UNIVERSITÉ PARIS XIII - SORBONNE PARIS NORD

École Doctorale Sciences, Technologies, Santé Galilée

Stochastic Financial Modeling and Machine Learning for Risk-Neutral and Real-World Market Indicators, Model Calibration and Data Quality

---

THÈSE DE DOCTORAT

présentée par

Djibril SARR

Laboratoire Analyse, Géométrie et Applications

pour l'obtention du grade de

DOCTEUR EN MATHEMATIQUES APPLIQUEES

Soutenue le 29 Novembre 2024 devant le jury d'examen composé de :

BA Mouhamadou Lamine, Université Cheikh Anta Diop de Dakar ..... Examinateur

BEN ALAYA Mohamed, Université de Rouen Normandie ..... Directeur de thèse

BOURY Frédéric, ... Encadrant CIFRE, Examinateur invité

HU Yueyun, Université Sorbonne Paris Nord ..... Examinateur

JIAO Ying, Institut de Science Financière et d'Assurances ..... Présidente du jury

KEBAIER Ahmed, Université d'Evry ..... Directeur de thèse

PERGAMENCHTCHIKOV Serguei, Université de Rouen Normandie ..... Rapporteur

PHAM NGOC Thanh Mai, Université Sorbonne Paris Nord ..... Examinatrice

SCOTTI Simone, Université Paris Diderot ..... Rapporteur

Remerciements

Je souhaite tout d'abord exprimer ma gratitude profonde et sincère envers mes directeurs de thèse, Pr. KEBAIER Ahmed et Pr. BEN ALAYA Mohamed. Depuis mon diplôme d'ingénieur jusqu'à ce doctorat, vous m'avez fait confiance, guidé et conseillé. Vous avez partagé avec moi vos connaissances remarquables et m'avez aidé à progresser, tant en tant qu'apprenti chercheur qu'en tant que scientifique en devenir, sur de nombreux aspects. Mais le plus précieux à mes yeux, c'est l'inspiration que vous m'avez insufflée. Merci.

Je tiens également à remercier mon encadrant industriel, BOURY Frédéric, fondateur du cabinet FBH Associés. D'abord maître de stage, puis manager, tu es aujourd'hui devenu un mentor et un grand ami. Tu es, sans aucun doute, l'une des personnes les plus modestes, bienveillantes et attentionnées que la vie ait placées sur mon chemin. Je remercie également l'ensemble des équipes de FBH Associés pour leur confiance et leur amitié.

Je tiens à exprimer ma profonde gratitude envers Pr. PERGAMENCHTCHIKOV Serguei et Pr. SCOTTI Simone, qui ont généreusement accepté de rapporter cette thèse. Je vous remercie sincèrement pour le temps précieux que vous avez consacré à la lecture et à l'évaluation de mon travail. Je remercie également les Professeurs BA Mouhamadou Lamine, HU Yueyun, JIAO Ying et PHAM NGOC Thanh Mai pour avoir accepté de faire partie du jury et d'examiner cette thèse.

Je tiens aussi à remercier tous les membres du LAGA de l'Université Sorbonne Paris Nord, avec une mention particulière à Leïla pour sa grande disponibilité et sa patience.

Je souhaite également adresser mes remerciements chaleureux à tous mes amis et proches, notamment, ceux de la MACS, de FBH et de la CEB, pour leur soutien et leur présence tout au long de cette aventure.

Aussi, je remrcie tous ces professeurs et enseignants qui ont éveillé ma curiosité et nourri ma soif de connaissance, depuis le Point E II jusqu'à Sup' Galilée, en passant par le merveilleux Lycée Thierno Saïdou Nourou Tall et Mariste Post-Bac. Votre enseignement et votre passion ont marqué mon parcours.

Enfin, un immense merci à ceux grâce à qui les thèses CIFRE existent et perdurent. Votre engagement est précieux pour la science et la recherche.

Ma ngi sant sama Borom SWT ndaxx bi ma juddoo bah leggi, yermande, suturë akk xewël rekk la ma sangg. Ma ngi Koy sant, di tuub jëmm ci Moom, di julli ci Yonentem bu tedd bi SWS, di ko denk samay wayjuur, sama njaboot akk sama bopp. Diko denk samay niit akk mboleem juliit. Yallah na ñu Yallah yerrëm te jeggal ñu, moom miiy jeggale lepp [39:53].

Sama yaay, sama baay, ma ngi lenn di gërëm, di lenn sant. Awma senn fay. Dajj ngenn lunekk ngir ma nitte, juub te mënnal sama bopp. Yallah na lenn ko suñu Borom fay. Niit yu baax ngenn, royukaay ngenn. Li may gënna magg, di gënna raañee, di gënna giss ne, bi may nekk tuut tank ba fa mi toll nii, coono yi ngenn dajj yëpp ci senn njaboot akk ci jaam bo lenn mënna taxawu, munu ma koo sax natt. Budee balla may tëdd, samay jiko melnañ ni seen boss, sama xell daal. Yallah na lenn Yallah fay, may lenn weergu yaram, fekke akk tall. Yallah na lenn Borom bi yerëm ni ngenn ma yerrëmee bi ma nekke gunne.

Sama soxna, sama xariit. Lijaasa bii ma donn uut, coono la wonn ci yow. Taaxirlu nga, muñ nga. Yallah na la ko suñu Borom fay. Begg naa ci yow, sant na sama Borom ci limuma bolle akk yow. Jamonoy studio Epinay, jamonoy "356" akk "361", jamonooy "Ilknour", jamonoy "Last H bi kañ la ?", bah jamonoy "Yaay jëlli Dud's ?", sant Yallah... Li ci kanam rawul i gëtt saxx. Gëmm naa nee, lii di ñëw moy dakkati ci katanu Sunu Borom SWT. Yaay sama waay. Yallah na ñu suñu Borom juboole, dalal su ñuy xell, dalal su ñuy xoll, xewëlal ñu te samm suñu njaboot. Sama jambaar ji, nakk, xammul li xeww, waaye ci mbir yi la bokk, naxx bërri na lumu ma nee "Parc ?", mane ko "Ecole.", waaye yenna ma gënna taxx di xey akk di daw. Ma ngi ñaan Yallah, lijaasa bii, ammal ñu njeriñ bu raww li ma ci yakaar.

Sama jigeen ñi, suñu taaw ji, magg ju baax, ñekka judd, ñëkka baax. Yaa ñu wann yonn bi. Ñëkka indi doctorat, mashaAllah. Yallah na la Yallah samm, yow akk sa njaboot. Sama xulowaale, waaye sama toffo, sama xarit, bennen docteur biiy waaj inshaAllah, Yallah na nga aggalek sa saggo. Ñaffekatt nga, ca kaw ca kanam rekk, Yallah na la Yallah dëgërëlal. Suñu caat, suñu artiste, gënn ño muuss ñuun ñëep te gënn ño rakaaju, yaay neexal kër gi nakk. Yallah na la suñu Borom deff magg bu amm njeriñ. Jamm akk xewëll ju saxx.

Sama xaarit yi, tamaleeko, ci sama gars yu Saïdou Nourou yii, jamonoy MADS. 6ème bah tay, konn dëgg la. Kenn ci ñuun, neewonn na "Buma lenn xammulwonn, xawma bann xeetu nitt laay donn". Waax nga lepp bro. Bokk ngenn ci lima deffar, Yallah na Yallah dëgëral andd bi te weeyal ko ci suñu njaboot. Ma ngi denkk ñaan yi, akk jërëjëff yii, suñu maame, muy sama niit, sama xarit, sama lepp, ki nga xamne dama koy xalaat, sama xoll tooy, ki nga xamne, daffay deff yiiw, ma ngiss ko topando ko. Ki ma woolu ci sama lepp. Mbaaxam akk jubam, rawee na lu bërri lennen luma giss fennen. Samay xaarit, waa prépa, waa Galilée, waa Cergy, waa Gagny, waa Palaiseau, akk kepp kuma bokkal yonn bii kuma tuddul, jërëjëff. Bo lenn fii nekkul wonn, mu gënna metti.

Samay maam, samay nijaay akk samay yaaye, samay bajën akk samay papë, akk seeni njaboot. Akk samay mbokk yëpp, ma ngi lenn di gërëmm di lenn santt, di ñaan Azawajal yerrëm ñi ci tëdd, te samm ñi fi nekk. Rawatina, sama xarit, sama pépé, "Yallah du juum du jawaale", bi nga ma donn waax, bokk na ci lima tax di goor goorlu akk di moytu, waaye di dunde yakaar. Bugonn naa nga fekke lii, Yallah na la Yallah yerëm, akk juliit ñi tëdd.

Ma ngi koy tejje nakk, ñëpp ñi may jappale ci sama diggante akk Buur bi, denk ko sama oustaz ca Ndakaru akk waa DKN, rawatina sama xarit, li nga ma yeenal, akk fi nga bëgg ma egg, leele, mann saxx du ma ko seña ñaan.

Jërë ngenn jëff.

Abstract

This thesis explores the integration of financial modeling and artificial intelligence (AI) to tackle challenges in the field. Financial models often rely on stochastic processes to simulate market indicators and evaluate pricing. However, with recent advances in AI, these models can be enhanced, providing new alternative approaches, useful in practice. The current work contributes to the development of stochastic models for credit spreads, frameworks for transitioning between risk-neutral (RN) and real-world (RW) measures, AI-driven calibration techniques, and a methodology for data quality measurement. These advancements aim to improve the precision and practical use of financial models.

In the first chapter, a stochastic model for credit spreads is introduced. This chapter develops a continuous term structure of credit spreads instead of using the factorial assumptions present in many existing models. As a by-product, the chapter proposes a calibration approach that aligns model and market volatilities of default intensities, enabling the model's capability to replicate realistic credit spread behavior. Given that most financial models are developed under the RN framework, the natural progression leads us to the second chapter, which extends stochastic financial models from RN to RW measures.

The second chapter provides a framework for transitioning diffusion models from the RN measure, to the RW measure, which better reflects real-world market behaviors. The methodology, based on Girsanov's theorem, is designed to incorporate real-world dynamics, making it crucial for risk management. The framework is built to be applicable to a variety of models, including models with non-additive noise, such as the CIR++ model. We validate the framework's versatility through case studies involving forecasts and stress tests. Since the quality of a model is tied to its calibration, a crucial but often laborious step, the aim of the third chapter is to address its challenges by leveraging AI.

In the third chapter, we focus on contributing to the calibration of financial models using alternative deep learning (DL) methods. We propose a systematic calibration process using fully connected and convolutional neural networks to estimate the parameters of the G2++ model. The obtained results indicate that our DL approach achieves better accuracy and efficiency. The chapter emphasizes that while AI is efficient in improving model calibration, it is heavily dependent on the quality of the data used for training. This brings us to the fourth chapter, which addresses the critical issue of data preprocessing.

In the fourth and final chapter, we develop a framework for automating data quality assessment and enhancement. Our methodology tackles common data defaults such as absence, redundancy, and incoherence by integrating statistical methods with machine learning algorithms. The framework strikes a balance between accuracy and explainability, ensuring transparency in the identification and correction of data anomalies.

Keywords: Quantitative finance, Risk measurement, Credit Risk, Market indicator modeling, Deep calibration, Real-world measure, Data quality.

Résumé

Cette thèse explore l'intégration de la modélisation financière et de l'intelligence artificielle (IA) pour relever les défis dans le domaine. Les modèles financiers utilisent souvent des processus stochastiques pour simuler les indicateurs de marché et évaluer les prix. Toutefois, les récentes avancées en IA permettent de les améliorer, offrant des approches alternatives pratiques. Ce travail développe de modèles stochastiques pour les *spreads de crédit*, des méthodes de transition entre les mesures risque-neutre (RN) et monde-réels (RW), des techniques de calibration basées sur l'IA, et d'une méthodologie pour évaluer la qualité des données. Ces avancées visent à améliorer la précision et l'utilisation des modèles financiers.

Le premier chapitre introduit un modèle stochastique pour la structure par terme continue des *spreads de crédit*. L'approche contourne les hypothèses factorielles présentes dans de nombreux modèles existants. Le chapitre propose une méthode de calibration qui aligne les volatilités des intensités de défaut du modèle avec celles du marché, permettant de reproduire un comportement réaliste des *spreads de crédit*. La plupart des modèles financiers étant développés sous RN, nous sommes dirigés vers le deuxième chapitre, qui étend les modèles financiers stochastiques de la mesure RN à la mesure RW.

Le deuxième chapitre propose une approche pour la transition des modèles de diffusion de RN à RW, qui reflète mieux les comportements réels du marché. La méthodologie, basée sur le théorème de Girsanov, est conçue pour intégrer les dynamiques du monde réel, ce qui est crucial pour la gestion des risques. L'approche est conçue pour être applicable à de nombreux modèles, y compris ceux avec un bruit non additif, comme le modèle CIR++. Nous validons la versatilité de l'approche à travers des scénarios de prévisions et des stress tests. Étant donné que la qualité d'un modèle dépend de sa calibration, étape cruciale mais souvent laborieuse, le but du troisième chapitre est de relever ces défis en exploitant l'IA.

Le troisième chapitre contribue à la calibration des modèles financiers par des méthodes alternatives d'apprentissage profond (DL). Nous proposons un processus systématique de calibration utilisant des réseaux neuronaux linéaires et convolutifs pour estimer les paramètres du modèle G2++. Les résultats obtenus montrent que notre approche atteint une meilleure précision et efficacité. Le chapitre souligne que, bien que l'IA soit efficace pour améliorer la calibration, elle dépend fortement de la qualité des données utilisées pour l'entraînement. Cela nous mène au quatrième chapitre, qui aborde leur prétraitement.

Dans le quatrième chapitre, nous développons une approche pour automatiser l'évaluation et l'amélioration de la qualité des données. Notre méthodologie aborde leurs défauts tels que l'absence, la redondance et l'incohérence en intégrant des méthodes statistiques avec des algorithmes d'apprentissage automatique. L'approche trouve un équilibre entre précision et explicabilité, garantissant la transparence dans l'identification et la correction des anomalies.

Mots clés: Finance quantitative, Mesure du risque, Risque de crédit, Modélisation des indicateurs de marché, Deep calibration, Mesure monde-réel, Qualité des données.

List of papers included in this thesis

This thesis consists of four chapters, each of which is (or is substantially founded on) a standalone research work.

- Chapter II, *Credit Spreads' Term Structure: Stochastic Modeling with CIR++ Intensity*, submitted, 2024, https://arxiv.org/abs/2409.09179.

- Chapter III, *Financial Stochastic Models Diffusion: From Risk-Neutral to Real-World Measure*, submitted, 2024.

- Chapter IV, *Deep Calibration of Interest Rate Models*, submitted, 2024, https://arxiv.org/abs/2110.15133.
  A first version of this work was published in the 2022 annual proceedings of the SFdS, *Société Française de Statistique* (53èmes Journées de statistique de la SFdS, pages 906-911 https://jds22.sciencesconf.org/).

- Chapter V, *Towards Explainable Automated Data Quality Enhancement without Domain Knowledge*, submitted 2024, https://arxiv.org/abs/2409.10139.
  This work is substantially founded on the work presented at the 2021 *Challenge mathématiques et entreprises* of AMIES, *Agence pour les Mathématiques en Interaction avec l'Entreprise et la Société*. This work was awarded the first prize in the data quality competition (https://challenge-maths.sciencesconf.org/?lang=fr).

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Generalities, Motivations and Structure

1.1.1 Generalities

Whether it is measuring risks, valuing products, or finding investment strategies, financial modeling often requires the use of stochastic analysis to construct models and equations that best reflect the realities of financial markets and macroeconomic contexts. Over the years, various models have been developed for this purpose. Some of these models are designed to simulate the behavior of financial and macroeconomic indicators, such as interest rates, credit spreads, exchange rates, and volatilities, which are crucial for achieving these objectives. Other models are specifically intended for evaluating the price of financial products (pricing models). In recent years, artificial intelligence (AI) has increasingly influenced financial modeling across different applications—whether for risk measurement, valuation, finding investment strategies, or other purposes—and across various types of models, including those for indicator simulation, pricing, and beyond. This thesis thus contributes to these questions with a dual aspiration: to advance both financial modeling and the integration of AI into this regard. First, we introduce an approach to model the entire term structure of credit spreads, a crucial market indicator for both default risk and pricing. This initial model is formulated under the Risk-Neutral (RN) probability measure. Subsequently, we present a generic approach for transitioning the diffusion of financial models from the RN measure to the Real-World (RW) measure, enabling, for instance, more sophisticated stress-testing. Accurate and reliable use of any financial model depends on proper calibration. To this end, we leverage the strengths of AI to introduce calibration techniques that facilitate the use of existing models. Finally, recognizing the importance and challenges of data preprocessing, especially when dealing with AI algorithms, we also develop a statistics- and AI-based methodology to automate this often tedious task.

1.1.2 Motivations

As we have briefly mentioned, many financial models aim to diffuse key market indicators over time, i.e., stochastically forecasting their future values. A thorough analysis of the state of the art in diffusion models reveals that not all indicators have received equal attention over time. For instance, while the literature is rich with studies on interest rate models (e.g., Hull-White, LIBOR Market Model, Cox-Ingersoll-Ross, etc.), the same cannot be said for credit spread models. More interestingly, there does not seem to be a unanimously agreed-upon modeling approach—whether factor modeling, stochastic modeling, or otherwise. Furthermore, many existing models do not directly provide the entire term structure of credit spreads, which is a crucial indicator, particularly when analyzing entities' creditworthiness. We therefore deem it interesting to address the following question:

Question 1.1.1. *Can we construct a stochastic diffusion model of the term structure of credit spreads that is both robust and representative of real market behaviors?*

We then observe that most financial models, including the one resulting from the answer to Question 1.1.1, are confined to the risk-neutral (RN) probability measure. This is primarily because most models, even those forecasting indicators, ultimately to address pricing issues, which necessitate the use of the RN measure. However, for other applications, the Real-World (RW) measure is better suited. Effective RW modeling maintains the arbitrage-free behavior of a stochastic model, produces realistic curves that are representative of the indicator being diffused, and, crucially, allows the replication of any arbitrarily given target curve. This last feature is particularly valuable for conducting stress tests and running "what-if" scenarios. Therefore, we are interested in addressing the following question:

Question 1.1.2. *Can we build a generic approach to consistently shift any financial model from the RN measure to the RW measure?*

Independently of the probability space under which financial models are defined, most models are described by parametric equations, often involving stochastic differentials. These models are all the more relevant and efficient as their parameters are chosen in a sufficiently relevant manner to be representative of a given market and macroeconomic context. This parameter assignment procedure is called calibration. Several calibration methods exist. For example, when it is possible to obtain the likelihood of an existing quantity, maximum likelihood estimators can be used. However, it is rather rare to have an expression of this likelihood, especially for complex models. In practice, the most common approach seems to be to minimize the difference between a market quantity and a quantity expressed in the model (price, rate, correlations, etc.). This is therefore an optimization procedure. This procedure is generally tedious, with many local minima, highly dependent on initializations, and sensitive to noise present in the market quantity used in the calibration. Therefore, we ask the following question:

Question 1.1.3. *By using deep learning (DL), can we achieve a systematic calibration*

approach, outperforming the performance of existing calibration methods in terms of ease of use, accuracy, and robustness?

The effective integration of AI enhances financial modeling as well as the use of financial models, as demonstrated by the response provided to Question 1.1.3. Nevertheless, thorough data preprocessing is indispensable. This critical stage ensures data quality and appropriateness for analysis. Starting with data collection from diverse sources, it involves a sequence of cleaning steps to rectify errors and inconsistencies while maximizing usable data. While often laborious and time-consuming, there are methods in literature aiming to automate these cleaning processes, frequently leveraging AI. However, these methods typically rely on users' familiarity with data types and content, making them somewhat specific and less universally applicable. Moreover, employing AI for preprocessing introduces challenges in explaining and interpreting algorithmic outcomes, especially problematic in data preprocessing where justifying the exclusion or modification of certain data points is challenging. Thus, we aim to address the following inquiry:

Question 1.1.4. *Using ML and DL, can we develop a process that automatically assesses the quality of a dataset and corrects entries that require it, without requiring prior knowledge about the dataset's content, while maximizing explainability and interpretability as much as possible?*

1.1.3 Plan

We start by addressing Question 1.1.1 and provide an answer in Chapter 2. This chapter proposes a stochastic model for credit spreads. The model ensures compliance with the no-arbitrage assumption and generates realistic term structure curves of credit spreads. It is aiming to enhance the modeling of credit risk by addressing certain gaps in the existing literature. First, the model is grounded in the diffusion of default intensities using a CIR++ intensity framework within a risk-neutral probability space. Unlike many traditional models that rely on discrete assumptions, this approach provides a continuous, stochastic basis for modeling credit spreads. Second, a key feature of the model is its ability to directly generate a term structure for credit spreads, as well as a term structure for the prices of defaultable bonds. This chapter also introduces a practical calibration method that aligns the model's theoretical volatilities of default intensities with historical ones. The model's accuracy and robustness are assessed against historical credit spread data, demonstrating its capability to produce realistic term structure curves and accurately match the initial implied survival probabilities. Furthermore, the model offers an explicit formula for determining the credit spread at any maturity and future time, providing a useful tool for credit risk or pricing analysis .

Chapter 3 then addresses Question 1.1.2. This chapter introduces a versatile framework for transitioning financial diffusion models from the RN measure, commonly used in derivative pricing, to the RW measure, which more accurately captures actual market behavior and investor preferences by incorporating risk premiums for instance. Leveraging Girsanov's

theorem from probability theory, the framework addresses key challenges in integrating RW dynamics into financial models. These challenges include remaining arbitrage-free, generating realistic term structures of market indicators, and fitting arbitrarily given market curves. The framework is broadly applicable to various diffusion models, including those with non-additive noise, such as the CIR++ model. The framework's robustness and practical relevance are validated through case studies, including Goldman Sachs' 2024 global credit outlook forecasts and the European Banking Authority's 2023 stress tests. Te work in this chapter contributes to the literature as a valuable tool for improving risk measurement. The framework's versatility extends to stress testing and scenario analysis, offering practitioners a powerful tool for evaluating various scenarios and making informed decisions in pricing and risk management strategies. calculating

In the chapter dedicated to Deep Calibration (DC) (Chapter 4), we address Question 1.1.3 by presenting an innovative approach to calibrate the five parameters of the G2++ model using neural networks. Our method begins with generating synthetic datasets based on calibrated parameters from market data for the G2++ model. From these data, we calculate zero-coupon rates and forward rates, as well as their covariances and correlations. Our first model relies on a Fully Connected Neural Network (FCN), utilizing only the covariances and correlations of zero-coupon and forward rates. We highlight that covariances are more suitable than correlations due to the unfeasible backpropagation, which we analyze in this thesis and is the subject of Theorem 4.3.1. The second approach involves a Convolutional Neural Network (CNN) that uses only zero-coupon rates without additional transformations. Our numerical tests demonstrate that our DL-based calibration approach outperforms the classical calibration method used as a reference. Our methods have shown better computational performance on a comparison dataset, with an overall error reduced by more than threefold and a calibration time of less than one second, whereas the classical method required 164 seconds.

Finally, Chapter 5 is dedicated to providing an answer to Question 1.1.4. We begin by providing a definition of valid data. This definition provides a structure for the framework for the pre-processing we wish to automate. We also recall the definitions of explainability and interpretability according to some literature, before adapting them to our context of data quality improvement and measurement. These recontextualized definitions will help define the tolerance towards the understanding of results that a user may have when employing our work developed throughout the chapter. We then propose a procedure for detecting invalid entries in a dataset. This involves a succession of statistical and ML algorithms, including duplicate elimination, detection of statistical anomalies (outliers), identification of typographical errors, and logic error detection. The procedure enables the localization of invalid entries, understanding why they were invalidated, and providing fixes if necessary. Our results show that statistical and ML techniques indeed allow for the evaluation of data quality while maintaining a good level of transparency and enabling automation. However, it is interesting to note that most of the required tools are mining tools which, by their very construction (especially high dimensionality and input transformation), reduce explainability and interpretability. Therefore, this chapter aims to find the right balance

between transparency and performance. This balance is achieved by combining standard statistical methods with ML-based methods rather than favoring only one approach.

1.2 Chapter II: Credit spreads' term structure: stochastic modeling with CIR++ intensity

Credit spreads are crucial in finance, primarily serving as measures of creditworthiness of entities, with high spreads indicating high perceived credit risk and low spreads suggesting lower risks. They are also essential for pricing market instruments, as discount factors, denoted $d(t,T)$ can depend on credit spreads, $\mathrm{Sp(t,T)}$ and zero-coupon rates, $\mathrm{Z(t,T)}$, with $d(t,T) = e^{-(T-t)[\mathrm{Z(t,T)} + \mathrm{Sp(t,T)}]}$. Reliable models for credit spreads are needed for risk assessments, including Credit Valuation Adjustment (CVA) (BCBS and BIS, 2011), and can serve as macroeconomic indicators (Akinci and Queralto, 2022; De Santis, 2016). This work introduces a new stochastic model for credit spreads using a CIR++ (shifted Cox Ingersoll Ross) intensity model to diffuse default intensities, leading to credit spread computation. The model, driven by a classic stochastic differential equation (SDE), retains favorable properties, providing a term structure of credit spreads and fitting the market-implied survival probability curve at $t = 0$. The model offers a closed-form credit spread formula, easily calibrated, and produces realistic dynamics akin to market behavior, as discussed in Section 2.4. Additionally, the model provides an analytical expression for defaultable bond pricing, critical for credit risk assessment and portfolio management. Existing literature primarily consists of factorial models that explain spread variations through other variables (Davies, 2008; Manzoni, 2002). Researchers have extensively analyzed various indicators to understand credit spread determinants. Notably, factors like T-bill rates, equity prices, and industrial production have been found useful, as demonstrated by studies such as (Davies, 2008). Additionally, auto-regressive models, incorporating factors like FTSE returns, term spreads, exchange rates, have been employed by researchers like those in (Manzoni, 2002) to further investigate credit spread determinants. These factorial models are advantageous for their ease of manipulation and interpretation but may lack the native term structures and continuous-time dynamics found in SDE-driven models. In contrast, SDE-driven models inherently provide these features, making them well-suited for applications in risk management and investment strategies. The native term structure in SDE-driven models allows for the continuous monitoring of the indicator being considered, over different maturities, providing a more granular view. This feature is particularly useful when entities need to evaluate potential shocks or movements at various points on said indicator. Therefore, it is a common requirement in risk management as well as in pricing issues.

1.2.1 General modeling setting and the CIR++ intensity model

The approach involves two primary steps: modeling default intensities using a CIR++ framework and deriving the corresponding credit spread equation.

In this model, let τ represent the default time and $T > 0$ the time horizon. The model operates in a risk-neutral framework within a filtered probability space $\left(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{Q}\right)$, where the filtration $(\mathcal{F}_t)_{t\in[0,T]}$ reflects the evolution of the default-free market, and $\mathbb{Q}$ is the risk-neutral measure. The risk-free rate is denoted by $r(t) = r_t$, and the default intensity by $\lambda(t) = \lambda_t$. Both processes are $\mathcal{F}_t$-measurable and conditionally independent. The filtration is extended to include default information as $\mathcal{G}_t = \mathcal{F}_t \vee \sigma(\{\tau < s\},\ s \leq t)$, in line with other models (Chiarella et al., 2011; Bielecki et al., 2011). Under the CIR++ intensity model, the default intensity $\lambda(t)$ is characterized as follow:

$$\begin{cases} \lambda(t) = y(t) + \psi(t) \\ dy(t) = \kappa\left(\theta - y(t)\right)dt + \sigma\sqrt{y(t)}dW_t, \end{cases} \tag{1.1}$$

where $(W_t)_{t\in[0,T]}$ is a $\mathbb{Q}$-Brownian motion, with $\kappa, \theta, \sigma > 0$ satisfying the Feller condition $2\kappa\theta \geq \sigma^2$ to ensure $y(t) = y_t$ remains positive. The function $\psi(t)$ is used to align with initial market-implied survival probabilities. The CIR model is advantageous as it naturally ensures positive default intensities and aligns with standard practices (Brigo and Alfonsi, 2005; Mbaye and Vrins, 2018). There is an equivalence between the survival probability $S(t,T)$ in this intensity model and the zero-coupon bond price $P(t,T)$ in short-rate models, both expressible as Laplace transforms of CIR-type processes (Brigo and Mercurio, 2006). Especially, the price of a risk-free zero-coupon bond is: $P(t,T) = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\middle|\mathcal{F}_t\right]$, and the survival probability is: $S(t,T) = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T \lambda_s ds\right)\middle|\mathcal{F}_t\right]$.

1.2.2 Credit spread term structure modeling

The developments conducted in Section 2.2 of Chapter 2 provide the analytical expressions for the credit spread, $\mathrm{Sp}(t,T)$, representing the yield difference between a risky bond and a risk-free bond. The following result is given by Theorem 2.3.1:

$$\begin{cases} \mathrm{Sp}(t,T) = -\dfrac{1}{T-t}\ln\left[\delta + (1-\delta)\dfrac{S^m(0,T)}{S^m(0,t)}\dfrac{A(0,t)}{A(0,T)}\dfrac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}}A(t,T)e^{-B(t,T)[\lambda(t)-\psi(t)]}\right] \\ \psi(t) = \lambda^m(t) + D(t) - y_0 E(t). \end{cases} \tag{1.2}$$

In both theorems, $S^m(0,T)$ is the initial market implied survival probability for the horizon T. The functions $A(t,T)$, $B(t,T)$, $D(t,T)$, and $E(t,T)$ are defined by:

$$A(t,T) = \left(\frac{2he^{\frac{1}{2}(\kappa+h)(T-t)}}{2h + (\kappa+h)\left(e^{(T-t)h}-1\right)}\right)^{\frac{2\kappa\theta}{\sigma^2}}, B(t,T) = \frac{2\left(e^{(T-t)h}-1\right)}{2h + (\kappa+h)\left(e^{(T-t)h}-1\right)},$$
$$h = \sqrt{\kappa^2 + 2\sigma^2},\ D(t) = \frac{d}{dt}ln(A(0,t)),\ \text{and}\ E(t) = \frac{d}{dt}B(0,t).$$

1.2.3 Calibration of the model

Calibrating the credit spread model is essential to ensure that its parameters accurately represent the specific context. Various calibration methods exist, including deep calibration, which we have applied to the CIR intensity model (see Section 4.5 of Chapter 4). Here, we propose a calibration approach focused on risk management. For pricing especially, calibration can involve fitting recent credit spreads or asset prices, similar to interest rate models' calibration. However, for risk assessment, it is crucial to capture the full range of historical credit spread variations. To achieve this, we propose calibrating the model based on the historical volatility of the market default intensity curve. This approach ensures that the model captures a broad range of market variations, aligning with the observed historical volatility of credit spreads. The model's default intensity variance is expressed as:

$$\mathrm{Var}_\lambda(T;\kappa,\theta,\sigma,y_0) = y_0\frac{\sigma^2}{\kappa}(e^{-\kappa T}-e^{-2\kappa T}) + \frac{\theta\sigma^2}{2\kappa}(1-e^{-\kappa T})^2. \tag{1.3}$$

Let $\Theta = \{\kappa,\theta,\sigma,y_0\}$ bet the set of parameters to calibrate and $\Theta^* = \{\kappa^*,\theta^*,\sigma^*,y_0^*\}$ the set of calibrated parameter. Let M be the number of maturities used for the calibration. We calibrate on the volatilities for each of the M maturities $\{T_i\}_{i\in[\![1,M]\!]}$. We denote by $\mathrm{Vol}_\lambda^m(T)$ the market volatility of the default intensity. The objective is to minimize the error between theoretical and market volatilities using the Sum of Squared Relative Error (SSRE). Setting $\mathrm{Vol}_\lambda(T;\dots) = \sqrt{\mathrm{Var}_\lambda(T;\dots)}$, we are interested in:

$$\Theta^* = \underset{\Theta}{arg\,min}\sum_{i=1}^{M}\left[\frac{\mathrm{Vol}_\lambda^m(T_i)-\mathrm{Vol}_\lambda(T_i;\,\Theta)}{\mathrm{Vol}_\lambda^m(T_i)}\right]^2. \tag{1.4}$$

For our analysis, we focus on the credit spread of the French bank Credit Agricole. We utilize daily data on Credit Agricole's risky bond yield and the French government bond yield. Data covers the period from January 2009 to January 2024, with CDS curves used to bootstrap default intensities and survival probabilities. We use the maximum observed weekly volatility over a 51-week period as a conservative estimate for historical volatility. Given the extensive dataset at our disposal and our risk management orientation, we conduct two calibrations:

- **Global Scenario**: Reflects fifteen years of market conditions, including periods of stress like the European/Greek Government Debt Crisis.
- **Present Scenario**: Focuses on the last two years, excluding extreme variations but encompassing the post-pandemic inflationary period.

Parameters in the Present Scenario show indeed lower volatility and convergence values compared to the global scenario. Calibration results are summarized in the table below:

Scenario	κ	θ	σ	y_0
Global	5.138×10^{-1}	1.497×10^{-2}	8.904×10^{-2}	4.348×10^{-2}
Present	9.186×10^{-2}	5.519×10^{-4}	1.006×10^{-2}	3.074×10^{-2}

Table 1.1: Parameters calibrated for *Global and Present scenario*

1.2.4 Diffusion of credit spreads and back-testing the model

Now that we have established and calibrated the model, we proceed to propagate the credit spreads. This propagation follows a straightforward process outlined by Equation (1.2). Initially, we diffuse the default intensity $\lambda(t)$, employing the actual distribution of the CIR equation solution, $y(t)$, and subsequently, we incorporate the function $\psi(t)$. The probability distribution of $y(t)$ is given in (Cox et al., 2005) and is the noncentral chi-square, $\chi^2[2cy(s); 2q+2, 2w]$, with $2q+2$ degrees of freedom and parameter of noncentrality $2w$, where $c := \frac{2\kappa}{\sigma^2\left(1-e^{-\kappa(t-s)}\right)}$, $w := cy(s)e^{-\kappa(t-s)}$, $v := cy(t)$, and $q := \frac{2\kappa\theta}{\sigma^2} - 1$. We initialize the market configuration as of January 1st, 2024. Employing the described simulation process, we generate 20,000 paths of credit spreads over a horizon of $T = 2$ years with a weekly time step. Figure 1.1 illustrates the average simulated credit (its expectation) spreads at the initial time (week 0) and at weeks 25, 50, 75, and 100. Additionally, Figure 1.2 displays the 10% and 90% quantiles of the simulated credit spread term structure at weeks 25, 50, 75, and 100. The graphics' values are in Basis Points (BP).

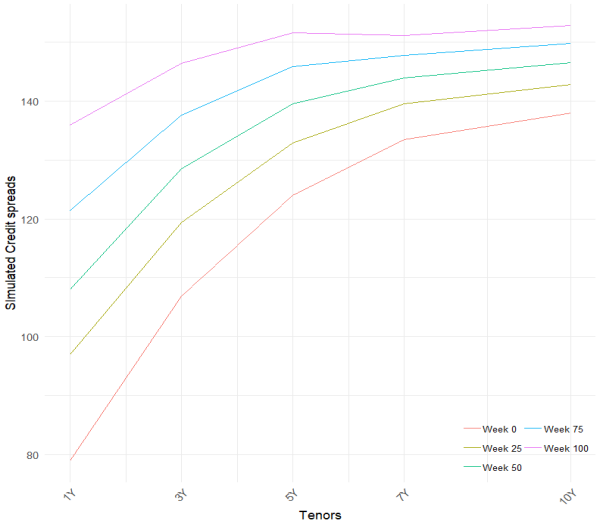

Figure 1.1: In BP, expectation of the term structure of simulated credit spreads

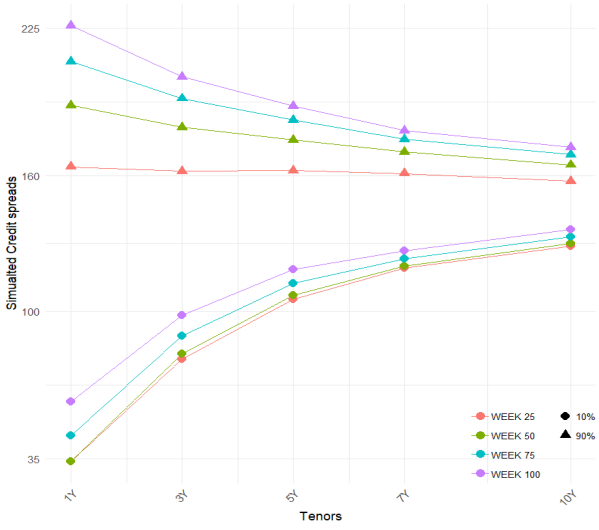

Figure 1.2: In BP, quantiles 10% and 90% of simulated term structured of credit spreads

These initial graphics serve as a preliminary back-test of our model. Indeed, they

demonstrate that our model produces realistic term structures of credit spreads. We also proceed to a more rigorous back-testing process to further validate the model's performance.

We selected a 200-week period from January 1, 2020, to January 1, 2024, to compare the actual observed credit spreads with the values generated by the model. Starting from the market conditions at the beginning of this period, we simulated 20,000 paths of credit spreads using the diffusion process described earlier, with the Gloabl Scenario's parameters. For a specific maturity, such as the 5-year tenor, we evaluated how various quantiles of the simulated spreads encompassed the actual observed values. Figure 1.3 shows the 5-year credit spreads and the quantiles 1%, 10%, 20%, 30%, 70%, 80%, 90%, and 99% of the simulated credit spreads. At the 99% quantile level, only one historical value exceeds our simulation, demonstrating the model's conservative nature and the realism of the credit spread values. However, it is important to note that this back-test is highly restrictive and demanding. Credit spread time series may exhibit jumps, drops, or significant slope changes that might not be fully captured over time, even if the model remained coherent in terms of credit spread term structures or value ranges. The favorable results observed in Figure 1.3 are largely due to the conservative calibration approach discussed in Section 2.4 of Chapter 2, including the selection of the calibration period and the representative volatility value.

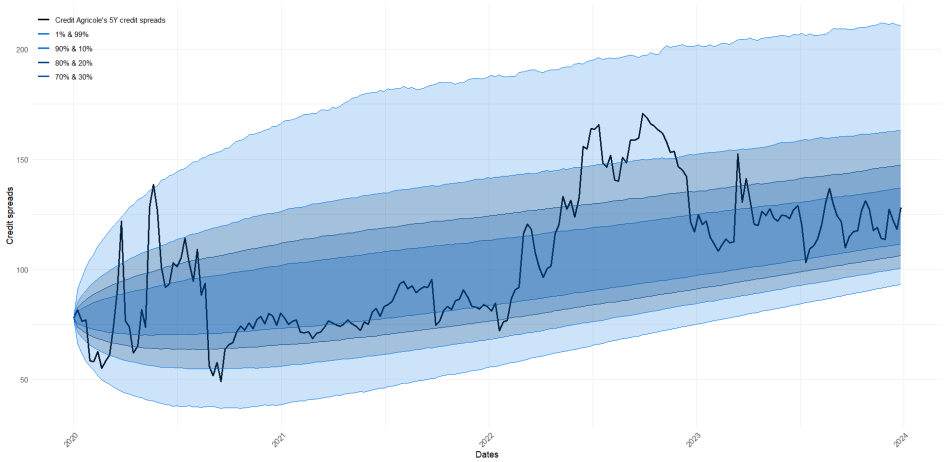

Figure 1.3: In BP, 5Y Credit Agricole's credit spreads observed values (black) and quantiles (shades of blue).

1.3 Chapter III: Financial Stochastic Models Diffusion: From Risk Neutral to Real World Measure

Stochastic financial modeling has traditionally employed the risk-neutral (RN) measure $\mathbb{Q}$, which is essential in financial mathematics in general. This approach is preferred for market-related applications, such as instrument pricing and market risk assessments,

because it ensures arbitrage-free modeling and allows discounted prices to be martingales (Black and Scholes, 1973). In contrast, the Real World (RW) measure $\mathbb{P}$ aims to reflect actual market behavior, for instance by including risk premiums, thus providing a more realistic representation of market dynamics (Berninger and Pfeiffer, 2021). The RW measure is crucial for applications requiring realistic scenarios, such as regulatory compliance, capital requirements, and long-term financial planning. For instance, the Basel II Framework recommends using RW probability for CVA computations, although RN models are often used due to the complexities of RW modeling (Hull et al., 2014). Desirable properties of RW models include being arbitrage-free, producing realistic dynamics, and being capable of matching predetermined curves, as developed in (Norman, 2009) in their work on the RW diffusion of the Brace-Gatarek-Musiela model (Brace et al., 1997). Our work introduces a general framework for transitioning from RN to RW using change-of-measure techniques. This approach is applicable to various diffusion models, including those with non-additive noise like the CIR model, and addressing a broad range of market indicators beyond just interest rates, towards which the literature on RW modeling has been primarily focused. For example, (Berninger and Pfeiffer, 2021) developed a framework for calibrating the G2++ model under both RN and RW measures, emphasizing time-dependent market prices of risk to improve long-term interest rate forecasts. Similarly, (Bruti-Liberati et al., 2010) explored RW interest rate term structure models incorporating jump-diffusion processes and real-world trends in derivative pricing, while (Barker et al., 2016) compared parametric and non-parametric approaches for RW measure simulation within Monte Carlo frameworks. A review of the literature highlights that RW modeling has not received as much attention as RN modeling has, and most of the work has been applied to interest rate forecasting, often focusing on calibrating the market price of risk. Our work aims to extend this by developing a generic framework and applying it to the credit spread model developed in Chapter 2.

1.3.1 The approach and theoretical results

The risk-neutral measure is a probability measure in which the discounted price of a financial asset is a martingale, essential for valuing derivatives under no-arbitrage conditions, in contrast to the real-world measure that incorporates investors' risk preferences and expected growth rates.

Expressing a chosen indicator under the RW measure can be complex, particularly with non-additive noise. Therefore, in our theoretical framework, we begin by eliminating non-additive noise through a Lamperti transformation. Our approach involves expressing the transformed indicator under the RW measure as a function of (i) the transformed indicator under the RN measure and (ii) a parametric function that requires calibration. After deriving these relations, we can reverse the transformation to express the indicator under the RW measure in terms of (i) the indicator under the RN measure, (ii) the calibrated parametric function, and (iii) any additional terms arising from the reverse transformation. Once this formulation is achieved, we can simulate the chosen indicator under the RW

measure by calibrating the parametric function and simulating the indicator under the RN measure, thereby fitting any specified values. The ultimate aim of this work is to derive the relationship between the indicator under RN and RW. This is done in Theorem 3.2.1.

The risk-neutral model is defined in a probability space $\left(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{Q}\right)$, with $(W_t)_{t\in[0,T]}$, a $\mathcal{F}_t$-Brownian motion under $\mathbb{Q}$. The corresponding Real World measure is denoted by $\mathbb{P}$. Consider the process $(Y_t)_{t\in[0,T]}$ in RN, satisfying:

$$dY_t = b(Y_t)dt + \sigma(Y_t)dW_t, \quad Y_0 = y \in \mathbb{R}, \tag{1.5}$$

where b and σ are locally Lipschitz continuous. We transform (Y_t) into (X_t), using a Lamperti transform. For $\phi(y) = \int_{y_0}^{y} \frac{1}{\sigma(x)}dx$, if $\sigma \in \mathcal{C}^1$, then by the Lamperti transform, $X_t = \phi(Y_t)$ satisfies the stochastic differential equation:

$$dX_t = L(X_t)dt + dW_t, \quad X_0 = \phi(y),$$

where $L(x) = \left(\frac{b}{\sigma} - \frac{\sigma'}{2}\right)(\phi^{-1}(x))$. While still keeping the relation $X_t = \phi(Y_t)$, we will consider the more generic framework $(X_t)_{t\geq 0}$, solution to the SDE with additive noise defined on $I = (c, +\infty)$, $c \in [-\infty, +\infty[$:

$$dX_t = L(X_t)dt + \zeta dW_t, \quad t \geq 0,\ X_0 = x \in I, \zeta \in \mathbb{R}.$$

L is assumed to meet the following monotonicity condition:

$$L : I \longrightarrow \mathbb{R} \ \textit{is}\ C^2, \ \text{such that}\ \exists\, K > 0,\ \forall x, x' \in I, x \leq x', L(x') - L(x) \leq K(x' - x). \tag{1.6}$$

Let us consider a process $(X_t^*)_{0\leq t\leq T}$ under $\mathbb{Q}$:

$$dX_t^* = \left[-\vartheta(X_u^* - X_u - \alpha_u) + L(X_u)\right]dt + \zeta dW_t, \tag{1.7}$$

where φ_t is an adapted function defined as:

$$\varphi_t = \frac{1}{\zeta}\left[-(L(X_t^*) - L(X_t)) - \vartheta(X_t^* - X_t - \alpha_t)\right], \tag{1.8}$$

with $\vartheta \in \mathbb{R}$ and α_t a deterministic function. If Novikov's condition holds, there exists a probability measure $\mathbb{P}^\varphi$ under which

$$W_t^* = W_t + \int_0^t \varphi_u du, \tag{1.9}$$

is a Brownian motion under $\mathbb{P}^\varphi$ and under this measure, the following two hold:

- The process $(X_t)^*$ is a solution of

$$dX_t^* = L(X_t^*)dt + \zeta dW_t^*, \quad t \geq 0,\ X_0^* = x \in I. \tag{1.10}$$

- The process $(X_t)^*$ verifies the following relation with (X_t)

$$X_t^* = X_t + \vartheta \int_s^t \alpha_u e^{-\vartheta(t-u)} du, \text{ for all } 0 \leq s < t \leq T. \tag{1.11}$$

Once this result is established, we can reverse the relation given by equation (1.11):

$$Y_t^* = \phi^{-1}\left(\phi(Y_t) + \vartheta \int_s^t \alpha_u e^{-\vartheta(t-u)} du\right), \tag{1.12}$$

thus, establishing a relation between (Y_t) and (Y_t^*).

