.7]$
- *time to maturity* $[1.0, 3.0]$.

Chebyshev interpolant

- *Anchor points/dimension* $= 10$
- *Total anchor points* $= 10^5$

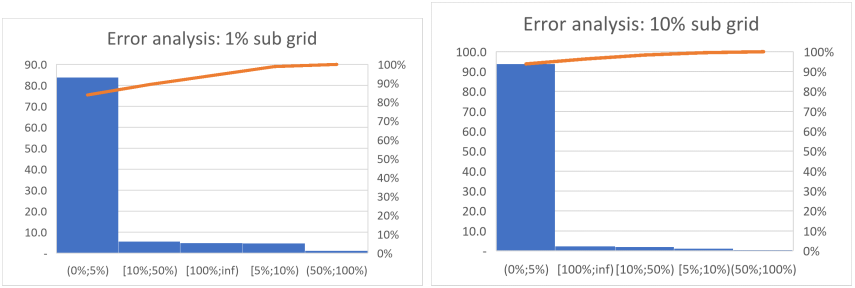
Subgrid size	1%	5%	10%	15%
No. test pts	74 403	74 392	74603	74434
Cheb. evaluation time (s)	57.6	58.9	57.5	54.6
Funct. eval. time (s)	11.6	11.4	11.3	11.7
Multiplier	0.20	0.19	0.20	0.21

Table 5.8: High dimensional Test 1 - using 5 dimensional Black Scholes model with 10 Chebyshev anchor points per dimension

Remark 5.2.1. Key take-aways in this subsection:

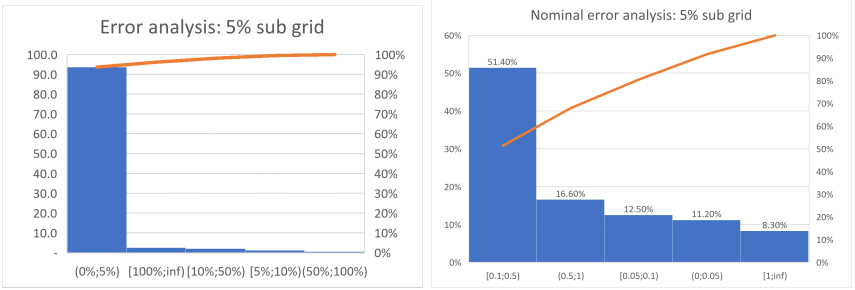
- *From Table* 5.8 *and Figure* 5.16a *we see that the TT-format allows us to sample a very small proportion of the full CT tensor to build our new proxy function. We sampled as little as 1% and obtained a fairly accurate and practical proxy function.*

- *However, we also notice from Table* 5.8 *that the multiplier is less than 1, which means that our proxy functions run considerabley slower that the original pricers, demonstrating that the TT-format method may not be ideal for simple analytical functions and that one maybe better off relying on the full Chebyshev tensor but lower anchor points, such as the one considered in Table* 5.6.
- *The graphs displayed in Figure* 5.16a, *5.16b and 5.17a show that more than 90% of the errors on our multidimensional surface (hyper-volume) are less than 5%. Furthermore when looking at the nominal values of the errors in Figure* 5.17b, *it can be that the majority of the errors (over 90%) are less than 1. Therefore, one can surmise that large errors most likely occur when the price function is very small and hence negligible.*



(a) (b)

Figure 5.16: High dimensional tests - Error analysis with subgrid size of 1% and 10%



(a) (b)

Figure 5.17: High dimensional tests - Error analysis with subgrid size of 5% , on the left we show the percentage error histogram and on the right we show the nominal errors

5.2.2 American option – lattice based

Here, we conduct more basic trials using a numerical model of an American put option. The model is lattice based and of the same type as that used in the low dimensional tests.

Table *5.9* and Figure *5.16a* to Figure *5.17b* show the results of the Mocax extended library which implements the TT-format low rank tensor approximations.

For these trials we use the analytic Black Scholes option model with the following parameters

Assumption 5.2.2. *Trial parameters*
American option – lattice based

- *spot* $= [5, 300]$
- *strike* $= [50, 150]$
- *interest rate* $= [0.01, 0.1]$
- *volatility* $= [0.05, 0.7]$
- *time to maturity* $= [1.0, 3.0]$.
- *time steps* $= 52$

Chebyshev interpolant

- *Anchor points/dimension* $= 10$
- *Total anchor points* $= 10^5$

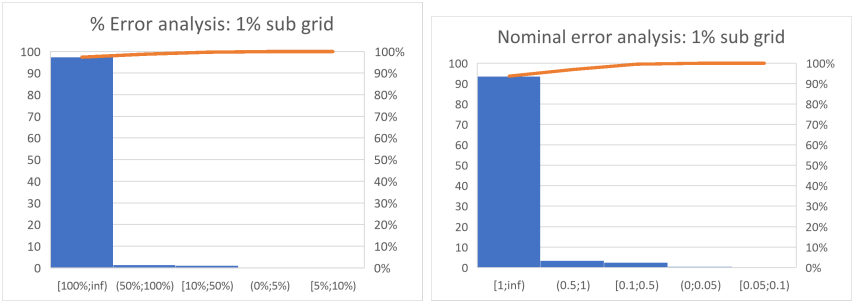
Subgrid size	1%	5%	10%
No. test pts	1000	10 000	10 000
Cheb. evaluation time (s)	0.8	9	8
Funct. eval. time (s)	1.3	12.0	12.6
Multiplier	1.68	1.34	1.58

Table 5.9: High dimensional Test 2 - using 5 dimensional lattice based American put option model with 10 Chebyshev anchor points per dimension

Remark 5.2.2. Key take-aways in this subsection:

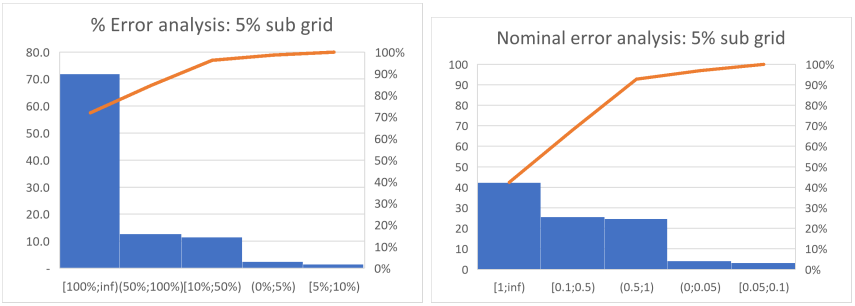
- *From Table* 5.9 *we can see that the multipliers are greater than one which implies that there is an improvement in run time due to the Chebyshev interpolants. Therefore, it is advantageous to consider the high dimensional Chebyshev technique when faced with computationally intensive numerical valuation models.*
- *The graphs displayed in Figure* 5.18a -5.18b *show that at the 1% sub-grid of the Chebyshev tensor the relative and nominal errors are very large. However, we see in Figure* 5.19a-*5.19a that at 5% of the original Chebyshev tensor, only under 60% of the nominal errors are less than one unit of currency. This is a considerable improvement, and for practical purposes, this would be an acceptable threshold for risk practitioners (at least for internal risk measurement purposes but not necessarily for regulatory reporting).*
- *The graphs displayed in Figure* 5.20a -5.20b *that at 10% sub-grid of the Chebyshev tensor the errors have improved considerably. Although on the percentage error basis, we see that approximately 60% of the errors are greater than 50, this is put into perspective when we refer to the nominal basis and observe that all difference are less than a unit of currency.*