1.3.2 Application to RW modeling of credit spreads

We start by recalling equation (1.1) that gives the dynamic of the default intensity $\lambda(t) = \lambda_t$ under the risk-neutral measure:

$$\begin{cases} \lambda(t) = y(t) + \psi(t) \\ dy(t) = \kappa\left(\theta - y(t)\right) dt + \sigma\sqrt{y(t)} dW_t. \end{cases}$$

Similarly, under the real-world measure $\mathbb{P}$, there exists a CIR++ diffusion of the RW default intensity, $\lambda^*(t) = \lambda_t^*$ written:

$$\begin{cases} \lambda^*(t) = y^*(t) + \psi(t) \\ dy^*(t) = \kappa\left(\theta - y^*(t)\right) dt + \sigma\sqrt{y^*(t)} dW_t^*, \end{cases} \tag{1.13}$$

with $y^*(0) = y_0^* > 0$. We can re-write for $x_t = \Upsilon\sqrt{y_t} = \Upsilon\sqrt{y(t)}$ (we will similarly consider $x_t^* = \Upsilon\sqrt{y_t^*} = \Upsilon\sqrt{y(t)^*}$), with $\Upsilon \in (0, 2^{\frac{1}{4}})$ and such that $\kappa\theta > \left[\frac{1-\Upsilon^2\frac{\sqrt{2}}{2}}{1-\Upsilon^2\sqrt{2}}\right]\frac{1}{2}\sigma^2$:

$$dx_t = \frac{1}{2}\left[\Upsilon^2\left(\kappa\theta - \frac{1}{4}\sigma^2\right)\frac{1}{x_t} - \kappa x_t\right] dt + \frac{1}{2}\sigma dW_t. \tag{1.14}$$

Note that, this new condition on $\kappa\theta$ slightly strengthens the previous Feller condition, $2\kappa\theta \geq \sigma^2$ since we can choose Υ as small as possible. Vis-à-vis the previous expressions, we can therefore identify $Y(t) = y(t)$, $X(t) = x(t)$, $L(x) = \frac{1}{2}\left[\Upsilon^2\left(\kappa\theta - \frac{1}{4}\sigma^2\right)\frac{1}{x} - \kappa x\right]$, $\zeta = \frac{1}{2}\sigma$, $\phi(x) = \Upsilon\sqrt{x}$ and for $\vartheta = \frac{1}{2}\kappa$:

$$\varphi_t = -\frac{\Upsilon^2}{\sigma}\left[\left(\kappa\theta - \frac{1}{4}\sigma^2\right)\left(\frac{1}{x_t^*} - \frac{1}{x_t}\right) - \kappa\alpha_t\right].$$

According to the previous developments, and especially based on the results in Theorem 3.2.1, after verifying Novikov's condition for the function φ_t, we will be able to derive the relation between $\lambda(t)$ and $\lambda^*(t)$, and we will even be able to go further and derive the

relation between the credit spread under RN and RW measures. We therefore want to verify that:

$$\mathbb{E}^{\mathbb{Q}}\left[e^{\frac{1}{2}\int_0^T \varphi_u^2 du}\right] < \infty.$$

Using the comparison theorem, we show that verifying Novikov's condition, for a positive φ_t and $\gamma \in \mathbb{R}$, can be done by verifying instead that

$$\mathbb{E}^{\mathbb{Q}}\left[\exp\left(\Upsilon^4\gamma^2\int_0^T \frac{1}{(x_u)^2}du\right)\right] < \infty.$$

Since $(x_u)^2 = y_u$, in practical terms, the problem consists of defining the domain of convergence $\mathcal{D}_t$ of the moment-generating function of the integral of a solution, $\hat{y}_t$, of equation (1.1). We show in Theorem 3.3.1 that this domain of convergence indeed contains positive values, including γ^2. Indeed, we show that

$$D_t = \left\{\mu \in \mathbb{R},\ \mathbb{E}\left[\exp\left(\mu\int_0^T \frac{1}{\hat{y}_u}du\right)\right] < \infty\right\} = \left\{\mu, \mu < \frac{1}{2\sigma^2}\left(\kappa\theta - \frac{1}{2}\sigma^2\right)^2\right\}.$$

Therefore, we can apply Theorem 3.2.1 and write:

$$x_t^* = x_t + \frac{1}{2}\kappa\int_s^t \alpha_u e^{-\frac{1}{2}\kappa(t-u)}du. \tag{1.15}$$

Noting $f(t) = \frac{1}{2}\kappa\int_s^t \alpha_u e^{-\frac{1}{2}\kappa(t-u)}du$, we can naturally draw the relation between the real world and risk neutral default intensities. For numerical tests, we take $\Upsilon = 1$ as it simplifies calculations and has no impact on the numerical results. It leads to $x_t = \sqrt{y_t}$ and $\lambda_t = y_t + \psi_t$, therefore:

$$\lambda_t^* = \lambda_t + f_t^2 + 2f_t\sqrt{y_t}. \tag{1.16}$$

Theorem 3.3.2 establishes the relationship between the real-world term structure of credit spreads, $\mathrm{Sp}^*(t,T)$, and the risk-neutral term structure, $\mathrm{Sp}(t,T)$, under our setup, given by:

$$e^{-(T-t)\,\mathrm{Sp}^*(t,T)} = e^{-B(t,T)F(t,\sqrt{y_t})}\left[-\delta\left(1 - e^{B(t,T)F(t,\sqrt{y_t})}\right) + e^{-(T-t)\,\mathrm{Sp}(t,T)}\right], \tag{1.17}$$

where $F(t,\sqrt{y_t}) := f_t^2 + 2f_t\sqrt{y_t}$. This relation ensures that our RW model is arbitrage-free, as it is built on an arbitrage-free RN model, following Theorem 2.7 of (Harrison and Pliska, 1981). To fit any given market credit spread curve, we transition through the cumulative hazard rate curve. This choice is made for two reasons: first, the credit spreads model introduced in Chapter 2 fits the initial term structure of survival probabilities, or identically, the cumulative hazard rate; the second reason is that it allows for a linear relation between RN and RW. The cumulative hazard rate at time t and for the time horizon T is related to the survival probability, $S(t,T)$, by the following equation: $S(t,T) = e^{-\Lambda(t,T)}$. Chapter 2

gives the expression of the RW survival probabilities in our credit spread model:

$$S^*(t,T) = \frac{S^m(0,T)}{S^m(0,t)} \frac{A(0,t)}{A(0,T)} \frac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}} A(t,T) e^{-B(t,T)[\lambda^*(t)-\psi(t)]}. \tag{1.18}$$

The expressions of $A(t,T)$ and $B(t,T)$ are the same as previously defined. This allows us to write under $\mathbb{Q}$ the relation between the credit spread and the cumulative hazard rate:

$$\mathrm{Sp}^*(t,T) = -\frac{1}{T-t} \ln \left[\delta + (1-\delta) e^{-\Lambda^*(t,T)} \right]. \tag{1.19}$$

Note that equations (1.18) and (1.19) are written for RW quantities but could have been identically written for quantities under RN. This is because, in our setting, both under RN and RW, the default intensity is described by a CIR++ intensity with the same parameters in both cases. Theorem 3.3.3 provides the linear relation between the real-world cumulative hazard rate $\Lambda^*(t,T)$ and its risk-neutral counterpart $\Lambda(t,T)$:

$$\Lambda^*(t,T) = \Lambda(t,T) + B(t,T) F(t, \sqrt{y_t}). \tag{1.20}$$

This equation allows us to calibrate the function $\alpha(t)$ to ensure that the targeted cumulative hazard rate is matched in expectation. To achieve this feature, we calibrate the function to ensure that the expectation of $\Lambda^*(t,T)$ in the real world space corresponds to cumulative hazard rate anticipation / stress values / forecasts, $\{c_i\}_{i \in [\![1,N]\!]}$. In other terms, we are willing to obtain an equation to solve for $\alpha(u)$ that would guarantee for a chosen maturity $\widetilde{T}$ and for a set of N dates $\{t_i\}_{i \in [\![1,N]\!]}$, $\mathbb{E}^{\mathbb{P}} \Lambda^*(t_i, \widetilde{T}) = c_i, \ \forall i \in [\![1,N]\!]$ and with $t_i \in [0,T], \ \forall i \in [\![1,N]\!]$. We therefore solve for $\alpha(t)$

$$\left[c_i - \mathbb{E}^{\mathbb{P}} \Lambda(t_i, \widetilde{T}) \right] \frac{1}{B(t_i, \widetilde{T})} = f^2(t_i) + 2 f(t_i) \mathbb{E}^{\mathbb{P}} \sqrt{y(t_i)}, \tag{1.21}$$

with $f(t) = \frac{1}{2}\kappa \int_s^t \alpha_u e^{-\frac{1}{2}\kappa(t-u)} du$. Berninger and Pfeiffer (2021) make a comparative analysis on the form to give to the function $\alpha(t)$. We employ a step constant function, which allows for time-dependent parameters without introducing too much complexity while achieving good and stable results. Practitioners can then simulate RW credit spreads by selecting target cumulative hazard rates (deduced from credit spreads target), simulating $\Lambda(t,T)$ under RN, inferring $\Lambda^*(t,T)$ using (1.20), and finally obtaining the RW credit spreads term structure via (1.19). This process is practical for regulatory stress scenarios and allows for precise fitting of arbitrarily chosen curves, enhancing risk management and forecasting capabilities.

1.3.3 Numerical results

We conducted two analyses to assess the behavior of our RW modeling of credit spreads. As in Chapter 2, we use Credit Agricole's credit spreads. We have two application scenarios that correspond to two different possible uses of our approach: Goldman Sachs' 2024 Global Credit Outlook and the European Banking Authority (EBA)'s 2023 EU-wide stress test.

The first case corresponds to a scenario where one has forecasts of credit spreads and wants to see how the term structure responds to these forecasts. In the second case, one receives stress scenarios and observes the deformation of the term structure of credit spreads in response. In both cases, these results could be complemented by measuring how these modifications in the term structure affect the value of a given portfolio, either by full revaluation or by sensitivities, for instance.

Goldman Sachs' 2024 Global Credit Outlook: We first applied the credit spreads forecasts from Goldman Sachs' 2024 report, which anticipates a slight decrease in European financial issuers' credit spreads. Although these forecasts are not specific to individual issuers like Credit Agricole, we adapted the relative changes to the 5-year credit spread for Credit Agricole. The current value in the fourth quarter of 2023 is 113, which decreases to 109 in the first quarter of 2024, 107 in the second quarter of 2024, 105 in the third quarter of 2024, and finally reaches 103 in the fourth quarter of 2024. These projections were translated into cumulative hazard rates and simulated over 20,000 paths with a weekly time step starting from January 1, 2024. Figures 1.4 and 1.5 show that the expected simulated 5-year credit spreads closely follow the target values, while the term structure of credit spreads consistently decreases, reflecting the anticipated real-world behavior. The model ensures that the resulting credit spreads remain realistic across the entire term structure.

EBA's 2023 EU-wide Stress Test: Next, we examined Credit Agricole's credit spreads under the EBA's 2023 EU-wide stress test, which imposes a stress value of 133 basis points (BPs) for French financial institutions like Credit Agricole. We modeled a linear progression of this stress over a one-year period. Similar to the Goldman Sachs scenario, we conducted 20,000 simulations with weekly intervals. Figures 1.6 and 1.7 illustrate that the simulated 5-year credit spreads align closely with the targeted stressed values, and the term structure shows realistic behavior even under stress, including an inversion of the term structure, which is observed during market stress conditions.

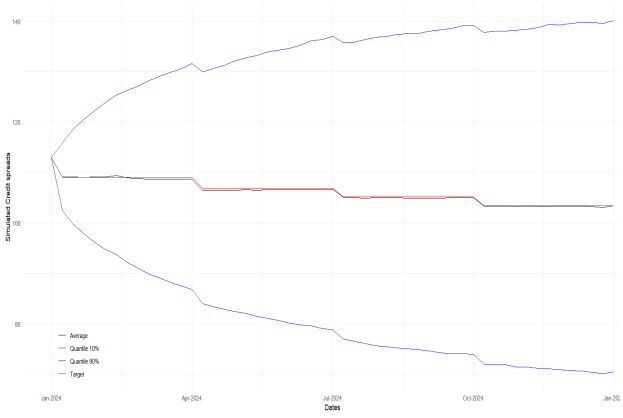

Figure 1.4: In BP, 5-year credit spreads of Credit Agricole: simulations (expectation, 10th and 90th percentiles) and targets (forecast).

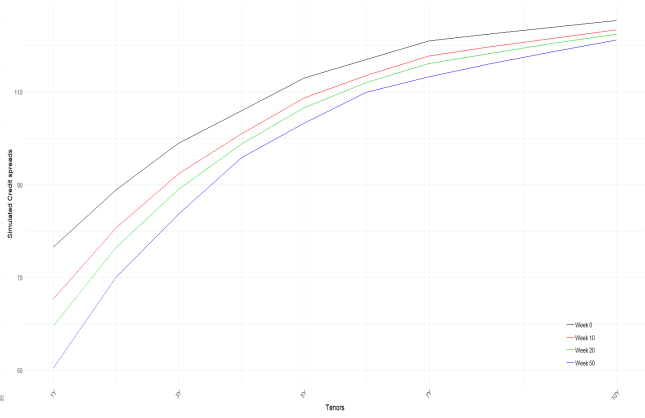

Figure 1.5: In BP, the expected term structure of credit spreads for Crédit Agricole under the forecast scenario at various future dates.

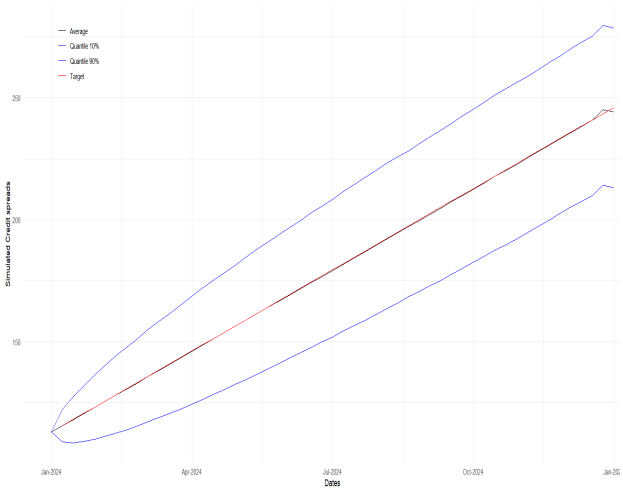

Figure 1.6: In BP, 5-year credit spreads of Credit Agricole: simulations (expectation, 10th and 90th percentiles) and targets (stress tests).

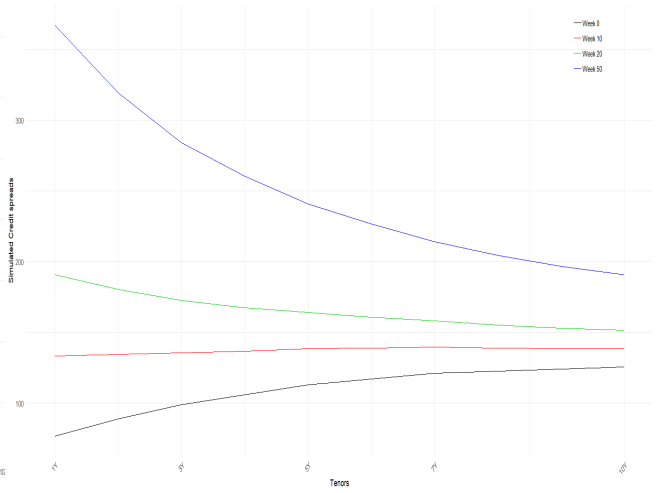

Figure 1.7: In BP, the expected term structure of credit spreads for Crédit Agricole under the stress test scenario at various future dates.

Both analyses demonstrate the model's ability to realistically simulate the behavior of credit spreads under different scenarios, whether it is a forecasted decline or a regulatory stress scenario. The results show that the model produces realistic and consistent term structures, confirming its robustness for stress testing and forecasting in real-world applications.

1.4 Chapter IV: Deep Calibration of interest rate models

Financial institutions face the challenge of managing market and macroeconomic indicators, necessitating a deep understanding of their behavior for risk management and investment optimization. Key drivers like interest rates (IR) profoundly impact assets, including IR derivatives such as swaps, swaptions, and cross-currency swaps. A variety of IR models, including the Vasicek model (Vasicek, 1977a), the Cox Ingersoll Ross Model (CIR) (Cox et al., 1985a), and the Gaussian model G2++ (Brigo and Mercurio, 2006), are utilized in financial institutions, with the latter known for its two-factor. This chapter explores the calibration of financial models, with a particular focus on the G2++ model, using deep learning (DL) techniques. We present two distinct approaches applied to relevant datasets for calibration, aiming to enhance accuracy, efficiency, and user-friendliness compared to traditional methods. Our systematic approaches are model-agnostic and can be applied to any financial model with appropriate data. Calibration is a critical

aspect of model accuracy. In calibrating the one-factor Hull-White model, Gurrieri et al. (2009) considers selecting constant or time-dependent parameters for mean reversion and volatility, determining whether to calibrate locally or globally using specific products, and deciding whether to optimize mean reversion and volatility together or separately. This approach is commonly found in the literature (Gurrieri et al., 2009; Hull and White, 2001; Russo and Torri, 2019; Schlenkrich, 2012). In finance, DL techniques, particularly neural networks (NNs), have found widespread applications, including forecasting, pricing, and risk management. These techniques can replicate financial behaviors and instruments, optimize investment strategies, and improve assets pricing accuracy (Heaton et al., 2016; Bloch, 2019; Heaton et al., 2017; Buehler et al., 2019; Chen et al., 2020; Jang et al., 2021). Our approach introduces DL-based calibration techniques for financial models, particularly IR models. Unlike most recent DL-based calibration methods that focus on or need to transit through pricing functions (Pironneau, 2019; Büchel et al., 2022), our approach centers directly on parameter calibration. We offer direct and indirect calibration methods, demonstrating their effectiveness in improving accuracy and reducing calibration times compared to conventional approaches.

1.4.1 The G2++ model

The G2++ model, an extension of the Vasicek model, is a widely used interest rate model in finance. The G2++ model finds extensive applications in finance, particularly in pricing interest rate derivatives and managing interest rate risk. Its dual-factor structure enables it to capture complex term structure dynamics observed in real-world interest rate markets while the Gaussian distribution allows for a very appreciable tractability and flexibility. The model, features two stochastic factors, $x(t)$ and $y(t)$. These factors follow stochastic differential equations (SDEs) driven by correlated Brownian motions. Consider a time horizon $T > 0$ and a probability space $\left(\Omega, \mathcal{F}, (\mathcal{F})_{t\in[0,T]}, \mathbb{Q}\right)$ where we have denoted by $\mathcal{F}_t$ the sigma-field generated by the pair (x, y) up to time t and $\mathbb{Q}$ is the risk-neutral measure.. The model's equations are as follows:

$$\begin{aligned} r(t) &= \phi(t) + x(t) + y(t) \\ dx(t) &= -K_x x(t)dt + \sigma_x dW_t^x, \ x(0) = x_0 \\ dy(t) &= -K_y y(t)dt + \sigma_y dW_t^y, \ y(0) = y_0 \text{ with } K_x, K_y, \sigma_x, \sigma_y > 0 \text{ and } \rho \in [-1, 1]. \end{aligned} \tag{1.22}$$

Here, $(W_t^x, W_t^y)_{t\in[0,T]}$ is a two-dimensional $(\mathcal{F}_t)_{t\in[0,T]}$-Brownian motion with correlation ρ. Parameters K_x, K_y, σ_x, σ_y, and ρ need calibration. The function $\phi(t)$ is a deterministic function, well defined in $[0, T]$, that will allow the model to fit the initial term structure of forward rates. In particular, $\phi(0) = r_0$.

For the different deep calibration approaches that will be introduced in Chapter 4, we will require six quantities, Zero-Coupon (ZC) rates, forward (FWD) rates and each of their correlations and covariances. The price at time t of a Zero-Coupon bond maturing at T,

noted $P(t,T)$ is given by:

$$P(t,T) = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r(s)ds\right)\middle|\mathcal{F}_t\right]. \tag{1.23}$$

In the G2++ model, it is well known that this price can be expressed analytically:

$$P(t,T) = \frac{P^M(0,T)}{P^M(0,t)}\exp[\mathcal{A}(t,T)] \tag{1.24}$$

With $\mathcal{A}(t,T) = \frac{1}{2}[V(t,T) - V(0,T) + V(0,t)] - \frac{1-e^{-K_x(T-t)}}{K_x}x(t) - \frac{1-e^{-K_y(T-t)}}{K_y}y(t)$, $P^M(0,T)$, the initial ZC market curve price at maturity T and $V(\cdot,\cdot)$ is specified in Section 4.2 of Chapter 4. ZC rates, $Z(\cdot,\cdot)$ satisfy $Z(t,T) = -\frac{1}{(T-t)}\ln P(t,T)$, and for FWD rates, $f(\cdot,\cdot)$, we have the following relation: $f(t,T) = -\frac{\partial}{\partial T}lnP(t,T)$. Additionally, we give in Section 4.2.1 of Chapter 4 the expressions of the so-called instantaneous covariances and correlations of ZC and FWD rates (see also chapter 4.2 of Brigo and Mercurio (2006)) that will be used as inputs to our models.

1.4.2 Deep Calibration

As already stated, we outline two distinct approaches for calibrating the G2++ model. These approaches are systematic, they can be applied to any financial model, provided there exists an Observable Quantity Of Interest (OQOI) that can be observed in the market and derived analytically from the model under consideration. In our strategy for developing DC models, a key priority is the ease and availability of data. Our goal is to leverage easily accessible market data without requiring complex preprocessing. To train our DL models, we opt to use synthetic data generated through numerical simulations. This choice offers several benefits that will be discussed later on.

Indirect Deep Calibration: Covariances and correlations as information providers In our initial approach, we recognize that ZC and FWD rates are readily observable in the market, with analytical expressions available in most Interest Rate (IR) models. However, employing them directly as our OQOI in a simple Fully Connected Network (FCN) architecture could be ineffective due to dimension issues. Hence, we opt for an intermediary approach, termed the *Indirect Deep Calibration*, which captures sufficient compressed information from ZC and FWD curves to facilitate the calibration process. Promising candidates for the OQOI include correlation and covariance matrices of ZC and FWD rates variations.

For our dataset construction process, we undertake the following steps: first, we calibrate reference parameters for the G2++ model using mean squared error (MSE) minimization on real-world market data. Next, we extend the parameter ranges to establish acceptance intervals and randomly draw 10,000 parameters within these intervals. Subsequently, we

compute covariances and correlations for ZC and FWD rates. Following this, we choose the min-max scaling transformation for the dataset. Finally, the dataset is split, keeping 80% of the simulated dataset for training and 20% for testing the model.

In pursuit of simplicity, we opt for a straightforward FCN architecture. Our NN comprises five linear layers: an input layer with nf neurons representing the number of features, three hidden layers (with 1,000, 1,500 and 1,000 neurons respectively), and an output layer with np neurons denoting the number of parameters to calibrate ($np = 5$). Activation functions are applied to the first three layers using Rectified Linear Units (ReLU), while the output layer serves as a prediction layer devoid of activation. To combat overfitting, we incorporate a dropout mechanism with a probability of 0.25 between the last hidden layer and the prediction layer. Despite the capability to simulate a wide range of inputs, we deliberately limit our dataset to 10,000 entries. In contrast, prior studies utilize millions of swaptions trained over thousands of epochs, thus, requiring more time and computational resources to train their models.

Direct Deep Calibration: Direct use of zero-coupon rates We now consider the utilization of ZC rate curves as they can be directly observed from the market, eliminating the necessity for additional transformations and simplifying computational demands. The training process will require us to capture historical trends by simulating ZC curves across different dates and maturities, yielding a matrix of ZC rates. Given the potential dimension of these input (indeed, each matrix can have tens of columns corresponding to the different maturities used and potentially hundreds of rows, corresponding to all the observed dates), a Convolutional Neural Network (CNN) seems more suitable than a FCN for this calibration approach.

The process closely resembles that of indirect DC, with the computation of the OQOI changing. For this purpose, we will simulate expectations of ZC rates. From equation (1.24), the relation between the ZC rates and their prices and by setting $x_0 = y_0 = 0$, we derive an explicit formula for the expectation ZC rates:

$$\mathbb{E}^{\mathbb{Q}}\left[Z(t,T)\right] = -\frac{1}{T-t}\log\left[\frac{P^M(0,T)}{P^M(0,t)}\right] - \frac{1}{2(T-t)}\left[V(t,T) - V(0,T) + V(0,t)\right]. \quad (1.25)$$

Using the expectation of ZCs was not the only solution. Indeed, instead of taking the expectation, we could have used random simulated paths of the ZC rate using the distributions of $x(t)$ and $y(t)$. This different approach to the dataset construction in the direct DC approach is also tested in 4.5 of Chapter 4 on CIR intensity, yielding similar results to the ones obtained here with expectations.

For the $P^M(0,t)$ curve, we adopt the Euro curve as of 2020/11/04. Unlike indirect calibration using an FCN, employing a CNN alleviates concerns regarding the number of tenors. We select the following maturity sets: {1 day, 1 week, 1 month, 2 months, 3 months, 6 months, 9 months}; [1 year, 12 years] with a 1-year increment; and [15 years, 50 years] with a

5-year increment. Propagation involves computing ZC rates expectations at time steps $t_i = i\Delta,\ i \in [\![1, nb_{\text{steps}}]\!]$, with the step size $\Delta = 1$ week, and we perform 105 steps of propagation (approximately 2 years). Thus, our inputs will be in $\mathbb{R}^{nb_{\text{steps}} \times nb_{\text{tenors}}}$ with $nb_{\text{steps}} = 106$ and $nb_{\text{tenors}} = 28$.

The architecture integrates convolutional and linear layers in a relatively shallow design. Commencing with a convolutional layer of dimensions (C=1, H=106, nf=28) employing a filter size of (7×7) and a stride of 2 with padding, followed by a pooling layer with a stride of 2. Subsequently, two linear layers ensue, the first comprising 100 neurons and the second, serving as the prediction layer, featuring $np = 5$ neurons. Here also, we have 10 000 observations. 80% serves the training phase with the remaining used for the test phase.

1.4.3 Deep calibration results

Which approach and which OQOI for the Deep Calibration?

In terms of accuracy, both direct and indirect DC algorithms yield notably strong results. Table 1.2 below provides a summary of the mean squared errors obtained on the validation set for each parameter set. The indirect DC method yields similar results for both FWD and ZC rate curves due to their similarities. Notably, using covariances results in about half the error compared to using correlations, indicating that covariances offer more information for calibration. This is supported by numerical analysis of derivatives, where correlations exhibit vanishing derivatives. This observation suggests a phenomenon akin to the vanishing gradient problem, impacting the efficacy of backpropagation on correlations (CORs). In the unfeasible backpropagation theorem, Theorem 4.3.1, we present a condition when backpropagation becomes unfeasible, a novel finding in the field, to the best of our knowledge.

Method	OQOI	K_x	K_y	σ_x	σ_y	ρ
Indirect DC	COV - ZC	3.5×10^{-4}	5.6×10^{-4}	7.7×10^{-4}	8.3×10^{-4}	8.2×10^{-2}
	COV - FWD	3.5×10^{-4}	5.5×10^{-4}	7.6×10^{-4}	8.3×10^{-4}	8.2×10^{-2}
	COR - ZC	7.5×10^{-4}	1.1×10^{-3}	1.3×10^{-3}	1.4×10^{-3}	2.0×10^{-1}
	COR - FWD	7.4×10^{-4}	1.1×10^{-3}	1.3×10^{-2}	1.4×10^{-2}	2.4×10^{-2}
Direct DC	ZCs	4.0×10^{-4}	6.3×10^{-4}	9.8×10^{-4}	1.0×10^{-4}	1.52×10^{-1}

Table 1.2: Calibration errors on the test set

In the table above, COV-ZC, COV-FWD, COR-ZC, and COR-FWD designate, in that order, covariances of ZC rates, covariances of FWD rates, correlations of ZC rates, and correlations of FWD rates. This table leads to a preference for using direct DC due to its

simpler application, despite slightly larger errors compared to indirect DC. The crucial comparison lies between DC and traditional calibration methods.

Deep Calibration opposed to Classic Calibration
We also compare our direct DC approach to classic calibration (CC) methods. As a benchmark classic calibration method we take the minimization of the error between the historical and the theoretical covariances and correlations of ZC and FWD rates in a noisy environment. The global error on each parameter is computed using the normalized root mean squared error (NRMSE) on $N^{sets} = 50$ sets of out-of-sample covariances of ZC and FWD rates (and correlations for the CC). Hence, for each parameter $i \in [|1, np|]$, here $np = 5$, we compute $\text{NRMSE}^{\text{i}} = \frac{\text{RMSE}^{\text{i}}}{\max(\theta^i) - \min(\theta^i)}$, with $\text{RMSE}^{\text{i}} = \sqrt{\frac{1}{N_{sets}} \sum_{j=1}^{N_{sets}} (\theta_j^i - \hat{\theta}_j^i)^2}$, where for a given set of parameters j, $\{\theta_j^i\}_{i \in [|1,5|]} = \{K_{x_j}, K_{y_j}, \sigma_{x_j}, \sigma_{y_j}, \rho_j\}$.

We can see that the DC outperforms or is equivalent to the CC for each of the five parameters, despite the fact that the comparison parameters are outside the training sets.

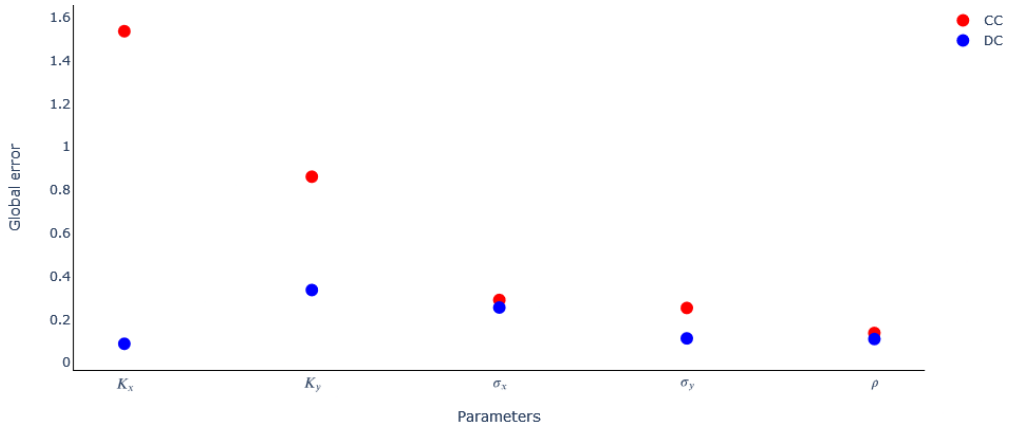

Figure 1.8: Global Error by parameter measured by NRMSEs for CC and DC

1.5 Chapter V: Towards explainable automated data quality enhancement without domain knowledge

Data is essential in modern business operations across various sectors, including commerce, entertainment, medicine, and the oil industry, driving efficiency and performance (Marr, 2016). The integration of AI, statistical methods, and probabilistic approaches facilitates the identification of complex patterns in large datasets, which would otherwise be difficult for humans to discern. However, the success of these methods heavily depends on the availability of high-quality data, necessitating extensive data preprocessing—a task that has historically consumed a significant portion of the analysis time, with estimates as high as 50% (Pyle, 1999). Despite advances in analytical techniques, the increasing complexity

and volume of data have exacerbated the challenges associated with data preparation (García et al., 2016). This phase, known as data preprocessing, is critical for ensuring data consistency and reliability before actionable insights can be extracteds. The growing rate of data generation across various industries calls for sophisticated analytical mechanisms, often requiring automation to address time and resource constraints (Han et al., 2022; Zaki and Meira, 2014). Key to this process is the identification and rectification of erroneous data values, which are central to data cleansing efforts (Han et al., 2012). Our framework, applicable to datasets containing numerical or textual data, identifies and corrects defects such as absence, redundancy, and inconsistency, leveraging a combination of AI-based techniques and statistical methods. For missing data, our approach favors classic imputation methods over AI-based solutions to preserve explainability. Similarly, for outlier detection and logic error correction, our framework strikes a balance between AI and statistical methods, focusing on transparency and interpretability. The literature review highlights a gap in existing data cleansing solutions that are both explainable and not dependent on domain knowledge. Our proposed framework aims to address this gap, especially explainability and interpretability—factors crucial for user confidence in the results. We demonstrate the effectiveness of this approach using a publicly available dataset, showcasing its ability to detect and correct errors without relying on domain-specific knowledge.

1.5.1 Key definitions

Valid data Data quality is often defined by the absence of defects, with high-quality data being free from such issues or minimally affected by them (Schelter et al., 2018; Corrales et al., 2018). In this chapter, we categorize defects into three main types: absence, redundancy, and inconsistency. Absence refers to missing data caused by errors during data collection, rendering the data invalid (Aydilek and Arslan, 2013). Redundancy involves duplicated data entries, while inconsistency covers discrepancies between a data item and other observations, including statistical outliers, typographical errors, and logical errors. Data is considered valid only when none of these defects are present, and the framework we propose is designed to detect and correct these issues while maintaining the explainability and interpretability of the methods used, without using domain-knowledge.

Explainability and interpretability The lack of explainability and interpretability, often resulting in a *black box* effect, is a significant concern limiting the broader adoption of ML or DL techniques (Escalante et al., 2018). Thus, ensuring these principles within our framework is a primary focus. In our context of statistical algorithms and ML, interpretability is commonly referred to as the ability of an algorithm to produce results understandable to humans (Doshi-Velez and Kim, 2018). Explainability, though more common, lacks a universally agreed-upon definition in algorithmic contexts. Some define it as making a process clear, while others (Keil, 2006; Ras et al., 2018) stress that all the very stages of the clarification process are what define explainability. In our framework, a result is explainable and interpretable if (i) its derivation is clear, (ii) its limits are understandable, and (iii) any identified defects are locatable within the dataset. Similarly, an algorithm

is explainable if its results meet these criteria. For brevity, we will refer to both concepts under the term explainability.

1.5.2 Data quality framework

In Chapter 5, we present the framework depicted in Figure 1.9. It is divided into two phases: the Pre-Quality Enhancement (PQE) phase and the Quality Enhancement (QE) phase. The PQE phase begins with identifying a primary key in the dataset (PQE1) and determining the columns requiring specific treatments (PQE2). These initial steps are crucial for automating the process without relying on domain knowledge. After this preparation, the QE phase focuses on enhancing data quality by addressing the three key areas identified earlier.

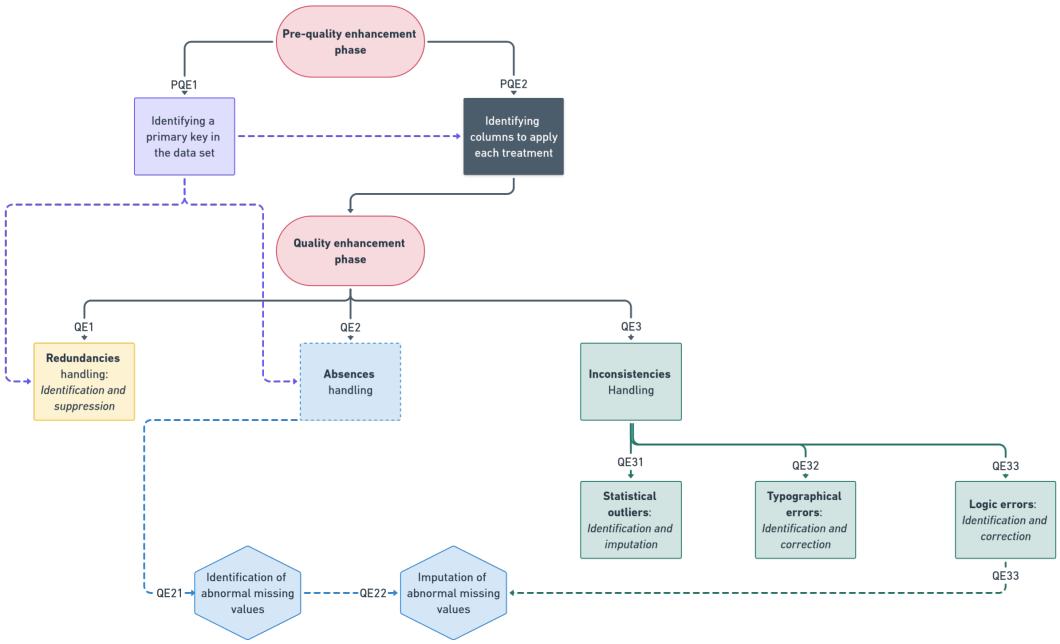

Figure 1.9: Data quality framework

1.5.3 Pre-quality Enhancement Phase

Identification of the Primary Key in the Dataset

The primary key in a dataset is a subset of attributes that uniquely identifies each record. To identify the primary key, we follow a two-step process involving pattern recognition and uniqueness analysis. Initially, we identify fields with less than 5% missing values and those containing terms like *ID*, *CODE*, or *KEY* in their names, as these are likely candidates for primary keys. If only one field remains after this pattern recognition, it is considered a potential primary key, subjected to further checks to ensure it has fewer than 5% duplicates and is not a numerical field. If this quick-win solution fails, we proceed

with a uniqueness analysis, examining combinations of candidate attributes to identify the primary key, ensuring minimal duplicates.

Mapping Processes to Specific Data Fields This phase assigns specific data quality processes to relevant fields without relying on domain knowledge. For redundancy detection, only primary key attributes are analyzed for duplicates. All fields are checked for missing values. Statistical outliers are considered only for numerical fields with over 50% data availability. Typographical errors are addressed for text fields, excluding those with less than half of the values available. Logic errors are examined in string fields with less than 75% missing values and at least five distinct observations. This ensures that the analysis focuses on fields where errors are likely and corrections will significantly enhance data quality.

1.5.4 Quality Enhancement Phase

Chapter 5 discusses in detail each step of the QE phase. Here, we briefly introduce our approach to address inconsistencies.

Statistical outliers
Various techniques have been devised to identify outliers, some relying on classical methods such as interquartile range (Corrales et al., 2018) and standard deviation-based approaches (Howell, 1998; Miller, 1991). However, these methods have limitations: they assume a normal distribution, are sensitive to outlier influence, and struggle with small sample sizes (Leys et al., 2013; Cousineau and Chartier, 2010). Despite these drawbacks, their simplicity and interpretability make them popular, especially with large datasets that approximate Gaussian distributions. Machine learning (ML) offers an alternative, with algorithms like Isolation Forest (IF) effectively isolating anomalies (Escalante, 2005; Liu et al., 2008). However, IF loses explainability in high-dimensional data. Thus, our approach combines statistical methods with IF in one dimension to retain interpretability. We define $\varphi_{outlier}$ as a function determining outlier status based on the Z-score of observed data X, i-e, $Z = \frac{X-\mu_X}{\sigma_X}$. The algorithm involves calculating skewness and kurtosis to determine outlier treatment. If both moments are below predefined thresholds, outliers are identified using standard deviation rules; otherwise, IF is applied.

$$\begin{aligned} \varphi_{\text{outlier}} &: \mathbb{R} \to \{0,\ 1\} \\ Z &\mapsto \varphi_{\text{outlier}}(Z). \end{aligned}$$

$$\varphi_{\text{outlier}}(Z) = \begin{cases} f_1(Z,\ \beta_1,\ \beta_2) & \text{if } |\mu_3| < \alpha_s \text{ and } |\mu_4| < \alpha_k \\ f_2(Z,\ \beta_1,\ \beta_2,\ \gamma) & \text{otherwise,} \end{cases} \tag{1.26}$$

where:

$$f_1(Z,\ \beta_1,\ \beta_2) = \begin{cases} 1 & \text{if } Z \notin]-\beta_1,\ \beta_2[\\ 0 & \text{otherwise,} \end{cases} \tag{1.27}$$

$$f_2(Z,\ \beta_1,\ \beta_2,\ \gamma) = f_1(Z,\ \gamma\beta_1,\ \gamma\beta_2) \times IF(Z), \tag{1.28}$$

and

$$IF(Z) = \begin{cases} 1 & \text{if the Isolation Forest algorithm detects an outlier} \\ 0 & \text{otherwise.} \end{cases} \tag{1.29}$$

μ_3 and μ_4 respectively define the 3^{rd} and 4^{th} standardized moment, skewness and kurtosis. α_s and α_k are two strictly positive moment acceptance thresholds. We take 6 and 30 respectively. β_1, β_2 are two strictly positive acceptance thresholds for the standard deviation-based tolerance interval. We take 3 for both coefficients. Finally, γ is a parameter to widen the tolerance interval on the value of Z; we choose 2.

The algorithm operates on the premise that a Gaussian approximation is feasible if the data distribution's skewness and kurtosis are within reasonable bounds. In such cases, the standard deviation rule is directly applicable. Alternatively, if the distribution exhibits significant distortion, the rule is applied with a widened tolerance interval. In instances of reduced reliability due to data distortion, outliers are identified based on consensus between the standard deviation rule and another algorithm, the IF. Therefore, this algorithm will not identify an extreme value as an outlier if it has other extreme values in its neighborhood.

Typographical outliers

The rectification of TPOs stands as a significant challenge in data processing readiness. Various solutions, often leveraging ML and DL techniques, have been proposed to address them (Moubayed et al., 2020; Neysiani and Babamir, 2019; Huang et al., 2013a). We introduce an algorithm whose quality lies in its efficient combination of basic statistical analysis and unsupervised machine learning. We introduce the Damerau-Levenshtein distance (DLD) (Levenshtein, 1966; Damerau, 1964), which calculates the minimum number of operations needed to change one character into another. These operations include substitution, insertion, deletion, and transposition. The DLD formula for letters i and j of words A and B is given by the recurrent equation:

$$d_{a,B}(i,j) = \min \begin{cases} 0 & \text{if } i = j = 0 \\ d_{A,B}(i-1,j) + 1 & \text{if } i > 0 \\ d_{A,B}(i,j-1) + 1 & \text{if } j > 0 \\ d_{A,B}(i-1,j-1) + 1_{(A_i \neq B_j)} & \text{if } i,j > 0 \\ d_{A,B}(i-2,j-2) + 1 & \text{if } i,j > 1 \text{ and } A_i = B_{j-1} \text{ and } A_{i-1} = B_j. \end{cases} \tag{1.30}$$

Practically, we utilize a scoring method derived from the DLD, rather than the DLD itself, as it mitigates the influence of word length on DLD interpretation. We denote it Damerau-Levenshtein Score (DLS). The DLS between words A and B is defined as $DLS(A,B) =: \frac{DMAX_{A,\,B} - DLD(A,B)}{DMAX_{A,\,B}}$, with $DMAX_{A,\,B}$ denoting the maximum possible distance between two words of the lengths of A and B.

The ideal solution of calculating DLSs for all entries is impractical due to computational

costs. We implement an alternative, cost-efficient approach. The TPO detection algorithm proceeds as follows: first, the list of words requiring TPO detection is sorted alphabetically, and consecutive DLSs are computed. Then, using DLS jumps, the initial similarity groups are identified. Each group is then represented by its dominant element (the element with the most occurrences). Clustering is conducted after determining the optimal number of clusters using a gap statistic method. Next, the algorithm checks for the presence of potentially valid entries within a cluster, either by comparing against a dictionary or by considering the proportion of observers with the identified TPO, depending, again, on the number of entries in the cluster. Finally, clusters, if applicable, are represented by their dominant element, serving as the correction for all other entries within the cluster. The strength of such an algorithm lies in its systematic approach to identifying and correcting TPOs in datasets. By sorting words alphabetically and computing consecutive DLSs, the algorithm efficiently detects similarities between entries. Utilizing DLS jumps allows for the initial grouping of similar entries, which are then represented by dominant elements for further analysis. Another important result is that the detection and correction of TPOs occur simultaneously.

Logic errors

Identifying logical errors can be challenging even through manual observation. However, our proposed algorithm aims to simplify this process by utilizing data mining techniques to uncover relationships between variables. Similar methods have been employed by (A.M.Rajeswari et al., 2014; Karthikeyan and Vembandasamy, 2015a). Specifically, we employ a modified version of the Apriori algorithm (Agrawal et al., 1994) to identify frequent sets within the dataset. The adjustments are made to improve efficiency, such as setting constraints on the minimum support and maximum length of item sets. This modified approach helps us identify potentially erroneous associations between variables, which we then scrutinize further to determine their validity. The algorithm's workflow involves running Apriori with specified constraints, obtaining all rules, filtering out high-confidence rules, and identifying and correcting observations that violate these rules. Overall, this process ensures a systematic and efficient approach to detecting and correcting LOGs in datasets.

Chapter 2

Credit Spreads' Term Structure: Stochastic Modeling with CIR++ Intensity

Abstract

This chapter introduces a novel stochastic model for credit spreads. The stochastic approach leverages the diffusion of default intensities via a CIR++ model and is formulated within a risk-neutral probability space. Our research primarily addresses two significant gaps in the literature. The first gap is the lack of credit spread models founded on a stochastic basis that enables continuous modeling, as many existing models rely on factorial assumptions. The second gap is the limited availability of models that directly yield a term structure of credit spreads. An intermediate result of our model is the provision of a term structure for the prices of defaultable bonds. We present the model alongside an innovative, practical, and conservative calibration approach that minimizes the error between historical and theoretical volatilities of default intensities. We demonstrate the robustness of both the model and its calibration process by comparing its behavior to historical credit spread values. Our findings indicate that the model not only produces realistic credit spread term structure curves but also exhibits consistent diffusion over time. Additionally, the model accurately fits the initial term structure of implied survival probabilities and provides an analytical expression for the credit spread of any given maturity at any future time.

2.1 Introduction

Credit spreads are a key indicator in finance. They are a versatile tool that effectively serve different purposes. Their most common use is probably as a creditworthiness measure of counterparties, as high credit spreads attributed to an entity (corporates, sovereigns, ...) indicate that the market perceives a high credit risk, while low values suggest lower risks. Credit spreads are also required when pricing most market instruments. For instance, discount factors, $d(t,T)$ at time t and maturity T are function of credit spreads, $\mathrm{Sp}(t,T)$ and zero-coupon rates, $\mathrm{Z(t,T)}$, as in its simplest expression (no adjustment spread for instance), we can write $d(t,T) = e^{-(T-t)[\mathrm{Z(t,T)}+\mathrm{Sp(t,T)}]}$. This implies that computing net present values for most products (securities, bonds, interest rate swaps, ...) requires credit spreads. Therefore, when performing full-revaluation risk computations, practitioners require models that can accurately propagate credit spreads. Similarly, regulatory risk computations like Credit Valuation Adjustment (CVA) often require credit spreads (see, for example, (BCBS and BIS, 2011)). In a less common use, they are also known to be macroeconomic indicators that can inform the state of an economy (Akinci and Queralto, 2022; De Santis, 2016).

For these reasons, financial institutions need to have reliable credit spread models. The purpose of this document is to introduce a new stochastic model for credit spreads. The model diffuses default intensities through a CIR++ (shifted Cox-Ingersoll-Ross) intensity model and, through an expression that is obtained in section 2.3 computes the credit spread. Our model's interest is first in the fact that it is derived from a classic stochastic model and therefore retains all the desirable properties from stochastic differential equation (SDE) driven models like most interest rate models. Indeed, the model natively provides a term structure of credit spreads, and it fits at $t = 0$ the term structure of the market implied survival probability curve. Also, the credit spread formula in the model is a closed formula. As we will discuss in section 2.4, the model can be easily calibrated as, from the SDE describing the default intensity or the equation deriving the spreads expression, one can infer different types of equations describing implied volatilities, instruments prices and so on to calibrate to. It produces realistic spreads curves dynamics that are similar to market observations.

The analytical development we conducted to obtain an explicit expression of credit spreads also yielded another very interesting result: an analytical expression for the price of a defaultable bond. Understanding the pricing of defaultable bonds is crucial for both investors and financial institutions, as it directly impacts the assessment of credit risk and the management of investment portfolios. Defaultable bonds, unlike risk-free bonds, carry the possibility of issuer default, necessitating sophisticated pricing models to accurately reflect their risk and return characteristics. The practical implications of defaultable bond pricing are vast. Accurate pricing models enable market practitioners to price and manage bond portfolios effectively, mitigating potential losses due to credit events. Furthermore, they support the development of hedging strategies that protect against the financial impacts of defaults. Additionally, these models aid in stress-testing financial institutions'

capital positions under adverse economic conditions. For instance, a model that values defaultable bonds based on macroeconomic variables such as foreign exchange rates can reveal how an institution's capital might be affected during market stress (Lo and Hui, 2000).

Various sophisticated models have been developed to price defaultable bonds. Early work by (Duffie and Singleton, 1999) is part of the reduced-form models family, also known as intensity-based models, to which our work also belongs. These models treat default as a random process with a specified hazard rate or intensity. They do not explicitly model the firm's assets but instead focus on the likelihood of default over time. Intensity-based models are particularly useful for capturing the term structure of defaultable bonds. They are often contrasted with structural models, such as the Merton model, which are based on the firm's asset value and its volatility. In structural models, the default event is typically modeled as occurring when the firm's asset value falls below a certain threshold (default barrier). This type of model was used by (Chen et al., 2009) to price defaultable bonds. More recently, (Russo et al., 2020) found closed-form solutions for pricing bonds under a two-factor Gaussian model. Structural models provide a different perspective by focusing on the firm's financial health and its impact on default risk.

The interest of our work regarding the pricing of defaultable bonds is twofold. First, the approach we develop does not introduce any complexity in the practical use of the model; the expression of the bond price is analytical and the calibration procedure is also provided, as it is the same as the one that will be introduced for the credit spread model. Second, to our knowledge, this is the first analytical expression of the defaultable bond price under a CIR++ intensity framework.

The primary challenge practitioners encounter when attempting to employ a credit spread model is the absence of a widely used and popular model. Unlike for interest rate modeling where models such as the Hull-White model (Hull and White, 1993), its two-factor extension (Hull and White, 1994), the Cox-Ingersoll-Ross model (Cox et al., 1985b), or the historical Vasicek model (Vasicek, 1977b) along with a list of other models are known to be very popular and were the subject of extensive studies analyzing their behavior and their performance (see for instance, Chan et al. (1992)), there does not seem to be one or many commonly used models for credit spreads. What made these models popular was their ease of use and effectiveness; our aim is to achieve the same: simplicity and efficiency.

The literature is rich with work on how to understand or forecast credit spreads. Most of the models in the literature seem to be factorial, meaning that they aim at explaining the variations of credit spreads via other variables. Such models have many advantages: they are, at least in the modeling part, easier to manipulate and use because, in general, the models are linear, their calibration is, in general, automatic (from factorial regressions or autoregressive models), and the interpretation of the model is more convenient as we deal with real meaningful indicators. However, they have drawbacks that motivate, for us, the use of SDE driven model: in general they do not allow native term structures, they do not allow continuous-time dynamics like SDEs, conducting stress tests is less flexible as we have

with SDEs many native features (for instance, stress testing via model parameters, typically model volatility, introduction of jumps or RW stress). It is worth noting that the native term structure in SDE-driven models allows for the continuous monitoring of the indicator being considered, over different maturities, providing a more granular view. This feature is particularly useful when entities need to evaluate potential shocks or movements at various points on said indicator. Therefore, it is a common requirement in risk management as well as in pricing issues.

(Davies, 2008) for instance, conduct a study using 85 years of corporate bonds data to find determinants of credit spreads. The author uses a self-extracting threshold model with the inflation characterizing the threshold, and the factors used are the levels series for the spread, US treasury bill (T-bill), equity and industrial production variables. (Manzoni, 2002) also uses autoregressive models that test both ARCH and GARCH models. The author uses FTSE returns, the term spread, the US dollar mark to the sterling pound and previous credit spreads values to describe CS. In (Avramov et al., 2007), the authors explain 67% of the variation of credit spreads using two types of indicators, common factors, they are used by all entities and include equity market return, changes in 5Y government rates, and company-level factors that include stock momentum or change in equity volatility. An interesting feature of their process is that they include in the set of common variables a dummy variable that is 1 when the US Federal funds rate increases and 0 otherwise. They find that this variable is significant only for the highest-rated counterparties where expansionary Fed policies (i.e., decreases in the rate value) reduce credit spreads.

One reason for the popularity of factorial model is probably the so-called "credit spread puzzle", which states that structural approach to credit risk and corporate bond pricing (like Merton, (Merton, 1974)) underestimates the credit spread for investment-grade counterparties. In (Feldhütter and Schaefer, 2018), the author introduces a new calibration methodology to use the Black-Cox model leading to more precise estimates of investment grade default probabilities. One key feature of the authors work is that instead of relying solely on the historical default rate for a specific maturity and credit rating as a proxy for default probability at that exact maturity and rating, they opt to utilize a wide cross section of default rates at different maturities and ratings. The primary characteristic of their approach that enables them to consolidate default rate data across various credit ratings and maturity periods is the underlying assumption that companies, regardless of their credit ratings and the maturity of their bonds, will still adhere to a common default threshold or boundary. An emphasis will therefore be put on back-testing the model and its calibration.

Also, authors have also introduced tree-based models for credit spreads or credit spread-related derivatives pricing (see for example, (Schönbucher, 1999; Xu and Li, 2009; Giacometti and Teocchi, 2005)).

Like in this work, in (Madan and Unal, 2000), the authors also use default intensity to obtain credit spreads. However, the two approaches are quite different. We do not require in our model an explicit derivation of the default intensity as the model is based

on the assumption that it can be described by a CIR++ model (just as we assume that instantaneous rates can be described by G2++, Vasicek, ... to obtain zero-coupon term structures). In the authors' work, the default intensity is expressed as a function of interest rates and market value of the assets of the firm. This requires the practitioner to hold information that is neither easily available for all types of counterparties (it might be accessible for large companies and some sovereigns, but not for most sub-sovereigns/regional governments and local authorities, small corporates, ...) nor easily proxied. It also requires having a model for interest rates and the market value of companies' assets. To our knowledge, this is the first modeling of credit spreads using default intensities described by a CIR++ model.

In what follows, we will start by deriving in Section 2.2 the risky bond price. Then, in Section 2.3 we obtain the expression for the credit spread under the risk neutral measure $\mathbb{Q}$. Section 2.4 then focuses on the calibration of the model. We discuss three calibration approaches before exploring in depth the one more suited for risk management. Section 2.5 illustrates the propagation of credit spreads using the model. Finally in Section 2.6 we perform the back-testing of the credit spread model.

2.2 Defaultable Bond Pricing under the CIR++ Intensity

The aim of this work is to provide practitioners with a stochastic model able to reliably reproduce and propagate the term structure of credit spreads. The framework has the following two steps:

- Representing default intensities by a CIR++ model.
- Finding the equation that defines credit spreads in that context.

The research for an analytical expression of credit spreads has yielded an analytical expression of defaultable bonds' price.

Model setting

In everything that follows, let τ be the instant of default and $T > 0$ the time horizon. The risk neutral model is set in a filtered probability space $\left(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{Q}\right)$, $(\mathcal{F}_t)_{t\in[0,T]}$ is the natural filtration of the default-less market, $\mathbb{Q}$ defines the risk neutral probability space and $Q(\cdot)$ the probability measure. Let $r(t) = r_t$ be the instantaneous risk-free rate of the market and $\lambda(t) = \lambda_t$ the default intensity. The default intensity process $(\lambda_t)_{t\in[0,T]}$ and the interest rates process $(r_t)_{t\in[0,T]}$ are $\mathcal{F}_t$-measurable. They are also taken to be $\mathcal{F}_t$-conditionally independent (see hypotheses (H3) and (H4)). Let $\mathcal{G}_t$ be the continuously enlarged filtration that includes default information, $\mathcal{G}_t = \mathcal{F}_t \vee \sigma(\{\tau < s\},\ s \leq t)$. This setup is similar to others from the literature (see for example, Chiarella et al. (2011); Bielecki et al. (2011)).

Density Hypothesis and Immersion property

We assume that for any time $t \geq 0$, the density of τ conditionally to $\mathcal{F}_t$ exists and is defined at any time t by an $\mathcal{F}_t \otimes \mathcal{B}(\mathbb{R}^+)$- measurable function $(\omega, \tilde{t}) \longrightarrow \alpha_t(\tilde{t})$. Consequently, for f, a bounded Borel function:

$$\mathbb{E}\left[f(\tau) \mid \mathcal{F}_t\right] = \int_0^\infty f(u)\alpha_t(u)\mathrm{d}u, \quad \text{a.s. .}$$

This assumption is known as the *density hypothesis*, see for instance El Karoui et al. (2010). We consider the particular case in which the reference filtration, observed after the default time, does not provide any additional insight into the conditional distribution of the default. This translates into the following equality for the conditional density function:

$$\alpha_t(\tilde{t}) = \alpha_{\tilde{t}}(\tilde{t}), \quad \forall t \geq \tilde{t} \quad \text{a.s. ,} \tag{H1}$$

Furthermore, it is well known that under hypothesis (H1), the *immersion property*, also known as the *H-hypothesis* holds. That is, for any fixed t and any bounded $\mathcal{G}_t$-random variable Y_t,

$$\mathbb{E}\left[Y_t \mid \mathcal{F}_\infty\right] = \mathbb{E}\left[Y_t \mid \mathcal{F}_t\right] \quad \text{a.s. ,} \tag{H2}$$

See for instance equations (11) and (12) in El Karoui et al. (2010) where a similar setup to the one described in this paragraph is presented.

Independence hypotheses

We make the following hypotheses regarding the independence between interest rates and the moment of default τ:

$$\exp\left(-\int_t^T r_s ds\right) \text{ is } \mathcal{F}_t\text{-conditionally independent of } \mathbb{1}_{\{\tau > T\}}, \tag{H3}$$

$$\exp\left(-\int_t^T r_s ds\right) \text{ is } \mathcal{F}_t\text{-conditionally independent of } \alpha_T(u),\ u > t. \tag{H4}$$

Default Intensity

The default intensity, also called in the literature hazard rate, $\lambda(t)$ verifies:

$$\lambda(t)dt = Q\left(\tau \in [t, t+dt] \mid \tau > t, F_t\right). \tag{2.1}$$

Equation(2.1) means that $\lambda(t)dt$ is the probability of default in $[t,\ t+dt]$. The result is derived from intensity models or reduced-form models that are in finance models that do not try to explain reason behind defaults (unlike factorial models, for instance). In such models, we often assume that default events correspond to jumps of inhomogeneous Poisson process where the previous relation is verified. That is to say, $\tau = \inf\{t \geq 0, \Lambda(0,t) \geq \Theta\}$ where Θ is a unit exponentially distributed random variable. We also introduce the cumulative

hazard rate that is defined by:

$$\Lambda(0,t) = \int_0^t \lambda_u du. \tag{2.2}$$

CIR++ Model

The dynamic of the CIR++ model is given by:

$$\begin{cases} \lambda(t) = y(t) + \psi(t) \\ dy(t) = \kappa\left(\theta - y(t)\right) dt + \sigma\sqrt{y(t)} dW_t, \end{cases} \tag{2.3}$$

where $(W_t)_{t\in[0,T]}$ is $\mathbb{Q}$-Brownian motion, $\kappa, \theta, \sigma > 0$. The Feller condition, $2\kappa\theta \geq \sigma^2$ and $y(0) = y_0 > 0$ guarantees $y(t) = y_t > 0$ under $\mathbb{Q}$. $\psi(t)$ is the deterministic function that we will use to fit the initial market-implied[1] survival probabilities, it will be explicited later on. Using a CIR-type model to diffuse default intensity is quite convenient, as we can easily ensure positive values. It actually is a common practice (see for instance, (Brigo and Alfonsi, 2005) or more recently, (Mbaye and Vrins, 2018)). When using the CIR model to diffuse default intensity, it is called the CIR intensity model. Many important works have been done regarding the relations between the CIR++ intensity and the classic CIR++ model for interest rates. The main result we will use is the equivalence between the survival probability, $S(\cdot,\cdot)$ in an intensity model and the zero-coupon bond price $P(\cdot,\cdot)$ in short rate models. This is because on one hand, as we already know about interest rates models, $P(0,t) = \mathbb{E}^{\mathbb{Q}}\left(e^{-\int_0^t r(u)du}\right)$. On the other hand, it is quite straightforward to show that $S(0,t) = Q(\tau > t) = \mathbb{E}\left[e^{-\int_0^t \lambda(u)du}\right]$. This means that both quantities can be written as Laplace transform of CIR-type processes. More detailed explanations on the meaning of the equivalence and the way to prove it can be found in sections 21.1.1.1 and 22.7.2 of (Brigo and Mercurio, 2006). The price of the risk-free zero-coupon bond is:

$$P(t,T) = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\middle|\mathcal{F}_t\right], \tag{2.4}$$

and the survival probability is given by:

$$S(t,T) = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T \lambda_s ds\right)\middle|\mathcal{F}_t\right]. \tag{2.5}$$

Defaultable bond price

We recall that in our setup we consider, $(r_t)_{0\leq t\leq T}$ to be a $(\mathcal{F}_t)_{0\leq t\leq T}$-measurable process that is $\mathcal{F}_t$-conditionally independent of the default intensity, and therefore τ-defined process.

[1]We present two solutions to imply the survival probabilities, the more common solution using CDS prices (Section 2.4) and a solution specific to our model using credit spreads (Section 2.5)

Then, the fair-price of a risky-bond (defaultable bond) with recovery rate $\delta \in (0,1)$ is

$$H(t,T) = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\left(\delta \mathbb{1}_{\{\tau \leq T\}} + \mathbb{1}_{\{\tau > T\}}\right)\middle| \mathcal{G}_t\right].$$

It is important to condition with respect to $\mathcal{G}_t$, as the bond's issuer may default.

Theorem 2.2.1. *Under the previous notations, and under hypotheses (H1-4), the price of the defaultable bond price of maturity T at time t, $H(t,T)$ as a function of the risk-free zero-coupon bond price $P(t,T)$ is given by:*

$$\begin{cases} H(t,T) = P(t,T)\left[\delta + (1-\delta)\dfrac{S^m(0,T)}{S^m(0,t)}\dfrac{A(0,t)}{A(0,T)}\dfrac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}}A(t,T)e^{-B(t,T)[\lambda(t)-\psi(t)]}\right] \\ \psi(t) = \lambda^m(t) + D(t) - y_0 E(t), \end{cases} \tag{2.6}$$

where,

$$A(t,T) = \left(\frac{2he^{\frac{1}{2}(\kappa+h)(T-t)}}{2h + (\kappa+h)\left(e^{(T-t)h}-1\right)}\right)^{\frac{2\kappa\theta}{\sigma^2}}, \; B(t,T) = \frac{2\left(e^{(T-t)h}-1\right)}{2h + (\kappa+h)\left(e^{(T-t)h}-1\right)}$$
$$h = \sqrt{\kappa^2 + 2\sigma^2}$$
$$D(t) = \frac{d}{dt}ln(A(0,t)), \; E(t) = \frac{d}{dt}B(0,t),$$

$S^m(0,t)$ is the initial market's implied survival probability, for horizon t and $\lambda^m(t)$ denotes the initial market default intensity for the horizon t.

Note that both $\lambda^m(t)$ and $S^m(0,t)$ are usually inferred from CDS prices. In the paragraph *Data Set* of Subsection 2.4 and in the paragraph *Fitting the initial market term structure* of Subsection 2.5, we provide more details on how they are obtained.

Proof. At first, we write

$$H(t,T) = \delta\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\mathbb{1}_{\{\tau\leq T\}}\middle|\mathcal{G}_t\right] + \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\mathbb{1}_{\{\tau>T\}}\middle|\mathcal{G}_t\right] =: H_1 + H_2. \tag{2.7}$$

For H_1, we have

$$\begin{aligned} H_1 &= \delta\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\middle|\mathcal{G}_t\right] - \delta\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\left(\mathbb{1}_{\{\tau>T\}}\right)\middle|\mathcal{G}_t\right] \\ &= \delta\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\middle|\mathcal{G}_t\right] - \delta H_2. \end{aligned} \tag{2.8}$$

Using Theorem 3.1 of El Karoui et al. (2010), we have:

$$\begin{aligned} H_{11} &= \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\Big|\mathcal{G}_t\right] \\ &= H_{11_t}^{bd}\mathbb{1}_{\{t<\tau\}} + H_{11_t}^{ad}\mathbb{1}_{\{\tau\leq t\}}. \end{aligned}$$

$H_{11_t}^{bd}$ and $H_{11_t}^{ad}$ respectively define the before default and after default components. Under hypothesis (H2), the immersion property, we have (see Subsection 3.1 of El Karoui et al. (2010)),

$$H_{11_t}^{ad} = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\Big|\mathcal{F}_t\right] = P(t,T). \tag{2.9}$$

Then for the component before default we have:

$$H_{11_t}^{bd} = \frac{\mathbb{E}\left[\int_t^\infty \exp\left(-\int_t^T r_s ds\right)\alpha_T(u)du\Big|\mathcal{F}_t\right]}{Q\left(\tau > t \mid F_t\right)}. \tag{2.10}$$

From hypothesis (H4) we write:

$$\begin{aligned} H_{11_t}^{bd} &= \frac{\mathbb{E}\left[\int_t^\infty \exp\left(-\int_t^T r_s ds\right)\alpha_T(u)du\Big|\mathcal{F}_t\right]}{Q\left(\tau > t \mid F_t\right)} \\ &= \frac{\mathbb{E}\left[\exp\left(-\int_t^T r_s ds\right)\Big|\mathcal{F}_t\right] \times \mathbb{E}\left[\int_t^\infty \alpha_T(u)du\Big|\mathcal{F}_t\right]}{Q\left(\tau > t \mid \mathcal{F}_t\right)} \\ &= \frac{P(t,T) \times \mathbb{E}\left[Q\left(\tau > t \mid \mathcal{F}_T\right) \mid \mathcal{F}_t\right]}{Q\left(\tau > t \mid \mathcal{F}_t\right)} \end{aligned}$$

$$H_{11_t}^{bd} = P(t,T). \tag{2.11}$$

Therefore from Equations (2.9) and (2.11), it follows that $H_{11} = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\Big|\mathcal{G}_t\right] = P(t,T)\mathbb{1}_{\{t<\tau\}} + P(t,T)\mathbb{1}_{\{\tau\leq t\}} = P(t,T)$. We then have $H(t,T) = \delta P(t,T) + (1-\delta)H_2$. For H_2 we recall that from the switching filtration theorem (see e.g. (Brigo and Mercurio, 2006, Section 22.5)), for any $\mathcal{G}_T$-measurable Z we have

$$\mathbb{E}\left(\mathbb{1}_{\{\tau>T\}}Z \mid \mathcal{G}_t\right) = \frac{\mathbb{1}_{\{\tau>t\}}}{Q\left(\tau > t \mid \mathcal{F}_t\right)}\mathbb{E}\left(\mathbb{1}_{\{\tau>T\}}Z \mid \mathcal{F}_t\right).$$

Using this result on H_2 leads to:

$$H_2 = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\mathbb{1}_{\{\tau>T\}}\Big|\mathcal{F}_t\right]\frac{\mathbb{1}_{\{\tau>T\}}}{Q\left(\tau > t \mid \mathcal{F}_t\right)}.$$

Since $\exp\left(-\int_t^T r_s ds\right)$ is $\mathcal{F}_t$-conditionally independent of $\mathbb{1}_{\{\tau>T\}}$ (hypothesis (H3)) and using

again $\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\Big|\mathcal{F}_t\right] = P(t,T)$, we write:

$$\begin{aligned} H_2 &= P(t,T)\mathbb{1}_{\{\tau>T\}}\frac{\mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{\{\tau>T\}} \mid \mathcal{F}_t\right]}{Q\left(\tau > t \mid \mathcal{F}_t\right)} \\ &= P(t,T)\mathbb{1}_{\{\tau>T\}}\frac{Q\left(\tau > T \mid \mathcal{F}_t\right)}{Q\left(\tau > t \mid \mathcal{F}_t\right)}. \end{aligned}$$

For the first time in our process, we now use the fact that we are in the context of intensity models $Q(\tau > t)$ is the survival probability and we know that $Q\left(\tau > T \mid \mathcal{F}_t\right) = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_0^T \lambda_s ds\right)|\mathcal{F}_t\right]$. It is well known that:

$$\frac{Q\left(\tau > T \mid \mathcal{F}_t\right)}{Q\left(\tau > t \mid \mathcal{F}_t\right)} = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T \lambda_s ds\right)\Big|\mathcal{F}_t\right],$$

which implies that

$$H_2 = P(t,T)\mathbb{1}_{\{\tau>T\}}\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T \lambda_s ds\right)\Big|\mathcal{F}_t\right]. \tag{2.12}$$

For instance, this is equivalent to result (3.3) of (Bielecki et al., 2011). Assuming that the default does not occur in within our observation window we take $\mathbb{1}_{\tau>T} = 1$, we find the expression of the risky bond price in a intensity modelisation context.We draw the reader's attention to the fact that, so far, we have not utilized the CIR++ model.

$$H(t,T) = P(t,T)\left\{\delta + (1-\delta)\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T \lambda_s ds\right)\Big|\mathcal{F}_t\right]\right\}. \tag{2.13}$$

The only component that remains to be determined is $\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T \lambda_s ds\right)\Big|\mathcal{F}_t\right]$. Its expression can be directly obtained through the identification we have already made between the CIR++ intensity model and the CIR++ model for short rates. Since the survival probability (described in our model by the expectation above) is equivalent to the zero-coupon bond price, they share the same expression. We recall that for interest rate modeling under the CIR++, we have the following dynamic for the short-term interest rates:

$$\begin{cases} r(t) = x(t) + \varphi(t) \\ dx(t) = \kappa\left(\theta - x_t\right)dt + \sigma\sqrt{x_t}dW_t. \end{cases}$$

The expression of the zero-coupon bond price in a CIR++ is very common in the literature (see for example Chapter 3.9 of (Brigo and Mercurio, 2006)) and is given by:

$$\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\Big|\mathcal{F}_t\right] = P(t,T) = \frac{P^m(0,T)}{P^m(0,t)}\frac{A(0,t)}{A(0,T)}\frac{e^{-B(0,t)x_0}}{e^{-B(0,T)x_0}}A(t,T)e^{-B(t,T)[r(t)-\varphi(t)]}.$$

In the above, $P^m(0,t)$ is the market quote of the initial zero-coupon bond price maturing at t and we have,

$$\begin{aligned} A(t,T) &= \left(\frac{2he^{\frac{1}{2}(\kappa+h)(T-t)}}{2h+(\kappa+h)\left(e^{(T-t)h}-1\right)} \right)^{\frac{2\kappa\theta}{\sigma^2}}; \\ B(t,T) &= \frac{2\left(e^{(T-t)h}-1\right)}{2h+(\kappa+h)\left(e^{(T-t)h}-1\right)}; \\ h &= \sqrt{\kappa^2+2\sigma^2}. \end{aligned} \tag{2.14}$$

Analogously, for our survival probability $S(t,T)$ we write :

$$\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T \lambda_s ds\right)\Big|\mathcal{F}_t\right] = S(t,T) = \frac{S^m(0,T)}{S^m(0,t)}\frac{A(0,t)}{A(0,T)}\frac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}}A(t,T)e^{-B(t,T)[\lambda(t)-\psi(t)]}. \tag{2.15}$$

where $S^m(0,\cdot)$ is the market's implied[2] survival probability at t_0.

The last step is to find the expression for $\psi(t)$. We have already stated that $\psi(t)$ should allow us to replicate the market's survival probabilities, $S^m(t)$. To achieve this, the model must satisfy $S^m(t) = S(0,t), \forall t \in [0,T]$. Here, we denote the market probability as $S^m(t)$, a uni-dimensional function, instead of $S^m(0,t)$ to emphasize that we are set at the initial time and S^m is therefore a market observation. We can then write:

$$\begin{aligned} S^m(t) = S(0,t) &= \mathbb{E}^{\mathbb{Q}}\left[e^{-\int_0^t \lambda_s ds}\right] \\ &= \mathbb{E}^{\mathbb{Q}}\left[e^{-\int_0^t y_s+\psi(s) ds}\right] \\ &= e^{-\int_0^t \psi(s) ds}\mathbb{E}^{\mathbb{Q}}\left[e^{-\int_0^t y_s ds}\right]. \end{aligned}$$

$\mathbb{E}^{\mathbb{Q}}\left[e^{-\int_0^t y_s ds}\right]$ is the price of a zero-coupon bond under a CIR model (or of the survival probability under a CIR intensity model). The literature (see the original CIR paper, (Cox et al., 1985b) for instance) gives $\mathbb{E}^{\mathbb{Q}}\left[e^{-\int_0^t y_s ds}\right] = A(0,t)e^{-B(0,t)y_0}$, so we can write:

$$S^m(t) = e^{-\int_0^t \psi(s) ds}A(0,t)e^{-B(0,t)y_0}.$$

From our definition of the default intensity, it is natural to write the survival probability as a function of the cumulative hazard rate $S^m(t) = e^{-\Lambda^m(t)}$, where analogously to equation (2.2), $\Lambda^m(t) = \int_0^t \lambda^m(u)du$, where $\lambda^m(t)$ denotes the initial market default intensity for the

[2]Unlike $P^m(0,t)$, survival probabilities are not usually directly quoted in the market. They must be inferred from relevant instruments such as CDS. This is discussed in more detail in the calibration section (2.4).

horizon t. Thus, we get:

$$\int_0^t \psi(s)ds = \Lambda^m(t) + \ln(A(0,t)) - B(0,t)y_0,$$

and consequently,

$$\psi(t) = \lambda^m(t) + D(t) - y_0 E(t)$$

where

$$D(t) := \frac{d}{dt} ln(A(0,t)) = \frac{2\kappa\theta}{\sigma^2}\left[\frac{1}{2}(\kappa+h) - \frac{h(\kappa+h)e^{th}}{2h+(\kappa+h)(e^{th}-1)}\right]$$
$$E(t) := \frac{d}{dt}B(0,t) = \frac{4h^2 e^{th}}{[2h+(\kappa+h)(e^{th}-1)]^2}.$$

Finally, combining these results with equations (2.13) and (2.15) we have our equation for the price of defaultable bonds:

$$\begin{cases} H(t,T) = P(t,T)\left[\delta + (1-\delta)\dfrac{S^m(0,T)}{S^m(0,t)}\dfrac{A(0,t)}{A(0,T)}\dfrac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}}A(t,T)e^{-B(t,T)[\lambda(t)-\psi(t)]}\right] \\ \psi(t) = \lambda^m(t) + D(t) - y_0 E(t). \end{cases}$$

□

Comment on the form of the defaultable bond price equation
From equation 2.13 in the proof of Theorem 2.2.1 we have that:

$$H(t,T) = P(t,T)\left[\delta + (1-\delta)S(t,T)\right].$$

This equation allows for an intuitive analysis of the expression. It guarantees that for any given recovery rate, the closer the survival probability is to 1, the closer the risky bond price $H(t,T)$ is to the risk-free bond price $P(t,T)$. In particular, if the survival probability is 1, meaning that the risky bond is virtually default-free, then the two bonds have the same price. Additionally, this equation implicitly highlights the risk premium that investors demand for holding a defaultable bond over a risk-free bond, as it ensures that $H(t,T)$ is always lower than $P(t,T)$. To exploit this equation to propagate the term structure of a defaultable bond price, one needs to combine equation (2.6) with a model for the pricing of the risk-free bond. Specifically, one needs to define the dynamics of the short-term interest rates. For instance, if one assumes the instantaneous rate to be modeled by a Hull-White model, then it is well known (Brigo and Mercurio, 2006) that we would have:

$$dr(t) = [\vartheta(t) - a^r r(t)]dt + \sigma^r dW^r(t),$$

where a^r and σ^r are positive constants and ϑ is chosen so as to exactly fit the term structure of interest rates being currently observed in the market and $W^r(t)$ is a $\mathbb{Q} - Brownian$

motion. The bond price $P(t,T)$ would be:

$$P(t,T) = A^{HW}(t,T)e^{-B^{HW}(t,T)r(t)},$$

where

$$B^{HW}(t,T) = \frac{1}{a}\left[1 - e^{-a(T-t)}\right],$$
$$A^{HW}(t,T) = \frac{P^m(0,T)}{P^m(0,t)} \exp\left[B^{HW}(t,T)f^m(0,t) - \frac{\sigma^2}{4a}\left(1 - e^{-2at}\right)B^{HW}(t,T)^2\right],$$

and $P^m(0,t)$ is the initial market bond price at time 0.

2.3 Modeling the Term Structure of Credit Spreads

Let us now focus on the main aim of this chapter: the term structure of credit spreads. We recall that our modeling framework represents default intensities using a CIR++ model and then finds the equation that defines credit spreads in that context. We maintain the same setting and definitions as previously established. As we have already stated, the price of the risk-free zero-coupon bond is:

$$P(t,T) = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\middle|\mathcal{F}_t\right].$$

We recall that in our setup we consider, $(r_t)_{0\leq t\leq T}$ to be a $(\mathcal{F}_t)_{0\leq t\leq T}$-measurable process, which is $\mathcal{F}_t$-conditionally independent of the default intensity and therefore any τ-defined process. The fair-price of a risky-bond with recovery rate $\delta \in (0,1)$ is defined by

$$H(t,T) = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s ds\right)\left(\delta \mathbb{1}_{\{\tau\leq T\}} + \mathbb{1}_{\{\tau>T\}}\right)\middle|\mathcal{G}_t\right],$$

and given explicitly by Theorem 2.2.1.

The credit spread represents the difference between the yield of a risky asset, denoted as $Z_r(t,T)$, and the yield of a risk-free asset, denoted as $Z(t,T)$. Typically, the risk-free asset can be government bonds, which are considered to have minimal default risk. In contrast, the risky asset could be a corporate bond or a bond issued by a financial institution such as a bank, which inherently carries a higher default risk. The relationship between the yields and the prices of these assets at time t for a bond maturing at time T is given by the following expressions:

$$P(t,T) = e^{-(T-t)Z(t,T)};\ H(t,T) = e^{-(T-t)Z_r(t,T)}.$$

The exponential function in these classical formulas reflects the continuous compounding of

interest rates. The credit spread, $Z_r(t,T) - Z(t,T)$, thus quantifies the additional yield that investors demand for taking on the additional risk associated with the corporate or bank-issued bond compared to a government bond. This spread is a crucial metric in credit risk analysis, reflecting the market's perception of the default risk and the overall creditworthiness of the issuer. Therefore, the credit spread Sp(t, T) of maturity T and evaluated at time t is given by

$$Sp(t,T) := Z_r(t,T) - Z(t,T) = -\frac{1}{T-t}\log\left[\frac{H(t,T)}{P(t,T)}\right]. \tag{2.16}$$

Note that this expression for the credit spread, which measures the compensation received for bearing additional risk, is similar to equation (60) in (Madan and Unal, 2000) and is consistent with previous work in the literature (see for instance, Davies (2008); Manzoni (2002); Giacometti and Teocchi (2005)).

Theorem 2.3.1. *Under the previous notations, the value of the credit spread of maturity T at time t, $Sp(t,T)$ is given by:*

$$\begin{cases} \mathrm{Sp}(t,T) = -\dfrac{1}{T-t}\ln\left[\delta + (1-\delta)\dfrac{S^m(0,T)}{S^m(0,t)}\dfrac{A(0,t)}{A(0,T)}\dfrac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}}A(t,T)e^{-B(t,T)[\lambda(t)-\psi(t)]}\right] \\ \psi(t) = \lambda^m(t) + D(t) - y_0E(t), \end{cases} \tag{2.17}$$

where, as in Theorem 2.2.1,

$$\begin{aligned} A(t,T) &= \left(\frac{2he^{\frac{1}{2}(\kappa+h)(T-t)}}{2h+(\kappa+h)\left(e^{(T-t)h}-1\right)}\right)^{\frac{2\kappa\theta}{\sigma^2}},\ B(t,T) = \frac{2\left(e^{(T-t)h}-1\right)}{2h+(\kappa+h)\left(e^{(T-t)h}-1\right)}, \\ h &= \sqrt{\kappa^2+2\sigma^2}, \\ D(t) &= \frac{d}{dt}ln(A(0,t)),\ E(t) = \frac{d}{dt}B(0,t), \end{aligned}$$

and $\lambda^m(t,T)$ and $S^m(0,t)$ are respectively the initial market's implied default intensity and survival probability for the horizon t.

Proof. Once the proof of Theorem 2.2.1 is given, the proof of Theorem 2.3.1 is quite straightforward. Indeed, on one hand, since

$$H(t,T) = P(t,T)\left\{\delta + (1-\delta)\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T \lambda_s ds\right)\middle|\mathcal{F}_t\right]\right\},$$

we naturally have

$$\mathrm{Sp}(t,T) = -\frac{1}{T-t}\ln\left[\delta + (1-\delta)\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T \lambda_s ds\right)\middle|\mathcal{F}_t\right]\right].$$

On the other hand, we had already justified for Theorem 2.2.1 that the survival probability analytical expression in our setup was:

$$\mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T \lambda_s ds\right)\Big|\mathcal{F}_t\right] = S(t,T) = \frac{S^m(0,T)}{S^m(0,t)}\frac{A(0,t)}{A(0,T)}\frac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}}A(t,T)e^{-B(t,T)[\lambda(t)-\psi(t)]}.$$

The above leads to our equation for the diffusion of the term structure of credit spreads:

$$\begin{cases} \mathrm{Sp}(t,T) = -\dfrac{1}{T-t}\ln\left[\delta + (1-\delta)\dfrac{S^m(0,T)}{S^m(0,t)}\dfrac{A(0,t)}{A(0,T)}\dfrac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}}A(t,T)e^{-B(t,T)[\lambda(t)-\psi(t)]}\right] \\ \psi(t) = \lambda^m(t) + D(t) - y_0 E(t). \end{cases}$$

□

2.4 Model Calibration

Once our credit spread model is well established, it seems natural to develop the associated calibration approach. A model can only be as good as its parameters are well chosen to be representative of a specific context. Many calibration approaches exist in the literature. An efficient approach that has already been tested on CIR intensity model can be the deep calibration developed in Chapter 4. We will introduce another calibration approach as suitable as the deep calibration but with a focus on a risk management perspective.

Calibration Process

A natural use of this model will be asset fair-value calculations, either for pricing issues or for market risk management. In the first case, the intuitive calibration might be to fit the last few credit spreads values or to fit market prices of some assets. This idea would be consistent with most calibration processes for interest rates models, where an indeed common practice is to fit the prices of some interest rate derivatives (see for example (Hull and White, 2001) or more recently (Russo and Torri, 2019)). However for a more risk assessment oriented calibration, finding the best parameters to only fit a set of given values may not guarantee the ability to cover all the historical variations of a credit spreads. Indeed, in this latter case, a conservative practice is to make sure the model is representative of the widest variations of credit spreads observed in the market. These variations are related to the historical volatility of credit spreads. We therefore choose a tractable approach that consists in calibrating on the historical volatility of the market default intensity curve. This choice is all the more relevant as the literature has already established the fact that in a CIR++ intensity configuration, the default intensity can be seen as an instantaneous credit spread (see for example Chapter 21.1 of Brigo and Mercurio (2006)). The model squared volatility (which is the variance) of the default intensity is given by:

$$\mathrm{Var}_\lambda(T;\kappa,\theta,\sigma,y_0) = y_0\frac{\sigma^2}{\kappa}(e^{-\kappa T} - e^{-2\kappa T}) + \frac{\theta\sigma^2}{2\kappa}(1-e^{-\kappa T})^2. \tag{2.18}$$

We denote by $\text{Vol}_\lambda^m(T)$ the market volatility of the default intensity. We will try to minimize the error between the theoretical and historical volatilities, and we set $\text{Vol}_\lambda(T;\dots) = \sqrt{\text{Var}_\lambda(T;\dots)}$. Regarding the choice of the objective function, it turns out that choosing the Sum of Squared Relative Error (SSRE)[3] achieves the best results as it leads to good and consistent calibrations (see figures 2.3 and 2.4). Let $\Theta = \{\kappa, \theta, \sigma, y_0\}$ be the set of parameters to calibrate and $\Theta^* = \{\kappa^*, \theta^*, \sigma^*, y_0^*\}$ the set of calibrated parameters. Let M be the number of maturities used for the calibration. We calibrate on the volatilities for each of the M maturities $\{T_i\}_{i\in[\![1,M]\!]}$. Therefore, we solve:

$$\Theta^* = \underset{\Theta}{arg\,min} \sum_{i=1}^{M} \left[\frac{\text{Vol}_\lambda^m(T_i) - \text{Vol}_\lambda(T_i;\,\Theta)}{\text{Vol}_\lambda^m(T_i)} \right]^2. \tag{2.19}$$

We have included y_0 in the set of parameters to calibrate. This is not mandatory. The only relation we need to ensure is that whatever the choice of y_0, and the way it is obtained, the equality $\psi(0) = \lambda^m(0) - y_0$ must hold. Adding it to the parameters to calibrate gives us another degree of freedom to achieve the best calibration possible. From equation (2.18) we get the theoretical volatility. To obtain the historical volatilities of default intensity, we apply the procedure described in (Rosenblum et al., 1983) for interest rates to each default intensity horizon. In that article, the authors use weekly data. For each week, the volatility calculation is made using that week and the previous 12 or 51 weeks and the volatility is represented by the standard deviation over the period for each week. We chose 51 weeks as we found it to give more stable results. This provides us with a time series of volatilities of default intensity for each horizon. Finally we need to choose one value driven by this time series to be representative of the whole volatility. Many solutions seem reasonable, for instance taking the average value. In this work, we make the more conservative choice—again, to adopt a more risk-oriented perspective—by taking the highest values. That is to say that the maximal weekly volatility obtained via the process described above will represent the historical volatility over that period. This choice is very conservative but it is the one that guarantees that any economic stress encountered within the calibration process will be taken into account. We recommend practitioners who want to be less conservative to take the median or the average value of the window. Also, as we have already stated when this model is used for pricing issues, calibrating on the last few values of a given quantity (prices for instance) would be more suited.

Data set

For the sake of visualization, calibration, and diffusion, we focus on the credit spread of the French bank, Credit Agricole. We start by observing its risky bond yield and the risk-free one, which we take as the French government bond. Both data sets can be obtained via private market data providers (Bloomberg, Refinitiv Eikon, ...). The data is daily and covers the period from January 01$^{\text{st}}$, 2009, to January 01$^{\text{st}}$, 2024. The values are in BPs.

[3]This error measurement metric is not common, however it is the sum and root-less version of the root mean squared relative error found in (Despotovic et al., 2016).

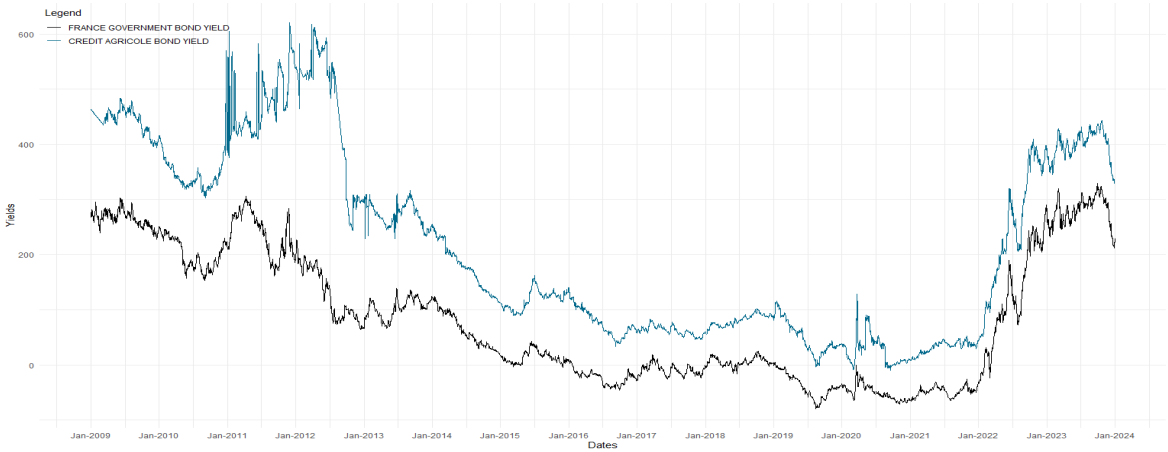

Figure 2.1: In BP, risk-free 5Y bond yield in blue (french government bond) and risky 5Y bond yield (Credit Agricole)

Then, we observe the 5Y credit spread. It is computed as the difference of yields between the French government bond and Credit Agricole's bond. Note that as of December 2023, S&P rates the French government as AA and Credit Agricole as AA-, as of September 2023.

Figure 2.2: In BP, Credit Agricole's 5Y credit spread

Since for our model we need both market implied default intensities, $\lambda^m(t)$ and market implied survival probabilities, $S^m(0,t)$ for the diffusion of credit spreads via equation (2.17) and for the calibration, we can use the CDS price curve of Credit Agricole to bootstrap said market implied default intensities and survival probabilities. The CDS curve has the same range as the yields and credit spreads (2009-01-01 to 2024-01-01) and is provided in Appendix 2.A. The bootstrapping procedure is not discussed in this work but is explained in Chapter 22.3 of (Brigo and Mercurio, 2006). Practically, for this work, we have used R's library *Credule*[4]. We also present in paragraph *Fitting the initial market term structure* of Subsection 2.5 a way to imply the survival probabilities from credit spreads. Figures 2.A.1, 2.A.2 and 2.A.3, provided in Appendix 2.A, show respectively, the CDS curve, the

[4] https://CRAN.R-project.org/package=credule

bootstrapped 5-year horizon default intensities and survival probabilities. Finally, we take the recovery rate to be $\delta = 40\%$.

Calibration Results

We use the data set described in the previous paragraphs to compute, for each maturity, the annual volatility of the credit spreads. We use the set of maturities {1Y, 3Y, 5Y, 7Y, 10Y}, which are reputed to be more reliable (regarding the availability of quotations and liquidity issues). As we have at our disposal a large data set, and since, as we have already stated, in this chapter—especially with the RW modeling—we are more risk management-oriented, we make two calibrations.

- *Global scenario*
  We aim for the model to be representative of the historical market situation. We take the last fifteen years as a reference to capture this historical behavior. Therefore, we calibrate using default intensities from 2009-01-01 to 2024-01-01. This data set ensures that we have sufficient data to represent the overall market behavior. This window includes stressed periods, such as the European/Greek Government Debt Crisis (see, for instance, (Baum et al., 2016)), as well as relatively calm periods.

- *Present scenario*
  This scenario is intended to represent the current behavior of credit spreads. We focus on the last two years, from 2022-01-01 to 2024-01-01. This calibration does not include any visually extreme variations in credit spreads, even though it captures part of the 2021-2022 post-pandemic inflation period (see, for example, Cline (2023)). As shown in Figure 2.1, the risk-free and risky yields seem to have reacted similarly to the crisis, resulting in relatively stable credit spreads, as depicted in Figure 2.2. This scenario is therefore expected to yield more favorable parameters.

The SSRE errors, as determined by equation (2.19), are 1.009×10^{-3} and 3.276×10^{-3}, respectively, for the *global scenario* and the *present scenario*. The parameters are given, respectively, by Tables 2.1 and 2.2. Figures 2.3 and 2.4 show the calibration's fittings.

κ	θ	σ	y_0
5.138×10^{-1}	1.497×10^{-2}	8.904×10^{-2}	4.348×10^{-2}

Table 2.1: Parameters calibrated for the *Global scenario*

κ	θ	σ	y_0
9.186×10^{-2}	5.519×10^{-4}	1.006×10^{-2}	3.074×10^{-2}

Table 2.2: Parameters calibrated for the *Present scenario*.

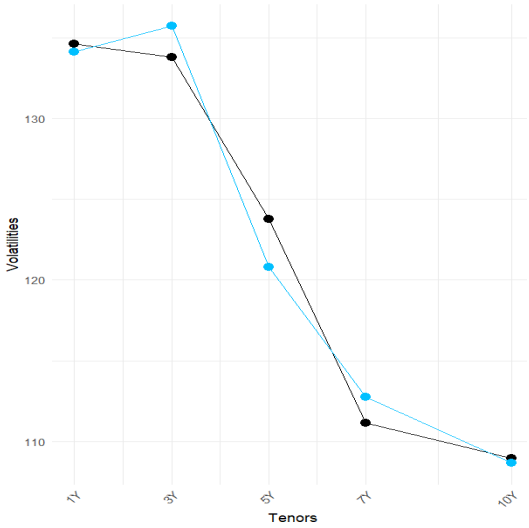

Figure 2.3: Calibration's fitting for the *global scenario*. Volatilities are expressed in BPs.

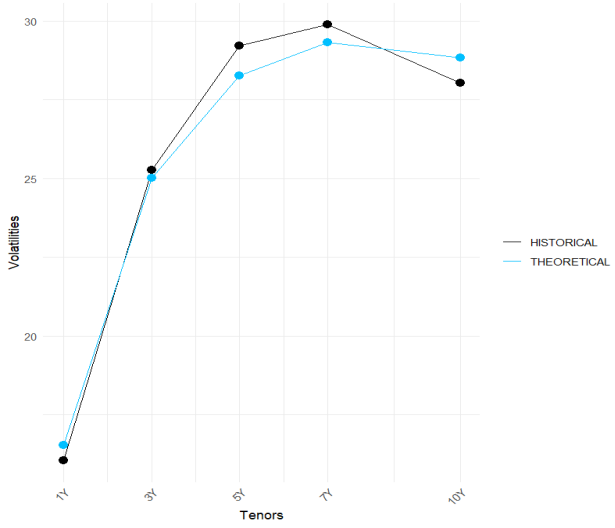

Figure 2.4: Calibration's fitting for the *present scenario*. Volatilities are expressed in BPs.

It is interesting to compare the parameters in the two scenarios. In fact, we observe a very appreciable behavior: both the volatility, σ, and the convergence parameter, θ, are considerably decreased for the present (stress-less) scenario. Practitioners can use any time range, expert-wisely chosen, to calibrate their parameters, whether it is for the base scenario, the present one, or even a stressed one. This is a considerable advantage of calibrating using quantities that are representative of the behavior over an entire period. Indeed, the significant increase in volatility for the global scenario, where volatilities range from 110 to 130 BPs as opposed to 20 to 30 BPs during the 2022-2024 period, is reflected in the calibration by the decrease of the parameters, especially σ. In the next sections, we will only use the global scenario parameters.

Comment on the shape of the volatilities

It is important to make a quick comment on the shapes of volatilities of default intensity. We observe a decreasing curve with respect to maturities for the global scenario and an increasing one for the present scenario. We have conducted many tests on different time frames, and the observation is always the same: if the time frame's main event is a stress period (meaning that the time frame is short and includes a minor stress or the time frame is large and includes a sufficiently large stress, like our global scenario), the volatility decreases with maturity; otherwise, it increases. As far as we know, this behavior is not studied in the literature, likely because it doesn't seem to have a significant impact. In the current work, we do not try to find the rationale behind this observed behavior. However, we are inclined to guess a link with the inversion of yield curves during stress periods (see, for instance, (Wright, 2006)), as we have also observed an inversion of the credit spread curve under stress periods.

2.5 Diffusion of credit spreads in the model

After having defined, then calibrated the model, we can now diffuse the credit spreads. Their diffusion is fairly straightforward using equation (2.17). The first step is to diffuse the default intensity $\lambda(t)$ and then apply equation (2.17). For the diffusion of $\lambda(t)$, we use the actual distribution of the solution of the CIR equation, $y(t)$, then we add the function $\psi(t)$. The probability density of $y(t)$ is given in (Cox et al., 2005) and is:

$$f(y(t), t\ ;\ y(s), s) = ce^{-w-v}\left(\frac{v}{w}\right)^{q/2} I_q\left(2(wv)^{1/2}\right),$$

where, $c := \frac{2\kappa}{\sigma^2\left(1-e^{-\kappa(t-s)}\right)}$, $w := cy(s)e^{-\kappa(t-s)}$, $v := cy(t)$, and $q := \frac{2\kappa\theta}{\sigma^2} - 1$ and $I_q(\cdot)$ is the modified Bessel function of the first kind of order q. The distribution function is the noncentral chi-square, $\chi^2[2cy(s); 2q+2, 2w]$, with $2q+2$ degrees of freedom and parameter of noncentrality $2w$ proportional to the current spot rate. In practice, in this Subsection for the simulations, we used R's library sde[5]. Note that this distribution is conditional on past values. Therefore, even when we are not using a diffusion scheme, we need to set a time step and keep track of at least one previous value. In all the simulations in this chapter, the time step is one week. We start from the configuration of the market at the simulation date, say January 1^{st}, 2024. Using the process described above, we simulate 20,000 paths of credit spreads for $T = 2(years)$ and with a time step of 1 week. Figure 2.5 shows the expectation (average) of credit spreads simulated at the origin (week 0) and at weeks 25, 50, 75, and 100. Figure 2.6 shows the 10% and 90% quantiles of the term structure of credit spreads simulated for weeks 25, 50, 75, and 100.

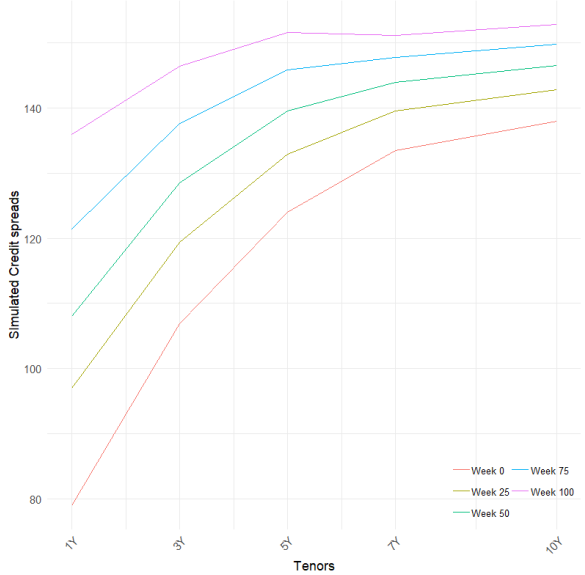

Figure 2.5: In BP, average term structure of simulated credit spreads

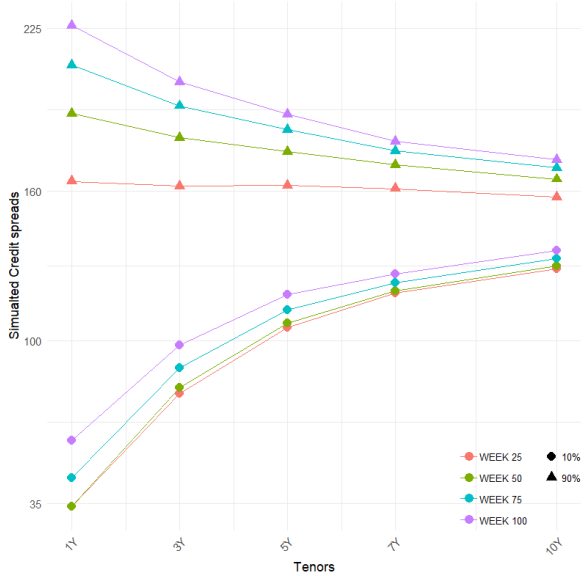

Figure 2.6: In BP, quantiles 10% and 90% of simulated term structure of credit spreads

[5] https://CRAN.R-project.org/package=sde

Figures 2.5 and 2.6 serve as the first back-tests of our model. They indeed show that our model produces realistic term structure of credit spreads over time and that the 10%-90% inter-quantile range seems coherent. Figure 2.7 shows the distribution of the 5Y credit spreads after about 1Y and 2Y (weeks 50 and 100). The figure also demonstrates a realistic shift in the credit spreads distribution over time. In subsection 2.6, we conduct a more robust back-test of our approach.