- *Overall, we note overall that the low-rank tensor approximation of Chebyshev Tensors can be applied effectively to the lattice based American option model. Although the speed multiplier is approximately 1.5, it offers very good accuracy (of less than a unit of currency) when only working with a 5% sub-grid.*



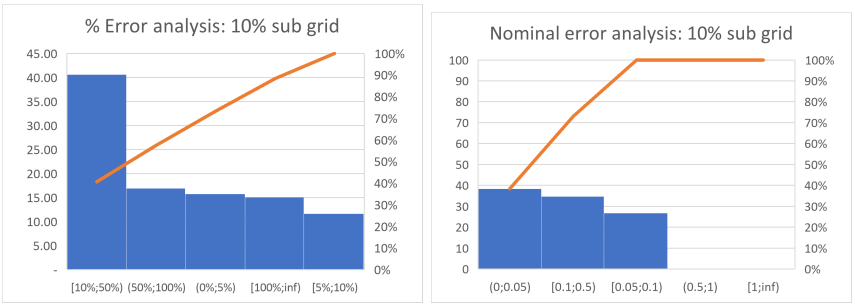
(a) (b)

Figure 5.18: High dimensional tests 2 - Error analysis with subgrid size of 1% (a) percentage error and (b) absolute nominal error



(a) (b)

Figure 5.19: High dimensional tests 2 - Error analysis with subgrid size of 5% (a) percentage error and (b) absolute nominal error



(a) (b)

Figure 5.20: High dimensional tests 2 - Error analysis with subgrid size of 10% (a) percentage error and (b) absolute nominal error

5.3 Derivative sensitivities

Herein we will look at some basic derivative sensitivity trials in low dimensional.

Note 5.3.1. *For clarity, when we refer to derivatives (math) of derivatives (finance) functions we will instead refer to the sensitivities of derivatives.*

5.3.1 Basic tests: 1-dimensional

Here we conducted some basic trials to assess the effectiveness of Chebyshev interpolants in approximating derivative sensitivities. It is worth reminding the reader that all the results of the various derivatives being calculated are computed directly from the interpolants and no additional information (Chebyshev anchor points) is required. The Chebyshev interpolants only have information or input about the subject function and nothing else. We seek to verify that the derivatives of the Chebyshev interpolants are:

- good approximators of the derivative sensitivity,
- converge in sub-exponentially, and
- are computationally faster.

Table *5.10* and Figure *5.21a* to 5.24c show the results of the Mocax library which uses the full Chebyshev tensors to approximate the functions and their derivatives in one build procedure. Figure *5.25* - 5.27 shows the Chebyshev interpolants results for the six anchor points and includes error analysis graphs.

For these trials we used the analytic Black Scholes option model with the parameters described below.

Assumption 5.3.1. *Trial parameters*
Black Scholes parameters

- *type (P/C)* - *Put*
- *spot* -$S_0 = [5, 150]$
- *strike* - $K = 100$
- *interest rate* - $r = 0.05$
- *volatility* - $\sigma = 0.2$
- *time to maturity* - $(T - t) = 1.$

Black Scholes Greeks/derivatives

- *Option value*

$$V_{Put} = Ke^{r(T-t)} \cdot N(-d_2) - S_t \cdot N(-d_1),$$

where $N(x)$ *denotes the standard normal cumulative distribution function:*

$$N(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-z^2/2} dz.$$

And $N'(x)$ *denotes the standard normal probability distribution function:*

$$N'(x) = \frac{\partial N(x)}{\partial x} = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$

, and d_1 *and* d_2 *are given by*

$$d_1 = \frac{1}{\sigma\sqrt{T-t}} \left[\ln(\frac{S_t}{K}) + (r + \frac{\sigma^2}{2})(T-t) \right]$$

$$d_2 = \frac{1}{\sigma\sqrt{T-t}} \left[\ln(\frac{S_t}{K}) + (r - \frac{\sigma^2}{2})(T-t) \right] = d_1 - \sigma\sqrt{T-t}$$

- *Delta*

$$\Delta_{Put} = \frac{\partial V}{\partial S} = -N(-d_1)$$

- *Gamma*

$$\Gamma_{Put} = \frac{\partial^2 V}{\partial S^2} = \frac{N'(d_1)}{S\sigma\sqrt{T-t}}$$

Chebyshev interpolant

- *Derivative order* $= 2$
- *Test grid size* $= 2500$

Test model runtime (s)	Chebyshev anchor points	Chebyshev build time (s)	Chebyshev run time (s)	Run speed multiplier	Reference
0.767463	3	0.005	0.038	20	Figure *5.21a - 5.21c*
0.750494	4	0.003	0.022	35	Figure *5.22a - 5.22c*
0.754702	6	0.008	0.039	19	Figure *5.23a - 5.23c*
0.778145	8	0.003	0.020	39	Figure *5.24a - 5.24c*

Table 5.10: Low dimensional derivative results 1 - runs times for derivatives of the Black Scholes option model

Remark 5.3.1. Key take-aways in this subsection:

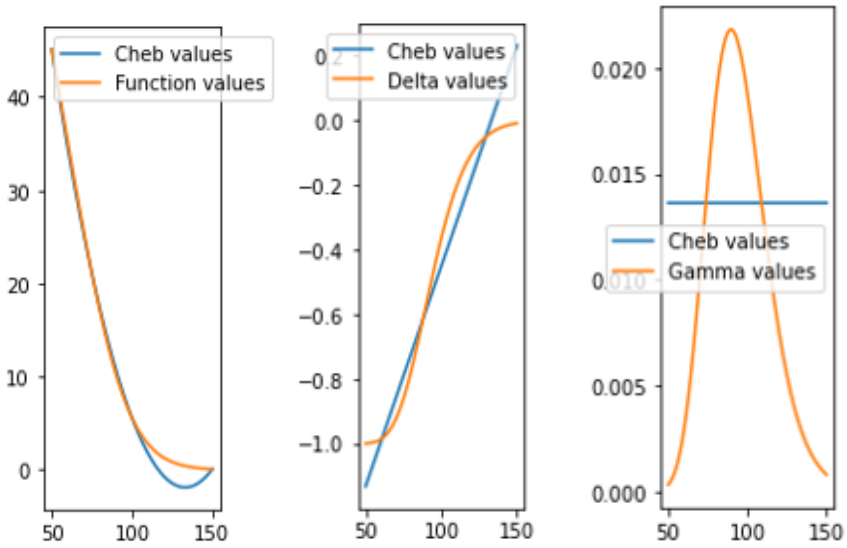
- *From Table* 5.10 *we see that the build time is almost indistinguishable from that observed in Table* 5.2. *This confirms that building derivative interpolants does not require much more effort.*
- *Table* 5.10 *also shows that the run speed multipliers are not affected by the additional requirement of building derivative interpolants/Chebyshev Tensors.*

- *Figure* 5.21a *to 5.24c show the actual function against the corresponding Chebyshev interpolants for the option price/value (a), delta(b) and gamma (c) for increasing anchor points. We observedthat the higher the number of anchor points the better the alignment between the Chebyshev approximator and subject function.*

  Furthermore, it is clear from the low anchor points that the Chebyshev approximation is polynomial based, as the first derivative is a linear function and second derivatives is constant function. As the anchor points increase the derivatives are approximated by higher degree polynomials, quadratic, cubic etc. and we observe that the alignment improves considerably.