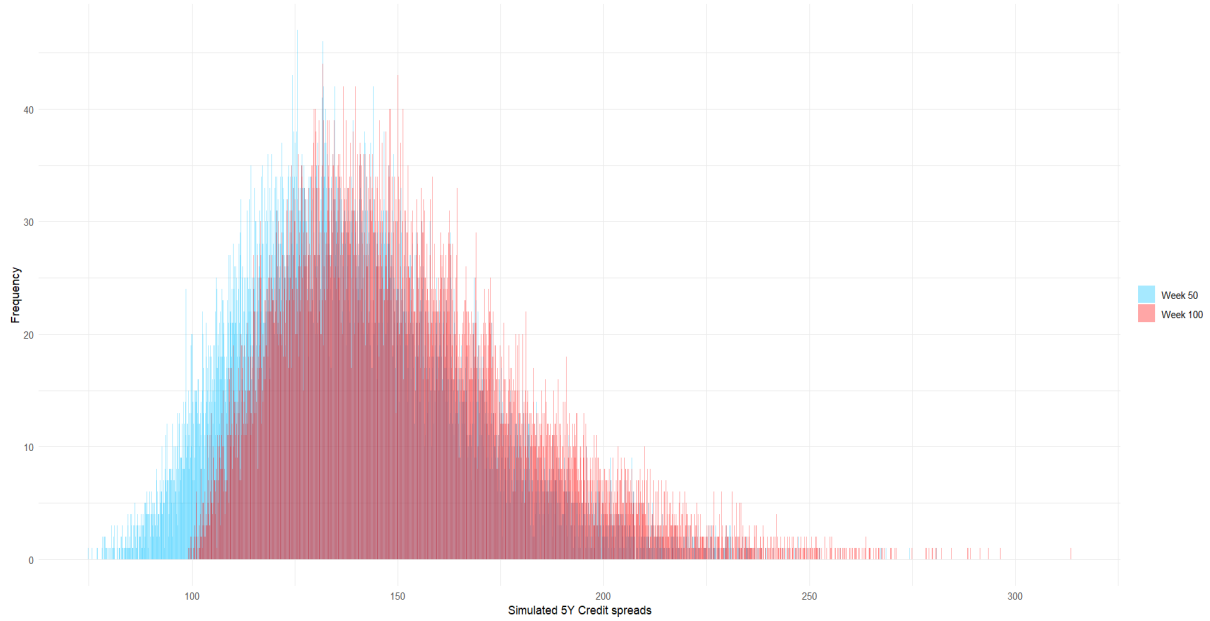

Figure 2.7: In BP, histogram of the simulated 5Y credit spreads after 50 and 100 weeks.

Fitting the initial market term structure

As previously discussed, the model fits the term structure of the cumulative hazard rate, which inherently corresponds to fitting the structure of survival probabilities. An advantageous consequence of this is the model's capability to fit the prices of Credit Default Swaps (CDS) for the risky asset, as survival probabilities are often implied from CDS values. Alternatively, in this model, one may opt to fit the initial credit spread values instead of CDS. In this scenario, the model permits the implied derivation of initial survival probabilities from the initial credit spreads. Equation (2.17) allows us to express the initial market survival probabilities, denoted as $S^m(0,T)$, as a function of the initial market credit spreads, $\mathrm{Sp}^m(0,T)$ and independently of the parameters of the model:

$$S^m(0,T) = \frac{e^{-T\,\mathrm{Sp}^m(0,T)} - \delta}{1-\delta} \tag{2.20}$$

It is noteworthy that these $S^m(0,T)$ values are always below 1 for positive credit spreads. However, ensuring their positivity requires a non-restrictive condition, which stipulates $\mathrm{Sp}^m(0,T) < -\ln(\delta)/T$. With the δ value of 40% used in this chapter, this condition remains

non-restrictive, allowing for credit spreads up to 900 basis points, even for maturities as long as 10 years.

2.6 Back-testing the model

As we have already stated, the figures in the previous section show that the model indeed produces a realistic term structure of credit spreads over time. However, we want to introduce another measure of our model's consistency. Specifically, we aim to evaluate how our model's extreme values behave with respect to the actual values of credit spreads. This back-test will also serve as a validation of the calibration. The procedure is as follows:

1. We have selected a suitable time horizon to analyze the actual observed credit spreads alongside the extreme values generated by our model. To ensure that the initial conditions adequately inform subsequent periods, we have capped the duration at 200 weekly intervals, roughly equivalent to a four-year time frame. Therefore, we take the period spanning from January 1, 2020, to January 1, 2024.
2. We start from the market conditions at the beginning of the period (2020-01-01). Using the diffusion process presented in the previous section, we simulate 20,000 paths of credit spreads.
3. For a specific maturity term, such as the 5-year tenor, we examine how various quantiles encompass the actual values of credit spreads.

Figure 2.8 shows the 5Y credit spreads and the quantiles 1%, 10%, 20%, 30%, 70%, 80%, 90% and 99% of the simulated credit spreads. We can observe how at the 99% level, only one historical credit spread value is beyond our simulation. This result demonstrates how conservative our model is and how much the credit spread values it gives are realistic. However it is important to note that this back-test algorithm is very restrictive and highly demanding. Credit spreads time series might have jumps or drops and especially important change of slopes that one might not be able to completely reproduce over time. We achieve the good results observed in figure 2.8 in large part thanks to the conservative calibration we have made in section 2.4, when choosing the calibration period as well as when choosing the representative volatility value. In fact we can actually see that the quantile 99% being close to the peak between 2020 and 2021 implies that it is way above the values that come after. Also if the window analysis had included significant changes in slopes, this back-test could have missed them while the model would have remained sufficiently coherent in terms of shapes of credit spreads term structures or of range of values.

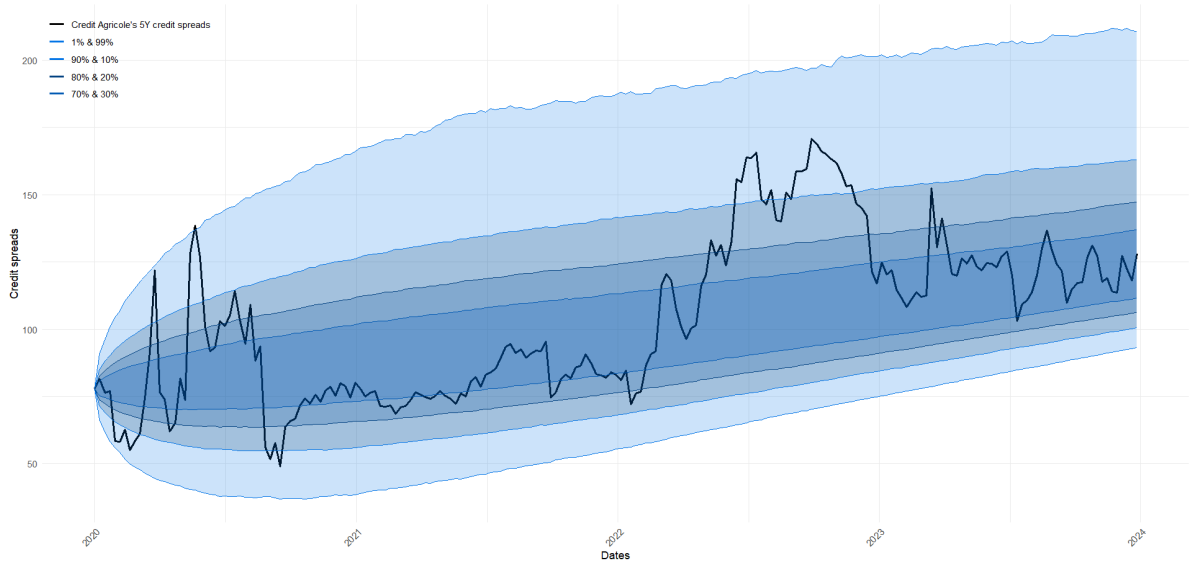

Figure 2.8: In BP, 5Y Credit Agricole's credit spreads observed values (black) and quantiles (shades of blue).

Note that for this back-test simulation, we have used equation (2.20) to fit the initial term structure of credit spreads. As we can see, the initial value of the simulated 5Y credit spread matches the initial market value.

2.7 Conclusion

In this chapter, we introduced a novel stochastic model for credit spreads using a CIR++ intensity model, addressing significant gaps in the current literature. Our model provides several key benefits that enhance its practical utility and theoretical robustness.

Firstly, the explicit expression of the credit spread under the CIR++ intensity model offers a powerful tool for practitioners. This explicit formulation simplifies the process of deriving credit spreads and allows for more straightforward applications in various financial computations and risk assessments. The ability to directly obtain the term structure of credit spreads is particularly valuable, as it enables a more accurate reflection of market conditions over different time horizons. This direct term structure derivation is crucial for pricing, risk management, and strategic financial planning, providing a comprehensive view of credit risk dynamics.

An important intermediary result of our research is the analytical expression for the price of a defaultable bond. This contribution not only enriches the understanding of defaultable bond pricing but also integrates seamlessly with our credit spread model. The simplicity and effectiveness of our calibration approach, which relies on the volatilities of default intensities, ensure that both models are robust and conservative. The calibration process was meticulously conducted by minimizing the Sum of Squared Relative Error

(SSRE) between historical volatilities of default intensities by horizon and the theoretical expression of the standard deviation of the default intensity. This approach ensured the accuracy and reliability of the model, as shown by the back-test. The latter revealed how the extreme quantiles of the simulated credit spreads encompassed the actual historic credit spreads, further validating the model's effectiveness.

In conclusion, the introduced stochastic model for credit spreads contributes to the literature by offering a robust, practical, and analytically sound framework. Future research could explore several exciting avenues to further enhance the model's applicability and depth. One potential direction is the application of the model to the pricing of options and the calculation of xVA (valuation adjustments), which are crucial for comprehensive risk management and derivative pricing. Additionally, incorporating more complexity into the model, such as introducing jumps or establishing correlations with other risk factors, could provide a more nuanced understanding of credit spreads and their interactions with broader market conditions.

Our work sets a solid foundation for more sophisticated credit spread modeling, promising significant advancements in financial risk management and economic forecasting. By continuing to refine and expand upon this model, researchers and practitioners alike can develop more effective strategies for navigating the complexities of credit risk.

Appendices

2.A Calibration Data

Figure 2.A.1 shows the historical values of Credit Agricole's CDS.

Figure 2.A.1: In BP, Credit Agricole's 5Y CDS

Figures 2.A.2 and 2.A.3 respectively show the bootsrapped default intensity and the survival probabilities.

Figure 2.A.2: In BP, Credit Agricole's Default Intensity for the 5Y horizon

Figure 2.A.3: In percents, Credit Agricole's Survival Probabilities for the 5Y horizon

Chapter 3

Financial Stochastic Models Diffusion: From Risk-Neutral to Real-World Measure

Abstract

This research presents a comprehensive framework for transitioning financial diffusion models from the risk-neutral (RN) measure to the real-world (RW) measure, leveraging results from probability theory, specifically Girsanov's theorem. The RN measure, fundamental in derivative pricing, is contrasted with the RW measure, which incorporates risk premiums and better reflects actual market behavior and investor preferences, making it crucial for risk management. We address the challenges of incorporating real-world dynamics into financial models, such as accounting for market premiums, producing realistic term structures of market indicators, and fitting any arbitrarily given market curve. Our framework is designed to be general, applicable to a variety of diffusion models, including those with non-additive noise such as the CIR++ model. Through case studies involving Goldman Sachs' 2024 global credit outlook forecasts and the European Banking Authority (EBA) 2023 stress tests, we validate the robustness, practical relevance and applicability of our methodology. This work contributes to the literature by providing a versatile tool for better risk measures and enhancing the realism of financial models under the RW measure. Our model's versatility extends to stress testing and scenario analysis, providing practitioners with a powerful tool to evaluate various what-if scenarios and make well-informed decisions, particularly in pricing and risk management strategies.

3.1 Introduction

Stochastic financial modeling has historically been done under the risk-neutral (RN) measure $\mathbb{Q}$, which is a fundamental concept in financial mathematics. This is mainly due to the fact that major stochastic models, especially those aiming at the diffusion of the instantaneous interest rate, are primarily used for market-related applications. Indeed, RN modeling is very useful when pricing instruments and when making classic risk computations (e.g., fair-value losses, CVA computation) as it ensures arbitrage-free modeling and that discounted prices of instruments (e.g., the discounted option price in the Black-Scholes framework, Black and Scholes (1973)) are martingales. Also, in general, RN modeling has many practical advantages. By assuming that all securities grow at the risk-free rate in the risk-neutral world, pricing derivatives becomes equivalent to finding the expected value of their discounted payoffs under this measure. Many financial models are analytically tractable under the risk-neutral measure, allowing for the derivation of analytical formulas. They are, in general, more straightforward to implement.

The Risk-Neutral (RN) probability measure is often contrasted with the Real-World measure (RW). The latter, often denoted as $\mathbb{P}$, tries to reflect the actual behavior of real financial markets. For instance, Berninger and Pfeiffer (2021) introduce the RW measure by including the market price of risks in the drift of the instantaneous interest rate described by a one-factor short rate RN-model, making the models more complex but also more realistic. Indeed, unlike the risk-neutral measure where the expected return is the risk-free rate, the real-world measure includes risk premiums. These are the additional returns that investors expect to compensate for the risk they undertake. This makes the real-world measure more reflective of actual investor behavior and market dynamics. In a more general definition, that goes beyond interest rate models, we can define the RW measure as the measure that takes into consideration the current observed behavior and/or the future expected behavior of market indicators (interest rates, risk indicators, investor preferences, etc.). It is becoming increasingly established that due to pricing needs, the focus has historically been on the $\mathbb{Q}$-measure, as we have already explained, it is often mandatory for pricing. However, for applications that require realistic scenarios, such as regulatory compliance, capital requirement calculations, long-term financial planning, risk management, macroeconomic forecasting, and performance scenario analyses, the $\mathbb{P}$-measure is crucial (e.g., (Berninger and Pfeiffer, 2021; Hull et al., 2014)). It helps in understanding how financial instruments and portfolios might perform under various real-world conditions. For instance, the authors in Hull et al. (2014) point out that the Basel II Framework[1] already admitted that when computing a counterparty's exposure for CVA (Credit Valuation Adjustment), the RW probability should be used instead of the risk-neutral one, but due to complexities arising from RW modeling, they tolerated using RN forecasting models. RW simulation should respect some properties to allow their optimal use. According to Norman (2009), in their work regarding RW modeling of the Brace Gatarek Musiela Model (BGM model, also known

[1]See footnote 240 on page 261 of International Convergence of Capital Measurement and Capital Standards

as the Libor Market Model, LMM, Brace et al. (1997)), RW models should be arbitrage-free, produce realistic dynamics, and be able to match a predetermined curve. These are all properties that we will ensure are verified in our different RW applications.

Our work aims to provide researchers and practitioners with a general approach that allows one to shift a model from RN to RW using change of measures results from probability theory. The generic framework we developed suits several diffusion models, including ones with non-additive noise like the CIR model Cox et al. (1985a). Moreover, our framework is not solely focused on the market price of risk and is designed to address a broad spectrum of market indicators, including interest rates and credit spreads. This work provides a tool to observe the changes in the entire term structure of a market indicator described by a diffusion model when values are anticipated for a certain maturity. These anticipations can come from regulatory authorities for region-wide stress tests or from internal forecasts or stress values. Numerical tests are performed, demonstrating the applications of our generic framework on regulatory stress scenarios and macroeconomic forecasting.

Despite significant advancements in incorporating RW measures in financial modeling to better reflect actual market dynamics and investor behavior, the literature is not as extensive as other topics in financial modeling. Most, if not all, of the work in the literature dealing with RW diffusion is focused on interest rates. To our knowledge, this work is the first to provide a generic framework intended to be applied to many risk indicators (like credit spreads) and to a wide range of models, including those with non-additive noise (like the CIR++ model).

Berninger and Pfeiffer (2021) conducted a study with many similarities to the work we will present in Section 3.2, but focused on zero-coupons and the G2++ model (also called the Gauss2++ or Gaussian two-factor model). The problem is therefore quite similar, but takes advantages of the regularity properties of Gaussian models, especially ones with additive noise. Their paper introduces a framework for calibrating the G2++ model under both RN and RW measures. They introduce a time-dependent market price of risk, whereas previous work considered a constant market price of risk (for instance, Dai and Singleton (2000)), which is known to be inefficient when forecasting interest rates (see Duffee (2002)). The authors also conduct a comparative analysis between step and linear functions to describe the market price of risk. These time-dependent market price of risk functions allow them to achieve more stable long-term interest rate forecasts.

Complementing this approach, (Bruti-Liberati et al., 2010) also study interest rate term structure models under the RW measure. In this context, they explore the integration of jump-diffusion processes, capturing both continuous and discrete market uncertainties. The particularity of their work is twofold. First, it is in their choice of the growth optimal portfolio as the numeraire for their pricing, and secondly, the fact that they take an interest in pricing under the RW measure. The authors aim to incorporate real-world trends into the pricing of interest rate derivatives.

From a numerical point of view, (Barker et al., 2016) tackle the simulation aspect of

RW measure estimation. They propose and compare two methodologies: a parametric approach using the Esscher transform and a minimum entropy, non-parametric approach for estimating the RW measure within Monte Carlo simulation frameworks. The authors also apply their work to interest rate models, specifically the 3-factor Hull-White model.

This analysis of the state of the art shows that most of the RW modeling has been performed to accommodate interest rate forecasting. This explains why an important part of RW modeling has, as a practical aim, the calibration of the market price of risk. In our attempt to address the limitations of traditional RN models and improve risk management and forecasting, the focus of our work goes beyond the calibration of the market price of risk. We are mainly interested in allowing RW distributions that can take our forecasts of the future into consideration by replicating any arbitrarily given curve for a specific maturity, and allowing one to observe how the whole term structure evolves as a consequence.

In what follows, we will start by deriving in Section 3.2 a generic framework to switch from RN to RW. Then we will show how the framework can be applied. In Section 3.3, we apply the model to the RW modeling of credit spreads via the CIR++ intensity model as developed in Chapter 2. Finally, Section 3.4 introduces two numerical results. We conduct two Monte Carlo simulations for credit spreads under the RW measure: the first application is for economic forecasting, and the second is for stress testing. In both examples, we demonstrate how the model can be used to exactly fit an arbitrarily given curve.

3.2 General framework and main results

3.2.1 Aim of this Section

Financial institutions as well as their regulatory authorities are continuously more demanding when it comes to risk assessment and assets and liabilities management. There is a real need of a better understanding of the future behavior of market risk indicators. The problem of modelling the Real-World (RW) evolution of the term structure of key risk indicators is an important one, yet it has not received much interest relatively to Risk-Neutral (RN) models. Also, most of the research conducted focuses on interest rates. Indeed, most term structure models have been developed in order to price interest rate derivatives, (swaptions, caplets, ...). In pricing problems, it is generally necessary to use risk-neutral probabilities, as they guarantee arbitrage free prices. However, Real-World probabilities are necessary to answer questions about the real-world distribution of market indicators in the future. Two main reasons can explain that:

- When forecasting, RN models do not take into account the fact that investors demand a risk premium for risky assets. This is well explained in (Lopes and Vázquez, 2018).
- RN model do not allow one to make *what-if* scenarios where we observe the deformation of the whole term structure of a risk indicators under a constraint on one or more maturities.

In this section, we introduce a generic framework, adapted to many diffusion models, to consistently switch from RW to RN. The features we want our RW model to exhibit are driven by the characteristics identified by Norman (2009), where the author states that a set of desirable properties for a RW model to verify is:

1. Being arbitrage-free;
2. Producing realistic term structures;
3. Being able to match any arbitrarily given curve.

The feature that will receive most of our attention in this Section and the following is the third, as it is the hardest to guarantee while being of high interest in risk management.

The task of expressing a chosen indicator under the RW measure can be very tedious, especially with non-additive noise. This is why, in the theoretical framework below, we start by eliminating non-additive noise through a Lamperti transformation. Our approach is based on expressing the transformed indicator under the RW measure as a function of (i) the expression of the transformed indicator under RN and (ii) a parametric function that needs to be calibrated. Once these relations are found, we can revert the transform to express the indicator under RW as a function of (i) the indicator under RN, (ii) a parametric function, and (iii) other terms that might arise from reverting the transformations. Once this is obtained, through numerical simulations involving the calibration of the parametric function and the simulations of the indicator under the RN measure, we can simulate the chosen indicator under RW, fitting any given value. Furthermore, as we will see in the applications, especially in Section 3.3, transitioning from a model defined in RN to a model in RW will be sufficient to justify the first property. The second property requires numerical simulations, which will be performed in Section 3.4. The aforementioned parametric function will be calibrated to allow the real-world diffusion of the indicator to fit any given values for the indicator.

The indicator we want to fit to given values is very rarely the *initial quantity* that is diffused by the model being considered. For instance, when working with interest rate models, the model's *initial quantity* is the instantaneous rate. However, the quantity that practitioners and researchers are mostly interested in will often be zero-coupon bond rates, zero-coupon bond prices, or swaption prices. In the application we will see in Section 3.3, the CIR++ intensity model describes the diffusion of the default intensity, but our indicators of interest will be: credit spreads cumulative hazard rates.

In the remainder of this section, we outline a general approach to establishing the relation between the *initial quantity* (the one diffused by the considered model) under RW as a function of the three aforementioned quantities (the indicator under RN, a parametric function, and other terms). Then, in the application in Section 3.3, we will consider a practical example using the CIR++ intensity model. We will move from the relations involving the *initial quantity* (default intensity) (see Theorem 3.2.1 and equation (3.20)) to the relations involving the real indicators of interest (credit spreads and cumulative hazard rates) (see Theorems 3.3.2 and 3.3.3).

3.2.2 Theoretical framework

The risk-neutral measure is a probability measure in which the discounted price process of a financial asset is a martingale. Formally, for an asset price S_t, a risk-free rate r, and a future time T, under the risk-neutral measure $\mathbb{Q}$, the following holds:

$$\mathbb{E}^{\mathbb{Q}}\left[e^{-r(T-t)}S_T \mid \mathcal{F}_t\right] = S_t,$$

where $\mathcal{F}_t$ represents the information available up to time t. This concept is fundamental in pricing theory, as it allows for the valuation of financial derivatives based on the assumption that there are no arbitrage opportunities. This definition aligns with previous work in the literature, as established early on by Harrison and Kreps (1979) or Geman et al. (1995). The risk-neutral measure we introduced contrasts with the real-world measure, denoted by $\mathbb{P}$, which represents the probability measure that aims to describe the actual observed probabilities of events in financial markets. Unlike the risk-neutral measure, which is used for pricing under no-arbitrage conditions, the real-world measure directly incorporates investors' risk preferences and market expectations. Under the real-world measure, the asset price process S_t generally follows a drift that includes the actual growth rate or expected return (see for instance, Berninger and Pfeiffer (2021)).

The risk-neutral model is set in a filtered probability space $\left(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{Q}\right)$. Here, $(\mathcal{F}_t)_{t\in[0,T]}$ is a given filtration, and $\mathbb{Q}$ defines the risk-neutral probability measure. Let $(W_t)_{t\in[0,T]}$ be a $\mathcal{F}_t$-Brownian motion under $\mathbb{Q}$. The probability space affiliated with the real-world measure will be denoted by $\mathbb{P}$. Now, let us consider in RN, the process $(Y_t)_{t\in[0,T]}$, which is the solution to

$$dY_t = b(Y_t)dt + \sigma(Y_t)dW_t, \quad Y_0 = y \in \mathbb{R}, \tag{3.1}$$

where the functions $b : \mathbb{R} \to \mathbb{R}$ and $\sigma : \mathbb{R} \to \mathbb{R}$ are locally Lipschitz continuous with $\frac{1}{\sigma(.)}$ being locally integrable. For $\phi(y) = \int_{y_0}^{y} \frac{1}{\sigma(x)}dx$, if $\sigma \in \mathcal{C}^1$, then by the Lamperti transform, $X_t = \phi(Y_t)$ satisfies the stochastic differential equation

$$dX_t = L(X_t)dt + dW_t, \quad X_0 = \phi(y),$$

$L(x) = \left(\frac{b}{\sigma} - \frac{\sigma'}{2}\right)(\phi^{-1}(x))$. In the sequel, while still keeping the relation $X_t = \phi(Y_t)$, we consider the same, more generic framework as in Alfonsi (2013); Derouich and Kebaier (2022), and consider $(X_t)_{t\geq 0}$ the solution to the SDE with additive noise defined on $I = (c, +\infty)$, $c \in [-\infty, +\infty[$:

$$dX_t = L(X_t)dt + \zeta dW_t, \quad t \geq 0,\ X_0 = x \in I, \zeta \in \mathbb{R}, \tag{3.2}$$

where the drift coefficient L is assumed to meet the following monotonicity condition:

$$L : I \longrightarrow \mathbb{R} \ \textit{is}\ C^2, \ \text{such that}\ \exists\, K > 0,\ \forall x, x' \in I, x \leq x', L(x') - L(x) \leq K(x' - x). \tag{3.3}$$

Additionally, for some $d \in I$, we assume that

$$v(x) = \int_d^x \int_d^y \exp\Big(-\frac{2}{\zeta^2} \int_z^y L(\xi) d\xi \Big) dz dy \ \text{ satisfies } \lim_{x \to c^+} v(x) = +\infty. \tag{H1}$$

According to Feller's test (see e.g., Karatzas and Shreve (1991)), the conditions (3.3) and (H1) guarantee that the SDE (3.2) has a unique strong solution $(Y_t)_{t \geq 0}$ on I, which never reaches the boundaries c and $+\infty$. Under this framework, we introduce the following theorem:

Theorem 3.2.1. *Let $(X_t^*)_{0 \leq t \leq T}$ be a process that under $(\Omega, \mathcal{F}, \mathbb{Q})$ is a solution to*

$$dX_t^* = [-\vartheta[(X_u^* - X_u) - \alpha_u] + L(X_u)] + \zeta dW_t. \tag{3.4}$$

Let $(\varphi_t)_{t \in [0,T]}$ be an adapted process defined by

$$\varphi_t := \frac{1}{\zeta} \left[-(L(X_t^*) - L(X_t)) - \vartheta(X_t^* - X_t - \alpha_t) \right], \tag{3.5}$$

with, $\vartheta \in \mathbb{R}$ and α_t is a deterministic function.

If

$$\mathbb{E}^{\mathbb{Q}} \left[\exp\left(\frac{1}{2} \int_0^T \varphi_u^2 du \right) \right] < \infty, \tag{H2}$$

then $\exists\, \mathbb{P}^{\varphi}$, a new probability measure with the following density

$$\frac{d\mathbb{P}^{\varphi}}{d\mathbb{Q}} |_{\mathcal{F}_T} = \exp\left(-\int_0^T \varphi_u dW_u - \frac{1}{2} \int_0^T \varphi_u^2 du \right),$$

such that

$$W_t^* = W_t + \int_0^t \varphi_u du, \tag{3.6}$$

is a Brownian motion under $\mathbb{P}^{\varphi}$, under this measure, the process $(X_t)^*$ is a solution of

$$dX_t^* = L(X_t^*) dt + \zeta dW_t^*, \quad t \geq 0,\ X_0^* = x \in I, \tag{3.7}$$

and verifies the following relation with (X_t)

$$X_t^* = X_t + \vartheta \int_s^t \alpha_u e^{-\vartheta(t-u)} du, \ \textit{for all } 0 \leq s < t \leq T. \tag{3.8}$$

Proof. Condition (H2) is Novikov's condition, which allows the application of Girsanov's theorem. This directly provides the first two results: the density of $\mathbb{P}^{\varphi}$ and the existence and expression of the $\mathbb{P}^{\varphi}$-Brownian motion $(W_t^*)_{t \in [0,T]}$.

Then we combine equations (3.5) and (3.4) to write the dynamic of $(X_t^*)_{t\in[0,T]}$ under $\mathbb{Q}$ as a function of φ_t, which is:

$$dX_t^* = [L(X_t^*) + \zeta\varphi_t]dt + \zeta dW_t.$$

Plugging (3.6) in the above equation easily gives the dynamic of $(X_t^*)_{t\in[0,T]}$ under $\mathbb{P}^\varphi$, $dX_t^* = L(X_t^*)dt + \zeta dW_t^*$.

We can then express $(X_t)_{t\in[0,T]}$ under $\mathbb{P}^\varphi$ which is $dX_t = [L(X_t) - \zeta\varphi_t]dt + \zeta dW_t^*$. We hence have:

$$\begin{aligned} dX_t^* &= L(X_t^*)dt + \zeta dW_t^* \\ dX_t &= [L(X_t) - \zeta\varphi_t]dt + \zeta dW_t^*, \end{aligned}$$

and we can write:

$$\begin{aligned} d(X_t^* - X_t) &= [L(X_t^*) - L(X_t) + \zeta_u]\, dt \\ \text{which implies } d(X_t^* - X_t) &= -\vartheta(X_t^* - X_t)dt + \vartheta\alpha_t dt \\ \text{which gives } e^{-\vartheta t}d[e^{\vartheta t}(X_t^* - X_t)] &= \vartheta\alpha_t dt, \end{aligned}$$

Hence the final result of the theorem:

$$X_t^* = X_t + \vartheta\int_s^t \alpha_u e^{-\vartheta(t-u)}du.$$

□

The probability measure $\mathbb{P}^\varphi$ is a real-world probability, and $(X_t^*)_{t\in[0,T]}$ is a RW process. In what follows, any process marked with an asterisk, $[.]^*$, denotes a RW process (that might be expressed under the risk-neutral or the real-world probability measure). The variable ϑ could be chosen to simplify the calculations as much as possible (see Section 3.3 for a practical illustration of the choice of ϑ). This leads to the following theoretical result.

Corollary 3.2.1.1. *Under the conditions of theorem 3.2.1, we have*

$$Y_t^* = \phi^{-1}\left(\phi(Y_t) + \vartheta\int_s^t \alpha_u e^{-\vartheta(t-u)}du\right). \tag{3.9}$$

From a practical point of view, we require Y_t^* in the following form:

$$Y_t^* = Y_t + R(\phi(Y_t), \vartheta, \alpha_t, t), \tag{3.10}$$

for some function $R(\dots)$ that can be as complex as ϕ requires it to be, see for example, section below.

At this stage, we have, as we intended, expressed the indicator under RW as a function of the indicator under RN and other terms. Depending on the type of function ϕ, this expression will be used to calibrate the parametric function α_t. We introduce in the next sections one example with a square root function arising in CIR-type models.

3.3 Application to CIR++ intensity model for credit spreads

3.3.1 Credit Spreads modeling under the Risk-Neutral measure

To apply the generic framework we introduced in Section 3.2 to modeling credit spreads, we use the model and the framework developed in Chapter 2. As stated in Section 3.1, to our knowledge, this is the first work that takes an interest in the RW diffusion of credit spreads.

The risk-neutral model is set in a filtered probability space $\left(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{Q}\right)$. Here, $(\mathcal{F}_t)_{t\in[0,T]}$ is the natural filtration of the default-less market, and $Q(.)$ is the probability function. Let $r(t) = r_t$ be the instantaneous risk-free rate of the market and $\lambda(t) = \lambda_t$ the default intensity. The default intensity process $(\lambda_t)_{t\in[0,T]}$ and the interest rate process $(r_t)_{t\in[0,T]}$ are $\mathcal{F}_t$-measurable. Also, let the default intensity (or hazard rate) $\lambda(t)$ satisfy:

$$\lambda(t)dt = Q\left(\tau \in [t, t+dt] \mid \tau > t, \mathcal{F}_t\right), \tag{3.11}$$

and be represented by the following dynamics:

$$\begin{cases} \lambda(t) = y(t) + \psi(t), \\ dy(t) = \kappa\left(\theta - y(t)\right)dt + \sigma\sqrt{y(t)}dW_t, \end{cases} \tag{3.12}$$

where $(W_t)_{t\in[0,T]}$ is a $\mathbb{Q}$-Brownian motion, and $\kappa, \theta, \sigma > 0$. The Feller condition, $2\kappa\theta \geq \sigma^2$, and $y(0) = y_0 > 0$ guarantee $y(t) = y_t > 0$ under $\mathbb{Q}$. $\psi(t)$ is a deterministic function. Under this framework, the credit spread of maturity T at time t is given by:

$$\begin{cases} \mathrm{Sp}(t,T) = -\dfrac{1}{T-t}\ln\left[\delta + (1-\delta)\dfrac{S^m(0,T)}{S^m(0,t)}\dfrac{A(0,t)}{A(0,T)}\dfrac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}}A(t,T)e^{-B(t,T)[\lambda(t)-\psi(t)]}\right] \\ \psi(t) = \lambda^m(t) + D(t) - y_0 E(t), \end{cases} \tag{3.13}$$

where,

$$\begin{aligned} A(t,T) &= \left(\frac{2he^{\frac{1}{2}(\kappa+h)(T-t)}}{2h + (\kappa+h)\left(e^{(T-t)h}-1\right)}\right)^{\frac{2\kappa\theta}{\sigma^2}}, \; B(t,T) = \frac{2\left(e^{(T-t)h}-1\right)}{2h + (\kappa+h)\left(e^{(T-t)h}-1\right)} \\ h &= \sqrt{\kappa^2 + 2\sigma^2}, \\ D(t) &= \frac{d}{dt}ln(A(0,t)), \; E(t) = \frac{d}{dt}B(0,t), \end{aligned} \tag{3.14}$$

and $\delta \in [0,1]$ is the recovery rate. An extensive proof and all the related elements can be found in Chapter 2.

Let us now derive our model's shift from RN to RW. The probability space affiliated

with the real-world will again be denoted for brevity by $\mathbb{P}$, instead of $\mathbb{P}^{\varphi}$. We introduce $(W_t^*)_{t\in[0,T]}$, a Brownian motion under the RW probability measure $\mathbb{P}$. We will apply the results from Theorem 3.2.1 and then justify how we obtain the three features mentioned earlier from our model.

$$\begin{cases} \lambda(t) = y(t) + \psi(t) \\ dy(t) = \kappa\left(\theta - y(t)\right)dt + \sigma\sqrt{y(t)}dW_t, \end{cases} \tag{3.15}$$

with $2\kappa\theta \geq \sigma^2$ and $y(0) = y_0 > 0$. $\psi(t)$ is the deterministic fitting function (for the survival probabilities).

Similarly in RW (real-world), under the measure $\mathbb{P}$, there exists a CIR++ diffusion written:

$$\begin{cases} \lambda^*(t) = y^*(t) + \psi(t) \\ dy^*(t) = \kappa\left(\theta - y^*(t)\right)dt + \sigma\sqrt{y^*(t)}dW_t^*, \end{cases} \tag{3.16}$$

with $y^*(0) = y_0^* > 0$.

Using a Lamperti transform, we re-write for $x_t = \Upsilon\sqrt{y_t}$, for $\Upsilon \in (0, 2^{\frac{1}{4}})$ and such that $\kappa\theta > \left[\frac{1-\Upsilon^2\frac{\sqrt{2}}{2}}{1-\Upsilon^2\sqrt{2}}\right]\frac{1}{2}\sigma^2$:

$$dx_t = \frac{1}{2}\left[\Upsilon^2\left(\kappa\theta - \frac{1}{4}\sigma^2\right)\frac{1}{x_t} - \kappa x_t\right]dt + \frac{1}{2}\sigma dW_t. \tag{3.17}$$

Note that, this new condition on $\kappa\theta$ slightly strengthens the previous Feller condition $2\kappa\theta \geq \sigma^2$ since we can choose Υ as small as possible.

Therefore regarding the notations in Theorem 3.2.1, we make the following identifications:

- $L(x) = \frac{1}{2}\left[\Upsilon^2\left(\kappa\theta - \frac{1}{4}\sigma^2\right)\frac{1}{x} - \kappa x\right]$;
- $\zeta = \frac{1}{2}\sigma$;
- $\phi(x) = \Upsilon\sqrt{x}$.

To apply Theorem 3.2.1, we need to very condition (H2).

3.3.2 Condition H2: Novikov criterion

Per condition (H2), we need to verify that our function φ_t respects Novikov's criterion. Equation (3.5) indicates that the function φ_t is given by:

$$\varphi_t = \frac{1}{\zeta}\left[-(L(x_t^*) - L(x_t)) - \vartheta(x_t^* - x_t - \alpha_t)\right],$$

which, for the identifications we made earlier corresponds to:

$$\varphi_t = \frac{2}{\sigma}\left[-\frac{\Upsilon^2}{2}\left(\kappa\theta - \frac{1}{4}\sigma^2\right)\left(\frac{1}{x_t^*} - \frac{1}{x_t}\right) + \frac{1}{2}\kappa(x_t^* - x_t) - \vartheta(x_t^* - x_t - \alpha_t)\right].$$

We stated earlier that the parameter $\vartheta \in \mathbb{R}$ should be chosen to simplify as much as possible the calculations. A good choice, seems to be $\vartheta = \frac{1}{2}\kappa$ as it gives:

$$\varphi_t = -\frac{\Upsilon^2}{\sigma}\left[\left(\kappa\theta - \frac{1}{4}\sigma^2\right)\left(\frac{1}{x_t^*} - \frac{1}{x_t}\right) - \kappa\alpha_t\right],$$

α_u is the deterministic function that will shift our RN diffusion drift to allow us to fit RW values. The function α_u could be for instance step-wise constant. We now need to show that our function φ_t respects Novikov's condition, that is $\mathbb{E}^{\mathbb{Q}}\left[e^{\frac{1}{2}\int_0^T \varphi_u^2 du}\right] < \infty$.

Let $\gamma = \frac{1}{\sigma}\left(\kappa\theta - \frac{1}{4}\sigma^2\right)$. We need to show that $\mathbb{E}^{\mathbb{Q}}\left[e^{\frac{1}{2}\int_0^T \left[\Upsilon^2\gamma\left(\frac{1}{x_u^*} - \frac{1}{x_u}\right)\right]^2 du}\right] < \infty$.

From the positivity of x_t and x_t^* we have

$$\mathbb{E}^{\mathbb{Q}}\left[\exp\left(\frac{1}{2}\int_0^T \left(\Upsilon^2\gamma\left(\frac{1}{x_u^*} - \frac{1}{x_u}\right)\right)^2 du\right)\right] \leq \mathbb{E}^{\mathbb{Q}}\left[\exp\left(\frac{\gamma^2\Upsilon^4}{2}\int_0^T \left(\frac{1}{(x_u^*)^2} + \frac{1}{(x_u)^2}\right) du\right)\right].$$

Remark that the dynamic of x_t under $\mathbb{P}$ is:

$$dx_t = \frac{1}{2}\left[\Upsilon^2\left(\kappa\theta - \frac{1}{4}\sigma^2\right)\frac{1}{x_t} - \kappa x_t - \sigma\varphi_t\right]dt + \frac{1}{2}\sigma dW_t^*.$$

Meanwhile, the dynamic of x_t^* under $\mathbb{P}$ is:

$$dx_t^* = \frac{1}{2}\left[\Upsilon^2\left(\kappa\theta - \frac{1}{4}\sigma^2\right)\frac{1}{x_t^*} - \kappa x_t^*\right]dt + \frac{1}{2}\sigma dW_t^*.$$

From these two expressions, the comparison theorem gives us that if $-\sigma\varphi_t < 0$, then $x_t^* > x_t$. The existence, expression, and sign of φ_t will be justified later on. Therefore, for a suitable (regarding Girsanov) function $\varphi_t > 0$, we can write:

$$\mathbb{E}^{\mathbb{Q}}\left[\exp\left(\frac{1}{2}\int_0^T \left(\Upsilon^2\gamma\left(\frac{1}{x_u} - \frac{1}{x_u^*}\right)\right)^2 du\right)\right] \leq \mathbb{E}^{\mathbb{Q}}\left[\exp\left(\Upsilon^4\gamma^2\int_0^T \frac{1}{(x_u)^2} du\right)\right].$$

Since $(x_u)^2 = y_u$, in practical terms, the problem consists of defining the domain of convergence $\mathcal{D}_t$ of the moment-generating function of the integral of a solution of (3.15).

Theorem 3.3.1. *Let $\hat{y}_t$ be the solution of the CIR equation defined in (3.15), the moment-generating function of $\int_0^T \frac{1}{\hat{y}_t} dt$ is well-defined on D_t defined by:*

$$D_t = \left\{ \mu \in \mathbb{R},\ \mathbb{E}\left[\exp\left(\mu \int_0^T \frac{1}{\hat{y}_u} du \right) \right] < \infty \right\} = \left\{ \mu, \mu < \frac{1}{2\sigma^2} \left(\kappa\theta - \frac{1}{2}\sigma^2 \right)^2 \right\}.$$

Proof. The expression of the Laplace transform is provided in Ben Alaya and Kebaier (2012a).

$$\begin{aligned} \mathbb{E}\left(e^{\mu \int_0^T \frac{du}{y_u}} \right) =& \frac{\Gamma\left(c + \frac{\nu}{2} + \frac{1}{2}\right)}{\Gamma(\nu+1)} \frac{1}{(y_0)^c \delta^c} \beta^{\frac{\nu}{2}+\frac{1}{2}} \\ & \times \exp\left(\frac{b}{2\sigma} \left[\kappa\theta T - \frac{2y_0}{e^{\kappa T} - 1} \right] \right) {}_1F_1\left(c + \frac{\nu}{2} + \frac{1}{2}, \nu + 1, \beta \right), \end{aligned} \tag{3.18}$$

where $\beta = \frac{2\kappa y_0}{\sigma^2(e^{\kappa T}-1)}$, $c = \frac{\kappa\theta}{\sigma^2}$, $\delta = \frac{2\kappa e^{\kappa T}}{\sigma^2(e^{\kappa T}-1)}$ and $\nu = \frac{2}{\sigma^2}\sqrt{-2\mu\sigma^2 + (\kappa\theta - \frac{1}{2}\sigma^2)^2}$. Following a similar approach to the proof of Proposition 1.2.4 in Alfonsi et al. (2015a), we give the set of convergence of D_t. $\left\{ \mu \in \mathbb{R}, \mathbb{E}\left[\exp\left(\mu \int_0^T \frac{1}{y_u} du \right) \right] < \infty \right\}$ obviously contains at least $\mathbb{R}_-$. Also, the characteristic function is analytic on the inside of its set of convergence (e.g. Widder (2015)). Then Γ is analytic on $\mathbb{R}\backslash\{-k, k \in \mathbb{N}^*\}$ (see for example Abramowitz et al. (1988)). From Abramowitz et al. (1988) we also get that $1/\Gamma$ is entire and therefore analytic in $\mathbb{R}\backslash\{-k, k \in \mathbb{N}^*\}$. We bring to the attention of the reader that even if $\kappa,\ \theta \in \mathbb{R}$, $c > 0$ since $2\kappa\theta > \sigma^2$. Then, we get from chapter 6 of Bateman (1953) that the ${}_1F_1$ function is analytic. Thus, the right hand side of (3.18) is analytic as soon as ν is well defined in $\mathbb{R}$, which requires $\mu < \frac{1}{2\sigma^2}\left(\kappa\theta - \frac{1}{2}\sigma^2\right)^2$ as stated in Theorem 3.3.1. The right hand side is analytic in $\mathbb{R} \cap \left\{ \mu, \mu < \frac{1}{2\sigma^2}\left(\kappa\theta - \frac{1}{2}\sigma^2\right)^2 \right\}$. We know from the identity theorem in complex analysis (see the first chapter of Gunning and Rossi (2022) for more details) that since the left hand side and the right hand side are both analytic and coincide in $\mathbb{R}_- \cap \left\{ \mu, \mu < \frac{1}{2\sigma^2}\left(\kappa\theta - \frac{1}{2}\sigma^2\right)^2 \right\}$, they must coincide in $\mathbb{R} \cap \left\{ \mu, \mu < \frac{1}{2\sigma^2}\left(\kappa\theta - \frac{1}{2}\sigma^2\right)^2 \right\}$ as well. Hence $\left\{ \mu \in \mathbb{R}, \mathbb{E}\left[\exp\left(\mu \int_0^T \frac{1}{y_u} du \right) \right] < \infty \right\} = \left\{ \mu, \mu < \frac{1}{2\sigma^2}\left(\kappa\theta - \frac{1}{2}\sigma^2\right)^2 \right\}$ and the expression of $\mathbb{E}\left[\exp\left(\mu \int_0^T \frac{1}{y_u} du \right) \right]$ is given by equation (3.18). □

To use Theorem 3.3.1 in our setup, we therefore require:

$$\Upsilon^4 \gamma^2 < \frac{1}{2\sigma^2}\left(\kappa\theta - \frac{1}{2}\sigma^2\right)^2,$$

$$\text{which is equivalent by definition of } \gamma \text{ to } \Upsilon^4 \frac{1}{\sigma^2}\left(\kappa\theta - \frac{1}{4}\sigma^2\right)^2 < \frac{1}{2\sigma^2}\left(\kappa\theta - \frac{1}{2}\sigma^2\right)^2,$$

$$\text{which is equivalent to } \sqrt{2}\Upsilon^2\left(\kappa\theta - \frac{1}{4}\sigma^2\right) < \kappa\theta - \frac{1}{2}\sigma^2.$$

This is satisfied thanks to our assumption $\kappa\theta > \left[\frac{1-\Upsilon^2\frac{\sqrt{2}}{2}}{1-\Upsilon^2\sqrt{2}}\right]\frac{1}{2}\sigma^2 > \frac{1}{2}\sigma^2$ with $\Upsilon \in \left(0, 2^{1/4}\right)$.

Under these conditions we can apply Theorem 3.2.1 and express:

$$x_t^* = x_t + \frac{1}{2}\kappa\int_s^t \alpha_u e^{-\frac{1}{2}\kappa(t-u)}du. \tag{3.19}$$

3.3.3 Application to credit spreads and cumulative hazard rate

For our numerical tests, we set $\Upsilon = 1$ as it simplifies the equations and does not impact the numerical results.

We now need to find the relation between the credit spreads term structure in RN and RW. In what follows, we note $f(t) = \frac{1}{2}\kappa\int_s^t \alpha_u e^{-\frac{1}{2}\kappa(t-u)}du$. We can naturally draw the relation between the real-world and risk-neutral default intensities. We recall that $x_t = \sqrt{y_t}$ and $\lambda_t = y_t + \psi_t$

$$\lambda_t^* = \lambda_t + f_t^2 + 2f_t\sqrt{y_t}. \tag{3.20}$$

We have so far described and justified, via $\sqrt{y_t}$ (which is x_t), a link between λ_t and λ_t^*. Theorem 3.3.2 gives the relation between credit spreads in RN and RW.

Theorem 3.3.2. *The real-world term structure of credit spreads,* $\mathrm{Sp}^*(t,T)$*, defined in the space* $\left(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P}\right)$*, is related to the risk-neutral term structure of credit spreads,* $\mathrm{Sp}(t,T)$*, defined by equation (3.13) in the space* $\left(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{Q}\right)$*, by the following equation:*

$$e^{-(T-t)\,\mathrm{Sp}^*(t,T)} = e^{-B(t,T)F(t,\sqrt{y_t})}\left[-\delta\left(1 - e^{B(t,T)F(t,\sqrt{y_t})}\right) + e^{-(T-t)\,\mathrm{Sp}(t,T)}\right], \tag{3.21}$$

where $F(t,\sqrt{y_t}) := f_t^2 + 2f_t\sqrt{y_t}$.

Proof. We set in this proof for notation simplicity $M(t,T) = \frac{S^m(0,T)}{S^m(0,t)}\frac{A(0,t)}{A(0,T)}\frac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}}A(t,T)$ and $F(t,x_t) = f_t^2 + 2f_t x_t$. We also have shown $x_t^* = x_t + f_t$ with $f_t = \frac{1}{2}\kappa\int_s^t \alpha_u e^{-\frac{1}{2}\kappa(t-u)}du$. Since, by construction, equation (3.13) can be similarly written for the credit spreads under

RW, we have:

$$\begin{aligned}
\mathrm{Sp}^*(t,T) &= -\frac{1}{T-t}\ln\left[\delta+(1-\delta)M(t,T)e^{-B(t,T)y_t^*}\right]\\
y_t &= x_t^2 \text{ this implies } y_t^* = y_t + f_t^2 + 2f_t x_t\\
&\text{this gives}\\
e^{-(T-t)Sp^*(t,T)} &= \delta+(1-\delta)M(t,T)e^{-B(t,T)y_t}e^{-B(t,T)(f_t^2+2f_t x_t)}\\
&= e^{-B(t,T)F(t,x_t)}\left[\delta e^{B(t,T)F(t,x_t)}+\delta-\delta+(1-\delta)M(t,T)e^{-B(t,T)y_t}\right]\\
&= e^{-B(t,T)F(t,x_t)}\left[-\delta\left(1-e^{B(t,T)F(t,x_t)}\right)+\underbrace{\left(\delta+(1-\delta)M(t,T)e^{-B(t,T)y_t}\right)}_{\exp(-(T-t)Sp(t,T))}\right]\\
e^{-(T-t)Sp^*(t,T)} &= e^{-B(t,T)F(t,x_t)}\left[-\delta\left(1-e^{B(t,T)F(t,x_t)}\right)+e^{-(T-t)Sp(t,T)}\right].
\end{aligned}$$

□

So far, we can justify two of the features that we wanted our model to hold.

- **Arbitrage-free**: The easiest way to justify the arbitrage-free property of this RW model is to use Theorem 2.7 of (Harrison and Pliska, 1981). Since we start from a risk-neutral measure, we ensure that the model is arbitrage-free. This argument is used, for instance, by (Norman, 2009).
- **Realistic Term structures**: The natural way to justify that our model produces a realistic term structure of credit spreads is to observe the said curves. This is what Figures 3.4.2 and 3.4.4 show. We can, however, have an intuitive hint that this property is verified. Let us consider equation (3.21). When considering the trivial case where $\alpha(u)$ is constant at 0, we indeed have the equivalence between $\mathrm{Sp}(.,.)$ and $\mathrm{Sp}^*(.,.)$. Therefore, the curves under RW are realistic as soon as RN curves are, which has already been ensured in Chapter 2. Then, when (t,T) is fixed, $e^{-(T-t)\,\mathrm{Sp}^*(t,T)}$ is a linearly shifted $e^{-(T-t)\,\mathrm{Sp}(t,T)}$. These comments on the realistic curve property are also intended to help having a better understanding and interpret equation (3.21).

Further developments are, however, needed for the third property, which is the ability to fit any arbitrarily chosen curve. We have given, through equation (3.21), the link between the credit spread at time t and maturity T under RN and RW probability measures. The model we consider fits the initial market survival probabilities or, identically, the initial market cumulative hazard rate curve. The cumulative hazard rate at time t and for the time horizon T is related to the survival probability, $S(t,T)$, by the following equation:

$$S(t,T) = e^{-\Lambda(t,T)}. \tag{3.22}$$

We also recall that in Chapter 2 we give the expression of survival probabilities in our

CIR++ intensity context:

$$S(t,T) = \frac{S^m(0,T)}{S^m(0,t)} \frac{A(0,t)}{A(0,T)} \frac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}} A(t,T) e^{-B(t,T)[\lambda(t)-\psi(t)]}. \tag{3.23}$$

In what follows, we derive the model to be able to fit any arbitrarily given market cumulative hazard rate curve. This feature is arguably the most convenient and of highest practical use. It will allow researchers practitioners to observe the full credit spread curve deformation as one maturity is set to any arbitrarily chosen value. This provides an efficient way to conduct regulatory or internal stress scenarios. It also provides analysts with a tool to better evaluate the behavior (with respect to credit spreads) of any product they are willing to acquire or sell, and therefore have a better understanding of its forthcoming valuation's variation.

To achieve this feature, we calibrate the function $\alpha(t)$ to ensure that the expectation of $\Lambda^*(t,T)$ in the real-world space corresponds to cumulative hazard rate anticipation / stress values / forecasts, $\vec{c}$. In other terms, we are willing to obtain an equation to solve for $\alpha(u)$ that would guarantee for a chosen maturity $\widetilde{T}$ and for a set of N dates $\{t_i\}_{i\in[\![1,N]\!]}$, $\mathbb{E}^{\mathbb{P}}\Lambda^*(t_i,\widetilde{T}) = c_i,\ \forall i \in [\![1,N]\!]$. Theorem 3.3.3 gives the relation between the cumulative hazard rate under RN and RW.

Theorem 3.3.3. *The real-world cumulative hazard rate, $\Lambda^*(t,T)$, defined in the space $\left(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P}\right)$, is related to the risk-neutral cumulative hazard rate, $\Lambda(t,T)$, defined by equations (3.22) and (3.23) in the space $\left(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{Q}\right)$, by the following linear equation:*

$$\Lambda^*(t,T) = \Lambda(t,T) + B(t,T)F(t,\sqrt{y_t}). \tag{3.24}$$

Proof. Equation (3.23) similarly describes the survival probability under RW:

$$S^*(t,T) = \frac{S^m(0,T)}{S^m(0,t)} \frac{A(0,t)}{A(0,T)} \frac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}} A(t,T) e^{-B(t,T)[\lambda^*(t)-\psi(t)]},$$

and we still have the relation $-\ln[S^*(t,T)] = \Lambda^*(t,T)$. These two equations lead to:

$$-\ln[S^*(t,T)] = \Lambda^*(t,T) = -\ln\left[\frac{S^m(0,T)}{S^m(0,t)} \frac{A(0,t)}{A(0,T)} \frac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}} A(t,T)\right] + B(t,T)[\lambda_t^* - \psi_t].$$

Since $\lambda^*(t) - \psi(t) = y^*(t)$ and $y_t^* = (x_t^*)^2 = (x_t + f_t)^2$, we finally write:

$$\Lambda^*(t,T) = \underbrace{-\ln\left[\frac{S^m(0,T)}{S^m(0,t)}\frac{A(0,t)}{A(0,T)}\frac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}}A(t,T)\right] + B(t,T)y_t}_{\Lambda(t,T)} +$$
$$B(t,T)\underbrace{[f_t^2 + 2f_t x_t]}_{F(t,x_t)}.$$

From which we finally have

$$\Lambda^*(t,T) = \Lambda(t,T) + B(t,T)F(t,\sqrt{y_t}).$$

□

Taking the expectations under the RW probability measure $\mathbb{P}$ on equation (3.24), it is straightforward to write:

$$\mathbb{E}^{\mathbb{P}}\Lambda^*(t,T) = \mathbb{E}^{\mathbb{P}}\Lambda(t,T) + B(t,T)[f_t^2 + 2f_t\mathbb{E}^{\mathbb{P}}\sqrt{y_t}],$$

Then we can derive the equation to solve $\forall t_i \in \{t_i\}_{i\in[\![1,N]\!]}$ for maturity $\widetilde{T}$:

$$\mathbb{E}^{\mathbb{P}}\Lambda^*(t_i,\widetilde{T}) = \mathbb{E}^{\mathbb{P}}\Lambda(t_i,\widetilde{T}) + B(t_i,\widetilde{T})\left[f^2(t_i) + 2f(t_i)\mathbb{E}^{\mathbb{P}}\sqrt{y(t_i)}\right],$$

which implies

$$\left[c_i - \mathbb{E}^{\mathbb{P}}\Lambda(t_i,\widetilde{T})\right]\frac{1}{B(t_i,\widetilde{T})} = f^2(t_i) + 2f(t_i)\mathbb{E}^{\mathbb{P}}\sqrt{y(t_i)}, \tag{3.25}$$

with $f(t) = \frac{1}{2}\kappa\int_s^t \alpha_u e^{-\frac{1}{2}\kappa(t-u)}du$. We are solving (3.25) for $\alpha(t)$. We can choose many types of functions for $\alpha(t)$. (Berninger and Pfeiffer, 2021) analyze the impact of choosing a constant, step, or linear function. In this work, we choose step functions as they are an easy way to obtain time-varying parameters. Other possibilities will not be discussed here. Therefore, $\alpha(u)$ is a step function that will have exactly N steps α^i, $i \in [\![1,N]\!]$ (the function will remain constant at its last value after the last constraint value).

$$\alpha(t) = \sum_{i=1}^{N-1}\mathbb{1}_{t\in[t_i,t_{i+1}]}\alpha^i + \alpha^N\mathbb{1}_{t>t_N}.$$

It comes the following expression for $f(t)$:

$$f(t) = \begin{cases} e^{-\frac{1}{2}\kappa t}\sum_{i=1}^{m}\alpha^i\left(e^{\frac{1}{2}\kappa t_{i+1}} - e^{\frac{1}{2}\kappa t_i}\right) & \text{if } t \le t_N \text{ and } m \text{ s.t. } t\in[0,t_m] \\ e^{-\frac{1}{2}\kappa t}\sum_{i=1}^{N-1}\alpha^i\left(e^{\frac{1}{2}\kappa t_{i+1}} - e^{\frac{1}{2}\kappa t_i}\right) + e^{-\frac{1}{2}\kappa t}\alpha^N\left(e^{\frac{1}{2}\kappa t} - e^{\frac{1}{2}\kappa t_N}\right) & \text{if } t > t_N. \end{cases} \tag{3.26}$$

The proof is given in appendix 3.A.

• **Solving for $\alpha(t)$**
Let us now explain how each element of equation (3.25) is obtained:

- The set of scalars $\{c_i\}_{i\in[\![1,N]\!]}$ are the expert-wise chosen targets of the cumulative hazard rate. They can be forecasts, stress values and so on. Each c_i is a target cumulative hazard rate value for the target date t_i.
- The expectation $\mathbb{E}^{\mathbb{P}}\Lambda(t_i,\widetilde{T})$ is the expectation of the risk-neutral cumulative hazard rate for the horizon $\widetilde{T}$ rate under the the real-world probability. It is obtained for each t_i via Monte-Carlo simulation.
- The scalar $B(t_i,\widetilde{T})$ is obtained via equation (3.14).
- The scalar $f(t_i)$ is obtained via equation (3.26).
- The expectation $\mathbb{E}^{\mathbb{P}}\sqrt{y(t_i)}$ is also simulated in the RW Monte-Carlo process using $y(t_i)$ that is also required to obtain $\mathbb{E}^{\mathbb{P}}\Lambda(t_i,\widetilde{T})$.

• **Credit spreads anticipation**
It is clear that practitioners willing to use a RW credit spreads model will focus on credit spreads rather than cumulative hazard rates. Similarly, regulatory stress tests are concerned with credit spreads and not cumulative hazard rates. In the previous sections, we presented the link between RW and RN credit spreads (equation (3.21)), but when it came to calibrating the function $\alpha(t)$ to fulfill the third property (reproducing any arbitrarily given curve), we used the cumulative hazard rate. This was due to two main reasons. First, it made sense from a modeling perspective, as the model was designed to fit the market structure of survival probabilities or, equivalently, cumulative hazard rates at the initial time. The other reason was more practical: the link between survival probabilities and cumulative hazard rates linearized the problem, simplifying the calculations. Fortunately, going back to credit spreads is relatively easy. From equation (3.13) describing credit spreads, we have in RW:

$$\begin{aligned}\mathrm{Sp}^*(t,T) &= -\frac{1}{T-t}\ln\left[\delta+(1-\delta)\frac{S^m(0,T)}{S^m(0,t)}\frac{A(0,t)}{A(0,T)}\frac{e^{-B(0,t)y_0}}{e^{-B(0,T)y_0}}A(t,T)e^{-B(t,T)[\lambda^*(t)-\psi(t)]}\right]\\ &= -\frac{1}{T-t}\ln\left[\delta+(1-\delta)S^*(t,T)\right]\end{aligned}$$

$$\mathrm{Sp}^*(t,T) = -\frac{1}{T-t}\ln\left[\delta+(1-\delta)e^{-\Lambda^*(t,T)}\right] \tag{3.27}$$

This equation can also be inverted to:

$$\Lambda^*(t,T) = -\ln\left[\frac{1}{1-\delta}\left(e^{-(T-t)\,\mathrm{Sp}^*(t,T)}-\delta\right)\right]^{2}. \tag{3.28}$$

[2]This equation is not valid for any chosen $\mathrm{Sp}(,.)$ as it requires $\mathrm{Sp}(t,T) < -\frac{1}{T-t}\ln(\delta)$. This condition is also required in Chapter 2, to infer the initial survival probabilities from credit spreads. As stated by the

• Practical use

The few steps below summarize how the model can be used to make simulation under the real-world probability space. We recall that we have justified that the model under RW is arbitrage-free, and Section (3.4), Figures 3.4.2 and 3.4.4 will show that we produce realistic curves.

1. First, one must choose the N target values $\{c_i\}_{i\in[\![1,N]\!]}$ for the cumulative hazard rate at dates $\{t_i\}_{i\in[\![1,N]\!]}$ and for the maturity $\widetilde{T}$:
   - They can be chosen directly based on expert judgment.
   - More likely, they will be obtained through targets from credit spreads and inverted to cumulative hazard rates using equation (3.28).
2. Then we simulate $\Lambda(t,T)$ in RN (like any classic random simulation) using survival probabilities via equation (3.23) and $\Lambda(t,T) = -\ln[S(t,T)]$.
3. Using equation (3.24), we infer $\Lambda^*(t,T)$. Note that this equation also requires the simulation of y_t.
4. Finally, we obtain the RW credit spreads term structure using equation (3.27).

3.4 Numerical tests: an application to economic forecasts and to stress tests

Let us now take a look at a very practical use of RW credit spreads modeling. We introduce two real-world configurations to present two ways the model can be used. The first configuration is a forecast scenario, and the second is a stress scenario.

In all the simulations that follow, we use the same data set and parameters as those used in Chapter 2. In particular, the simulations are based on Credit Agricole's credit spread, survival probability, default intensity, and cumulative hazard rate. The initial time is taken as of January 1, 2024. The differentiation is made between its bond yield and the French Government's bond yield. The parameters are those calibrated in the so-called *global scenario* (see Table 3.B.1 in Appendix 3.B). Similarly, the diffusion of the cumulative hazard rate under RN follows the same logic as in the referenced paper. We start by diffusing the default intensity using its distribution, and using equation (3.23), we get the cumulative hazard rate. We recall that a solution of the CIR model follows a noncentral chi-square, $\chi^2[2cy(s); 2q+2, 2w]$, with $2q+2$ degrees of freedom and parameter of noncentrality $2w$ (see (Cox et al., 2005)). The density of this distribution is given by:

$$f(y(t),t\ ;\ y(s),s) = ce^{-w-v}\left(\frac{v}{w}\right)^{q/2} I_q\left(2(wv)^{1/2}\right),$$

authors, this condition is not restrictive. δ is the recovery rate and is often taken as 40%, as we did in all our numerical simulations. For a $[t,T]$ gap as wide as 10, this condition allows for credit spreads up to 900 basis points (BPs).

where $c := \frac{2\kappa}{\sigma^2\left(1-e^{-\kappa(t-s)}\right)}$, $w := cy(s)e^{-\kappa(t-s)}$, $v := cy(t)$ and $q := \frac{2\kappa\theta}{\sigma^2} - 1$.

and $I_q(\cdot)$ is the modified Bessel function of the first kind of order q. To perform the simulations, we have used R's library sde[3]. Note that this distribution is conditional to the past values. Therefore, it requires setting a time step and keeping track of previous values. We take a weekly time step.

Goldman Sachs' 2024 Global Credit Outlook

We first use the credit spreads forecasts of Goldman Sachs from their 2024 global credit report.[4] The report anticipates a slight decrease in the values of credit spreads. The behavior expected for European financial issuers is described in Table 3.4.1 below:

Current (Q4 2023)	Q1 2024	Q2 2024	Q3 2024	Q4 2024
201	194	190	187	184

Table 3.4.1: In BPs, Goldman Sachs' Euro financial credit spreads forecast

These forecasts do not cover individual counterparties, including Crédit Agricole. However, we can easily transform them into relative changes and apply them to the behavior of Crédit Agricole's credit spreads. They are also not differentiated by maturity. We apply these forecasts to the 5-year tenor:

Current (Q4 2023)	Q1 2024	Q2 2024	Q3 2024	Q4 2024
113	109	107	105	103

Table 3.4.2: In BPs, Goldman Sachs' Euro financial credit spreads forecasts, translated to Crédit Agricole's 5-year tenor.

Now, per the previous section, this stress should be transformed into a cumulative hazard rate stress. Applying equation (3.28), it gives the following stress:

[3] https://CRAN.R-project.org/package=sde

[4] https://www.goldmansachs.com/intelligence/pages/gs-research/2024-global-credit-outlook-back-in-the-saddle/report.pdf

Current (Q4 2023)	Q1 2024	Q2 2024	Q3 2024	Q4 2024
0.096	0.093	0.091	0.089	0.088

Table 3.4.3: Credit Agricole 5Y-credit spreads forecasts translated to cumulative hazard rate forecasts

For the simulation, we make 20,000 draws. The simulation starts on January 1, 2024, with the actual initial credit spreads. The time step is 1 week. We assume that the first target (Q1 2024) is attained after the first time step (January 8, 2024); then, between two quarters, the credit spread remains constant until the next forecast value, and so on. This means that we have one target c_i for each time step t_i, and thus, the number of time steps N is 52. In other terms we have for $I_0 = [01/01/2024, 1^{st}\ \text{week}[$, $I_1 = [1^{st}\ \text{week}, \text{Q1 2024}[$, $I_2 = [\text{Q1 2024}, \text{Q2 2024}[$, $I_3 = [\text{Q2 2024}, \text{Q3 2024}]$ and $I_4 = [\text{Q3 2024}, \text{Q4 2024}]$:

$$\{t_i\}_{i\in[\![1,52]\!]} = [\![1^{st}\ \text{week}, \text{Q1 2024}]\!] \cup]\!] \text{Q1 2024}, \text{Q2 2024}\, 4[\![\cup [\![\text{Q3 2024}, \text{Q4 2024}]\!]$$

$$c_i = \begin{cases} 0.096 \text{ if } t_i \in I_0 \\ 0.093 \text{ if } t_i \in I_1 \\ 0.091 \text{ if } t_i \in I_2 \\ 0.089 \text{ if } t_i \in I_3 \\ 0.088 \text{ if } t_i \in I_4 \end{cases}$$

Figures 3.4.1 and 3.4.2 below respectively show the behavior of the 5-year credit spread during the RW simulation and the behavior of the entire term structure.

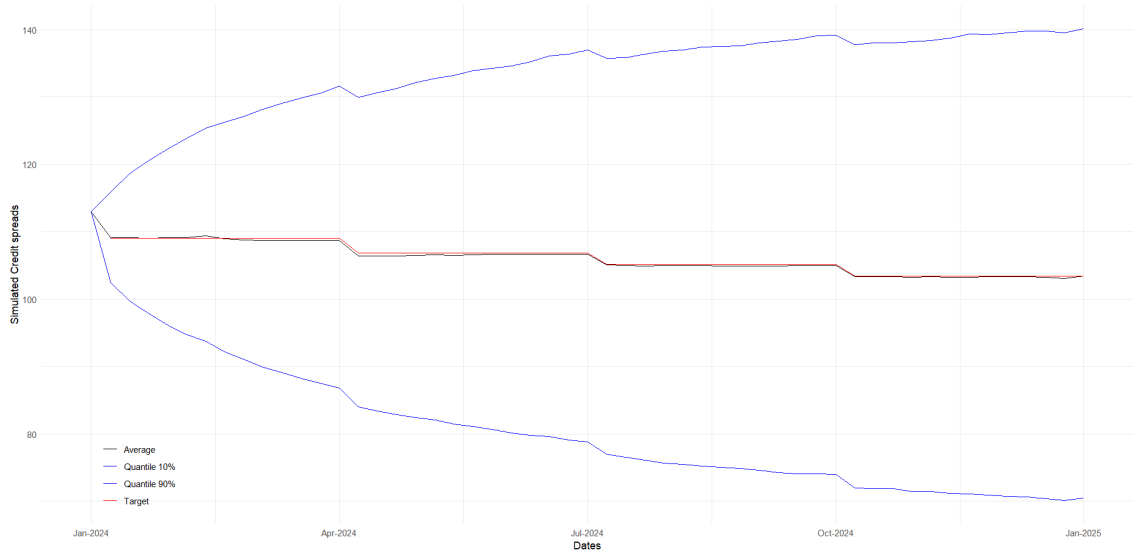

Figure 3.4.1: In BPs, 5-year Credit Agricole's credit spreads, simulated values and target values (Goldman Sachs's forecast). We observe the expectation and the 10th and 90th percentiles.

We observe that the expectation of the simulated 5-year credit spreads (black curve) almost exactly matches the target values (red curve), while the quantiles (blue curves) encompass the expectation curve.

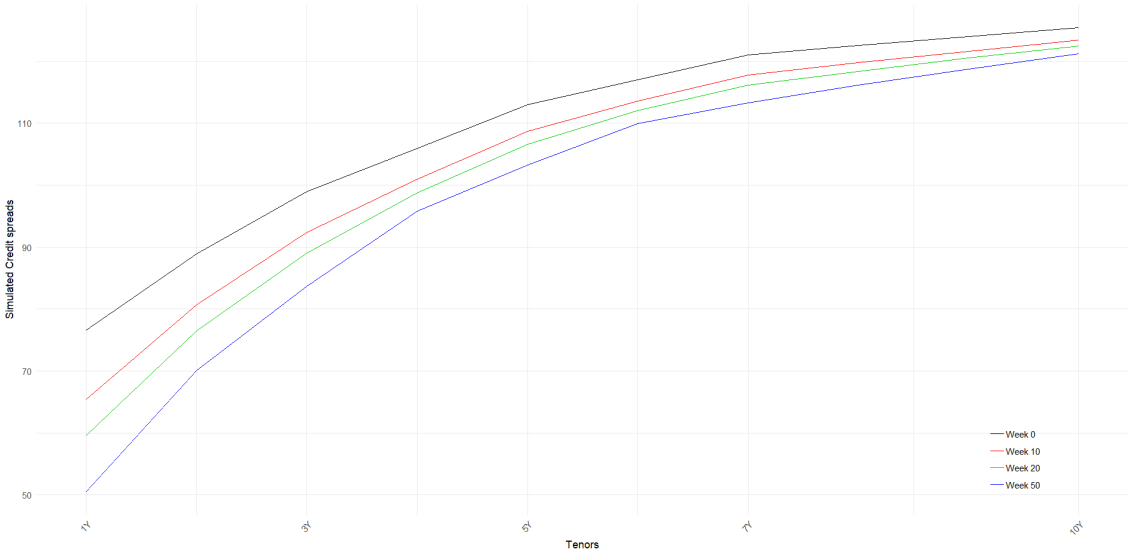

Figure 3.4.2: In BPs, the expectation of the term structure of Credit Agricole's credit spreads under the forecast scenario at different dates in the future.

We observe how the entire term structure decreases as a consequence of the RW behavior of the credit spreads. This is indeed the intuitive result one would expect. Another very important observation is that the behavior imposed on the 5-year credit spread still results in very realistic credit spreads across the entire term structure.

EBA's 2023 EU-wide stress test

The second configuration uses the European Banking Authority (EBA)'s regulatory stress test.[5]. The stress test provides an absolute change to apply to issuers' credit spreads by region and sector, based on credit rating. For French financial counterparties rated between 1 and 2 by ECAIs (External Credit Assessment Institutions), which is the case of Credit Agricole as of January 1, 2024, the stress value is 133 BPs. While the regulator usually requires an instantaneous stress test, we consider here a linear progression of the stress value over a one-year period, for the sake of illustration. We still take the 5-year maturity. The simulation still involves 20,000 draws with a 1-week time step. Therefore, we have $N = 52$ and $\forall i \in [\![1, N]\!]$:

$$t_i = i\frac{1}{52}$$
$$c_i = 113 + 133t_i$$

[5] https://www.eba.europa.eu/publications-and-media/press-releases/eba-launches-2023-eu-wide-stress-test

Figures 3.4.3 and 3.4.4 below, respectively show the behavior of the 5-year credit spread during the RW simulation and the behavior of the entire term structure.

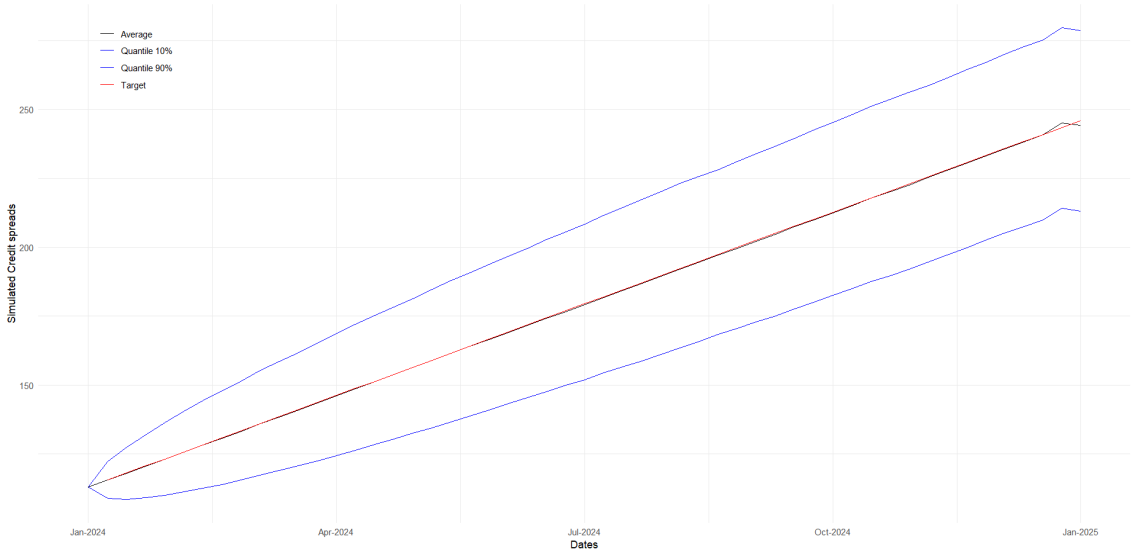

Figure 3.4.3: In BPs, 5-year Credit Agricole's credit spreads, simulated values and target values (EBA stress test). We observe the expectation and the 10th and 90th percentiles.

We observe here that, once again, we fit the target set for the 5-year maturity almost perfectly throughout the simulation. We include in Appendix 3.C the graphics showing the behavior of the 5-year horizon cumulative hazard rate in both scenarios.

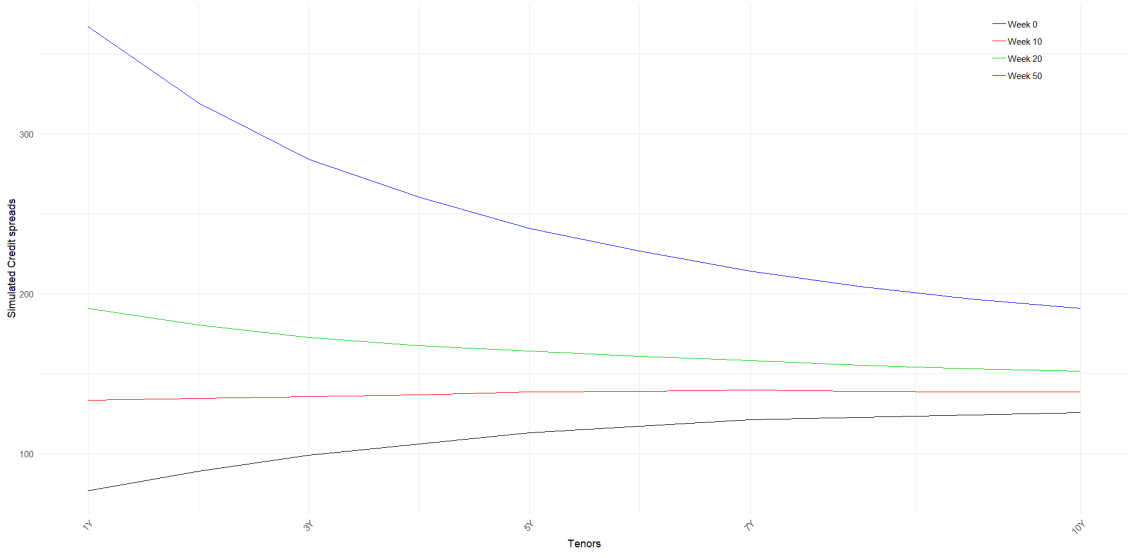

Figure 3.4.4: In BPs, the expectation of the term structure of Credit Agricole's credit spreads under the stress test scenario at different dates in the future.

We can observe that, for this example, we have again, indeed produced a very realistic

term structure of credit spreads. A notable feature is the inversion of the term structure as the stress increases. This behavior is observed in stressed market conditions, and we provide a few examples in Appendix 3.D.

The process we have conducted in this section and the previous one can be generalized to other models. One needs to apply Theorem 3.2.1 after verifying condition H2. Then, the one needs to find the equivalent of Theorem 3.3.3 for their model. This will provide them with the equation to solve in order to calibrate α_t. Note that this is straightforward for models with additive noise, as the function $\phi(x)$ is just the identity function.

3.5 Conclusion

In this research, we developed a comprehensive novel methodology for transitioning financial diffusion models from the risk-neutral measure to the real-world measure, leveraging probability theory results. Our framework has been applied to various models, including those with non-additive noise, such as the CIR++ model. The framework's utilization has been successfully demonstrated through numerical simulations on credit spreads' forecasts and stress tests. Drawing upon Credit Agricole's credit spread data relative to France's Government yield, we analyzed the robustness of our theoretical framework. Through applications to Goldman Sachs' 2024 global credit outlook forecasts and the European Banking Authority (EBA) 2023 stress tests, we validate the practical relevance and applicability of our model. The transition resulted in an arbitrage-free model capable of generating realistic credit spread term structures and accurately replicating arbitrarily chosen credit spread curves.

Our methodology offers practitioners and researchers a robust tool that enhances the precision and ease of conducting stress tests and forecasts. It allows for a detailed observation of the term structure of key risk indicators after applying forecasts or stress values to a specific maturity. Furthermore, the model accurately captures the behavior of the term structure in response to changes under high-stress conditions; the term structure of simulated RW credit spreads exhibits inversion of the curve, mirroring a behavior observed in the market. These modified term structures can be used for various risk calculations, including assessing the impact of interest rate variations on portfolio valuations through full revaluation of securities or derivatives. It can also be used for XVA calculations to estimate forward rates for determining future variable cash flows. The potential applications of this framework are extensive and varied, providing significant benefits for stress testing and risk management.

It is noteworthy that the framework does come with some complexities, particularly the need to verify conditions like the Novikov criterion, which can be tedious when dealing with models with non-additive noise. While our current model provides a solid foundation, future research avenues could explore enhancements such as incorporating jumps, generalizing the framework to dimensions greater than one, and applying it to other models and

risk indicators, such as credit default swaps (CDS) premiums and foreign exchange (FX) rates. This would further expand the utility and applicability of the framework, making it a versatile tool in financial risk management and forecasting. Furthermore, the main advancement that could enhance the general applicability of the framework would be to generalize the approach by introducing a linearization for equation (3.9) and an error measurement for that linearization. For some models, the relation described by equation (3.9) might be very complex, making it difficult to express the indicator of interest under the real-world probability space as a function of the indicator under the risk-neutral probability space, as we do with Theorem 3.3.3.

Appendices

3.A Expression of $f(t)$

We give here the proof of equation (3.26) which gives:

$$f(t) = \begin{cases} e^{-\frac{1}{2}\kappa t} \sum_{i=1}^{m} \alpha^i \left(e^{\frac{1}{2}\kappa t_{i+1}} - e^{\frac{1}{2}\kappa t_i} \right) & \text{if } t \leq t_N \text{ and } m \text{ s.t. } t \in [0, t_m] \\ e^{-\frac{1}{2}\kappa t} \sum_{i=1}^{N-1} \alpha^i \left(e^{\frac{1}{2}\kappa t_{i+1}} - e^{\frac{1}{2}\kappa t_i} \right) + e^{-\frac{1}{2}\kappa t} \alpha^N \left(e^{\frac{1}{2}\kappa t} - e^{\frac{1}{2}\kappa t_N} \right) & \text{if } t > t_N \end{cases}$$

Proof.

- $t > t_N$, let m be taken, s.t. $t \in [0, t_m]$

$$\begin{aligned} f(t) &= \frac{1}{2}\kappa e^{-\frac{1}{2}\kappa t} \int_0^t \alpha(u) e^{\frac{1}{2}\kappa u} du \\ &= \frac{1}{2}\kappa e^{-\frac{1}{2}\kappa t} \int_0^t \sum_{i=1}^{N} \mathbb{1}_{u \in [t_i, t_{i+1}]} \alpha^i e^{\frac{1}{2}\kappa u} du \\ &= \frac{1}{2}\kappa e^{-\frac{1}{2}\kappa t} \sum_{i=1}^{m} \int_{t_i}^{t_{i+1}} \alpha^i e^{\frac{1}{2}\kappa u} du \\ f(t) &= e^{-\frac{1}{2}\kappa t} \sum_{i=1}^{m} \alpha^i \left(e^{\frac{1}{2}\kappa t_{i+1}} - e^{\frac{1}{2}\kappa t_i} \right) \end{aligned}$$

- $t \leq t_N$, let m be taken, s.t. $t \in [0, t_m]$

$$\begin{aligned} f(t) &= \frac{1}{2}\kappa e^{-\frac{1}{2}\kappa t} \int_0^t \alpha(u) e^{\frac{1}{2}\kappa u} du \\ &= \frac{1}{2}\kappa e^{-\frac{1}{2}\kappa t} \int_0^{t_N} \alpha(u) e^{\frac{1}{2}\kappa u} du + \frac{1}{2}\kappa e^{-\frac{1}{2}\kappa t} \int_{t_N}^t \alpha(u) e^{\frac{1}{2}\kappa u} du \\ &= \frac{1}{2}\kappa e^{-\frac{1}{2}\kappa t} \int_0^{t_N} \sum_{i=1}^{N-1} \mathbb{1}_{u \in [t_i, t_{i+1}]} \alpha^i e^{\frac{1}{2}\kappa u} du + \frac{1}{2}\kappa e^{-\frac{1}{2}\kappa t} \int_{t_N}^t \alpha(u) e^{\frac{1}{2}\kappa u} du \\ &= \frac{1}{2}\kappa e^{-\frac{1}{2}\kappa t} \sum_{i=1}^{N-1} \int_{t_i}^{t_{i+1}} \alpha^i e^{\frac{1}{2}\kappa u} du + \frac{1}{2}\kappa e^{-\frac{1}{2}\kappa t} \int_{t_N}^t \alpha^N e^{\frac{1}{2}\kappa u} du \\ f(t) &= e^{-\frac{1}{2}\kappa t} \sum_{i=1}^{N-1} \alpha^i \left(e^{\frac{1}{2}\kappa t_{i+1}} - e^{\frac{1}{2}\kappa t_i} \right) + e^{-\frac{1}{2}\kappa t} \alpha^N \left(e^{\frac{1}{2}\kappa t} - e^{\frac{1}{2}\kappa t_N} \right) \end{aligned}$$

□

3.B Parameters of the CIR++ Intensity model

We take the parameters that resulted from the global calibration described in Chapter 2.

κ	θ	σ	y_0
5.138×10^{-1}	1.497×10^{-2}	8.904×10^{-2}	4.348×10^{-2}

Table 3.B.1: Parameters calibrated for the *Global scenario*

3.C 5Y horizon cumulative hazard rate behaviour in the RW simulations

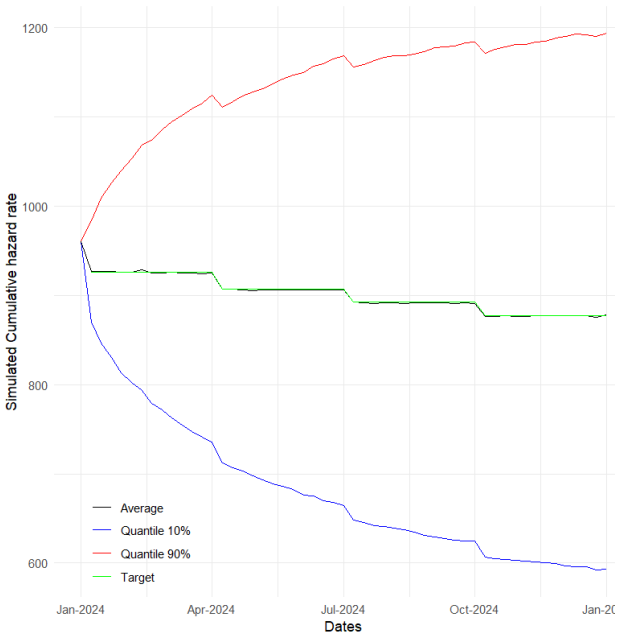

Figure 3.C.1: In BPs, 5Y Credit Agricole's cumulative hazard rate under the forecasts for the 5Y scenario

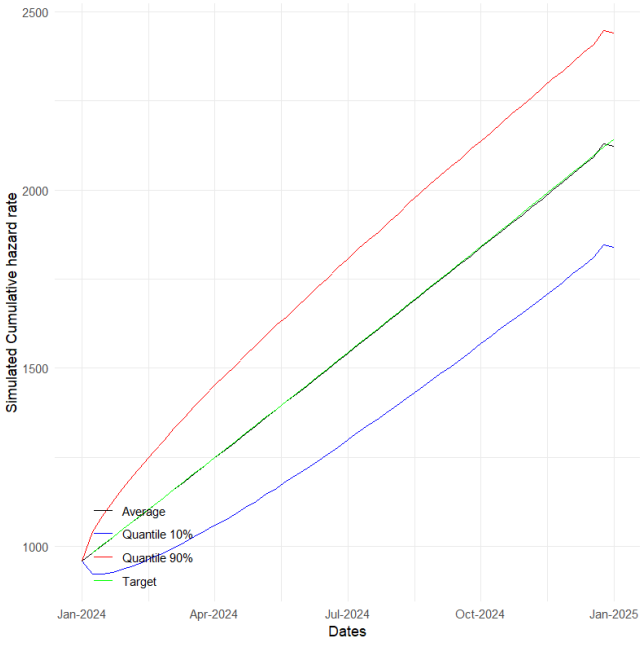

Figure 3.C.2: In BPs, 5Y Credit Agricole's cumulative hazard rate and target values (EBA stress test)

3.D Inversion of the credit spreads curve

As mentioned in Section 3.4, we observed an inversion of the credit spread curve during periods of stress. Figure 3.D.1 shows inverted credit spreads during the stress period (dates within the Eurozone sovereign debt crisis in 2012 and 2013) compared to standard condition credit spreads (2022 and 2023).

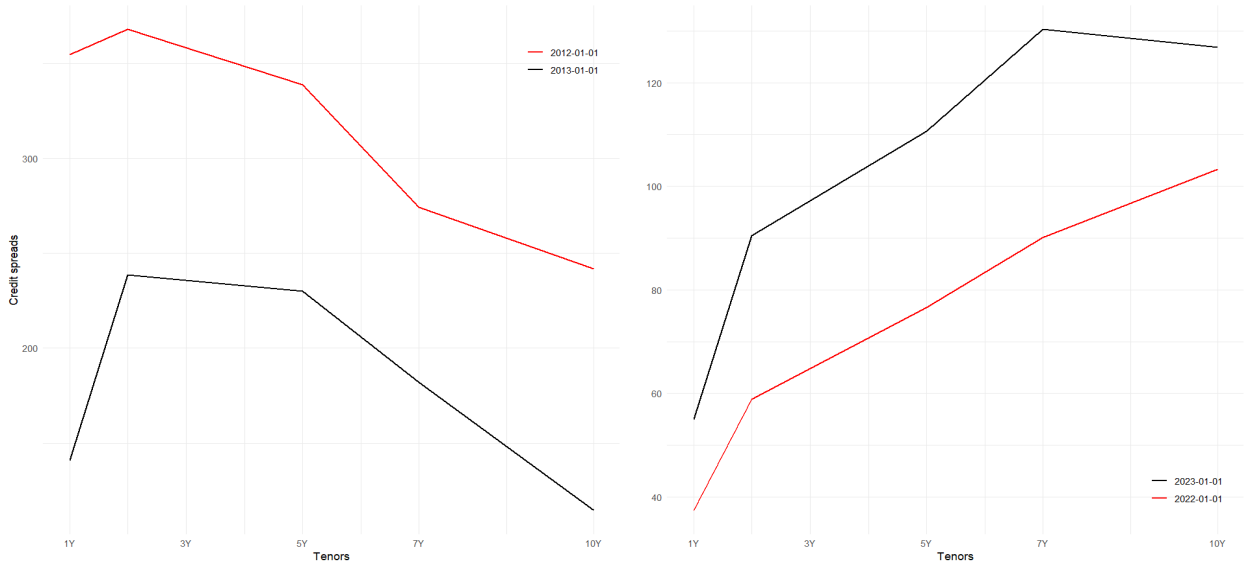

Figure 3.D.1: In BPs, Credit Agricole's credit spread term structure inverted during stress periods (2012/01/01 and 2013/01/01) and regular under standard conditions (2022/01/01 and 2023/01/01).

Chapter 4

Deep Calibration of Interest Rate Models

Abstract

For any financial institution, it is essential to understand the behavior of interest rates. Despite the growing use of Deep Learning, for many reasons (expertise, ease of use, etc.), classic rate models such as CIR and the Gaussian family are still widely used. In this chapter, we propose to calibrate the five parameters of the G2++ model using Neural Networks. Our first model is a Fully Connected Neural Network and is trained on covariances and correlations of Zero-Coupon and Forward rates. We show that covariances are more suited to the problem than correlations due to the effects of the unfeasible backpropagation phenomenon, which we analyze in this chapter. The second model is a Convolutional Neural Network trained on Zero-Coupon rates with no further transformation. Our numerical tests show that our calibration based on deep learning outperforms the classic calibration method used as a benchmark. Additionally, our Deep Calibration approach is designed to be systematic. To illustrate this feature, we applied it to calibrate the popular CIR intensity model.

4.1 Introduction

Governments, industries, and banks have to manage the behavior of various financial indicators. Whether it is to manage risks or optimize investment returns, it is necessary to understand, forecast, and stress drivers of fair value assets, for example. Among those drivers are interest rates (IR), which can affect IR derivatives such as IR swaps, swaptions, cross-currency swaps, and so on. IR models are widely used in banks and financial institutions to assess IR behavior. There are many models. The Vasicek model paved the way for these models (Vasicek, 1977a) and was later improved by many others. The Cox-Ingersoll-Ross Model (CIR) (Cox et al., 1985a) is often used as it is quite simple to use and calibrate. However, it is unifactorial, which limits its use. The Gaussian model G2++ allows for two factors as it is a deterministically shifted two-factor Vasicek model.

This document focuses on the calibration of the G2++ model (also called the Gauss2++ or Gaussian two-factor model) using deep learning (DL) techniques. More precisely, we will introduce two different approaches applied to different sets of relevant data to calibrate the G2++ model using Neural Networks (NN). These approaches are expected to be at least as accurate as the *classic* ones (see the next paragraphs of the current section for the state of the art), to perform faster, and above all, to be easier to use regarding the treatments necessary on the data. In what follows, we will present two different approaches to calibrate IR models.

We consider a time horizon $T > 0$. We consider a probability space $\left(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{Q}\right)$. The sample space Ω represents the set of all possible outcomes. $\mathcal{F}$ is a σ-field representing the set of events $A \subset \Omega$. $\mathbb{Q}$ is the risk-neutral probability that assigns a probability to each event in $\mathcal{F}$ and under which discounted obligation prices are martingales. In this setup, we recall the equations defining the G2++ model:

$$\begin{aligned} r(t) &= \phi(t) + x(t) + y(t) \\ dx(t) &= -K_x x(t)dt + \sigma_x dW_t^x, \ x(0) = x_0 \\ dy(t) &= -K_y y(t)dt + \sigma_y dW_t^y, \ y(0) = y_0 \text{ with } K_x, K_y, \sigma_x, \sigma_y > 0. \end{aligned} \tag{4.1}$$

In the above equation, $(W_t^x, W_t^y)_{t\in[0,T]}$ is a two-dimensional $(\mathcal{F}_t)_{t\in[0,T]}$-Brownian motion with correlation ρ, i.e., $[dW_t^x, dW_t^y] = \rho\, dt$. The function $\phi(t)$ is, as usual, deterministic and allows the model to fit the initial term structure of forward rates. In the remainder of this work, the function $\phi(T) = f^{Market}(0,T) + \frac{\sigma_x^2}{2K_x^2}\left(1 - e^{-K_x T}\right)^2 + \frac{\sigma_y^2}{2K_y^2}\left(1 - e^{-K_y T}\right)^2 + \rho\frac{\sigma_x\sigma_y}{K_x K_y}\left(1 - e^{-K_x T}\right)\left(1 - e^{-K_y T}\right)$, where $f^{Market}(0,T)$ is the market initial forward rate (see Section 4.2.1 of Brigo and Mercurio (2006) for more details). We then require the (x,y)-model to fit the market *as best as possible* via calibration and then update the function $\phi(t)$ using the optimal parameters (see discussion after Lemma 2 of Mbaye and Vrins (2022)). The scalars $K_x, K_y, \sigma_x, \sigma_y$ are positive parameters, and ρ is in $[-1, 1]$. These are the parameters of the model and require calibration.

Despite our work being applied to G2++, the approaches are actually systematic,

meaning that they are not model-dependent as long as some requirements, which will be detailed in Section 4.2, are met. In Section 4.5, we show another possible application with the CIR intensity model. Hence, the Deep Calibration process we introduce here is a contribution to the calibration of IR models and to the application of DL to finance. The first matter is a historical one, and many solutions have been introduced over the years.

Indeed, the accuracy of the model used for IR is highly dependent on the quality of the calibration. For the one-factor Hull-White (HW) model, Gurrieri et al. (2009) defines the calibration by these three elements:

1. The choice of constant or time-dependent mean reversion and volatility.
2. The choice of products to calibrate to, and whether to calibrate locally or globally.
3. Whether to optimize the mean reversion and the volatility together or separately, and in the latter case, how to estimate one independently of the other.

We can generalize this definition by extending the first and third points to include not only the mean reversion and volatility but any parameters required in the model. Each of these three points is important and will be discussed in a moderate to thorough manner.

Independently of these points, in general, the calibration process involves minimizing the distance between a theoretical value obtained from the model and the actual corresponding market observation. Hull and White (2001), for example, uses numerical optimization techniques such as the Levenberg-Marquardt algorithm to find the set of volatility parameters that minimize the error between the model's prices for cap and floor options and swaptions. The error used is the sum of the squares of the differences (SSD), that is, $\theta^* = \underset{\theta}{argmin} \sum_{i=1}^{N} \left(OP_i(\theta) - OP_i^{market}\right)^2$ where $OP(\cdot)$ defines an option price, N is the number of options included in the *calibration basket*, and θ is the set of parameters of the model to calibrate. The same idea of an optimization process on swaption prices can also be seen in Russo and Torri (2019), where, instead of using the SSD, the relative error is used, that is, $\theta^* = \underset{\theta}{argmin} \sqrt{\sum_{i=1}^{N} \left(\frac{OP_i(\theta) - OP_i^{market}}{OP_i^{market}}\right)^2}$. Schlenkrich (2012) also calibrates the HW model on swaptions. The optimization process is done using Gauss-Newton and Adjoint Broyden quasi-Newton methods. The required derivatives are approximated by automatic differentiation techniques.

Other calibration methods address more specific and advanced questions. For instance, in a macroeconomic context with low to negative IRs, the methods above may not be well suited. Russo and Fabozzi (2017) addresses the problem of negative IRs with a focus on the second point of our definition, the choice of products to calibrate to. More precisely, while they still use swaptions, they aim at finding the best swaption quotation. They conclude that calibrating the Hull-White model, the shift-extended CIR model, or the shift-extended squared Gaussian model using the shifted log-normal or normal volatility results in more stable quotations, which in turn provides more stable parameters.

Orlando et al. (2019) tackle the calibration of the CIR model with negative or near-zero

IR values. Their approach consists of translating the IRs to positive values, thereby retaining the initial volatility. The rate is shifted by a constant α (allowing one to keep the initial dynamics), i.e., $r_{shift}(t) = r_{real}(t) + \alpha$. They use the 99th percentile of the empirical IR probability distribution as the constant. For the calibration process, instead of minimizing the error between theoretical and market data, the authors take advantage of the fact that parameters' likelihoods in the CIR model are known (for example, Kladívko (2007); Ben Alaya and Kebaier (2013)).

Regarding the second matter, the use of DL in finance, DL techniques, especially neural networks (NNs), are increasingly being applied in various fields, including finance, where many applications have been found. According to Huang et al. (2020), for example, feedforward neural networks (FNNs), which consist of multiple layers of fully connected perceptrons (Pal and Mitra, 1992), and Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997), which are recurrent networks (NNs that can process information in two directions instead of one like FNNs), are the architectures mainly used in forecasting (exchange rates, option prices, risks, stock markets, etc.).