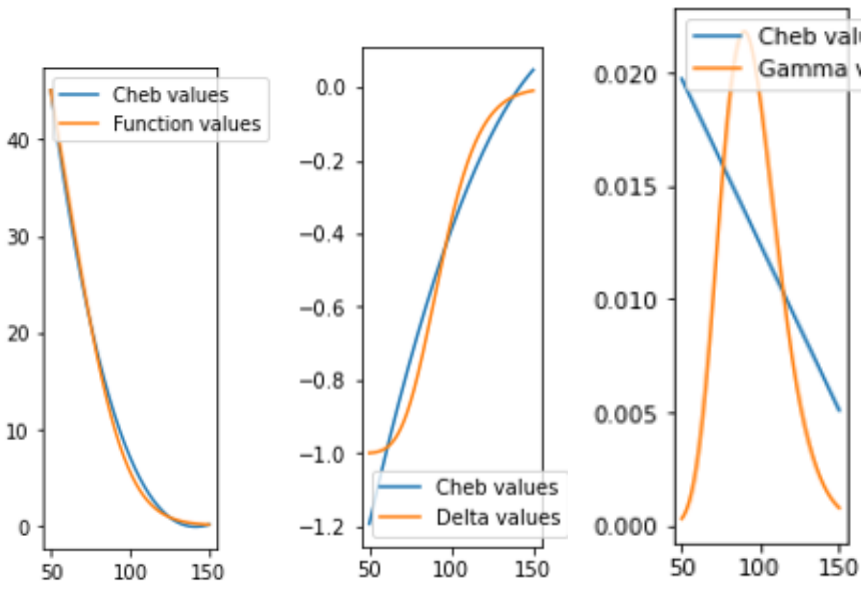
- *The error analysis for the derivative approximations is presented in Figure* 5.25 *- 5.27 for six anchor points only. For each function the absolute nominal and percentage errors are presented. We note the following:*

  - *As expected, the fit was best for the subject function, and progressively worsened for each derivative level. This can be seen by the scale of percentage errors that increases with higher derivatives (the range of the percentage errors for the price is 0-4%, for Delta it is 0-50% and for Gamma it is 0-5000%).*
  - *However, we note that the errors are bounded in nominal terms, as can be seen from the absolute error graphs. The highest deviations occur near the extrema or when the subject function approaches zero, and when both occur, the deviation is exacerbated.*
  - *In relative terms, we observe from the percentage errors plot that the relative errors in general tend to be very small, except when the subject function approaches zero.*



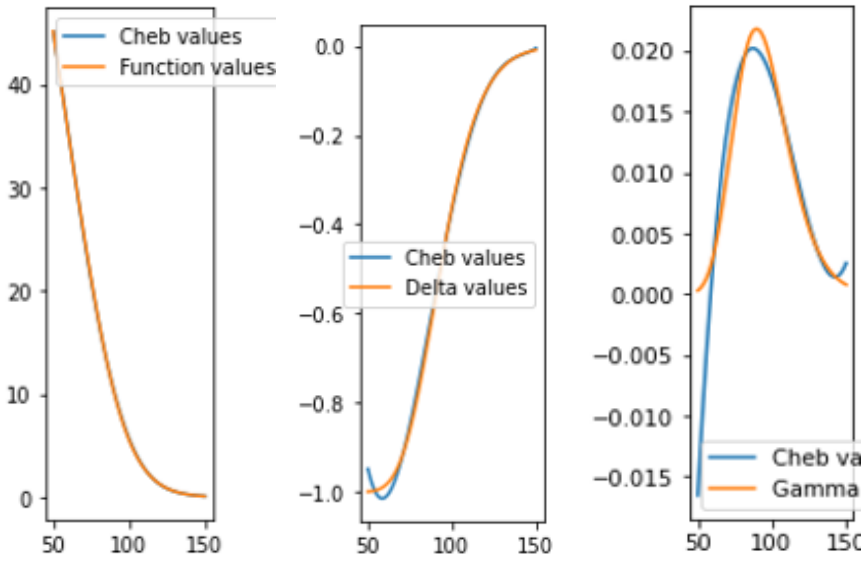
(a) (b) (c)

Figure 5.21: Black Scholes option model with 3 Chebyshev anchor points: (a) Value (b) Delta (c) Gamma



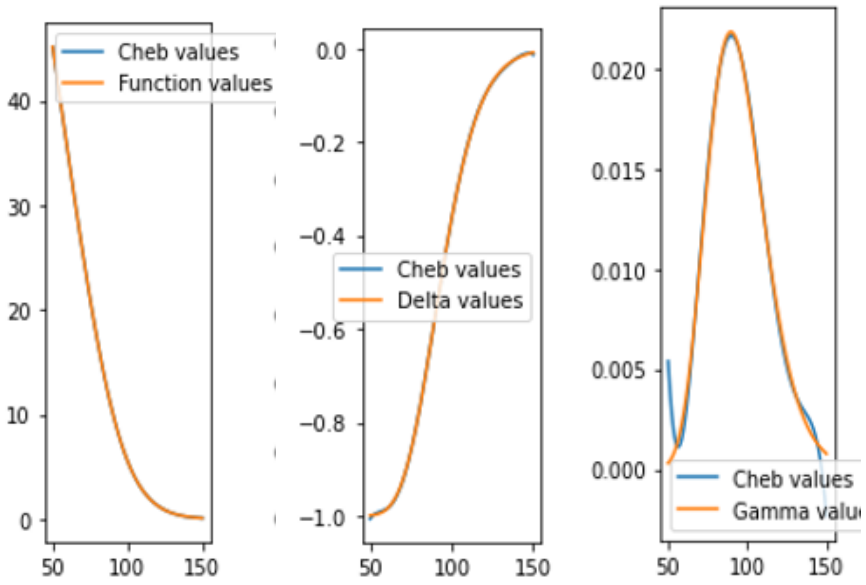
(a) (b) (c)

Figure 5.22: Black Scholes option model with 4 Chebyshev anchor points: (a) Value (b) Delta (c) Gamma



(a) (b) (c)

Figure 5.23: Black Scholes option model with 6 Chebyshev anchor points: (a) Value (b) Delta (c) Gamma



(a) (b) (c)

Figure 5.24: Black Scholes option model with 8 Chebyshev anchor points: (a) Value (b) Delta (c) Gamma