Additionally, DL can be used to replicate financial instruments or characteristic behaviors. Heaton et al. (2016), for example, use autoencoders Ng et al. (2011), which are neural architectures originally designed to automatically learn features from unlabeled data to replicate the inputs. The authors' purpose is to replicate indexes such as the S&P 500. A similar idea of replication is found in Bloch (2019), where the authors use a neural network with dimensionality reduction through Principal Component Analysis (PCA) to learn the dynamics of the implied volatility surface. Replicating the dynamics of indexes or volatilities can be a first step in choosing an investment or hedging strategy. However, DL can also be used directly for both of these purposes. Regarding investment strategies, Heaton et al. (2017) introduces an approach called *Deep Portfolios*, which aims to understand the key factors driving asset prices before generating the mean-variance efficient portfolio. As for hedging, Buehler et al. (2019), with the *Deep Hedging* approach, uses reinforcement learning—which employs a reward system so the algorithm can learn from its experience (in-depth details can be found in Kaelbling et al. (1996))—to find the optimal hedge given available instruments in a hypothetical market driven by the Heston model.

Another application of DL in finance that is gaining attention is asset pricing. For example, Chen et al. (2020)'s approach combines three different architectures with distinct purposes: an FNN aimed at understanding non-linearities, an LSTM network to identify sets of economic state processes that help the model understand macroeconomic conditions relevant to asset pricing, and a generative adversarial network (GAN) (for more on GANs, see Creswell et al. (2018)) to identify the strategies with the most unexplained pricing information. The added value of the authors' work is also in their inclusion of the no-arbitrage assumption via a stochastic discount factor, which they demonstrate helps improve performance. Jang et al. (2021) use a deep FNN to predict market option values. Their performance is enhanced by pre-training the network with data generated using parametric option pricing methods (Black-Scholes, Monte Carlo, finite differences, and binomial trees)

to overcome the lack of market data and unbalanced datasets.

Finally, another application of DL to finance, which is the main focus of this chapter, is the calibration of financial models. Pironneau (2019), for example, calibrated the Heston model for European options using FNNs. While the author achieves good precision with a relatively shallow architecture, performance stalls at a certain point. Hernandez (2016) also uses FNNs to calibrate financial models in general and applies this approach to the one-factor Hull-White model using swaption prices. The author outperforms classic methods in terms of calibration times. In Horvath et al. (2021), the authors calibrate the volatility surface using neural networks. They draw inspiration from implied volatility and option prices, conceptualizing them as a set of pixels. After achieving good performance in terms of computation time and accuracy, the authors analyze the use of deep calibration algorithms to identify the stochastic model that most accurately represents a given dataset, aiming to discern which model best characterizes the market.

Very recently, Büchel et al. (2022) conducted a comparison of the calibration of an interest rate term structure model based on the Trolle and Schwartz volatility model, with and without the integration of artificial neural networks (NNs). Their findings indicate that employing NNs leads to faster and more stable results over time. Notably, their methodology utilizes NNs to learn the model itself. Consequently, to obtain the calibrated parameters, a final optimization step is required. The calibration framework they used is based on the work of (Liu et al., 2019), where the authors use NNs to calibrate the Heston pricing model and the Bates model. In that paper, the authors also have an initial step that involves learning to map parameters to prices (referred to as the forward pass in both references) before proceeding to the actual calibration. In comparison, our approach involves a neural network architecture that directly outputs the parameters of the models, thereby completing the calibration process directly.

The approach we present differs from the aforementioned calibration processes using DL in both the input data (we do not use options prices) and the architectures we propose. Indeed, depending on the input data, we use either a Feedforward Neural Network (FNN) or a Convolutional Neural Network (CNN) (see (Albawi et al., 2017) for details). Notably, in the previously mentioned works, FNNs are primarily used. Overall, one of the primary distinctions between the research presented in this work and much of the existing literature on the subject, is that we focus directly on interest rate models and their parameters instead of pricing functions. Additionally, this work introduces two types of calibration: direct and indirect calibration.

Moreover, in the current work, as part of our efforts to assess the robustness of the calibration process, we conduct an analysis where we deliberately introduce noise into the inputs. Additionally, in our systematic approach, the models we introduce are applied to the simultaneous calibration of the five parameters of the G2++ model but can be applied to any model, as long as some non-restrictive conditions, detailed in Section 4.2, are met. Our work also differs from the current literature regarding the types of inputs since we train our first neural network on covariance and correlation matrices of Zero-Coupon (ZC)

rates and forward (FWD) rates. We show that for feedforward calibration neural networks, covariances provide richer information compared to correlations. To our knowledge, feeding neural networks using covariance matrices to calibrate models has not yet been explored in the literature, and our research fills this gap. This leads us to the introduction of the unfeasible backpropagation phenomenon (see Theorem 4.3.1), which supports the choice of covariances over correlations—a concept that, to the best of our knowledge, had not been previously addressed in existing literature.

In what follows, Section 4.2 will describe the models we introduce in this chapter. For each model, we will also explain how the dataset used for training was constructed. The results obtained, both in terms of accuracy and computational performance, will be presented in Section 4.3. Interesting intermediary results are also provided in this section, where we demonstrate that covariances of ZC rates are more suited for the Deep Calibration process than correlations due to vanishing gradients. More specifically, we present a theorem explaining when backpropagation becomes unfeasible (see Theorem 4.3.1), a result that, to our knowledge, is not commonly found in the literature. Section 4.4 will compare our results to classic calibration methods, showing better accuracy and a significant reduction in calibration time. Finally, Section 4.5 will illustrate how the work on the G2++ model can be easily applied to a CIR intensity model.

4.2 A systematic approach with Deep Calibration (DC) for interest rate models

In what follows, we will present two different approaches to calibrate IR models. Both approaches are not model-dependent (and thus systematic) and could be applied to any model (financial or not) as long as what we call the Observable Quantity of Interest (OQOI), which must be a relevant information bearer, can be both observed (in the market, in our financial context) and obtained through an analytical expression from the model that one wants to calibrate.

One of the main concerns of our DC models is the ease of use regarding data. We aim to use easily available market data that require minimal transformations. For the training of our DL models, we decided to use synthetic data based on numerical simulations in both approaches. In more detail, we generate the synthetic datasets from a set of parameters calibrated from real market data (see Sections 4.2.1.2 and 4.2.2.2). There are many reasons in favor of this approach:

- The dataset becomes of unlimited size.
- We can modify the data to be more representative of a specific context (a stressed one, for example), or we can decide to include many different contexts.
- Unlike the other calibration papers mentioned previously, we can measure our error directly vis-à-vis the real parameters.
- Our DL architectures gets to actually learn what *drives* the IR model. This means

that when performing out-of-sample calibration (outside of the learning process), the DL model will find the parameters that truly suit the given IR model. If, instead, real-world data were used, we would have found the parameters corresponding to the real-world data closer to the IR model. In other words, using synthetic data allows us to replicate the behavior of the model we are currently calibrating.

The last point is important as it also helps us stay consistent with our aim of calibrating IR models rather than forecasting IR curves, which is a different task that has also received attention in the literature (see, for example, Oh and Han (2000); Jacovides (2008); Vela (2013)). The main reasons to support this choice are that, as of today, professionals and researchers using IR forecasts have more mastery over IR models than they do over DL models, since the former have been widely used for many decades. Additionally, IR models have many advantages; for instance, it is quite straightforward to stress IR forecasts via classic models (through real-world simulations or alterations of parameters).

The following two subsections present each approach by explaining how the analytical expressions allowing synthetic data generation are obtained, how the dataset is constructed, and which architecture is chosen. Both approaches—in different ways—use the risk-neutral values of ZC and FWD rates.

We recall the expression of the ZC bond price $P(\cdot,\cdot)$ at time $t \in [0,T]$ and maturity T under the G2++ model (described by equation (4.1)):

$$\begin{aligned} P(t,T) =& \frac{P^M(0,T)}{P^M(0,t)} \exp\{\mathcal{A}(t,T)\}, \\ \text{with } \mathcal{A}(t,T) =& \frac{1}{2}[V(t,T) - V(0,T) + V(0,t)] - \frac{1-e^{-K_x(T-t)}}{K_x} x(t) - \frac{1-e^{-K_y(T-t)}}{K_y} y(t), \end{aligned} \tag{4.2}$$

and $P^M(0,T)$, the initial ZC bond price at maturity T, and $V(\cdot,\cdot)$ verifying:

$$\begin{aligned} V(t,T) =& \frac{\sigma_x^2}{K_x^2}\left[T-t+\frac{2}{K_x}e^{-K_x(T-t)} - \frac{1}{2K_x}e^{-2K_x(T-t)} - \frac{3}{2K_x}\right] \\ &+ \frac{\sigma_y^2}{K_y^2}\left[T-t+\frac{2}{K_y}e^{-K_y(T-t)} - \frac{1}{2K_y}e^{-2K_y(T-t)} - \frac{3}{2K_y}\right] \\ &+ 2\rho\frac{\sigma_x\sigma_y}{K_xK_y}\left[T-t+\frac{e^{-K_x(T-t)}-1}{K_x} + \frac{e^{-K_y(T-t)}-1}{K_y} - \frac{e^{-(K_x+K_y)(T-t)}-1}{K_x+K_y}\right]. \end{aligned} \tag{4.3}$$

Since the ZC rate $Z(.,.)$ satisfies $P(t,T) = e^{-(T-t)Z(t,T)}$, we have:

$$Z(t,T) = -\frac{1}{T-t} ln\left\{\frac{P^M(0,T)}{P^M(0,t)}\right\} - \frac{1}{T-t}\{\mathcal{A}(t,T)\}. \tag{4.4}$$

Finally, as for the FWD rate $f(.,.)$, we have $f(t,T) = -\frac{\partial}{\partial T} lnP(t,T)$, then:

$$\begin{aligned} f(t,T) &= -\frac{\partial}{\partial T}\ln P(t,T) = -\frac{\partial}{\partial T}A(t,T) \\ &= -\frac{\sigma_x^2}{2K_x^2}\left[1-2e^{-K_x(T-t)}+e^{-2K_x(T-t)}\right] \\ &+ \frac{\sigma_y^2}{2K_y^2}\left[1-2e^{-K_y(T-t)}+e^{-2K_y(T-t)}\right] \\ &+ \rho\frac{\sigma_x\sigma_y}{k_xk_y}\left[1-e^{-K_x(T-t)}+e^{-K_y(T-t)}-e^{-(K_x+K_y)(T-t)}\right] \\ &+ e^{-K_x(T-t)}x(t)+e^{-K_y(T-t)}y(t). \end{aligned} \tag{4.5}$$

The proofs of these expressions are given in the fourth chapter of (Brigo and Mercurio, 2006).

Let us also define, for the remainder of this document, the set of parameters to calibrate, $\theta = \{\theta^i\}_{i\in[\![1,5]\!]} = \{K_x,\ K_y,\ \sigma_x,\ \sigma_y,\ \rho\}$.

4.2.1 Indirect Deep Calibration of the G2++ model

4.2.1.1 Covariances and Correlations as OQOI

For our first approach, we start with the observation that ZC and FWD rates are indeed easily observable in the market and that an analytical expression can be derived from most, if not all, IR models. However, if we want to maintain a simple Fully Connected Network (FCN) architecture, directly using them as our OQOI might be time-consuming and less efficient. We decide to use intermediary quantities (hence the term *Indirect Deep Calibration*) that bear enough compressed information from the ZC and FWD rates curves to allow for efficient calibration. Good candidates for the OQOI could be the correlation and covariance matrices of ZC and FWD rates variations.

From equation (4.5) we can re-write:

$$f(t,T) = \psi(t,T)\ + e^{-K_x(T-t)}x(t)\ + e^{-K_y(T-t)}y(t),\ t\in[0,T] \tag{4.6}$$

with

$$\begin{cases} x(t) = e^{-K_xt}x_0 + \sigma_x e^{-K_xt}\int_0^t e^{K_xs}dW_s^x \\ y(t) = e^{-K_yt}y_0 + \sigma_y e^{-K_yt}\int_0^t e^{K_ys}dW_s^y. \end{cases}$$

As the integrands in the above equations are deterministic, the covariance and the Itô's

bracket match and for two given tenors T_i and $T_j \in [0, T]$, we have

$$d\text{Cov}(f(t,T_i), f(t,T_j)) = \Big[\sigma_x^2 e^{-K_x(T_i+T_j-2t)} + \sigma_y^2 e^{-K_y(T_i+T_j-2t)} + \rho\sigma_x\sigma_y \left(e^{-K_xT_i-K_yT_j+(K_x+K_y)t} + e^{-K_yT_i-K_xT_j+(K_x+K_y)t}\right)\Big] dt. \tag{4.7}$$

From now on, following (Brigo and Mercurio, 2006) (see chapter four) and adopting their notations, we rather focus on the so-called instantaneous covariances and correlations defined by the following equation:

$$\begin{aligned}
&\text{Cov}\left(df(t,T_i), df(t,T_j)\right) := \\
&\quad \left[X(t,T_i)X(t,T_j) + Y(t,T_i)Y(t,T_j) + \rho\left(X(t,T_i)Y(t,T_j) + X(t,T_j)Y(t,T_i)\right)\right] dt \\
&\text{Cor}\left(df(t,T_i), df(t,T_j)\right) := \\
&\quad \frac{\left[X(t,T_i)X(t,T_j) + Y(t,T_i)Y(t,T_j) + \rho\left(X(t,T_i)Y(t,T_j) + X(t,T_j)Y(t,T_i)\right)\right]}{\sqrt{\left[X^2(t,T_i) + Y^2(t,T_i) + 2\rho X(t,T_i)Y(t,T_i)\right] . \left[X^2(t,T_j) + Y^2(t,T_j) + 2\rho X(t,T_j)Y(t,T_j)\right]}},
\end{aligned} \tag{4.8}$$

where $X(t,T) = \sigma_x e^{-K_x(T-t)}$ and $Y(t,T) = \sigma_y e^{-K_y(T-t)}$. Note that in practice, $df(t,T)$ is interpreted as $f(t+\varepsilon,T) - f(t,T)$ with $dt = \varepsilon$. Equation (4.7) defines covariances for FWD rates. We want a similar expression for ZC rates. One just needs to re-write (4.4) in a way that splits the stochastic and the deterministic parts of the equation

$$Z(t,T) = \widetilde{\psi}(t,T) - \frac{1-e^{-K_x(T-t)}}{TK_x}x(t) - \frac{1-e^{-K_y(T-t)}}{TK_y}y(t), \tag{4.9}$$

with

$$\widetilde{\psi}(t,T) = -\frac{1}{T}\ln\left\{\frac{P^M(0,T)}{P^M(0,t)}\right\} - \frac{1}{2T}[V(t,T) - V(0,T) + V(0,t)].$$

In the same way as for the FWDs, we use (4.9) to introduce the instantaneous covariances and correlations for ZCs defined by

$$\begin{aligned}
&\text{Cov}\left(dZ(t,T_i), dZ(t,T_j)\right) := \\
&\quad \left[\widetilde{X}(t,T_i)\widetilde{X}(t,T_j) + \widetilde{Y}(t,T_i)\widetilde{Y}(t,T_j) + \rho\left(\widetilde{X}(t,T_i)\widetilde{Y}(t,T_j) + \widetilde{X}(t,T_j)\widetilde{Y}(t,T_i)\right)\right] dt \\
&\text{Cor}\left(dZ(t,T_i), dZ(t,T_j)\right) := \\
&\quad \frac{\left[\widetilde{X}(t,T_i)\widetilde{X}(t,T_j) + \widetilde{Y}(t,T_i)\widetilde{Y}(t,T_j) + \rho\left(\widetilde{X}(t,T_i)\widetilde{Y}(t,T_j) + \widetilde{X}(t,T_j)\widetilde{Y}(t,T_i)\right)\right]}{\sqrt{\left[\widetilde{X}^2(t,T_i) + \widetilde{Y}^2(t,T_i) + 2\rho\widetilde{X}(t,T_i)\widetilde{Y}(t,T_i)\right] . \left[\widetilde{X}^2(t,T_j) + \widetilde{Y}^2(t,T_j) + 2\rho\widetilde{X}(t,T_j)\widetilde{Y}(t,T_j)\right]}},
\end{aligned} \tag{4.10}$$

where $\widetilde{X}(t,T) = \sigma_x\frac{1-e^{-K_x(T-t)}}{K_xT}$ and $\widetilde{Y}(t,T) = \sigma_y\frac{1-e^{-K_y(T-t)}}{K_yT}$.

In what follows, COV-ZC, COV-FWD, COR-ZC, and COR-FWD will designate, in that order, covariances of ZC rates, covariances of FWD rates, correlations of ZC rates, and correlations of FWD rates. A fortiori, when a pair of maturities (T_1, T_2) is appended, it will designate the function applied to that specific pair of maturities; e.g., COV-ZC(5Y, 7Y) is the covariance of the ZC rate of maturity 5Y and the ZC rate of maturity 7Y.

4.2.1.2 Data set construction

The dataset construction is done in five steps:

1. Calibrating reference parameters on real-world market data for the G2++ model using MSE (Mean Squared Error) minimization. The MSE is given by:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2\,, \tag{4.11}$$

   where, y_i is the actual value for the ith observation, $\hat{y}_i$ is the predicted value for the ith observation and n is the total number of observations.
2. Extending the reference parameters to define a range of acceptance and drawing the parameters within the corresponding intervals.
3. Using equations (4.8) and (4.10) to compute the set of covariances and correlations for ZC and FWD rates.
4. Choosing the best scaling transformations.
5. Finally, splitting the dataset into training and test sets.

Step 1: Choice of reference parameters
Table 4.2.1 shows the parameters obtained from a calibration on real market Euro ZC rates data ranging from June 2019 to November 2020. The calibration leading to these parameters is performed on ZC correlations and covariances as described in Section 4.4.1. These reference parameters are obtained without any use of NNs and are only used to infer a wide range of realistic parameters that will be used to train our NNs.

K_x	K_y	σ_x	σ_y	ρ
0.07173132	0.08930784	0.09465584	0.094675523	-0.999318

Table 4.2.1: Reference parameters

Step 2: Extension of reference parameters and random draws
The idea here is, for a given $\gamma > 0$, to define intervals for our parameters ranging from $\theta^i - \gamma\theta^i$ to $\theta^i + \gamma\theta^i$ for each reference parameter θ^i, except for the correlation, where we choose to take ρ in the interval $[-0.999318,\ 0.999318]$. In what follows, we take $\gamma = \frac{2}{3}$. We obtain the following intervals:

	K_x	K_y	σ_x	σ_y	ρ
MIN	0.02391044	0.02976928	0.03155195	0.03155851	-0.999318
MAX	0.1195522	0.1488464	0.15775973	0.15779254	0.999318

Table 4.2.2: Range of parameters

We decide to draw $N = 10,000$ parameters uniformly from each interval.

Step 3: Computing covariances and correlations for ZCs and FWDs
The only question here is which maturities are chosen for the computations. Since in the real market, especially for short terms, liquidity can affect the rate curves (see, e.g., (Covitz and Downing, 2007)), we decide to use only tenors from 1 year to 12 years with a 1-year step. The covariance and correlation matrices are our features. They are transformed into vectors by stacking each column of the lower triangular matrix without the diagonal for correlations. Hence, we obtain an $\mathbb{R}^{nf}$ array with $nf = 66$ for correlations and $nf = 78$ for covariances. For three sets of randomly selected parameters, Figure 4.2.1 below shows the covariance of ZC rates, and Figure 4.2.2 shows the correlations of FWD rates. Appendix 4.A shows similar correlations and covariances from market ZC ratess.

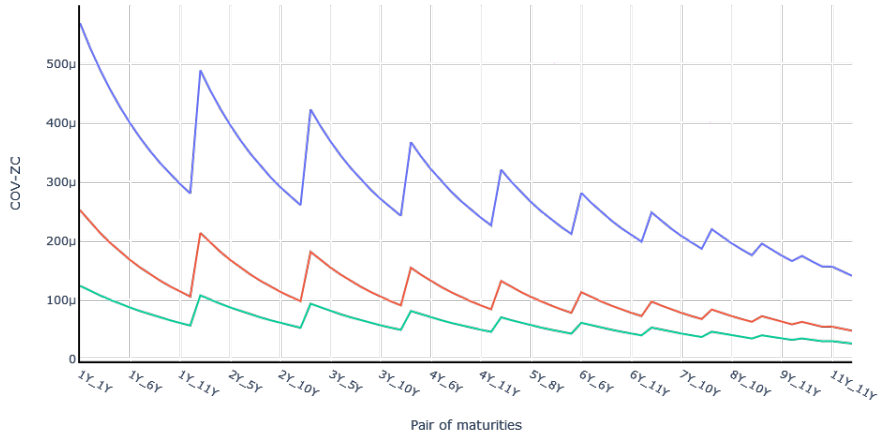

Figure 4.2.1: Covariances of ZC rates for random sets of parameters.

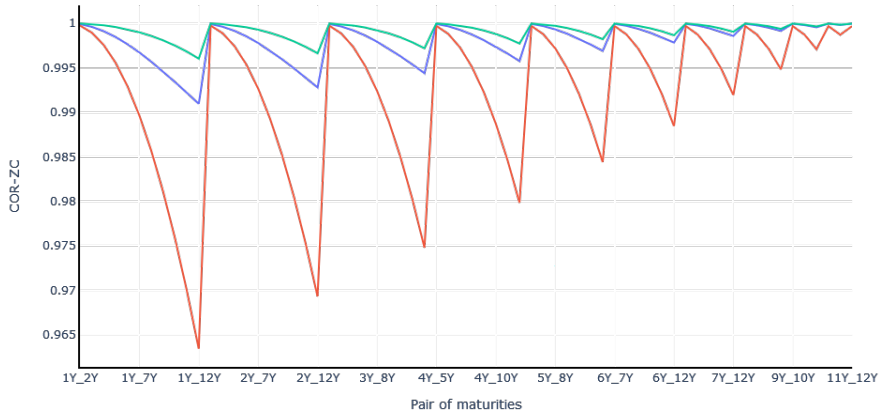

Figure 4.2.2: Correlations of FWD rates for random sets of parameters

In the two graphics above, for the blue curve, $K_x = 0.03$, $K_y = 0.14$, $\sigma_x = 0.1$, $\sigma_y = 0.09$ and $\rho = 0.84$; for the red curve, $K_x = 0.04$, $K_y = 0.1$, $\sigma_x = 0.06$, $\sigma_y = 0.1$ and $\rho = 0.18$; finally for the green curve, $K_x = 0.05$, $K_y = 0.06$, $\sigma_x = 0.05$, $\sigma_y = 0.13$ and $\rho = -0.87$.

Step 4: Scaling transformations
According to our numerical tests, the best transformation is obtained using the min-max scaler applied to both features and targets: $u_{\text{transformed}} = \frac{u - \min(u)}{\max(u) - \min(u)}$.

Step 5: Splitting the data set
We now have at our disposal four different datasets (correlations and covariances for each ZC and FWD rates), each with 10,000 entries. For each of these four sets, we randomly allocate 80% to the training set and 20% to the test set. The training sets will be used for training the neural networks, while the test sets will be reserved for post-training analysis of our models' performance. Additionally, the test sets will be used to conduct comparisons between the results obtained from the different datasets.

4.2.1.3 The neural network architecture

As mentioned previously, we aim to keep a simple FCN architecture. Our neural network has five linear layers: one input layer with nf neurons (which corresponds to the number of features—66 for correlations or 78 for covariances, as explained above), three hidden layers, the first with $h_1 = 1\,000$ neurons, the second with $h_2 = 1\,500$, and the last hidden layer with $h_3 = 1\,000$ neurons, and finally, one output layer with np neurons (the number of parameters to calibrate, where $np = 5$). The first three layers are activated using a ReLU function, and the last one, as a prediction layer, has no activation. To help prevent

overfitting, we also include a dropout with a probability of 0.25 between the last hidden layer and the prediction layer.

Note that while artificial neural networks are susceptible to overfitting issues, these concerns may be mitigated, especially when employing synthetic data that allows for the creation of vast training sets without constraints, and when the data is devoid of sample noise due to simulation via deterministic functions of the parameters to calibrate (see Büchel et al. (2022)). However, in our study, despite the ability to simulate an extensive range of inputs, we have opted to limit the dataset to 10 000 entries, with only 80% utilized for training purposes. This decision has been made to accelerate training procedures and reduce the number of epochs required. Furthermore, it is worth noting that in Section 4.4, we intentionally introduce noise to the inputs to evaluate the resilience of our model. Therefore, we cannot disregard the potential for overfitting, prompting the incorporation of dropout mechanisms. The architecture can be represented as follows:

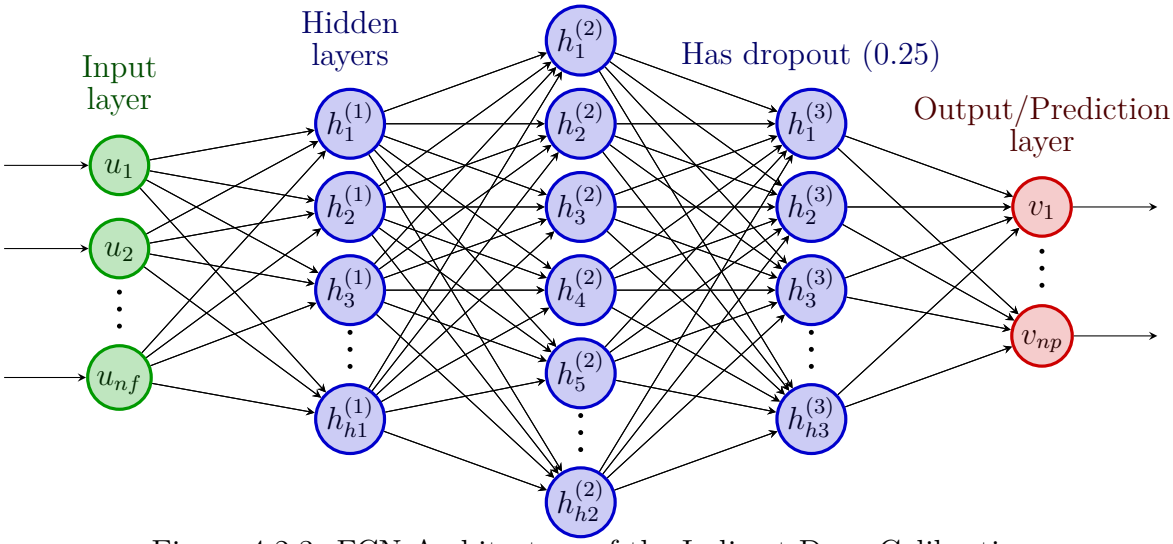

Figure 4.2.3: FCN Architecture of the Indirect Deep Calibration

Additionally, we use mini-batches of size 1 000 and train for 1 000 epochs. The Adam algorithm (Kingma and Ba, 2014) is used for optimization, without weight decay, and with a learning rate of 0.001. The error to minimize is the MSE (see Equation (4.11)).

4.2.2 Direct Deep Calibration of the G2++ model

4.2.2.1 ZCs as OQOI

The aim of this subsection is to consider a calibration process that directly uses ZC rate curves[1] as they are direct market observations, unlike correlations and covariances.

[1]We could have similarly used FWD rates, but the idea of the Direct DC is to reduce data transformations as much as possible. It is worth noting that ZC rates are directly observed in the market, which is not necessarily the case for FWD rates.

This approach makes the process easier to use and reduces computational costs since no further transformation is needed. Indeed, ZC rates can be directly plugged into our model. For calibrations, we generally want to capture historical behavior. To do so, we consider simulations of ZC rates on several dates and maturities. This means we have to deal with matrices, where the number of rows corresponds to the temporal depth, and the number of columns corresponds to the number of maturities. Due to the nature of these inputs, it seems that for this purpose, a CNN is better suited than an FCN.

4.2.2.2 Data set construction

The process is mostly the same as that for indirect DC. Only step 3 (actual computation of the OQOI) changes[2].

Computing ZC rates:
From equations (4.4) and (4.1), taking $x_0 = y_0 = 0$, we obtain an explicit formula for the expectation of ZC rates given by:

$$\mathbb{E}\left[Z(t,T)\right] = -\frac{1}{T-t}\log\left[\frac{P^M(0,T)}{P^M(0,t)}\right] - \frac{1}{2(T-t)}\left[V(t,T) - V(0,T) + V(0,t)\right]. \quad (4.12)$$

Taking the expectation of ZC rates as the input for our direct DC model allows us to obtain a typical path that could be observed in the market (see Figure 4.B.1). Note that in practice, real market ZC rate curves will be used as input when calibrating with our model.

For the initial market bond price, $P^M(0,t)$, we take the Euro curve as of 2020/11/04, and $V(.,.)$ is explicitly defined in (4.3). Unlike the indirect calibration using an FCN, with a CNN we do not have issues with the number of tenors. As stated before, when considering the initial market curve, we can observe an inhomogeneous behavior in the short-term tenors compared to the long-term ones. Taking enough maturities into account significantly reduces the influence of this behavior. Hence, we take the following sets of maturities: {1 day, 1 week, 1 month, 2 months, 3 months, 6 months, 9 months}; [1 year, 12 years] with a 1-year step; and [15 years, 50 years] with a 5-year step. Finally, for the propagation, expectations of ZCs will be computed for time steps $t_i = i\Delta,\ i \in [|1, nb_{\text{steps}}|]$, with the step size $\Delta = 1$ week, and we make 105 steps of propagation (about 2 years). Thus, our inputs will be in $\mathbb{R}^{nb_{\text{steps}} \times nb_{\text{tenors}}}$ with $nb_{\text{steps}} = 106$ and $nb_{\text{tenors}} = 28$.

Figure 4.2.4 shows, for a randomly selected set of parameters, the generated ZC rate curves for a set of projection dates. Appendix 4.B shows actual market ZC rates. We can observe similar curves.

[2]Regarding the scaling transformations, using the min-max scaler only on the targets gave better results.

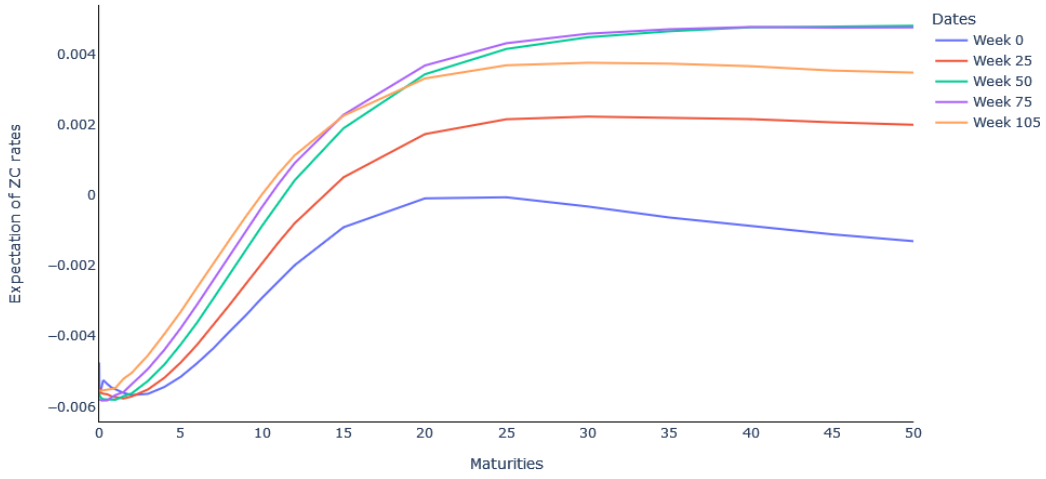

Figure 4.2.4: Example of expectations of simulated ZC rate curves for different dates. $K_x = 0.07$, $K_y = 0.09$, $\sigma_x = 0.09$, $\sigma_y = 0.09$ and $\rho = -0.99$.

4.2.2.3 The neural network architecture for direct DC

The architecture combining convolution and linear layers is still quite shallow. We start with a convolution layer of dimension (C=1, H=106, $nb_{\text{tenors}} = 28$), with a filter of size (7×7). The stride of the convolution is 2, and we also allow padding. Then, we add a pooling layer with a stride of 2. Finally, two linear layers follow, the first with 100 neurons and the second one (which is the prediction layer) with $np = 5$ neurons. Here as well, we include a dropout with a probability of 0.25 in the second-to-last layer.

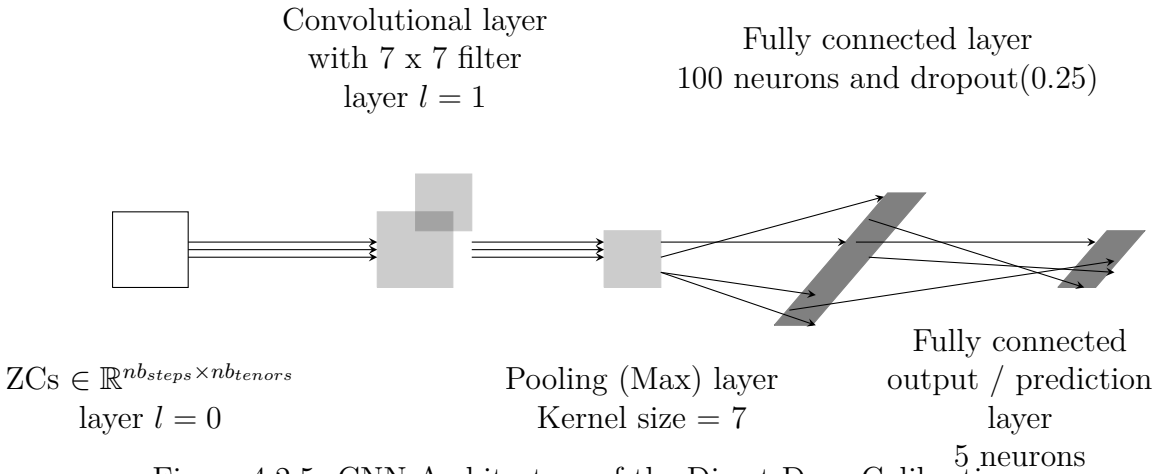

Figure 4.2.5: CNN Architecture of the Direct Deep Calibration

Similarly to the indirect DC, we use mini-batches of size 1,000, and the Adam algorithm

without weight decay is applied to minimize the MSE. The learning rate is 0.0002, and the number of epochs is 4,000.

4.3 Results for indirect and direct DC models

Regarding accuracy, both algorithms produce good results. Time efficiency is also remarkable, with calibration on the test set (2,000 entries) performed in less than 0.30 seconds for the indirect DC model and less than 0.10 seconds for the direct DC model. Table 4.3.1 below summarizes the MSEs obtained on the n_{test} observations of the test set for each parameter, given by:

$$\text{MSE}(\theta^i, \hat{\theta}^i) = \frac{1}{n_{test}} \sum_{j=1}^{n_{test}} (\theta_j^i - \hat{\theta}_j^i)^2, \text{ for i} \in [\![1,\ np]\!]. \tag{4.13}$$

Method	OQOI	K_x	K_y	σ_x	σ_y	ρ
Indirect DC	COV-ZC	3.5×10^{-4}	5.6×10^{-4}	7.7×10^{-4}	8.3×10^{-4}	8.2×10^{-2}
	COV-FWD	3.5×10^{-4}	5.5×10^{-4}	7.6×10^{-4}	8.3×10^{-4}	8.2×10^{-2}
	COR-ZC	7.5×10^{-4}	1.1×10^{-3}	1.3×10^{-3}	1.4×10^{-3}	2.0×10^{-1}
	COR-FWD	7.4×10^{-4}	1.1×10^{-3}	1.3×10^{-2}	1.4×10^{-2}	2.4×10^{-2}
Direct DC	ZCs	4.0×10^{-4}	6.3×10^{-4}	9.8×10^{-4}	1.0×10^{-4}	1.52×10^{-1}

Table 4.3.1: Calibration errors on the test set

Indirect DC

As one might expect, given the similarities between the FWD and ZC rate curves, the results with the indirect DC are very close for FWDs and ZCs rates. The most interesting result we observe is that the error using covariances is about half the error using correlations. It appears that covariances provide significantly more information for the calibration process. We can verify this numerically by observing the derivative curves of covariances and correlations with respect to the parameters, computed by finite differences. We arbitrarily select the maturity pair (5Y, 7Y) and a subset of 100 parameters for each of K_x, σ_y, and ρ, and compute the derivatives. Figures 4.3.1, 4.3.3, and 4.3.5 show the derivatives of ZC rates covariances for the sample of parameters, while Figures 4.3.2, 4.3.4, and 4.3.6 show the derivatives of ZC rates correlations for the same sample. We can directly see that the correlation derivatives quickly vanish for the three parameters. Meanwhile, for covariances, only the derivative for K_x vanishes, and it does so less quickly. For σ_y and ρ, we do not observe this vanishing behavior at all. Similar results are obtained when observing the other parameters K_y and σ_x.

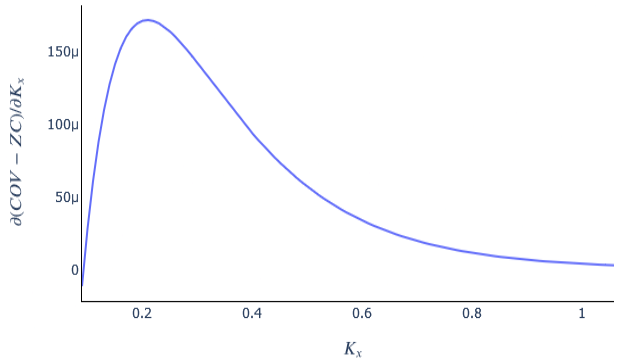

Figure 4.3.1: Derivative of COV-ZC (5Y, 7Y) w.r.t. K_x

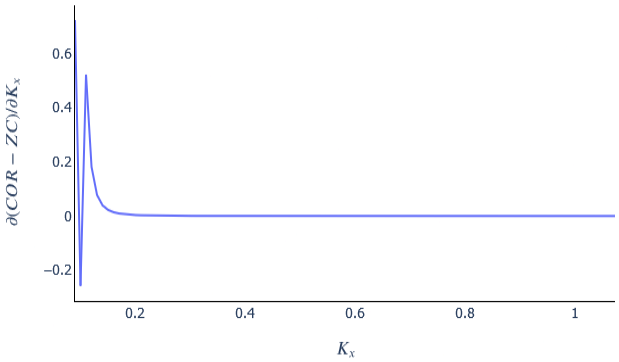

Figure 4.3.2: Derivative of COR-ZC (5Y, 7Y) w.r.t. K_x

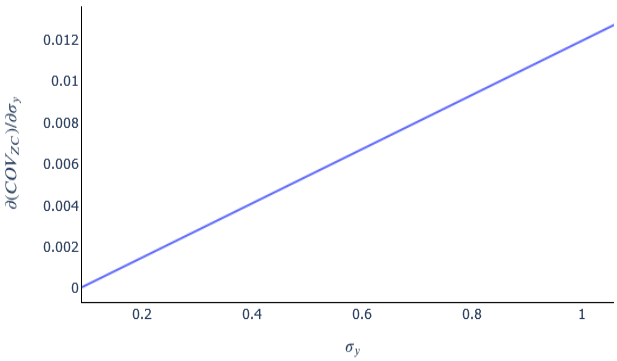

Figure 4.3.3: Derivative of COV-ZC (5Y, 7Y) w.r.t. σ_y

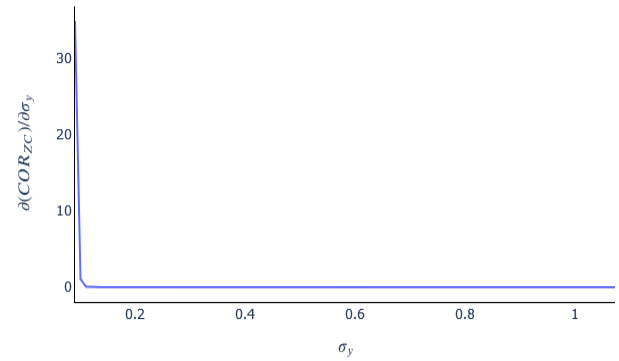

Figure 4.3.4: Derivative of COR-ZC (5Y, 7Y) w.r.t. σ_y

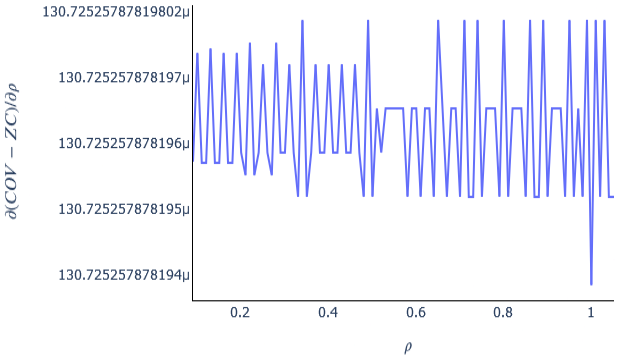

Figure 4.3.5: Derivative of COV-ZC (5Y, 7Y) w.r.t. ρ

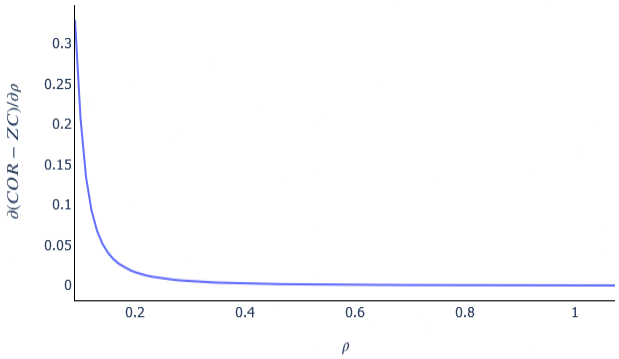

Figure 4.3.6: Derivative of COR-ZC (5Y, 7Y) w.r.t. ρ

Thanks to this analysis, we observe a phenomenon with effects similar to the vanishing gradient. When this phenomenon occurs, it prevents an effective learning process using backpropagation on correlations. In fact, we present a theorem explaining when backpropagation becomes unfeasible. To our knowledge, such an intuitive result does not appear in the literature.

Theorem 4.3.1 (Unfeasible backpropagation)**.** *In a feedforward neural network architecture, if the derivatives of the inputs with respect to the targets vanish, then the derivatives of the loss function with respect to the weights also vanish.*

The proof of this theorem is provided in Appendix 4.C. A straightforward consequence of this theorem is that when the derivatives of the loss function with respect to the weights vanish, it prevents gradient descent by backpropagation, subsequently stalling the training process. Our indirect DC provides a practical illustration of the theorem. The derivatives of the correlation with respect to the parameters (which are the targets of our NN architecture) vanish, as shown in Figures 4.3.2, 4.3.4, and 4.3.6. Consequently, the gradient descent will stall. Thus, this theorem clarifies why covariances are more suitable than correlations for our DC purposes. More generally, it provides practitioners with a rule of thumb to avoid some cases of stalling gradients.

Direct DC

Compared to the indirect DC with ZC rates covariances, the direct method leads to slightly less accurate calibrations. Although the differences are not significant, this outcome makes sense. The indirect DC benefits from information already extracted from the ZC rate curve, taking advantage of the deeper insights provided by the covariance matrices.

Fitting curves

While MSE analysis allows for the comparison of different models, visualizations enable the observation of the fitting behavior. Figure 4.3.8 shows the fitting on the test dataset for the indirect DC with ZC rates covariances, including a zoomed-in view on a randomly selected interval, and Figure 4.3.7 shows the fitting for the correlations. Finally, Figure 4.3.9 shows the fitting for the direct DC with ZC rates.

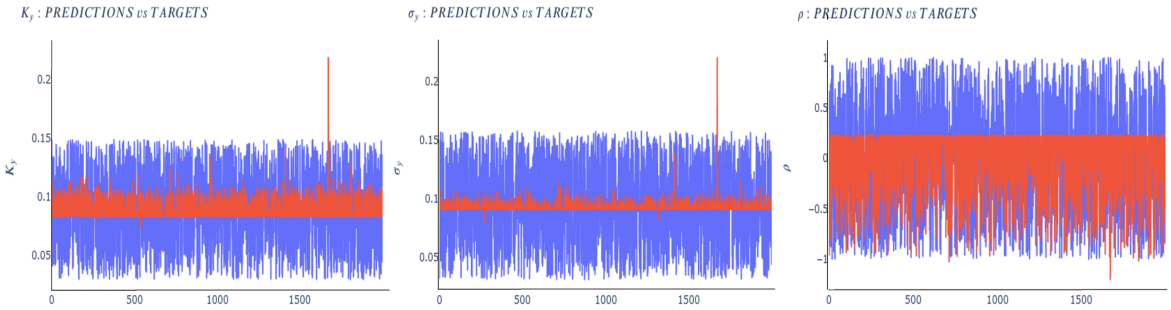

Figure 4.3.7: Indirect DC on FWD rates correlations: fitting curve on K_y, σ_y, and ρ. The red curves are predictions and the blue curves actual values.

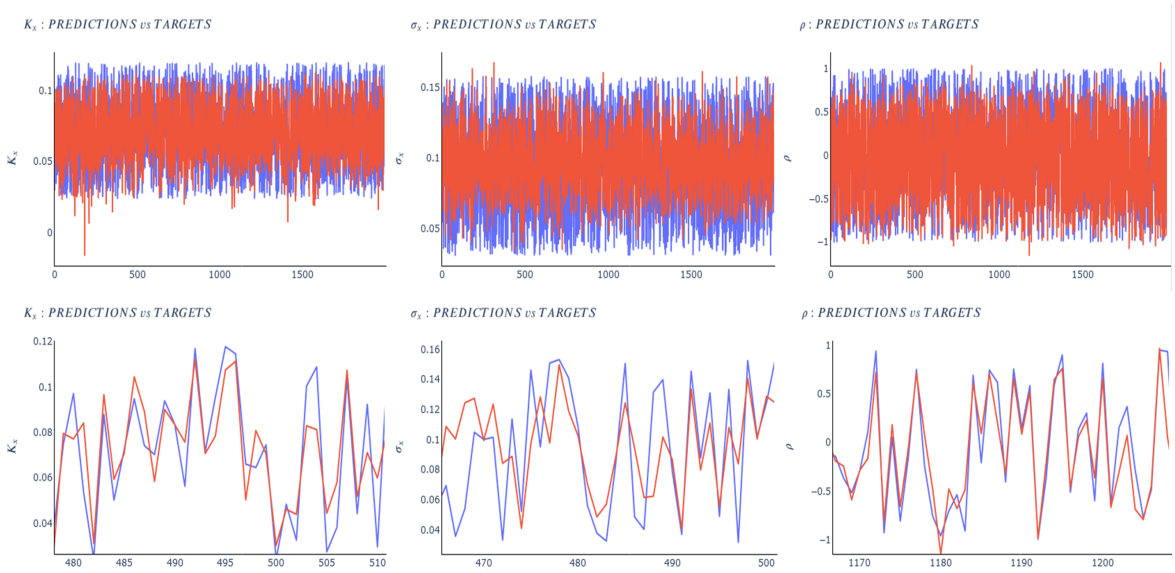

Figure 4.3.8: Indirect DC on ZC rates covariances: fitting curve on K_x, σ_x, and ρ (original and zoomed-in). The red curves are predictions and the blue curves actual values.

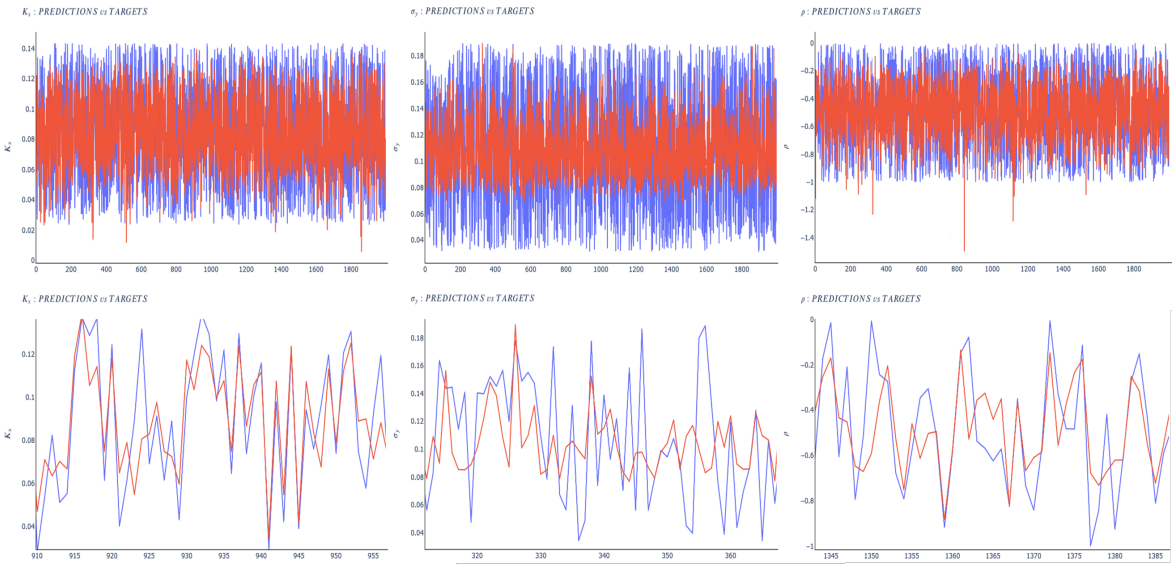

Figure 4.3.9: Direct DC on ZC rates: fitting curve on K_x, σ_x, and ρ (original and zoomed-in). The red curves are predictions and the blue curves actual values.

These visualizations further support the fact that covariances are better than correlations for indirect DC and that direct DC's performance is close to that of indirect DC. It is worth noting that, from a practical point of view, using direct DC, which works on direct market observations (unlike covariances and correlations), is easier to use and has a reduced computational cost since no further transformation is needed.

4.4 Deep Calibration vs Classic Calibration

4.4.1 Comparison Methodology

It is not a trivial task to perform a comparison between our DC models and classic optimization methods for calibration. Assume that we want to compare a model using swaptions to perform a calibration with our DC model (regardless of whether the DC is indirect or direct). We would obtain one set of parameters from those swaptions and another set of parameters from our calibration using ZC or FWD rates (directly or indirectly). There is no guarantee that the two sets of parameters will be alike since we are not trying to fit the same quantities, even if the calibration were perfect in both cases. This similarly happens when choosing between swaptions or caplets to calibrate a classic model. Consequently, it is impractical to use data not related to our OQOIs. Therefore, we focus on calibrations based on ZC and FWD rates and their correlations and covariances.

A common calibration method widely used in practice consists of finding the set of parameters that minimizes the error between the theoretical covariances/correlations of ZC and FWD rates obtained from equation (4.10) and the historical ones. We choose this method as the benchmark classical calibration (CC) method to compare against our DC method. To do so, we use a synthetic set of parameters. Indeed, using real market data would make it difficult to compare the efficiency of both calibration methods since the real set of parameters targeted would be unknown. Below, we outline the different steps of our comparison process between DC and CC.

1. First, we choose N_{sets} sets of parameters $\left\{k_x^i, k_y^i, \sigma_x^i, \sigma_y^i, \rho^i\right\}_{i\in[|1,N_{sets}|]}$ for the comparison.
2. We compute ZCs with equation (4.4) and add white noise and jumps to replicate real market data behavior, including stress periods.
3. Next, we compute (numerically) the covariances and correlations from the obtained ZC rates, which will only be useful for the CC.
4. We proceed with the direct DC and CC.
5. Finally, we compare the two calibrations.

To measure how sensitive both methods are to the noise added in step two above and how they perform in a non-stressed environment, we will also make the comparison with the data without the noise. It is worth noting that in this process, we chose to use the direct DC instead of the indirect DC for two main reasons:

- We have already compared the direct DC and the indirect DC in the previous section. The latter showed very similar calibration performances, and as we will see in Subsection 4.4.2, the direct DC performs better than the CC.
- As explained in Section 4.2.2.1, the direct DC is clearly advantageous not only in terms of computational cost compared to the indirect DC but also in terms of practical use. Hence, the choice of performing the comparison using the direct DC, which would be the natural first choice for any practical user.

Step 1: Choice of the set of parameters
To make a fair comparison, the parameters must be outside the training dataset and ideally even the test set. We keep the same range of parameters defined in Table 4.2.2 and uniformly draw N_{sets} sets of parameters. Setting $N_{sets} = 50$ would be a reasonable choice, especially from a computational point of view, and would make sense in a real market situation. It could correspond to a scenario where a market risk unit needs to perform G2++ calibrations for ten currencies in five different stress scenarios.

Step 2: Adding noise to ZC rate curves to mimic real market stressed data
We compute the expectation of ZC rates (see equation (4.12)) at times $t_i = i\Delta$, $i \in [|1, nb_{steps}|]$ using the results in Subsection 4.2.2.2. To approach real-world data, we decide to add two types of noise. First, we add jumps, and then we add Gaussian white noise. For a given tenor T, we denote $\tilde{Z}_i$ as the ZC rate at time t_i with added jumps and $\hat{Z}_i$ as the ZC rate with both jumps and Gaussian noise. Assuming positive ZC rates, the process is as follows:

- Following a Bernoulli distribution with parameter p, we decide whether a jump will occur at time t_i (we take $p = 2\%$). If a jump occurs, it will have a magnitude α_i, which is a uniform random variable in $\left[\frac{1}{a},\ a\right]$.
- If a jump occurs at time t_i, there will be no jump at times t_{i+1} and t_{i+2}. Instead, we will have a correction with $\tilde{Z}_{i+1} = \beta_i Z_i$ and $\tilde{Z}_{i+2} = \gamma_i Z_i$, where β_i is uniformly drawn between α_i and 1. Similarly, γ_i is drawn from a uniform distribution between β_i and 1.

The first step is motivated by the fact that we know jumps occur in the market, but they are not *extremely* frequent. The second step allows us to better reflect market behavior, where jumps are generally not followed by an immediate return to normality but often require some time before stabilizing. The process described above, in the case of a jump, can be written as:

$$\begin{cases} \tilde{Z}_i = \alpha_i \times Z_i \text{ with } \alpha_i \sim U_{\left[\frac{1}{a},\ a\right]} \\ \tilde{Z}_{i+1} = \beta_i \times Z_i \text{ with } \beta_i \sim U_{[min(1,\ \alpha_i),\ max(1,\ \alpha_i)]} \\ \tilde{Z}_{i+2} = \gamma_i \times Z_i \text{ with } \gamma_i \sim U_{[min(1,\ \beta_i),\ max(1,\ \beta_i)]}. \end{cases} \tag{4.14}$$

We chose $a = \frac{3}{2}$. To take different tenors into consideration, one needs to use different intensities for the jumps. To do so, we define the sets $\left\{\alpha_i^0,\ \ldots,\ \alpha_i^{M-1}\right\}$, $\left\{\beta_i^0,\ \ldots,\ \beta_i^{M-1}\right\}$, and $\left\{\gamma_i^0,\ \ldots,\ \gamma_i^{M-1}\right\}$, where M defines the number of tenors (typically, based on the previous notations when describing the NN architecture for the DC in Subsection 4.2.2.3, $M = nb_{tenors}$). We then apply the previous process to the first tenor with $\alpha_i^0 = \alpha_i$, $\beta_i^0 = \beta_i$, and $\gamma_i^0 = \gamma_i$, and subsequently get the coefficients for $j \in [\![1,\ M-1]\!]$ using the following process.

$$\begin{cases} \alpha_i^j = \max(1,\ \alpha_i^0\ U_i^j) \text{ if } \alpha_i^0 >= 1 \text{ and } \alpha_i^j = \min(1,\ \alpha_i^0\ U_i^j) \text{ otherwise} \\ \beta_i^j = U_{[min(1,\ \alpha_i^j),\ max(1,\ \alpha_i^j)]} \\ \gamma_i^j = U_{[min(1,\ \beta_i^j),\ max(1,\ \beta_i^j)]}, \end{cases} \tag{4.15}$$

with U_i^j drawn for each j such that, $U_i^j \sim U_{[b,\ c]}$. We choose $b = \frac{1}{2}$ and $c = \frac{3}{2}$. This leads us to avoid having jumps of same magnitude for all tenors while keeping same signs (increasing or decreasing stress). For negative ZC rates, Z_i, at any considered time step t_i and for a given tenor, the related coefficients, α_j, β_j, or γ_j is inverted. Regarding the Gaussian noise, we will add M centered Gaussian random variables with a standard deviation proportional to the one observed on the ZC rate curve at time t_i, (i-e, considering the whole term structure) , we denote it σ_{Z_i} . For the ZC rate of tenor j at time t_i, we hence write $\hat{Z}_i^j = \tilde{Z}_i^j + N(0,\ (d\sigma_{Z_i})^2)$. We choose $d = 10\%$. Figure 4.4.1 shows for one set of parameters original and noisy ZCs curves.

We choose $b = \frac{1}{2}$ and $c = \frac{3}{2}$. This helps avoid having jumps of the same magnitude for all tenors while maintaining the same signs (increasing or decreasing stress). For negative ZC rates, Z_i, at any considered time step t_i and for a given tenor, the related coefficients, α_j, β_j, or γ_j, are inverted. Regarding the Gaussian noise, we will add M centered Gaussian random variables with a standard deviation proportional to that observed on the ZC rate curve at time t_i (i.e., considering the whole term structure). We denote it σ_{Z_i}. For the ZC rate of tenor j at time t_i, we thus write $\hat{Z}_i^j = \tilde{Z}_i^j + N(0,\ (d\sigma_{Z_i})^2)$. We choose $d = 10\%$. Figure 4.

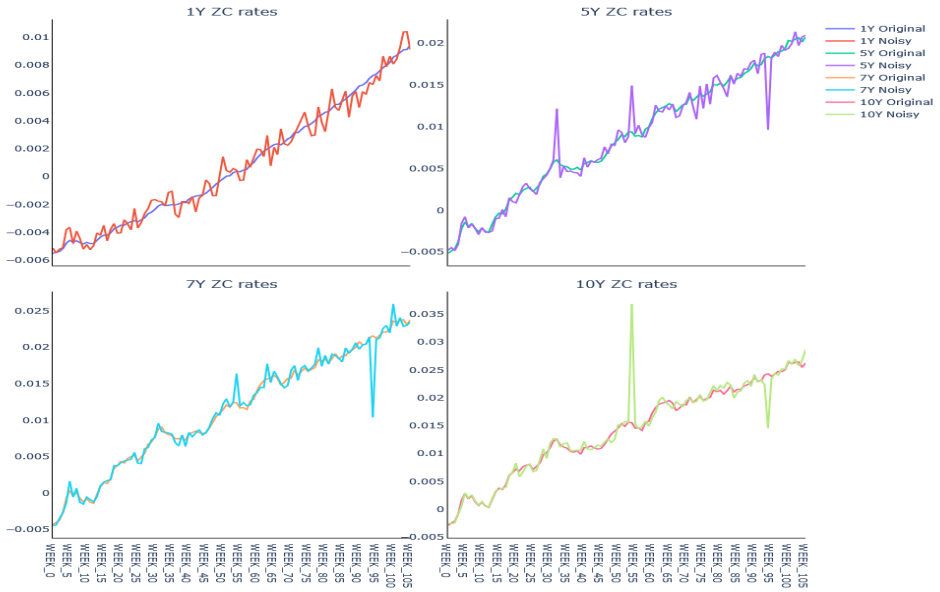

Figure 4.4.1: Stressed and not stressed ZC rate curves. $K_x = 0.09$, $K_y = 0.08$, $\sigma_x = 0.13$, $\sigma_y = 0.06$ and $\rho = -0.96$.

Step 3: Numerical computation of COVs and CORs
As Figure 4.4.2 shows (on covariances), the noise added to ZC rates has an impact on correlations and covariances as they also become less smooth.

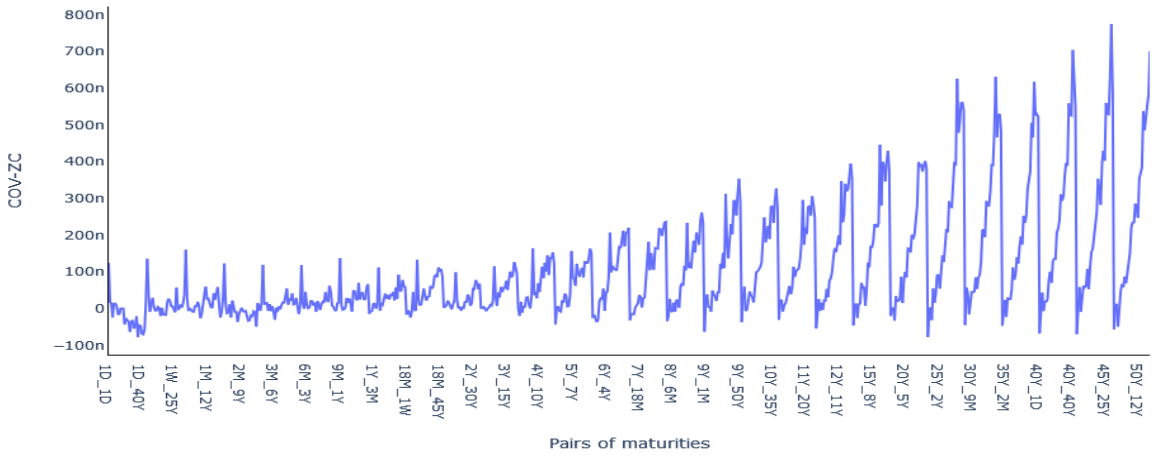

Figure 4.4.2: Covariance curve of ZC rates after add of noise. $K_x = 0.07$, $K_y = 0.07$, $\sigma_x = 0.03$, $\sigma_y = 0.03$ and $\rho = -0.42$.

Step 4: CC and DC
For CC, the purpose is to find the set of parameters that minimizes the error between real correlations and covariances and their estimates via equation (4.8). We proceed with the following minimization on each of our $N_{sets} = 50$ sets of parameters. Let us denote $\theta_i = (K_x^i, K_y^i, \sigma_x^i, \sigma_y^i, \rho^i)$, where $K_x^i, K_y^i, \sigma_x^i, \sigma_y^i > 0$, and $\rho^i \in [-1, 1]$, with $i \in [|1, N_{sets}|]$. Then,

$$\theta_i^* = \underset{\theta_i}{argmin} \frac{1}{M} \sum_{i,j=1}^{M} \left[W_{cov} \left| \frac{\text{Cov}(T_i, T_j;\, \theta_i) - \text{Cov}(T_i, T_j)^{noisy}}{\text{Cov}(T_i, T_j)^{noisy}} \right| + W_{cor} \left| \frac{\text{Cor}(T_i, T_j;\, \theta_i) - \text{Cor}(T_i, T_j)^{noisy}}{\text{Cor}(T_i, T_j)^{noisy}} \right| \right],$$

with, W_{cov} and W_{cor} verify $W_{cov} + W_{cor} = 1$. The optimization is performed using SciPy's (Virtanen et al., 2020) *optimize* package under constraints[3]. Except for internal reports, we have not been able to find any paper in the literature dealing with this calibration procedure.

Meanwhile, for the DC, we use our already trained direct DC model, which is applied to the fifty sets of parameters. To simulate a real-life situation, the NN outputs are capped

[3]Many sets of tenors have been tested for the calibration, and it turned out that selecting a set of medium tenors from 3Y to 12Y led to better results for the CC. Additionally, using $W_{\text{cov}} = 3/4$ and $W_{\text{cor}} = 1/4$ gave better performance. For the initialization of the optimization process, we uniformly draw our initial parameters from the range defined in Table 4.2.2.

or floored when needed (e.g., if the model outputs a correlation $\rho > 1$, it will be capped at 1; for other parameters, we set floors at 10^{-8} if negative values are outputted). The DL model used (a classic CNN) is not expected to learn the constraints of the financial model. The caps and floors used are the same as those in the constrained optimization process of the CC.

4.4.2 Results' comparison

Computational performances

Once trained on our wide range of parameters (see Table 4.2.2), the time needed for calibration is remarkably reduced by the DC. The fifty CCs were completed in 480.1 seconds, while the DC ran in less than a second—specifically, in 0.063 seconds. This drastic reduction in processing time easily justifies the time spent beforehand to train the NN. In fact, the strong time efficiency of the DC lies in the fact that once the neural network is trained, it operates as a deterministic function, directly yielding parameters from the OQOI.

Calibration accuracy

We measure the error in three different ways.

- **The global error on each parameter:** We use a normalized root mean squared error (NRMSE) for each parameter. Hence, for each parameter $i \in [\![1, np]\!]$, where $np = 5$, we write:

$$\text{NRMSE}^{\text{i}} = \frac{\text{RMSE}^{\text{i}}}{\max(\theta^i) - \min(\theta^i)}, \text{ with } \text{RMSE}^{\text{i}} = \sqrt{\frac{1}{N_{sets}} \sum_{j=1}^{N_{sets}} (\theta_j^i - \hat{\theta}_j^i)^2}. \quad (4.16)$$

.

- **The overall error:** This measures the performance of calibrating the five parameters across the fifty sets (see step 1 of Subsection 4.4.1):

$$\text{Overall Error} = \sqrt{\sum_{i=1}^{np} (\text{NRMSE}^{\text{i}})^2}. \quad (4.17)$$

- **The local error:** This measures the global error for each of the $N_{sets} = 50$ sets of parameters. For set $j \in [|1, N_{sets}|]$, we then write:

$$\text{Local Error}_j = \sqrt{\sum_{i=1}^{np} \left(\frac{\theta_j^i - \hat{\theta}_j^i}{\max(\theta^i) - \min(\theta^i)} \right)^2}. \quad (4.18)$$

The global errors on each parameter and the overall error are presented in Table 4.4.1 and Figure 4.4.3 below:

Errors	K_x	K_y	σ_x	σ_y	ρ	Overall Error
CC	1.536	0.862	0.291	0.254	0.138	1.809
DC	0.0088	0.337	0.256	0.113	0.110	0.460

Table 4.4.1: Global error by parameter for CC and DC, and overall errors

We can see that the DC outperforms or is equivalent to the CC for each of the five parameters, even though the comparison parameters are outside the training sets.

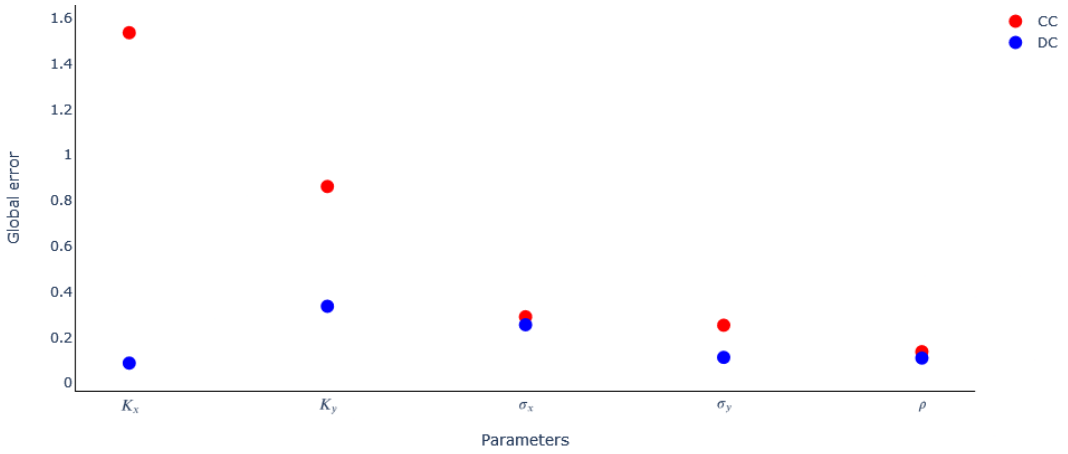

Figure 4.4.3: Global error by parameter for CC and DC

Finally, Figure 4.4.4 shows the distribution of local errors for each of the fifty sets. It highlights how the DC outperforms the CC in terms of global error. The DC method does not produce as many high errors as the CC.

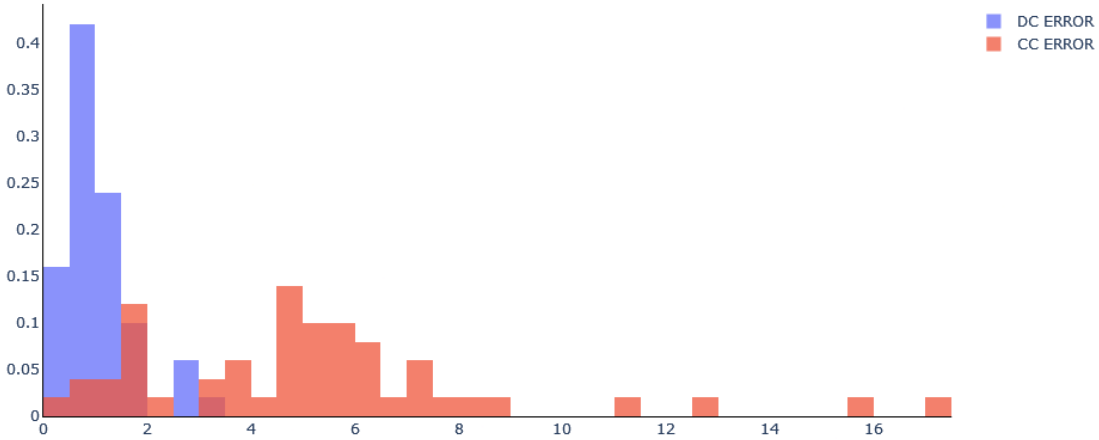

Figure 4.4.4: Histogram of local errors for each of the N_{sets} sets

The previous comparisons between DC and CC were performed on a noisy synthetic data set, representative of a stressed market context. We also provide the same comparisons in a non-stressed environment, using expectations of ZC rates defined in equation (4.12) and the related covariances and correlations. As expected, both models achieve lower errors since no noise is added. In this context, the DC still outperforms the CC. The results are presented in Table 4.4.2 below. It is also interesting to note that the DC's overall error does not change much from the noisy sets, indicating that the DC calibration is more resilient to noise.

Errors	K_x	K_y	σ_x	σ_y	ρ	Overall Error
CC	1.288	0.765	0.290	0.257	0.135	1.553
DC	0.099	0.330	0.243	0.104	0.106	0.447

Table 4.4.2: Global error by parameter for CC and DC, and overall errors in the noise-free environment

4.5 Beyond interest rate models calibration: CIR intensity calibration for credit risk problems

Motivated by the fact that our DC approach is designed to be systematic, meaning that it can be applied to any model as long as the conditions outlined in Section 4.2 are met, we now apply it to a different model (the CIR intensity) and in a different context (credit risk problems). More precisely, the default intensity (DI), which can be stripped from CDS prices in a standard way (see, e.g., Chapter 22 of Brigo and Mercurio (2006)), is now used for another application of our indirect DC process.

4.5.1 DIs as OQOI

Let us consider the CIR intensity model described by the following equation:

$$d\lambda(t) = (a - b\lambda(t))\, dt + \sigma\sqrt{\lambda(t)}\, dW_t, \tag{4.19}$$

with $2a > \sigma^2$, $b > 0$, and $\lambda(0) = \lambda_0 > 0$ to ensure strictly positive DIs. Here, $(W_t)_{t\geq 0}$ denotes a Brownian motion, and $\lambda(t)$ denotes the DI at time t (for more details, see, e.g., Alfonsi et al. (2015b)). In what follows, we use the indirect DC calibration procedure introduced in Section 4.2.2. Only the OQOI changes from COV / COR-ZC to DIs. For this new application of our work, we calibrate only the drift parameters (a and b). It is

worth noting that maximum likelihood estimators can be built for these drift parameters (see, e.g., Ben Alaya and Kebaier (2012b, 2013)).

4.5.2 Data set construction

Once again, the process remains the same as the one described in Subsection 4.2.1.2 as we still use synthetic data sets, except for the third step, where we apply the drift implicit discretization scheme to the stochastic differential equation. More precisely, we consider a uniform grid $t_i = iT/n,\ n \in \mathbb{N}^*$, and for $4a > \sigma^2$ and $T/n < 2b$, the drift implicit scheme introduced by (Alfonsi, 2005) is given by:

$$\lambda_{t_{i+1}} = \left(\frac{\frac{\sigma}{2}\left(W_{t_{i+1}} - W_{t_i}\right) + \sqrt{\lambda_{t_i}} + \sqrt{\left(\frac{\sigma}{2}\left(W_{t_{i+1}} - W_{t_i}\right) + \sqrt{\lambda_{t_i}}\right)^2 + 4\left(1 + \frac{bT}{2n}\right)\frac{a-\sigma^2/4}{2}\frac{T}{n}}}{2\left(1 + \frac{bT}{2n}\right)} \right)^2 . \tag{4.20}$$

Note that, we could have also used the actual distribution of a solution of the CIR model (see for instance, Cox et al. (1985a)).

We define for each of the parameters a range of 3 equidistant values: from 0 to 5 for b, 0% to 5% for a, and 0.1% to 3% for σ. We then take all combinations of values verifying $2a > \sigma^2$, which, since $a > 0$, ensures $4a > \sigma^2$. Using equation (4.20), we compute a thousand paths of DIs for each set of parameters. We keep the same time range as previously: 105 steps of size 1 week, which is about 2 years. The value of λ_0 is set to 0.5%. Similarly to ZC rates, DI simulations lead to many random paths. We can either consider one typical path that would be representative of all the others (as we do in Subsection 4.2.2.1 by taking the expectation of ZCs) or consider the entire set of simulated paths. In what follows, we use the latter option to illustrate this alternative approach, as in Subsection 4.2.2.1 we applied the first solution. This approach leads to a dataset of 18,000 entries with 107 features (106 time steps and σ). Applying an 80%-20% split results in training and test sets of 14,400 and 3,600 entries, respectively.

4.5.3 The Neural Network Architecture

The architecture is similar to the indirect DC's; we have a shallow FCN with only 3 layers in this case. The input layer has 106 neurons (nf, the number of features), the hidden layer has 1,500 neurons (h), and the output layer has 2 neurons (np, for the 2 parameters a and b to calibrate). The activation functions are ReLUs, except for the prediction layer, which does not have an activation function. The Adam optimizer is used to minimize the MSE loss with a learning rate of 0.001. The training is performed over 1,000 epochs with mini-batches of size 256. The architecture can be represented as follows:

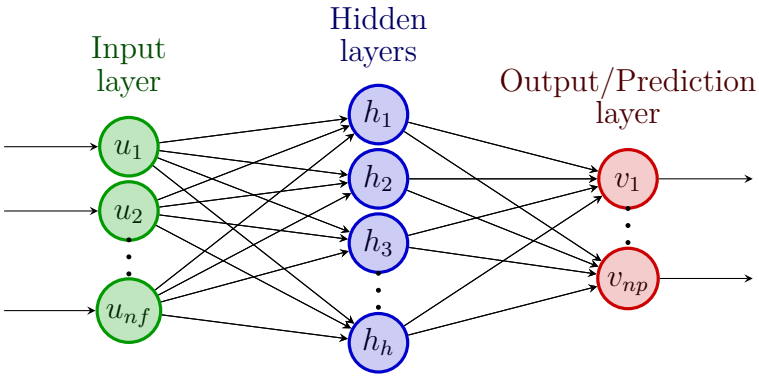

Figure 4.5.1: FCN Architecture of the indirect DC for the CIR intensity model

4.5.4 Calibration Results

Both accuracy and time efficiency are good, as calibration on the test set is performed in under a second. The MSE on a is 2×10^{-5} and 0.5776 on b. Figure 4.5.2 shows the fitting curves.

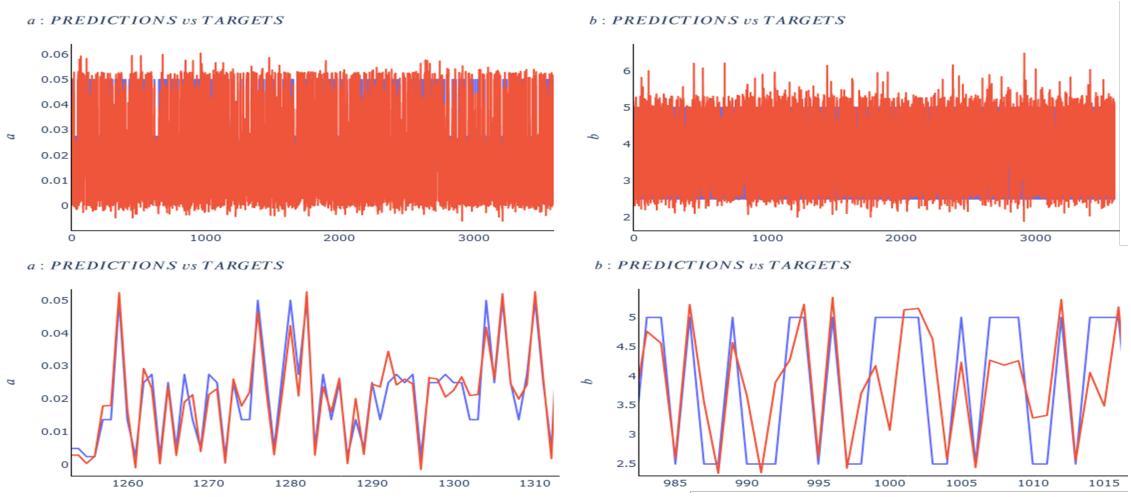

Figure 4.5.2: Indirect DC on the CIR intensity model: Fitting Curves with and without zoom on parameters a and b. The red curves are predictions and the blue curves actual values.

4.6 Conclusion

Despite the growing use of deep learning and related artificial intelligence techniques in finance, analytical models, such as interest rate models like the G2++, remain commonly used due to the expertise that professionals and researchers have developed over the last few decades. However, this does not mean that DL and AI, in general, should not be

leveraged to improve our use of these models. In this chapter, we have demonstrated that DL can be effectively used to calibrate IR and intensity models. Starting with parameters from a classic market calibration of the G2++ model, we generated a synthetic dataset of parameters by uniformly sampling around those reference parameters. Using results from the literature and our reference parameters, we generated two types of datasets. The first, for the so-called indirect DC, consists of correlations and covariances of ZC and FWD rates. The second set serves the direct DC and consists of ZC and FWD rate curves. For the first model, we used a shallow fully connected network and for the second, a shallow convolutional neural network.

We have shown that covariances provided more information to the neural networks, and, as a consequence, the indirect DC using correlations made about double the errors of the covariance-based indirect DC. The direct DC with ZC rates, like the indirect DC with covariances, achieved good accuracy. The errors and computational performance obtained allow us to conclude that using DL leads to very fast calibration with low errors for the G2++ model. Our DC approach is designed to be systematic, meaning that it can be applied to any model with data fitting our requirements for the observable quantity of interest, i.e., it must be observable in the market and have an analytical expression. Indeed, we achieved a global error significantly lower than the classical optimization method in less than a second, compared to the 480 seconds required by the classic method. Additionally, as we intended, the models are straightforward to use and require only easily available data with minimal to no transformation.

As a next step, we could improve the results of the indirect DC by focusing on the covariances, allowing more tenors, and applying dimension reduction techniques, such as auto-encoders or Principal Component Analysis (PCA). Additionally, deepening the network could also enhance performance, but this might require more complex architectures to avoid overfitting, such as those incorporating regularization techniques. It would be of high interest to apply our pretrained NNs to other models, such as option pricing models or those with path-dependent parameters. This could be the focus of future work.

Appendices

4.A Market ZC Correlations and Correlations

We can observe ZC rates' covariance and correlation curves similar to those simulated for our synthetic data sets[4].

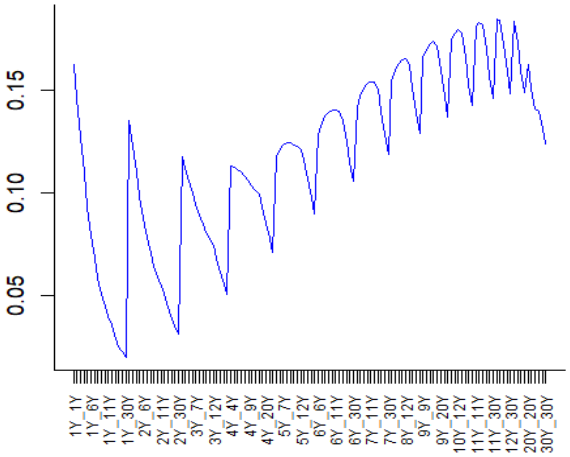

Figure 4.A.1: Covariance curve of the US Treasury ZC rate curve; the data used ranges from January 2020 to October 2021.

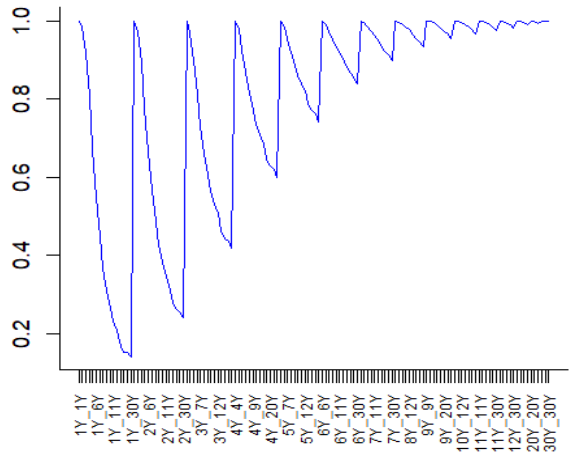

Figure 4.A.2: Correlation curve of the US Treasury ZC rate curve; the data used ranges from January 2020 to October 2021.

4.B Market ZC Curve

We also observe ZC rates' graphs similar to those simulated for our synthetic data set[5].

[4]The data used to obtain the CORs and COVs can be found at https://data.nasdaq.com/data/FED/SVENY-us-treasury-zerocoupon-yield-curve.

[5]The data can be retrieved from https://www.ecb.europa.eu/stats/financial_markets_and_interest_rates/euro_area_yield_curves/html/index.it.html.

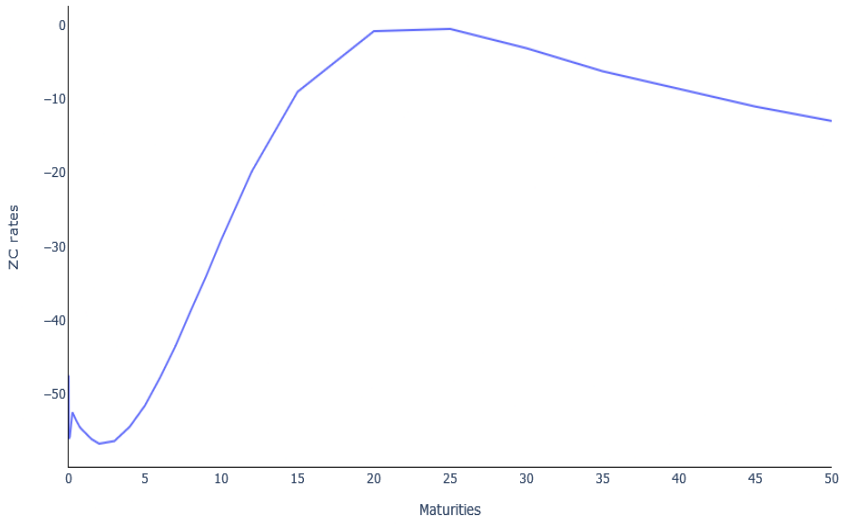

Figure 4.B.1: Initial Euro ZC rates in basis points as of 2020-11-04

4.C Unfeasible backpropagation theorem's proof

Proof of Theorem 4.3.1. Without loss of generality, we consider an unbiased NN with inputs and outputs of dimension 1 and with one hidden layer. The NN is described in Figure 4.C.1, where $u \in \mathbb{R}$ is the input, h is the result of applying the hidden layer's activation function to u, and y is the output of the NN.

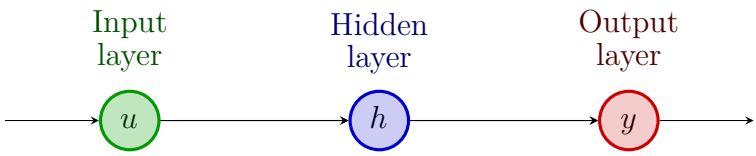

Figure 4.C.1: Simplified feed-forward neural network

Let $w = (w_0,\ w_1) \in \mathbb{R}^2$ define the weights, where w_0 is the weight on the hidden layer and w_1 is the weight on the output layer. These unbiased weights are updated between optimization steps k and $k+1$ by gradient descent: $w^{k+1} = w^k - \varepsilon \nabla_w L(w^k)$, $k \in \mathbb{N}$, where L is any smooth loss function and ε is the learning rate. The backpropagation uses the chain rule and is written as:

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial w_1}. \tag{4.21}$$

Let us focus on $\frac{\partial L}{\partial y}$. Let f be the function linking the inputs u of the neural network to the outputs y. The important step is to use the fact that our inputs are functions of the targets (and consequently of the outputs). In fact, $u = u(y)$ and by the theorem's

hypothesis, $\frac{\partial u}{\partial y} = 0$. So we have:

$$\frac{\partial L}{\partial y} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial y} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial u}\underbrace{\frac{\partial u}{\partial y}}_{=0} = 0. \tag{4.22}$$

Hence, by (4.21) $\frac{\partial L}{\partial w_1} = 0$. On the other hand, when backpropagating to w_0 as we will write, $\frac{\partial L}{\partial w_0} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial h}\frac{\partial h}{\partial w_0}$ and we have already shown $\frac{\partial L}{\partial y} = 0$, then we also deduce $\frac{\partial L}{\partial w_0} = 0$. Consequently we have $\nabla_w L(w^k) = 0_{\mathbb{R}^2}$ which implies that $w^{k+1} = w^k,\ k \in \mathbb{N}$.

These arguments can easily be generalized for more complex architectures, since the basic principle of the backpropagation which is to start from the rightmost partial derivative and propagate through, still remains. Hence, for outputs $(y_i)_{i\in[|1,\,d|]}$ of dimension $d \geq 1$, the only requirement is that all the partial derivatives of the inputs w.r.t to the outputs must be null, that is $\forall i \in [|1, d|], \frac{\partial u}{\partial y_i} = 0$.

These arguments can easily be generalized for more complex architectures, since the basic principle of backpropagation, which is to start from the rightmost partial derivative and propagate through, still remains. Hence, for outputs $(y_i)_{i\in[\![1,\,\dim]\!]}$ of dimension $\dim \geq 1$, the only requirement is that all the partial derivatives of the inputs with respect to the outputs must be zero, that is

$$\forall i \in [\![1, \dim]\!], \frac{\partial u}{\partial y_i} = 0.$$

□

Chapter 5

Towards Explainable Automated Data Quality Enhancement without Domain Knowledge

Abstract

In the era of big data, ensuring the quality of datasets has become increasingly crucial across various domains. We propose a comprehensive framework designed to automatically assess and rectify data quality issues in any given dataset, regardless of its specific content, focusing on both textual and numerical data. Our primary objective is to address three fundamental types of defects: absence, redundancy, and incoherence. At the heart of our approach lies a rigorous demand for both explainability and interpretability, ensuring that the rationale behind the identification and correction of data anomalies is transparent and understandable. To achieve this, we adopt a hybrid approach that integrates statistical methods with machine learning algorithms. Indeed, by leveraging statistical techniques alongside machine learning, we strike a balance between accuracy and explainability, enabling users to trust and comprehend the assessment process. Acknowledging the challenges associated with automating the data quality assessment process, particularly in terms of time efficiency and accuracy, we adopt a pragmatic strategy, employing resource-intensive algorithms only when necessary, while favoring simpler, more efficient solutions whenever possible. Through a practical analysis conducted on a publicly provided dataset, we illustrate the challenges that arise when trying to enhance data quality while keeping explainability. We demonstrate the effectiveness of our approach in detecting and rectifying missing values, duplicates and typographical errors as well as the challenges remaining to be addressed to achieve similar accuracy on statistical outliers and logic errors under the constraints set in our work.

5.1 Introduction

Data assumes an escalating significance in contemporary business operations. Across diverse sectors such as commerce, entertainment, medicine, and the oil industry, data emerges as a potent catalyst for augmenting business efficacy (Marr, 2016). An abundant reservoir of qualitative data, coupled with Artificial Intelligence (AI), statistical methodologies, and probabilistic approaches, facilitates the discernment of intricate behavioral patterns or associations that would be arduous for humans to discern. However, the efficacy of these methodologies depends on the availability of high-quality data, often necessitating voluminous datasets. Notably, the preparatory phase, which entails rendering data compatible with analytical algorithms, has long been estimated to consume an important part of the process of exploiting the data. Two decades prior, Pyle (1999) already estimated the proportion of time spent to be over 50% of the overall processing time. Presently, notwithstanding the advancements in analytical techniques, the increasing volume and intricacy of data sources have exacerbated the challenges associated with data preparation (García et al., 2016). This critical aspect, known as data preprocessing, encompasses a suite of techniques executed prior to the extraction of actionable insights from the data, constituting a pivotal facet of data exploitation. Due to the inherent anomalies in data, initiating data mining endeavors directly is impractical without addressing inconsistencies and redundancies. Furthermore, the exponential surge in data generation rates across multiple sectors necessitates the adoption of increasingly sophisticated analytical mechanisms (Han et al., 2022; Zaki and Meira, 2014). In light of constraints pertaining to time and human resources, there exists a pressing imperative to automate and streamline this preparatory phase. Central to this process is the identification and rectification of erroneous data values. They constitute a crucial component of the data cleansing process (refer to, for instance, Section 3.2 of Han et al. (2012)).

For the reasons outlined above, it is imperative for practitioners to possess efficient tools for measuring and enhancing data quality. This study focuses on optimal strategies for providing automated tools for this phase. We also assert that without explainability and interpretability, the effectiveness of data quality assessment algorithms is compromised, as users may lack confidence in the reliability of the results. For instance, if an entry is deemed invalid due to the detection of a statistical outlier on a specific line of a data table, but the problematic field(s) cannot be precisely identified by the algorithm, it becomes challenging to fully rely on the algorithm for making informed decisions. We propose a framework comprising five major steps that can automatically identify defects in a dataset without prior knowledge of its contents. This is accomplished while ensuring the ability to explain and justify each conclusion reached by our framework. This feature enables the framework, which consists of a series of algorithms, to provide native corrections for all errors identified in a given dataset. Returning to the earlier example, the ideal framework should not only identify the presence of a statistical outlier but also pinpoint the problematic field and propose a correction. Our framework is applicable to any table containing numerical values, text, or both. It addresses three types of defects that can affect data: absence, redundancy,

and inconsistency. The first two steps of the five-steps algorithm we introduce deal with the first two types of errors and the last three steps all handle inconsistencies as we consider statistical outliers, typographical errors and logic errors. While the first two defects may be self-explanatory, indicating respectively missing and duplicated values, the third is less straightforward. In our framework, inconsistencies can impact any type of observation, whether numerical or textual, occurring when an input in a given field appears abnormal relative to others in the same field. While Artificial Intelligence (AI)-based techniques may offer potentially superior performance, they often lack the transparency necessary for thorough understanding and validation of results. Therefore, the algorithms within our comprehensive framework attempt to combine AI-based techniques and statistical results to identify and correct defects in given datasets. This characteristic, specifically its outcome of being able to explain all results, is the main distinction between our work and existing literature.

The literature contains several algorithms for analyzing the validity of a data set. As in this work, many of them leverage Machine-Learning (ML), Deep-Learning (DL) and statistical methods. Considering specifically the handling of missing values, they do not constitute per se a tough challenge as they can be identified in pretty straightforward ways. However, in the process of handling them, two challenges arise. The first one is analyzing the rightfulness of the missing data. Indeed, a missing value does not necessarily designate a defect in the dataset as it can, in various situations, be a legitimate and information-providing entry. As developed in Sainani (2015), the most straightforward and commonly employed approach to managing missing data involves excluding incomplete observations entirely from the analysis. That solution, while very easily applicable, is drastic. As already stated, it might delete the valuable information provided by the missing value, but it obviously also discriminates all the other valid entries. The author also presents the very common solution, which is to impute the missing value with a representative value of the field. It can be the average value (for instance, if it is heights of people), it can be a *local* average, for instance the weight of the males in a specific study, and so on and so forth. Other researchers have instead used ML and DL to handle missing values. For instance, Savarimuthu and Karesiddaiah (2021) imputation method on time series uses an iterative imputation algorithm that clusters univariate time series data, taking into account the data's trend, seasonality, and cyclical patterns. Then within the cluster, the authors employ a similarity-based nearest neighbor imputation method within each cluster to fill in missing values. More recently, Kachuee et al. (2020) used a generator network to generate imputations that a discriminator network is tasked to distinguish; this method allows also to properly estimate the distribution of the targets. While methods using AI are shown by their authors to be very efficient, they hardly provide the level of explainability that we aim for. For these reasons, in section 5.4.2.2, we will only use a classic imputation method for handling missing numerical values, ensuring explainability and interpretability. However, for character entries, when addressing logic errors in Subsection 5.4.2.5, we will employ ML algorithms for imputation while maintaining transparency. Our strategy will guarantee that legitimate missing values will not be discriminated, thanks to the subsequent

application of the logic error detection algorithm. Our approach also guarantees that every choice will be explainable and interpretable. Research has also explored the combination of neural networks and statistics specifically for outlier detection. For example, Small et al. Dai et al. (2018) trained a neural network to replicate a specific field within their dataset to identify outliers. They compared the neural network's output with the actual values and employed statistical indicators similar to those discussed in Subection 5.4.2.3. The authors calculated the difference between the output and actual values of the field under consideration, constructing a tolerance interval based on the distribution of errors between the two values. If the error fell outside the tolerance interval, the actual value was deemed to have been incorrectly attributed. Therefore, their approach necessitates building and training a neural network for identifying statistical outliers, which can be costly, particularly for high-dimensional datasets. The reliability of this method relies on the neural network's performance, as the entire approach depends on its ability to accurately reproduce actual values. Another commonly explored approach in the literature is the use of DBSCAN, Density-Based Spatial Clustering of Applications with Noise (Çelik et al., 2011; Pandya et al., 2020). However, the challenge with this method lies in selecting the parameters ϵ, which defines the maximum distance between points in the same neighborhood, and $minPts$, the minimum number of points needed to form a dense region. This parameter selection issue has been extensively discussed in the literature (Karami and Johansson, 2014; Thang and Kim, 2011). Moreover, considering our goal of achieving maximum interpretability and explainability, it is essential to note that DBSCAN is designed for clustering and density-based outlier detection in multi-dimensional datasets, making it challenging to maintain explainability while identifying outliers, particularly as defined in Section 5.2.
Regarding typographical errors, they have long been a focus of researchers, particularly in text analysis. Early scholars, such as those referenced in Morris and Cherry (1975), were among the first to address this issue. As discussed in Subection 5.4.2.4, their approach, similar to ours, drew insights directly from the dataset, enabling the algorithm to adapt dynamically to the text under scrutiny. For example, they utilized statistical analysis of word trigrams within their documents. In contrast, our method, as explained later, utilizes word frequencies extracted from the table we aim to rectify. Our typographical error detection and correction algorithm incorporates machine learning techniques. This integration of AI and text analysis is well-documented; previous researchers have employed machine learning and deep learning methods to rectify typographical errors, often leveraging contextual cues or domain-specific knowledge (e.g., Huang et al. (2013b)). In our study, we opt for unsupervised machine learning approaches, particularly clustering algorithms, to avoid dependence on domain-specific knowledge and to expedite the identification of typographical errors. Furthermore, much of the existing literature in this field relies on external datasets. For instance, one step in the algorithm proposed by Bassil and Alwani (2012) involves cross-referencing each word with Google Web 1T 5-Gram extensive words' unigrams dataset. The algorithm we introduce doesn't require external information. It can however be used when desired to increase the speed of the process. The application we will provide will solely use information that is provided within the data sets.
Analyzing and correcting logic errors is not as frequently addressed in the literature as

the other types of errors we have previously discussed, particularly with the definition provided in Subsection 5.4.2.3. However, as we will see in details, identifying logic errors can be likened to recognizing anomalous behavior across an entire observation (taking into account all fields of the data table) in comparison to others. This can include tasks such as detecting credit card fraud or identifying cancer cells. One of the model families commonly employed for this purpose is rule-based algorithms, as demonstrated by Karthikeyan and Vembandasamy (2015b) in diagnosing Type II diabetes. Similarly, this is the model family we will employ for logic error detection. Nonetheless, numerous alternative solutions also exist. For instance, in Xu et al. (2018), variational auto-encoders are utilized to detect seasonal anomalies in key metrics (such as the number of views, online users, and orders) of web applications. Alternatively, Support Vector Machines (SVMs) have been explored, as demonstrated in Rezapour (2019) for credit fraud detection. A common characteristic among all these approaches, including ours, is their reliance on utilizing all available fields for a specific observation. This requirement was not necessarily present in the handling of other defects discussed in this document.

After conducting an extensive review of the current state of the art, it appears that the current works aims to address a gap in the data cleansing literature: the need for an explainable, interpretable, and automatic solution. While numerous efficient solutions are available, they do not have the dual constraints emphasized in our work. By applying our approach to a publicly available agricultural equipment sales data set, we illustrate its effectiveness, in detecting and rectifying missing values, duplicates and typographical errors while being able to keep a maximal explainability and without using domain knowledge. Similarly, we discuss the challenges remaining to be addressed to achieve similar accuracy on statistical outliers and logic errors under the same constraints.

In Section 5.2 of this document, we present fundamental definitions that underpin the selection of algorithms and strategies. These definitions primarily address the concepts of data validity and the explainability of an algorithm, and consequently, its results. Following this, in Section 5.3, we introduce the dataset on which we will apply the framework. This will allow us to provide concrete illustrations when presenting the algorithms. Then, Section 5.4 will be the core of this work as we introduce the framework. Initially, we offer a broad overview of the process, followed by a detailed exposition of each of its five steps. Subsequently, in Section 5.5, we present the results obtained after applying the framework to the dataset.