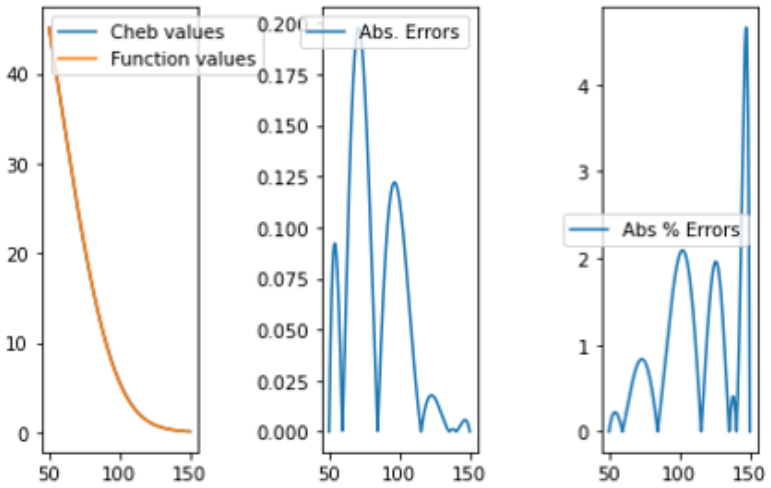


Figure 5.25: Derivative results: Test 1- Value Error Analysis for 6 anchor points

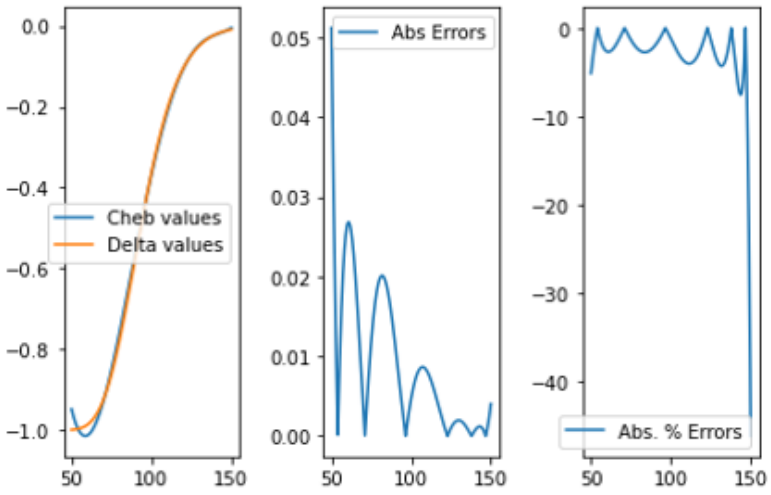


Figure 5.26: Derivative results: Test 1- Delta Error Analysis for 6 anchor points

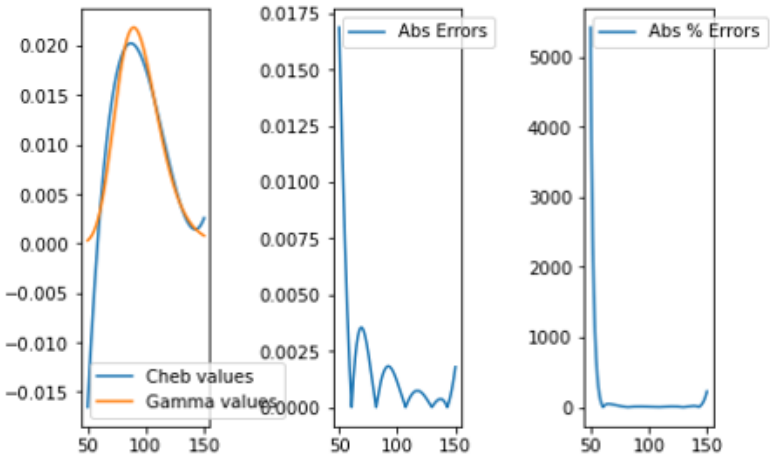


Figure 5.27: Derivative results: Test 1- Gamma Error Analysis for 6 anchor points

5.3.2 Basic tests: 2-dimensional

Here, we extend the trials from *Section 5.3.1* to 2-dimensions. We have the same objectives in mind; therefore, we will not repeat them here.

For these trials, we use the analytic Black Scholes option model with the parameters described below.

Assumption 5.3.2. *Trial parameters*
Black Scholes parameters

- *type (P/C)* - *Put*
- *spot* -$S_0 = [50, 150]$
- *volatility* - $\sigma = [0.05, 0.7]$
- *strike* - $K = 100$
- *interest rate* - $r = 0.05$
- *time to maturity* - $T = 1.$

Black Scholes Greeks/derivatives

- *Option value* V *(as above)*
- *Delta* $\Delta = \frac{\partial V}{\partial S}$ *(as above)*
- *Gamma* $\Gamma = \frac{\partial^2 V}{\partial S^2}$ *(as above)*
- *Vega*
$$\nu = \frac{\partial V}{\partial \sigma} = K\mathrm{e}^{-r(T-t)}N'(d_2)\sqrt{T-t}$$
- *Vanna*
$$Vanna = \frac{\partial^2 V}{\partial S \partial \sigma} = N'(d_1)\frac{d_2}{\sigma}$$

Chebyshev interpolant

- *Derivative order* $= 2$
- *Test grid size* $= 2500$

We consider the results for one trial in which we choose six Chebyshev anchor points per dimension. As we saw in *Section 5.3.1*, six anchor points are sufficient to approximate 2nd order sensitivities of the Black Scholes option model. One can infer from the results of *Section 5.3.1* the anticipated behaviour with fewer or more anchor points. The results are presented in Table *5.11* , Figure *5.28* to 5.32 and discussed in the *Remark 5.3.2*. The graphs displayed in Figure *5.28* to *5.32* show the subject function against the corresponding Chebyshev interpolants, with a focus on the error analysis. In each figure the first graph shows the option value or sensitivity (left), the second graph shows the absolute error (middle) and the third graph shows the absolute relative error(right).

Test model runtime (s)	Chebyshev anchor points	Chebyshev build time (s)	Chebyshev run time (s)	Run speed multiplier	Reference
2.780	$6^2 = 36$	0.011	0.069	40	Figure *5.28 - 5.32*

Table 5.11: Low dimensional derivative results 2 - runs times for derivatives of the Black Scholes option model

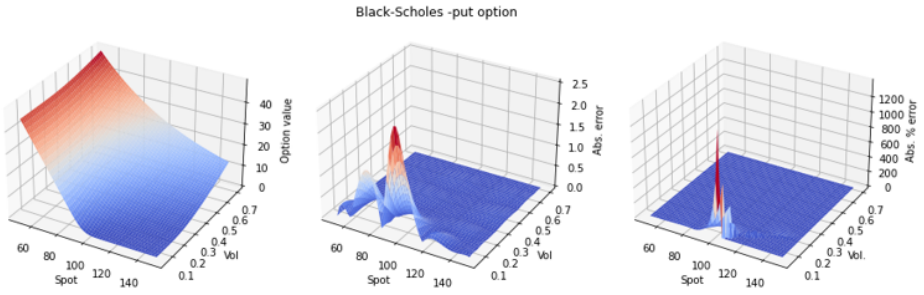


Figure 5.28: Derivative results: Test 2- Value and error analysis for 6 anchor points

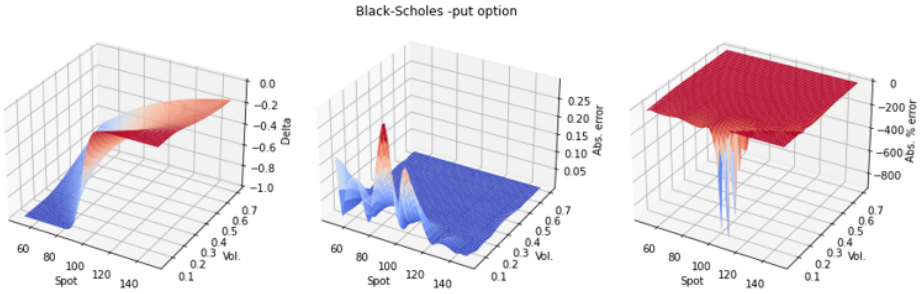


Figure 5.29: Derivative results: Test 2- Delta and error analysis for 6 anchor points

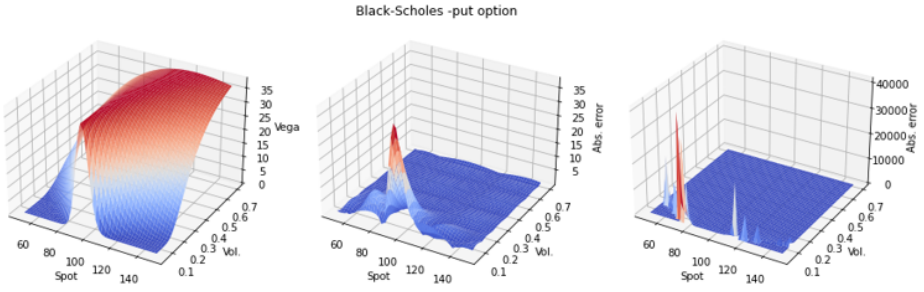


Figure 5.30: Derivative results: Test 2- Vega and error analysis for 6 anchor points

Remark 5.3.2. Key take-aways in this section:

- *From Table* 5.11 *we can see that run run time of the Chebyshev interpolant is 40 times faster than that of the original pricing models.*
- *In Figure* 5.28 *the subject function is the option value. The absolute errors are bounded above 2.5 units and mainly occur for very low volatility levels. The relative error surface shows that the largest deviations occured in the region near*

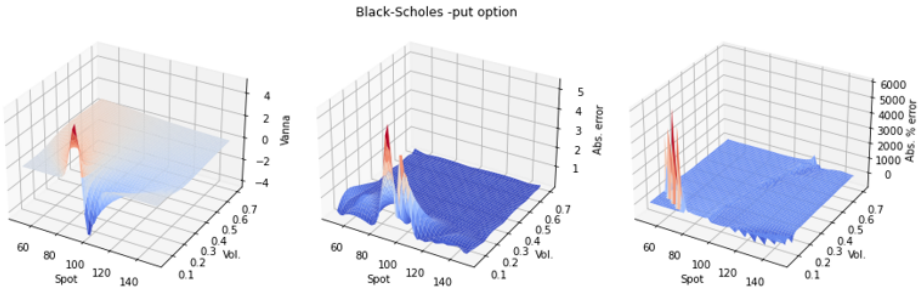


Figure 5.31: Derivative results: Test 2- Vanna and error analysis for 6 anchor points

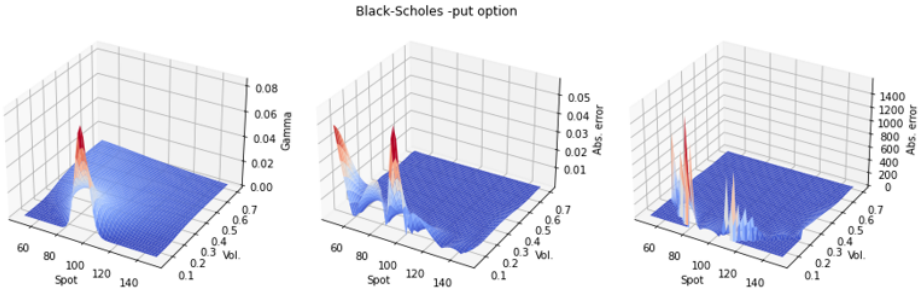


Figure 5.32: Derivative results: Test 2- Gamma and error analysis for 6 anchor points

the strike (100) and towards the lower bound of the volatility. This is expected since at low volatility the valuation function has sharp change in slope. Overall the errors are very small which shows the good accuracy of the Chebyshev interpolant.

- *In Figure* 5.29*, 5.30, 5.31 and 5.32, the subject functions are delta, vega, vanna and gamma of the option, respectively. Across all figures we notice that the absolute and relative errors are mostly very small; hence, the error surfaces are mostly flat or have very small ripples. Spikes on the error surfaces are mostly observed where the subject function has regions of rapidly changing gradients, convexity, turning points, inflection points or regions where the function approaches zero.*

Observation 5.3.2.1. *There are two ways to improve the accuracy of Chebyshev interpolants for functions with known or anticipated regions of discontinuity, non-differentiability as observed in the above tests. The first and naive method is to simply increase the number of Chebyshev anchor points so that there are more points around the region of concern. Alternatively, one can apply the domain-splitting technique, this takes advantage of the fact that the Chebyshev points are dense near the bounds of the domain. Therefore by splitting the domain into two (or more) regions, one can ensure a higher concentration of anchor points near the region and hence better accuracy.*

The better the knowledge and insights about the financial instrument or portfolio one has, the better can intelligently apply Chebyshev techniques to improve the efficiencies of the proxy function and overall risk metric calculation.

5.4 Summary and discussion of results

Proxy Function Tests in Low Dimensions: The results chapter began with proxy function tests in low dimensions (see *Section 5.1.1*) which serves as a foundational step in understanding the effectiveness of Chebyshev techniques. We consider three different classes of valuation models analytic (or closed form solution), lattice based, and Monte Carlo based. We observed that Chebyshev proxies could run approximately one, two and four orders of magnitude faster than the original pricer functions for the above classes respectively. All the while maintaining absolute errors below 1 unit of currency which an acceptable threshold for risk management in practice.

These tests demonstrate that the Chebyshev approximation has the potential to significantly reduce the computational burden while maintaining the accuracy of the risk metric calculations.

Counterparty Credit Risk Trials: The study then moved on to counterparty credit risk trials, which were divided into two parts. The first part (CCR trial I), see *Section 5.1.2*, focuses on replicating the full CVA calculation comprising the scenario generation and valuation averaging into one function.The second part (CCR trial II), *Section 5.1.3*, deploys the Cebyshev approximations in the valuation of the IRS at each time step within the MC simulated paths.

In CCR trial I, we observed extremely high run speed multipliers of approximately six orders of magnitude and again coupled with good accuracy levels (maximum percentage error of 10% which only occurred when the risk metric value were very small).

In CCR trial II, considered the most likely use of the Chebyshev techniques as it offers more flexibility in that the Chebyshev proxies only depend on the number of time steps and can handle any desired number of MC simulations. We noticed that with only a few (between 6 and 9) Chebyshev anchor points were required per time step to deliver errors within 1% accuracy. Furthermore, the low number of anchor points meant that the computational efficiency was improved by 99.994% for a single IRS and by 99.5% for a netting set portfolio, relative to the traditional method using the original pricer function. Furthermore, the total run speed multiplier was approximately one order of magnitude faster than that to the traditional method.

The results from these trials highlight the robustness of the Chebyshev techniques in handling counterparty credit risk calculations. The approximation method provesd to be effective in reducing the computational time required for these calculations, which is crucial for a timely risk assessment.

Curse of Dimensionality: A significant challenge with the Chebyshev approximation technique is the curse of dimensionality, where the computational cost increases exponentially with the number of dimensions. Furthermore, we saw that high dimensionality also implies slow run times and hence lower run speed multipliers. We provided two example to test this phenomenon and demonstrated the need for low rank tensor approximation techniques for Chebyshev Tensors.

This section also demonstrates that a good understanding of the portfolio and risk metric being calculated is critical in obtaining the most benefit from applying the Chebyshev techniques. The application of Chebyshev techniques has to be done with some foresight from experience or careful considerations of the structure of the problem or through incremental trials to establish the best fit for the Chebyshev interpolants.

Proxy Function Tests in High Dimensions: To test the applications of the Chebyshev tensor in TT-format (which is designed to overcome the curse of dimensionality problem), we applied the method to standard pricing models: the European option and a lattice based American option.

We observed that for the analytic European option model, the TT-format method yielded very favourable results, with 90% of error less than 1 unit of currency, when only using 5% of the full Chebyshev Tensor. However, we observed that the speed multipliers where below 1, which meant that the Chebyshev Tensors were slower to run than the original Black Scholes option pricing model. This suggests that it may not be optimal to apply Chebyshev tensors in the TT-format to high dimensional analytic functions.

For lattice based American options, the TT-format method required between 5% and 10% for errors to well below 1 unit of currency. In addition, the speed multipliers are above one which means that there is a gain in computational time from Chebyshev Tensors in TT-format.

Our results, although based on relatively elementary trials, do indicate that Chebyshev techniques remain effective in higher dimensions, providing accurate approximations while significantly reducing computational time. This is particularly relevant for complex financial instruments that employ numeric methods and are high-dimensional.

Derivative sensitivity results: The final section of *Section 5* presents the results of applying Chebyshev techniques to determine the derivative sensitivities. For these trials we focused on analytic Black Scholes model, in one and two dimensions. Our results revealed tow remarkable observations. First, the build times of the Chebyshev Tensors that approximate the option sensitivities (such as delta and gamma), are almost indistinguishable from the build time we observed in time observed in the low dimensional case. This confirms that building sensitivity interpolants requires only marginal additional effort. Second, we observed that the run speed multipliers remain relatively high even for the sensitivity proxy functions.

These results indicate that Chebyshev techniques may be very effective in improving not onlythe efficiency of the risk calculations but also their sensitivity. This is a highly desirable property, as the practice of risk management in banking is often also concerned with the sensitivities of risk metrics.

Chapter 6

Discussion

In this chapter, we discuss the outcome of the results, present some concluding remarks and discuss areas of further research. This chapter will be presented with the following headings

- Overview and objectives
- Evaluation of results
- Conclusions
- Limitations and further research

6.1 Overview and objectives

this study tests the effectiveness of these techniques in reducing the computational burden and improving the inefficiencies of risk metric calculations in the banking sector. In this study, we provide a detailed review of the theory underpinning theChebyshev approximations and discuss their practical application to banking risk calculation.

Our approach follows and attempts to verify several results published by, among others, Zeron and Ruiz with the aid of adapted codes from Mocax Intelligence open source Python libraries. The goal is to test the effectiveness of Chebyshev methods in reducing the computational burden and improving the inefficiencies of risk metric calculations in the banking sector.

We set ourselves the following objectives, in assessing the resulting efficiencies of Chebyshev techniques:

1. the original pricing function in the building phase should be called considerably fewer times than that in the first principle approach;
2. the precision of the interpolating object should be very good; and
3. the computational cost of evaluating the interpolating object should be low.

6.2 Evaluation of results

The results of our trials are detailed and discussed *Chapter 5*. In this section we review observations from the results against the objectives outlined above.

1. Efficient building phase: This study demonstrates that Chebyshev techniques can significantly reduce the computational burden associated with the risk calculations. This is evident from counterparty credit risk trials, where the computational efficiency was improved by 99.994% for a single IRS and by 99.5% for a netting set portfolio, relative to the first principles method that uses the original pricer function.

2. Maintaining accuracy: Despite the reduction in computational effort, the study found that Chebyshev techniques maintain a high level of accuracy in risk calculations. Again this is best illustrated by the CCR results, where a few (between six and nine) Chebyshev anchor points were required per time step to deliver errors within 1% accuracy for CVA. This is particularly important for financial institutions that rely on precise risk metrics for decision-making and regulatory reporting.

3. Efficient execution: The run speed multipliers achieved by the Chebyshev proxy functions are on average well over 100 times faster than typical numeric based pricer functions on a stand alone basis. When the Chebyshev techniques are applied to a complete risk metric calculation such as CVA or PFE, we find that the overall run speed multipliers are at least an order of magnitude higher than those of first principles methods.

4. Secondary observations: This study shows that Chebyshev techniques can be applied with minimal additional effort to approximating risk metrics and their sensitivities. The implementation of new numerical techniques in a real-world environment can be challenging. This study discussed potential challenges, such as the need for specialized knowledge and the integration with existing systems, and provided solutions to address these issues.

The results presented in Chapter 5 demonstrate that Chebyshev techniques can significantly reduce the computational burden associated with risk calculations while maintaining a high level of accuracy, which is consistent with Zeron and Ruiz and other related literature on the subject.

6.3 Conclusion

In conclusion, we have demonstrated through empirical trials that Chebyshev numerical techniques can improve the efficiency and accuracy of risk calculations in the banking sector at a minimal computational cost. This study provides a comprehensive assessment of the effectiveness of these techniques, highlighting their potential for enhancing risk management practices.

6.4 Limitations and further research

Our study performed only basic valuation trials for high dimensional cases. Performing CVA and PFE calculations for a portfolio of interest rate swaps and cross currency swaps and deploying Chebyshev Tensors in TT-format would be a valuable area for research. One could also study the application of these techniques to new regulatory standards, such as the FRTB-Internal Models Approach and standardized initial margin model (SIMM)

Chapter 7

Bibliography

[1] Grzelak L. A. *Sparse Grid Method for Highly Efficient Computation of Exposures for xVA*. arXiv https://arxiv.org/abs/2104.14319, 2022.

[2] Rodomanov A. Introduction to the tensor train decomposition and its applications in machine learning, 2016.

[3] Philippe Artzner, Freddy Delbaen, Jean-Marc Eber, and David Heath. *Coherent Measures of Risk*. Mathematical Finance, Wiley, 2001.

[4] Hull J. C. and White A. Libor vs. ois: The derivatives discounting dilemma. 2013.

[5] Merton R. C. *Theory of Rational Option Pricing*. The Bell Journal of Economics and Management Science, Rand Corporation, 1973.

[6] Heath D., Jarrow R., and Morton A. *Bond Pricing and the Term Structure of Interest Rates: A New Methodology for Contingent Claims Valuation*. Econometrica, The Econometric Society, 1992.

[7] O'Kane D. *Modelling Single-name and Multi-name Credit Derivatives*. John Wiley & Sons Inc., 2008.

[8] Afanas'eva E. and Golberg A. *Absolute Continuity in Higher Dimensions – Function Spaces, Theory and Applications*. Fields Institute Communications, Springer, 2023.

[9] Eberlein E., Keller U., and Prause K. *New Insights into Smile, Mispricing, and Value at Risk: The Hyperbolic Model*. Journal of Business, University of Chicago Press, 1998.

[10] SalzerH. E. *Lagrangian interpolation at the Chebyshev points*. The Computer Journal, Oxford University Press, 1972.

[11] Black F. and Scholes M. *The Pricing of Options and Corporate Liabilities*. Journal of Political Economy, University of Chicago Press, 1973.

[12] Saita F. *Value at Risk and Bank Capital Management*. Academic Press, 2007.

[13] Bank for International Settlements. *83rd Annual Report, 1 April 2012 – 31 March 2013*. Bank for International Settlements, available at https://www.bis.org/publ/arpdf/ar2013e.htm, 2013.

[14] West G. *South African Financial Markets*. Retrieved from Financial Modelling Agency: www.finmod.co.za, 2009.

[15] L.A. Grzelak, J.A.S. Witteveen, M. Suárez-Taboada, and C.W. Oosterlee. *The stochastic collocation Monte Carlo sampler: highly efficient sampling from "expensive" distributions.* Quantitative Finance, Routledge, 2019.

[16] Markowitz H. *The Early History of Portfolio Theory: 1600–1960.* Financial Analysts Journal, Taylor & Francis, 1999.

[17] Glyn A. Holton. *Capital Adequacy Computations.* Contingency Analysis, available at http://www.contingencyanalysis.com, 2002.

[18] Ruiz I. and Zeron M. *Machine Learning for Risk Calculations: A Practitioner's View.* Wiley, The Wiley Finance Series, https://books.google.co.za/books?id=j_BMEAAAQBAJ, 2021.

[19] Gregory J. *The xVA Challenge: Counterparty Risk, Funding, Collateral, Capital and Initial Margin.* John Wiley & Sons, isbn 978-1-119-50900-4, 2020.

[20] Hull J. and A. White. *Pricing interest rate derivative securities.* Review of Financial Studies, 1990.

[21] Hull J.C. *Risk Management and Financial Institutions.* Prentice-Hall, 2007.

[22] Hull J.C. *Options, Futures & Other Derivatives (10th Ed).* Prentice-Hall, 2018.

[23] Glau K., D. Kressner, and F. Statti. *Low-rank tensor approximation for Chebyshev interpolation in parametric option pricing.* arXiv,https://arxiv.org/abs/1902.04367v1, 2019.

[24] Glau K., R. Pachon, and C. Potz. *Speed-up credit exposure calculations for pricing and risk management.* Quantitative Finance, 21:481-499, 2020.

[25] Heston S. L. *A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options.* The Review of Financial Studies, Oxford University Press, 1993.

[26] Grzelak L.A. and C.W. Oosterlee. *From Arbitrage to Arbitrage-free Implied Volatilities.* Journal of Computational Finance, 20(3):31-49, 2016.

[27] Trefethen L.N. *Spectral Methods in MATLAB.* Society for Industrial and Applied Mathematics., 2000.

[28] Trefethen L.N. *Approximation theory and approximation practice.* Society for Industrial and Applied Mathematics., 2018.

[29] Bianchetti M. *Two Curves, One Price.* Risk Magazine, 2010.

[30] Gaß M. *PIDE Methods and Concepts for Parametric Option Pricing.* PhD thesis, Technical University of Munich, https://arxiv.org/abs/1505.04648, 2016.

[31] Gaß M., K. Glau, M. Mahlstedt, and M. Mair. *Chebyshev Interpolation for Parametric Option Pricing.* https://arxiv.org/abs/1505.04648, 2016.

[32] Morini M. *Solving the Puzzle in the Interest Rate Market.* Risk, 2009.

[33] Pistorius M. and Stolte J. *Fast Computation of Vanilla Prices in Time-Changed Models and Implied Volatilities Using Rational Approximations.* International Journal of Theoretical and Applied Finance, 2012.

[34] Wheatley M. *The Wheatley Review of LIBOR: Final Report.* HM Treasury, 2012.

[35] Zeron M. and Ruiz I. *Chebyshev Methods for Ultra-Efficient Risk Calculations*. http://dx.doi.org/10.2139/ssrn.3165563, 2018.

[36] Zeron M. and I. Ruiz. *Tensoring Dynamic Sensitivities and Dynamic Initial Margin*. Risk Magazine and arXiv preprint arXiv:2011.04544, 2021.

[37] Bernstein S. N. *Sur l'approximation des fonctions continues par des polynomes*. Compt. Rend. Acad. Sci., 1911.

[38] Vasicek O. *An Equilibrium Characterization of the Term Structure*. Journal of Financial Economics, Elsevier, 1977.

[39] Federal Reserve Bank of New York. *Interest Rate Derivatives and Monetary Policy Expectations*. Liberty Street Economics, December 2014.

[40] Basel Committee on Banking Supervision. *International Convergence of Capital Measurement and Capital Standards*. Amendments to the 1988 Basel Capital Accord, Bank for International Settlements https://www.bis.org/publ/bcbs04a.htm, 1995.

[41] Basel Committee on Banking Supervision. *Targeted Revisions to the Credit Valuation Adjustment Risk Framework*. Bank for International Settlements, available at https://www.bis.org/bcbs/publ/d507.pdf, 2020.

[42] Carr P. and Madan D. *Option Valuation Using the Fast Fourier Transform*. Journal of Computational Finance, Incisive Media, 1999.

[43] Jorion P. *Value at Risk: The New Benchmark for Managin Financial Risk*. The McGraw-Hill Companies, 2007.

[44] Zangari P. *Value at Risk: A Model for Measuring Portfolio Risk*. RiskMetrics Group, Available at SSRN: http://ssrn.com/abstract=141226, 1996.

[45] Pachon R. *Numerical Pricing of European Options with Arbitrary Payoffs*. Preprint, Available at SSRN: http://ssrn.com/abstract=2712402, 2016.

[46] Raible S. *Lévy Processes in Finance: Theory, Numerics, and Empirical Facts*. PhD Thesis, University of Freiburg, 2000.

[47] Smolyak S.A. *Quadrature and interpolation formulas for tensor products of certain classes of functions*. Dokl. Akad. Nauk SSSR, 4:240-243, 1963.

[48] E. Sachs and Schu M. *Reduced Order Models in PIDE Constrained Optimization*. Control and Cybernetics, 2010.

[49] Sauter S. A.and C. Schwab. *Boundary Element Methods*. Springer Series in Computational Mathematics. Springer Berlin, Heidelberg, 2013.

[50] Oosterlee C. W. and Grzelak L. A. *Mathematical Modeling and Computation in Finance*. Springer, 2019.

Appendix A

Additional results

A.1 Proxy function tests in low dimensions –tables

A.1.1 Composition method: pricing models

Pricing Function	Pricing Method	Test model runtime (s)	Chebyshev Anchor Points	Chebyshev build time (s)	Chebyshev run time	Accuracy/Max Error (abs diff)	Run time multiplier
BS call/put	Analytical	0.410119	42	0.01851797	0.014868498	0.854	28
American put (binom_tree1)	Binomial tree	3.264929	42	0.05690789	0.010220528	0.937	319
American put (binom_tree2)	Binomial tree	2.956927	42	0.05405045	0.008870602	0.935	333
American put-antithetic (MC_Laguerre)	Monte carlo	41.41481	42	58.3337464	0.001661539	0.558	24 926
American put - antithetic (MC_Monomial)	Monte carlo	50.56867	42	21.7754235	0.001420975	0.556	35 587
American put (MC_Laguerre)	Monte carlo	130.9352	42	44.8213425	0.00150156	0.555	87 199
American put (MC_Monomial)	Monte carlo	56.63965	42	20.717967	0.000997305	0.561	56 793

Table A.1: Low dimensional (2D) performance results, with each trial using 42 (6×7) Chebyshev anchor points - internal research.

A.1.2 Composition method: CCR trials I

Risk metric	Pricing method	Test grid (pts)	Test model runtime	Chebyshev anchor points	Chebyshev build time (s)	Chebyshev run time (s)	Max. Error (%)	Run speed multiplier	Figure
CVA - 1FHW model IRS	Monte Carlo	100	5 252.64	36	2 261.88	0.002	8.00%	2 679 863.13	Figure *5.2*
CVA - 1FHW model IRS	Monte Carlo	100	11 242.22	63	3 401.80	0.0006	12.10%	20 133 765.05	Figure *5.3*

Table A.2: Low dimensional performance results, here we consider a proxy for full CVA.

A.1.3 Composition method: CCR trial II

Anchor pts/ variables	MC sims & time steps	Test model run-time	Chebyshev anchor points/ time step	Build evaluations/Total valuation (extra effort)	Chebyshev build time (s)	Chebyshev run time (s)	Total Chebyshev model run time (s)	CVA Error (%)	Run speed multiplier	Run + build speed multiplier	Total speed multiplier
2x2	10,000; 41	18	4	0.040%	0.013	1.69	2.36	11.897%	10.27	10.19	7.82
2x3	10,000; 41	19	6	0.060%	0.018	1.77	2.38	-0.704%	9.84	9.74	7.79
3x3	10,000; 41	19	9	0.090%	0.024	1.80	2.41	-0.054%	9.98	9.85	7.95
2x2	100,000; 41	263	4	0.004%	0.016	15.83	21.44	16.606%	15.60	15.59	12.26
2x3	100,000; 41	248	6	0.006%	0.016	22.01	29.27	-0.994%	10.62	10.62	8.48
3x3	100,000; 41	209	9	0.009%	0.053	17.95	24.41	-0.066%	10.94	10.91	8.56

Table A.3: Low dimensional performance results: CVA for IRS

Chebyshev anchor pts/ dim	MC sims & time steps	Test model eval time	Test model run-time	Total Chebyshev anchor points	Build evaluations /Total valuation (extra effort)	CVA value	Chebyshev build time (s)	Chebyshev run time (s)	Total Chebyshev model run time (s)	CVA Error (%)	Run speed multiplier	Run + build speed multiplier	Total speed multiplier
2x2	1k; 55	16	17	164	0.40%	288 089.03	0.06	0.16	0.71	9.63%	104	78	24
2x3	5k; 60	138	142	360	0.12%	324 792.01	0.15	1.01	3.80	-2.00%	137	123	37
3x3	2k; 55	49	51	495	0.45%	53 163.43	0.21	0.37	1.60	-0.05%	133	88	32
4x4	10k; 40	271	281	640	0.16%	371 239.90	0.13	2.44	9.81	-0.05%	111	110	29

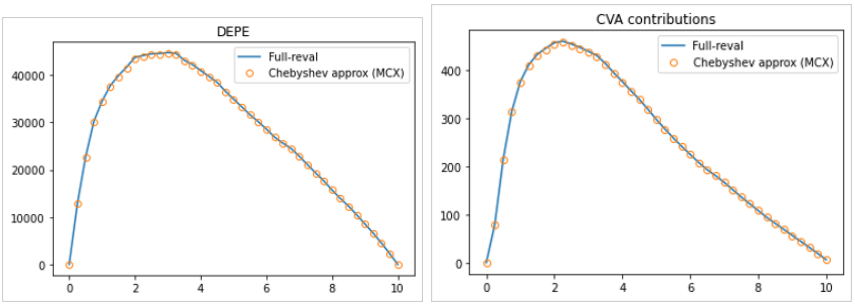
Table A.4: Low dimensional performance results: CVA for IRS

A.2 CCR IIb

A.2.1 Graphs from *Section 5.1.4.2*

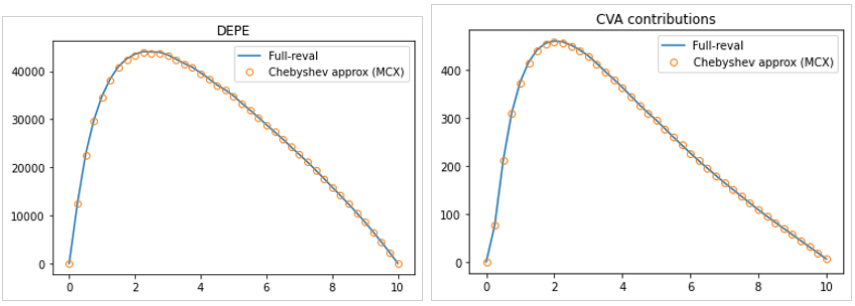
Figure *A.1a*, Figure *A.1b*, Figure *A.2a*, Figure *A.2b*, Figure *A.3a* and Figure *A.3b* show the accuracy of the Chebyshev interpolants at each time step on the determination of the Expected Positive Exposure and CVA contributions relative to the full evaluation method.

Although this method is inefficient implementation of Chebyshev techniques on CVA for IRS, the figures below do demonstrate the accuracy of the Chebyshev interpolants.



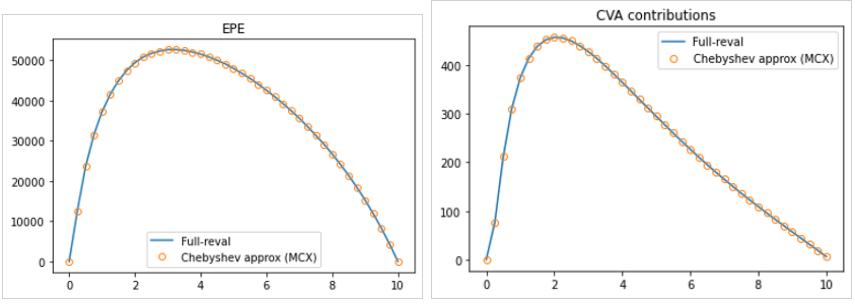
(a) (b)

Figure A.1: Composition methods at each time step: Example 4 with 10k monte carlo runs for CVA calculation



(a) (b)

Figure A.2: Composition methods at each time step: Example 5 with 100k monte carlo runs for CVA calculation



(a) (b)

Figure A.3: Composition methods at each time step: Example 6 with 1M monte carlo runs for CVA calculation

This dissertation reflects the authors' opinions and not necessarily those of their employers or the University of Pretoria.

This report was generated with the MikTeX, version 2.4.1704, and TeXnicCenter, version 1 Beta 7.01, freeware products.