5.2 Definitions

We begin by presenting key definitions that will guide the selection of algorithms and shape the focus of the framework that we will introduce. Specifically, we will define what we will consider in this work to be valid data and we will provide a contextualised definition of explainability and interpretability. It is worth emphasizing that these definitions might not achieve unanimous agreement in the literature, as they do not always capture clear and

universally objective concepts.

In what follows, a *data set* D is defined as a finite collection of n records $\{r_1, r_2, \ldots, r_n\}$, where each record r_i is a tuple consisting of m attributes. Formally, we can represent the data set as:

$$D = \{r_1, r_2, \ldots, r_n\},$$

where $r_i = (a_{i1}, a_{i2}, \ldots, a_{im})$ for $i = 1, 2, \ldots, n$. Each attribute a_{ij} is an element of a predefined attribute domain A_j. This formalization allows us to consider a data set as a structured collection of multi-dimensional data points, where the dimensionality is determined by the number of attributes m. Note that, any element a_{ij} can be a numerical value as well as a text value.

5.2.1 Valid data

In the literature, data quality is frequently characterized by the presence or absence of defects. According to this perspective, data is considered to be of high quality if it is devoid of such defects or if their impact is minimal. This conceptualization can be seen for instance in the work of Schelter et al. (2018) and Corrales et al. (2018). We will also adhere to this approach in this chapter. It is also important to note that throughout this document, *the data* is not defined explicitly when unambiguous. It encompasses both individual observations which corresponds to a specific column of a row within a data array and objects of dimensions greater than one denoted as $p > 1$. They refer to a row or subsets thereof within a data array. Also, obviously the observations can be text as well as numerical values. Let us now define the defaults that will form the basis of the framework.

1. **Absence:**
   This refers to the absence of data or its constituent elements, which can arise during the data collection process due to technical failure, error, or insufficient information (Aydilek and Arslan, 2013). In such instances, the data is deemed invalid.
2. **Redundancy:**
   Data (where $p \geq 1$) that is replicated across multiple entries is regarded as invalid and is often denoted as duplicates in the literature.
3. **Inconsistency:**
   We define inconsistency as the lack of agreement between a data item and other observed data. Consequently, it encompasses several types of defects:
   (a) **Statistical outliers:** These are data points that are significantly and unjustifiably distant from other observations. An abnormal value may result from inconsistency or aberrant behavior during the measurement process (Barnett and Lewis, 1994; Johnson and Wichern, 2014).
   (b) **Typographical errors:** These errors encompass all text-related issues, including typing errors (e.g., "France" vs. "Frange") and formatting discrepancies (e.g., "Citroën" vs. "CitroÃ?n").
   (c) **Logical errors:** These errors, more challenging to define, denote incompatibilities

between different variables (columns) within a dataset.

Data will, therefore, be considered valid only when it exhibits none of these defects. The forthcoming framework will be crafted to detect and address these varied defaults. It will strive to rectify such anomalies where feasible, all the while prioritizing the preservation of explainability and interpretability of both the algorithms used and their results.

5.2.2 Explainability and Interpretability

The lack of explainability and Interpretability and the *black box* character it entails is one of the main arguments against a more widespread and systematic use of ML or DL techniques (Escalante et al., 2018). Therefore, one of our focal point is to uphold these principles of explainability and interpretability to the fullest extent we can realize within this framework.

To facilitate our comprehension of the concept of interpretability, let us begin with a more literary perspective. According to the *Larousse*, the verb **interpret** refers to the act of "seeking to make a text, an author intelligible, explaining them, commenting on them." In the context of statistical algorithms and machine learning, as emphasized by Doshi-Velez and Kim (2018), this intelligibility is directed towards humans. An algorithm is considered interpretable if the results it generates can be articulated in terms understandable to humans. Meanwhile, the concept of **explainability** is more commonly encountered and appears inherently more intuitive. However, the literature does not always converge on a formal definition of this concept within an algorithmic context. Beginning again with a literary perspective, *Larousse* defines explainability as "making someone understand (a question, an enigma), clarifying them by providing the necessary elements." (Keil, 2006) emphasizes that every stage of the clarification process contributes to explainability. To provide a comprehensive view, (Ras et al., 2018) defines explainability (particularly in a deep learning context) as the combination of interpretability and transparency. Transparency, in this context, refers to the degree to which an explanation renders a specific result acceptable. It is important to note that acceptance of the result does not necessarily imply that the practitioner perceives it as right or wrong, but rather, that the path leading to that specific result is clear.

Drawing from both the literary perspective and insights from the literature, it becomes evident that the concepts of interpretability and explainability play pivotal roles in making sense of algorithmic outputs, especially within the context of statistical algorithms and ML. We can now, in our context provide the following definitions.

1. **A result that can be explained and interpreted will refer to:**
   (a) A result for which we can understand the path, regardless of its correctness. That is, we must be capable of elucidating the inputs, outputs, rules, etc., through which a defect (as defined in the previous section) has been identified.
   (b) A result whose boundaries can be recognized and comprehended is imperative. That is, one must be able to understand, explain, and deliberate on the scenarios

where a result, whose path has already been understood, is meaningful or not. In addition to these two elements that respectively address explainability and interpretability, we introduce a third equally significant constraint.

(c) A result that we can clearly identify and locate. That is, within a data quality context, if a particular row is flagged as problematic, we must be able to identify which columns are deemed incorrect.

2. **An algorithm that can be explained and interpreted is:**
   (a) An algorithm whose results can be explained and interpreted.

For the sake of conciseness, we will refer to the duality of explainability and interpretability simply as explainability.

5.3 Data set for the application of the framework

In the following sections, we will introduce each of the algorithms comprising the framework outlined in Section 5.4. To provide clarity on how these algorithms operate, we propose starting by introducing the dataset. This approach will enable us to reference the dataset when necessary to illustrate an algorithm's function.

The data set used in this work is build upon the Blue Book for Bulldozers data set publicly available[1]. The data set is made available by French agency AMIES (Agency for Mathematics in Interaction with Industry and Society[2]) for their 2021 competition[3].

The database contains 100,000 observations described by 53 parameters briefly defined in the table 5.3.1. This table categorizes various attributes of auction machinery, grouping them primarily under unique identifiers, descriptive information, configuration details, and location-specific data. Key identifiers such as SalesID, MachineID, ModelID, datasource, and auctioneerID are crucial for tracking and analyzing sales transactions. Descriptive attributes like YearMade, MachineHoursCurrentMeter, and UsageBand provide insights into the machine's age, usage level, and operational status. The table also extensively details machine configurations, highlighting features such as Drive_System, Enclosure, Forks, and numerous others that specify the machine's physical and operational setup, which is critical for potential buyers to assess the machinery's capabilities and condition. Furthermore, geographical and market segmentation is addressed through variables like State and ProductGroupDesc, which help in understanding the categorization of the machinery. This structured data arrangement aids in the comprehensive analysis and valuation of the machinery at the point of sale. Table 5.3.2 below summarizes all the types of errors that we will be identifying and correcting in this chapter. Obviously, none of the information provided by Tables 5.3.2 or 5.3.1 will be used in the algorithms.

[1] https://www.kaggle.com/c/bluebook-for-bulldozers/data

[2] Agence pour les Mathématiques en Interaction avec l'Entreprise et la Société

[3] The work presented in this chapter won the first prize of the data quality 2021 competition (https://challenge-maths.sciencesconf.org/resource/page/id/1).

Variables	**Description**
SalesID	Unique identifier of a particular auction machine sale.
MachineID	Identifier of a particular machine.
ModelID	Model identifier.
datasource	Unique machine identifier.
auctioneerID	Identifier of a particular auctioneer.
YearMade	Machine manufacturing year.
MHCurrentMeter	Current machine usage in hours at time of sale.
UsageBand	Value (low, medium, high) calculated by comparing the hours of this particular sales machine with the average usage of the base model.
Saledate	Sale date.
Saleprice	Cost in USD.
fiModelDesc	Description of a unique model of machine.
fiSecondaryDesc	fiModelDesc disaggregation
fiModelSeries	fiModelDesc disaggregation
fiModelDescriptor	Disaggregation of fiModelDesc
ProductSize	Size class grouping for a product family.
ProductClassDesc	Description of the 2nd level hierarchical grouping.
State	U.S. state in which the sale took place.
ProductGroup	Identifier for first-level hierarchical grouping.
ProductGroupDesc	Description of high-level hierarchical grouping.
Drive_System	Machine setup 1.
Enclosure	Machine configuration - whether the machine has an enclosed cab or not ?
Forks	Machine configuration - lifting accessory.
Pad_Type	Machine configuration - type of tread on a tracked machine.
Ride_Control	Machine configuration - optional feature on loaders.
Stick	Machine configuration - control type.
Transmission	Machine configuration - describes the transmission type.
Turbocharged	Machine configuration - naturally aspirated or turbocharged engine.
Blade_Extension	Machine configuration - standard blade extension.
Blade_Width	Machine configuration - blade width
Enclosure_Type	Machine configuration - does the machine have an enclosed cab or not?
Engine_Horsepower	Machine configuration - motor power.
Hydraulics	Machine configuration - type of hydraulics.
Pushblock	Machine configuration - option.
Ripper	Machine configuration - tool attached to the machine to work the ground.
Scarifier	Machine configuration - tool attached to the machine for soil conditioning

Tip_control	Machine configuration - type of blade control.
Tire_Size	Machine configuration - primary tire size.
Coupler	Machine configuration - type of machine interface.
Coupler_System	Machine configuration - machine interface type.
Grouser_Tracks	Machine configuration - describes ground contact interface.
Hydraulics_Flow	Machine configuration - describes the ground contact interface.
Track_Type	Machine configuration - type of tread on a tracked machine.
Undercarriage_Pad_W	Machine configuration - width of track treads.
Stick_Length	Machine configuration - length of machine digging tool.
Thumb	Machine configuration - accessory used for input.
Pattern_Changer	Machine configuration - operator control configuration can be adapted to the user.
Grouser_Type	Machine configuration - type of tread on a tracked machine.
Backhoe_Mounting	Machine configuration - optional interface used to add an excavator accessory.
Blade_Type	Machine configuration - describes the blade type.
Travel_Controls	Machine configuration - describes the operator control configuration.
Differential_Type	Machine configuration - differential type.
Steering_Controls	Machine configuration - describes the operator control configuration.

Table 5.3.1: Fields in the data set

The data set is altered with different types of errors.

Correspondence	**Types of error**	**Number of cases**	**Subtotal**
Redundancy	Duplicates	20	20
Absence	Missing Value	161	161
Inconsistency (Outliers)	Aberrant Value	200	200
Inconsistency (Typography)	Entry Error	100	200
	Uppercase	50	
	Lowercase	50	
Inconsistency (Logic)	Wrong Category	25	450
	Incoherent Machine	100	
	Incoherent Drive System	200	
	Incoherent Product Group Description	100	
	YearMade > saledate	25	
Total			1031

Table 5.3.2: Summary of error types and correspondence

Now that the dataset is properly introduced, we can proceed with presenting the algorithms within the data quality measurement and enhancement framework.

5.4 Automatic data quality enhancement framework

The framework illustrated in Figure 5.4.1 consists of two phases: one preceding the commencement of actual quality enhancement (Pre-Quality Enhancement, PQE phase), and the other constituting the actual data quality enhancement process (Quality Enhancement Phase, QE phase). Initially, the PQEP involves identifying a primary key in the dataset (PQE1) and determining the columns that will receive each of the required treatments (PQE2). These two steps are fundamental in enabling an automated process that does not rely on domain knowledge. Following this preparation, the QEP targets the three key areas that were mentioned earlier:

- **Redundancies handling** encompasses the identification and elimination of duplicate entries (QE1). As depicted in the flowchart, this step necessitates the utilization of the primary key identified in the PQE phase.
- **Absences Handling** concentrates on the imputation of abnormal missing values to preserve data integrity (QE12). This step is divided into two actions: firstly, the identification of abnormal missing values, and secondly, their imputation. Indeed, as explained in Subsection 5.4.2.2, the process of identifying missing values and their imputation is not simultaneous. Similar to the preceding step, handling absences will also leverage the key identified during step A1 of the PQE phase.
- **Inconsistencies Handling** is divided into three sub-steps: the identification and imputation of statistical outliers (QE31), the detection and correction of typographical errors (QE32), and the identification and correction of logic errors (QE33). None of these sub-steps necessitates splitting the actions, as for these tasks, the processes of identifying the problem and resolving it occur simultaneously. These steps do not require the identification of a primary key. They only benefit from the PQE phase by targeting the right columns to deal with.

These steps collectively enhance data quality by systematically removing errors and inconsistencies, thereby preparing the dataset for accurate and reliable analysis.

It is noteworthy that in Figure 5.4.1, there is an arrow linking logic errors to the imputation of missing values. This is because, as we will see in Sections 5.4.2.2 and 5.4.2.5, some specific types of missing values will be handled as logic errors.

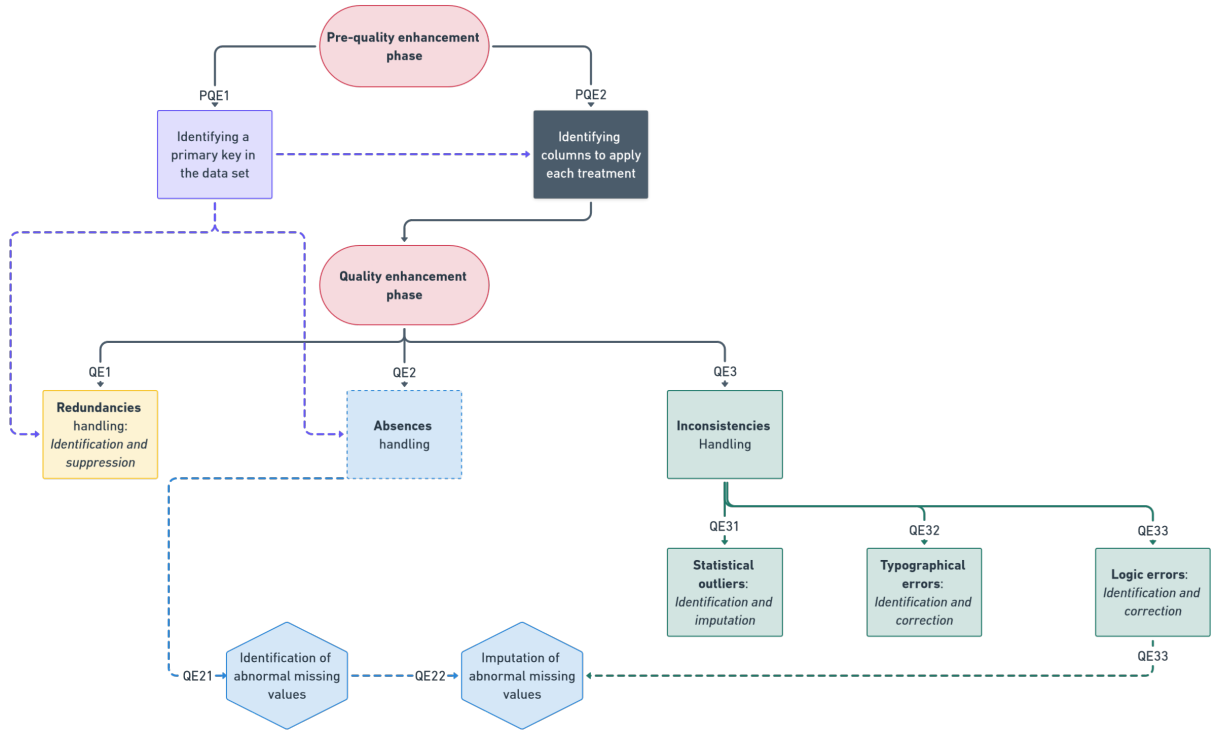

Figure 5.4.1: Invalid data detection framework

5.4.1 Pre-quality enhancement phase

5.4.1.1 Identification of the primary key in the data set

Given a data set D consisting of n records $\{r_1, r_2, \ldots, r_n\}$, where each record r_i is a tuple $(a_{i1}, a_{i2}, \ldots, a_{im})$ of m attributes, a *primary key* is a subset of the set of attributes (columns) $\{A_j\}_{j\in[\![1,m]\!]}$ that uniquely identifies each record in the data set. Let $P = \{p_1, p_2, \ldots, p_k\} \subseteq \{A_1, A_2, \ldots, A_m\}$ be the set of k fields corresponding to the primary key attributes and $P_{ind} \subseteq \{1, 2, \ldots, m\}$ the respective indices of each element of P. The primary key P must satisfy the following two properties:

1. **Uniqueness**
   For any two distinct tuples (rows) r_i and r_j in the table where $i \neq j$, their projections on the primary key attributes must be different, it thus, must hold that

$$r_i^P \neq r_j^P \quad \forall\, r_i, r_j \in D \text{ with } i \neq j,$$

   Here, r^P denotes the projection of tuple r onto the subset of attributes P which is

$$r_i^P = (a_{ij})_{j\in P_{ind}}.$$

   This ensures that no two rows can have the same value for P, providing a unique

identifier for each record.

2. **Minimality**
   The set P must be minimal with respect to the uniqueness property. This means that no proper subset of P satisfies the uniqueness condition. If any attribute is removed from P, the remaining attributes no longer uniquely identify every row in the table. It must therefore hold that if $Q \subset P$ and $Q \neq \emptyset$, then there exist at least two records r_i and r_j in D such that:
   $$r_i^Q = r_j^Q,$$
   where r_i^Q denotes the projection of record r_i onto the subset Q.

Primary keys in a database play a crucial role in ensuring data integrity and efficient data management. They are fundamental because they uniquely identify each record within a table, which is essential for maintaining the uniqueness and accuracy of the data stored. This unique identification facilitates efficient data retrieval, management, and manipulation processes, ensuring that each record can be accessed, updated, or related to other data in the database without ambiguity (Köhler et al., 2015; Le et al., 2012; Zhou et al., 2014). They will be pivotal to finding duplicates and will shorten processing times when finding missing values.

The approach we introduce combines two strategies of keys identification. Pattern recognition and uniqueness analysis. The procedure is summarized as follow:

1. Theoretically, the set of attributes that constitute a primary key should never have missing values, particularly if the data is stored in a database software. However, since this framework is designed to accommodate any dataset, such an assumption cannot be made. Nevertheless, we can anticipate primary key candidates to exhibit only a few missing values. Therefore, we identify all fields with less than 5% percent missing values.
2. We then identify each column name that could potentially designate a primary key, as they typically contain terms such as *ID*, *CODE*, or *KEY*. From there, two possibilities arise:
   (a) **Pattern recognition**
   We begin by attempting a quick-win solution. If this process has resulted in only one field remaining, then we have a good candidate for being the primary key. We conduct a set of sanity checks to ensure that the choice is justified. We verify that we do not have more than 5% of duplicates (see below). Additionally, we ensure that the field is not a numerical valued field. While theoretically, a numerical valued field could serve as a primary key, it is rather unlikely due to format changes, uniqueness issues, etc. If all the sanity checks are successfully passed, we have identified our unique primary key.
   (b) **Uniqueness analysis**
   If not, we proceed with candidates obtained from step 1 and conduct a uniqueness analysis for all combinations of variables, ranging from the smallest to the largest

set. Similar to the fact that we cannot assume the fields comprising the primary keys will be devoid of missing values, we cannot assume they will lack duplicates. However, we can anticipate that only a few duplicates will occur. Again, we limit them to 5%. Therefore, as soon as a combination of candidate attributes, with its number of duplicates falling below the threshold, is identified, it is considered as a primary key for the entire dataset. As fields labeled with the expressions *ID*, *CODE*, or *KEY* are still likely candidates to form sets of primary keys, we initiate the process with them.

We can now analyze the complexity of this algorithm. It is clear that if we are in a case where the pattern recognition was not conclusive (step 2.(a)), the complexity is dominated by the uniqueness analysis step. Also, it is clear that the advantages obtained from the identification of the primary key, in terms of structuring the data set and accelerating other processes, might be outweighed by the resources deployed to obtain said primary key if that process is not controlled. For that reason, it is more computationally reasonable to limit the analysis of combinations analyzed for uniqueness. For instance, when considering only pairs, the worst-case complexity is $O(m^2 n \log n)$, where n is the number of records and m is the number of attributes. Meanwhile, if the pattern recognition is successful, the time complexity is $O(mn)$. In both cases, the space complexity is $O(mn)$, primarily due to the storage requirements for intermediate results during uniqueness checks. In practice, however, as we will see in section 5.5.1.1, the worst-case scenario is not reached as we start with the fields identified in the pattern recognition step. Indeed, if those fields can constitute the primary key, let q be their number, then the complexity would be $O(q^2 n \log n)$.

- **Time complexity analysis**
  - The initial step of identifying fields with less than 5% missing values involves iterating through each field (column) and checking for missing values across all rows. This process results in a time complexity of $O(nm)$, where n represents the number of rows and m represents the number of columns. This step is essential to filter out columns with excessive missing data, ensuring that potential primary key candidates are viable.
  - Next, the process of identifying columns that could potentially designate a primary key by checking for specific terms such as "ID," "CODE," or "KEY" in their names has a time complexity of $O(m)$. This is because each column name is examined individually. This step narrows down the list of potential primary keys based on naming conventions, serving as a quick heuristic check. The pattern recognition step offers a quick-win solution where, if only one field remains, sanity checks are conducted to verify its suitability as a primary key. However, if the quick-win solution is not applicable, the algorithm proceeds to the more intensive uniqueness analysis.
  - The uniqueness analysis step is the most computationally expensive part of the algorithm. It involves examining combinations of candidate attributes to ensure their uniqueness. If we consider combinations of size s, the number of

such combinations is given by $\binom{m}{s}$, which is $O(m^s)$. The uniqueness check for each combination can be performed by sorting records, which has a $O(n \log n)$ time complexity. Thus, the overall time complexity is $O(m^s) \times O(n \log n) = O(m^s n \log n)$. In the case of pairs and focusing on candidate fields, the time complexity is $O(q^2 n \log n)$.

- **Space complexity analysis**
  - The space complexity for identifying fields with less than 5% missing values is $O(m)$, as it requires storing a list of columns that meet the criteria. This step ensures that only relevant columns are considered for primary key candidacy without consuming excessive space.
  - For identifying columns based on specific terms in their names, the space complexity remains $O(m)$. The list of potential primary key columns is stored, which is proportional to the number of columns in the dataset.
  - The uniqueness analysis step has a space complexity of $O(mn)$. This is because the algorithm needs to store combinations of columns and intermediate results for each uniqueness check. The space required for these combinations and results can grow significantly with an increasing number of columns, making this step the most space-intensive part of the algorithm.

5.4.1.2 Mapping processes to specific data fields

As our work aims to enhance and measure data quality without relying on domain knowledge or knowledge of the current table's content, it is crucial to be able to precisely target each analysis and correction to maintain reasonable computer resource demands. A similar approach is taken by Schelter et al. (2018). While the authors do not specifically aim to make their work domain-knowledge-free, their software can propose a set of constraints to verify for each column in the absence of user-provided constraints. This resembles our problem, but according to the authors, their constraint suggestion process is designed to involve human intervention. While the automated process we introduce may be time-consuming, it ultimately saves time and computational resources when considering the entire process. The defined rules for each type of error considered in this work are as follows:

- **Redundancies**
  Only the subset of attributes comprising the primary key will be analyzed for duplicates. This is one of the justifications for why it is important to begin by identifying it.
- **Absences**
  All fields will be analyzed for missing values, as they can affect any field. However, as we will see in Subsection 5.4.2.2, not all fields' missing values will be imputed.
- **Statistical outliers**
  Only fields with numerical values will be considered. We also assert that without sufficient information on the field, accurately apprehending its statistical properties is challenging. Hence, we will not attempt to detect statistical outliers unless more than half of the values are available.

- **Typographical errors**
  The algorithm introduced in Subsection 5.4.2.4 functions for both real words and non-words. However, it is not intended for application to entries containing both numerical and text. For instance, the field *fiBaseModel* includes values like *ZX160*. Without domain knowledge, it is at best very difficult to ascertain whether another entry, such as *ZX161*, is a typographical error or a legitimate model. For this treatment, we also exclude fields with less than half of the values available. Indeed, we argue again that with more than half of the data missing, any analysis or insights derived from this field are already significantly compromised. The presence of typographical errors in a minority of the data is less likely to have a substantial impact on the overall analysis compared to the large proportion of missing values.
- **Logic errors**
  These fields should contain strings, have less than 75% of their values missing, and be represented by at least five different observations. Allowing up to 75% missing values in these columns considers the unique nature of logic errors—errors that might not necessarily be about the absence of data but about data being present when it shouldn't be, or missing when it should be present. Therefore, when analyzing logic errors, it is not conservative enough to disqualify fields solely because they do not have enough data. Moreover, the requirement for at least five different observations ensures a diversity of data entries that can help identify inconsistencies and errors that a smaller number of observations might miss. By setting these parameters, we aim to enhance the reliability and validity of the corrections, focusing on fields where the scope for logic errors is significant and where corrections can substantially improve data quality.

5.4.2 Quality enhancement phase

5.4.2.1 Redundancies handling

Methodology

Given our data set D consisting of n records $\{r_1, r_2, \ldots, r_n\}$, where each record r_i is a tuple $(a_{i1}, a_{i2}, \ldots, a_{im})$ of m attributes, we aim to identify duplicates based on the primary key attributes. We recall that in Subsection 5.4.1.1 we defined $P \subseteq \{A_1, A_2, \ldots, A_k\}$, the set of attributes defining the primary key. The primary key P must satisfy the properties of uniqueness and minimality, ensuring that each record in D is uniquely identifiable by P. Two records r_i and r_j are considered duplicates if their projections on the primary key attributes are identical, i.e., $r_i^P = r_j^P$. Therefore, the set of duplicates $\mathcal{DUP}$ is defined as:

$$\mathcal{DUP} = \{(r_i, r_j) \in D \times D \mid i \neq j \text{ and } r_i^P = r_j^P\}.$$

To generalize this concept to sets of more than two records, we define a set $S \subseteq D$ as duplicates if all records in S have identical projections on the primary key attributes which

is: $\forall\, r_i, r_j \in S, \quad r_i^P = r_j^P$. We hence have, the more generalized expression:

$$\mathcal{DUP} = \{S \subseteq D \mid \forall\, r_i, r_j \in S,\, r_i^P = r_j^P \text{ and } |S| > 1\}.$$

To identify all duplicate sets in D, we follow these steps:

1. **Projection**
   We start by computing the projection of each record onto the primary key attributes:
   $$\{r_1^P, r_2^P, \ldots, r_n^P\}.$$
2. **Grouping**
   Then, we group records by their primary key projections. Let G be the collection of L groups, where $\forall\, l \in [\![1, L]\!]$ each group G_l, contains records with the same primary key projection:
   $$G = \{G_1, G_2, \ldots, G_L\},$$
   where $G_l = \{r_i \in D \mid r_i^P = v_l\}$ for some unique primary key projection v_l.
3. **Identifying duplicate sets**
   We now can identify all groups G_l in G that contain more than one record. These groups represent sets of duplicate records:
   $$\mathcal{DUP} = \{G_l \mid |G_l| > 1\}.$$
4. **Dropping duplicates**
   Finally, for all subsets G_l, we drop all records but one, we typically keep the first instance.

Complexity analysis
This generalized approach ensures that any set of records with identical primary key attribute values is identified as duplicates[4]. Let's analyze its time and space complexity. It can be analyzed in terms of the number of records n in the data set D and the number of attributes k.

- **Time complexity**
  - First, we consider the **projection** step, where we compute the projection of each record onto the primary key attributes. This involves iterating over all n records and projecting each record onto the k primary key attributes. If k is relatively small compared to n, the projection step has a time complexity of $O(n)$, otherwise if k is large, the time required for each projection increases. Specifically, the total time complexity of the projection phase is $O(nk)$.
  - Next, in the **grouping** step, we group records by their primary key projections.

[4]In practice, in our implementation, we used the *drop_duplicated* method from Python's pandas library (McKinney, 2010)

This step involves inserting n projected records into a suitable data structure for grouping (typically a dictionary or a hash table). Inserting each projected typically has an average-case time complexity of $O(1)$. Therefore, the grouping step has an average-case time complexity of $O(n)$.
  - Then, in the **duplicate identification** step, we identify all groups that contain more than one record. This step involves iterating over the groups formed in the previous step. In the worst case, there could be up to n groups (if all records have unique primary key projections). Checking the size of each group and collecting groups with more than one record has a time complexity of $O(n)$.
  - Finally, for each group G_l that contains more than one record, we keep only the first record and drop the rest. Iterating over all groups takes $O(L)$ time. Again, in the worst case, we have n groups. For each group G_l, dropping the records but the first one takes $O(|G_l|)$ where $|G_l|$ is the number of records in the group l. The sum of all records in the different groups being n. The time complexity of this step is therefore $O(n)$.

  Therefore, the overall time complexity is $O(nk + n + n + n) = O(nk)$ when k is large. This complexity is reduced to $O(n)$ when $n >> k$.
- **Space complexity**
  The space complexity of the algorithm is also important to consider. The projections of the records require $O(n)k$ space for a large k and $O(n)$ otherwise. The hash table used for grouping requires $O(n)$ space. Collecting the duplicate sets requires $O(n)$ space in the worst case (if all records are duplicates). Dropping the duplicates does not require any additional storage. Therefore, the overall space complexity is $O(n)k$.

This analysis indicates that if the number of attributes k defining the primary key PP is reduced, both time and space complexity become $O(n)$. However, for a large value of k, the time and space complexities of the algorithm become $O(nk)$, which could impact performance for large datasets. The typical scenario occurs when no previous work has been done to identify a primary key, resulting in k being exactly equal to the number of fields in the database (53 in our case). This underscores the importance of the pre-quality enhancement phase, as it not only allows for better structuring of the dataset but also reduces the computational cost of the algorithms in terms of both time and space complexities.

5.4.2.2 Missing values handling

As we have already stated, identifying missing values is not a tedious task in itself. The real challenge lies in distinguishing justifiably missing values from others and correcting only these values. Without domain knowledge, this task becomes even more difficult. In this section, we decide to implement a rather simple two-step solution.

- We identify fields that have missing values which might be unjustified. These fields fall into two categories: first, the attributes of the primary key; and second, the attributes that don't have many missing values. We consistently maintain the same

95% threshold used for primary key candidates (see Subsection 5.4.1.1). Any field with more than 95% missing values will be considered to have justified missing values, and we will not attempt to impute these missing values. In an industrialized implementation, such fields would be flagged by *warnings*.

- If the field being analyzed is part of the primary key, the missing value is automatically deemed abnormal. Since the projection of any record on the primary key cannot be duplicated, we cannot use the distribution of values in the same field for this case (even if the primary key might be comprised of multiple fields, this approach is not conservative enough). Therefore, we automatically generate a placeholder character that will act as the record for the remainder of the process. Then if the field being analyzed is not part of the primary key, instead of attempting to determine whether a missing value is justified or not, we assume that any observation, including missing values, could be valid and there are two possibilities:
  - If the field is a character-valued field, no imputation is made, and we assume that this missing value is justified. In the remainder of the process, specifically in Subsection 5.4.2.5, this will be checked as a logical error. For instance, if the *Enclosure* field in one row is empty, the algorithm introduced later will determine if it is likely for the model description (field *fiModelDesc*) in the considered row to have an empty Enclosure - without using domain knowledge -.
  - If the field contains numerical values, we apply a simple and classical method by replacing the missing values using linear interpolation with the two closest present values (one with a lower index and one with a higher index). In this work, where we do not have detailed information about the dataset, it is not advisable to replace missing values with a statistic (typically the average or the median) of the observations of the specific field. Indeed, the dataset could, for instance, contain time series with seasonality. In that case, the average might not be suited.

5.4.2.3 Processing statistical outliers

Let us now focus on identifying and correcting statistical outliers. We recall that we had defined a dataset D as a finite collection of n records $\{r_i\}_{i\in[\![1,n]\!]}$ with each r_i having k attributes, i.e., $r_i = (a_{i1}, \ldots, a_{ik})$. We are interested in analyzing the outliers for a given attribute. That is to say, we want to determine whether, for a given attribute j of record r_i, the observation a_{ij} is homogeneous with the rest of the observations $\{a_{lj}\}_{l\in[\![1,n]\!]\setminus\{i\}}$. Of course, depending on the meaning given to *homogeneous*, the identification and handling of statistical outliers may vary.

As we have discussed in Section 5.1, several techniques have been developed to identify outliers. Among them, many use classical results on interquantile intervals. This is notably the case with (Corrales et al., 2018), which decides that valid states a_{ij} are those that verify $a_{ij} \in [Q_1 - 1.5(Q_3 - Q_1),\ Q_3 + 1.5(Q_3 - Q_1)]$ where Q_1 and Q_3 define respectively the first and third quartiles of the distribution of attribute j's observations $\{a_{lj}\}_{l\in[\![1,n]\!]}$. Another

common method (see for instance Howell (1998)) involves invalidating all values that lie outside an interval based on the standard deviation. In other words, a_{ij} is invalidated as soon as $a_{ij} \notin [-\alpha\sigma + \mu,\ \alpha\sigma + \mu],\ \alpha \in \mathbb{R}_+^*$. μ designates the mean of the distribution and σ its standard deviation. The most common practice seems to be to choose $\alpha = 3$ (Miller (1991) for example).

There are several limits to these methods:

- Firstly, they assume that the distribution of observations is normal (Leys et al., 2013).
- Secondly, the mean and standard deviation are strongly influenced by the outliers we want to detect (Leys et al., 2013), leading to distorted measures of central tendency and dispersion.
- Finally, as Cousineau and Chartier (2010) points out, this method has very little chance of detecting outliers in small samples.

Therefore, applying these methods can lead to incorrect identification of outliers. In skewed distributions, one might either miss true outliers or falsely identify regular data points as outliers. For multimodal distributions, these methods might fail to recognize outliers within each mode. Overall, there is a non-negligible chance of falsely identifying normal data points when the assumptions required by these methods are not met. Despite these disadvantages, there are good reasons to continue using these methods. They are very simple to use and interpret, and when the amount of data used is large, the sample can approach a Gaussian distribution.

Isolation Forest

As others have done in the literature (see section 5.1), we want to leverage machine learning's ability to overcome the above limitations. A widely utilized and highly effective algorithm for anomaly detection that takes a novel approach to identifying outliers is Isolation Forest (IF) (Liu et al., 2008). Unlike traditional methods that first establish what constitutes normal data and then identify deviations from it, Isolation Forest focuses directly on isolating anomalies. The core idea behind this algorithm is that anomalies are easier to isolate than normal points because they are *few and different*. By recursively partitioning the data set, the algorithm creates a series of decision trees designed not to group similar data points but to isolate individual points. Anomalies, due to their rarity and distinctiveness, require fewer partitions to be isolated compared to normal data points. This process makes anomalies stand out more clearly and be detected more efficiently. The process and efficiency of Isolation Forest in isolating outliers is visually depicted in Figures 5.4.2, 5.4.3 and 5.4.4, where the partitions highlight the isolation of anomalies from the rest of the data set. This method is particularly advantageous in large datasets and real-time anomaly detection scenarios due to its linear time complexity and scalability. As this method seems less common than other used in this work, we give a more detailed explanation of the algorithm's functioning which can be described by the following steps:

1. **Subsampling:**

- The algorithm randomly selects a subset of the data points from the dataset. This step helps make the algorithm scalable and reduces computational complexity.

2. **Tree Construction:**
   - Recursive Partitioning: For each selected subset, the algorithm recursively partitions the data points by randomly selecting a feature and then randomly selecting a split value for that feature.
   - Node Creation: Each partition creates a node in the tree. The process continues until each data point is isolated in its own leaf node, the maximum tree height is reached, or the node contains a single data point.
3. **Forest Building:**
   - The tree construction process is repeated multiple times to build a collection of isolation trees, forming the Isolation Forest.
4. **Path Length Calculation:**
   - For each data point in the dataset, the algorithm computes the average path length from the root node to the terminating node (leaf) across all the trees in the forest. The path length is the number of edges traversed from the root to the leaf.
5. **Anomaly Score Computation:**
   - The anomaly score for each data point is calculated based on the average path length. The score is defined such that shorter average path lengths correspond to outliers (since outliers tend to be isolated quickly), while longer average path lengths correspond to normal data points.
6. **Anomaly Thresholding:**
   - The algorithm determines a threshold for the anomaly score. Data points with anomaly scores above this threshold are classified as outliers, while those below the threshold are considered normal.

For sake of illustration, let us consider a dummy data set, with two features (attributes) *Feature 1* and *Feature 2* represented by figure 5.4.2.

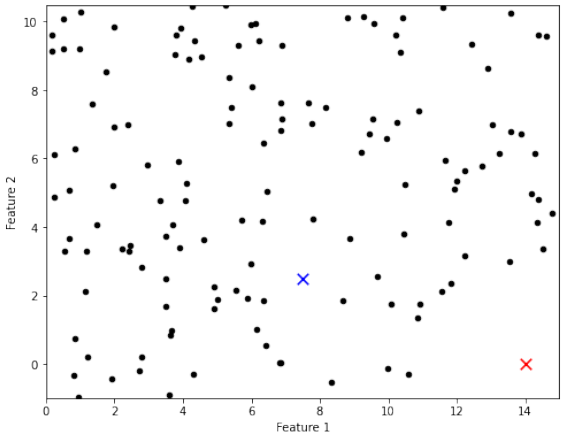

Figure 5.4.2: Dummy data set, the red and blue points are the two points we will focus our illustration on.

Now let us consider two isolation trees focusing respectively one the blue and the red point.

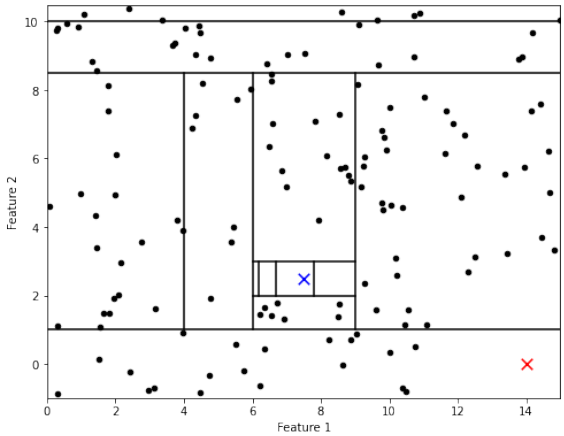

Figure 5.4.3: Isolation tree on the blue point

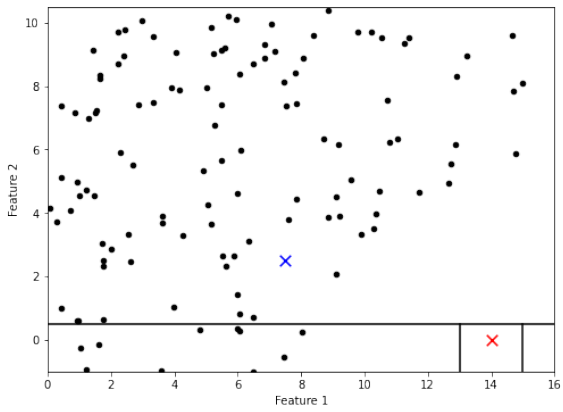

Figure 5.4.4: Isolation tree on the red point

Figures 5.4.3 and 5.4.4 illustrate the isolation process for two observations: one that seems to be valid (the blue cross) and one that seems to be an outlier (the red cross). We can easily understand how the isolation forest algorithm works. The blue cross is isolated after many steps while only three steps are enough to isolate the red cross[5]. This indicates that the red cross is an outlier. This process is repeated over a large number of random trees (leading to the forest). Then, the number of steps (branches) before the splitting are averaged (step 4 above), and a comparative analysis is performed with the threshold (steps 5 and 6 above). Figure 5.4.5 shows the average number of steps for different sizes of forests.

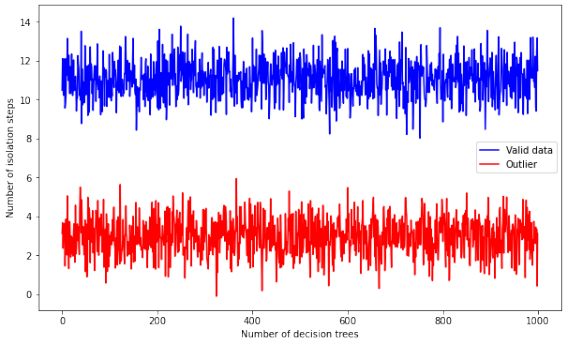

Figure 5.4.5: Number of steps for a valid data and a statistical outlier

This algorithm has proven its efficiency, particularly on large datasets. However, this is

[5]Note that these graphics are for the sake of illustration; the dataset and tree parameterization are chosen to perfectly and only match the isolation of the red and blue crosses. In practice, the isolation analysis is performed for all observations simultaneously

precisely where the limitation lies in our context. As this method is more interesting in high dimensions, it naturally loses its explainability, particularly in point 1.c of the definition (error identification). In fact, if the dataset contains only one or two columns, the algorithm loses its efficiency (as can be intuited from Figures 5.4.3 and 5.4.4 above). On the other hand, if it uses all the variables or a subset of them, it becomes impossible to know which column really poses a problem. We therefore choose not to use this powerful technique in high dimensions. Instead, we propose to combine statistical methods using standard deviation and IF in one dimension.

We work with Z the Z-score of observation a_{ij}, $Z = \frac{a_{ij}-\mu_j}{\sigma_j}$. Where μ_j and σ_j designate the unbiased estimators of respectively the expectation and the standard deviation of the distribution of attribute j's observations $\{a_{lj}\}_{l\in[\![1,n]\!]}$. Let $\varphi_{outlier}$ be the function defining whether the input X is an outlier, returning 1 in that case and 0 otherwise. Let us also assume that our IF is a function that returns 1 when given an outlier and 0 otherwise.

$$\begin{aligned} \varphi_{\text{outlier}}&: \mathbb{R} \to \{0,\ 1\} \\ &Z \mapsto \varphi_{\text{outlier}}(Z). \end{aligned}$$

$$\varphi_{\text{outlier}}(Z) = \begin{cases} f_1(Z,\ \beta_1,\ \beta_2) & \text{if } |\mu_3| < \alpha_s \text{ and } |\mu_4| < \alpha_k \\ f_2(Z,\ \beta_1,\ \beta_2,\ \gamma) & \text{otherwise,} \end{cases} \tag{5.1}$$

where:

$$f_1(Z,\ \beta_1,\ \beta_2) = \begin{cases} 1 & \text{if } Z \notin]-\beta_1,\ \beta_2[\\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

and

$$f_2(Z,\ \beta_1,\ \beta_2,\ \gamma) = f_1(Z,\ \gamma\beta_1,\ \gamma\beta_2) \times IF(Z), \tag{5.3}$$

where

$$IF(Z) = \begin{cases} 1 & \text{if the Isolation Forest algorithm detects an outlier} \\ 0 & \text{otherwise.} \end{cases} \tag{5.4}$$

We define the terms [6]:

- μ_3 and μ_4 respectively define the 3^{rd} and 4^{th} standardized moment, skewness and kurtosis.
- α_s and α_k are two strictly positive moment acceptance thresholds. We take 6 and 30 respectively.
- β_1, β_2 are two strictly positive acceptance thresholds for the standard deviation-based tolerance interval. We take 3 for both coefficients.
- Finally, γ is a parameter to widen the tolerance interval on Z, we choose 2.

[6]The large number of parameters may seem like an obstacle to easy automation, but in reality, these parameters are quite simple to choose and correspond to fairly intuitive and flexible choices based on the desired tolerance.

The idea behind this algorithm is to suggest that if we have enough data with a distribution that is not too distorted, with reasonable kurtosis and skewness, the Gaussian approximation is not a significant assumption. In this case, we can apply the standard deviation rule directly. If, on the opposite, this is not the case, we apply the same rule but with a tolerance interval γ times larger (e.g., twice as large). Since we are losing reliability, we seek a *second opinion* using the IF algorithm and consider as outliers only those entries where both techniques agree on the invalidity of the data. The main advantage of this method is that we benefit from the standard deviation's ability to discriminate values that are too high or too low, while the IF not only looks at extreme values but also identifies outliers as values that are isolated from the remaining dataset. Indeed, this algorithm will not identify an extreme value as an outlier if it has other extreme values in its neighborhood.

Figures 5.4.6 and 5.4.7 below illustrate the Kernel Density Estimation (KDE) of two attributes: *Saleprice* and *YearMade*. They help easily understanding the algorithm's process. For the attribute *YearMade*, the data is centered around the year 2000, with some noise around the mode of the distribution. We also observe that the support of the density reaches negative values as well as very high values, indicating the presence of invalid values. In this case, due to the relatively correct appearance of the density, we do not need the computational power required for the IF algorithm. We can be optimistic in our ability to successfully identify outliers using only the standard deviation rule, that is to say, $\varphi_{outlier}(Z) = f_1(Z, \dots)$. On the other hand, looking at the attribute *Saleprice*'s density, we observe that the rightmost part of the density has high values. Using only the standard deviation rule, these values would probably be wrongfully discarded. Therefore, the algorithm widens the tolerance of the standard deviation rule to ensure that we focus on extreme values, and then the IF identifies isolated extreme values, ensuring that a price is not discarded for being high but for being in a high region where no other product has a close price. That is to say, $\varphi_{outlier}(Z) = f_2(Z, \dots)$. If we had domain knowledge, the solution we would apply would be very close to the one our algorithm applies. Indeed, we would have known that the prices of the products will generally be reasonably close, but for some specific products, the prices could be very high (hence the behavior of the distribution).

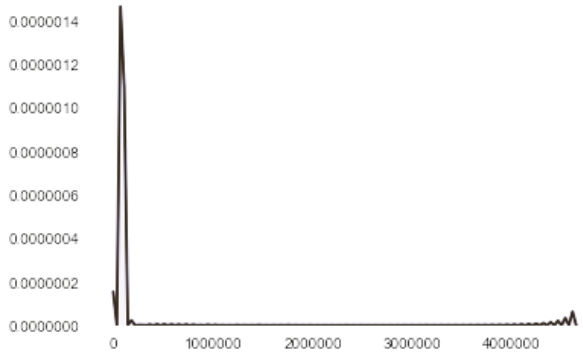

Figure 5.4.6: KDE of attribute *Saleprice*

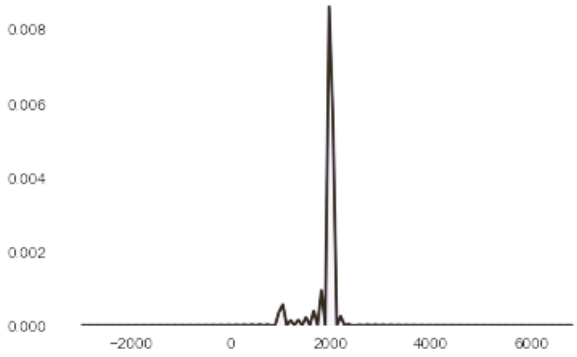

Figure 5.4.7: KDE of attribute *YearMade*

Once outliers have been rigorously identified as invalid, they are treated like any other numerical missing value. They are therefore imputed using linear interpolation, as described in Subsection 5.4.2.2.

On the choice of α_s and α_k
The choice of the tolerance parameters for the skewness and the kurtosis, respectively α_s and α_k, is important. One must keep in mind that the framework should work without domain knowledge and on any given dataset. Therefore, the values chosen must provide a practical, flexible, and robust framework for handling real-world data distributions, which often deviate from normality. We chose to select our parameters based on a known skewed and heavy-tailed distribution. We used a chi-squared distribution with 1 degree of freedom. The skewness threshold of 6 is approximately double the skewness of that distribution, indicating a substantial but reasonable allowance for asymmetry. The kurtosis threshold of 30, being double the kurtosis of the reference distribution, suggests tolerance for heavy-tailed distributions. These thresholds enable the primary outlier detection mechanism (based on standard deviation) to be broadly applicable, ensuring minor deviations in distribution shape do not overly influence outlier detection. This approach and the chosen values balance sensitivity and specificity, reducing false positives. The high thresholds ensure robustness, making the algorithm adaptable to various scenarios without frequent false alarms, while still being stringent in cases of extreme skewness or kurtosis. These parameters can be tuned depending on one's tolerance to false positives.

Complexity analysis
The complexity of this algorithm is dominated by the IF component. A thorough analysis on its time and space complexity is conducted by (Liu et al., 2008). Their work demonstrates a time complexity of the IF algorithm in $O(t\psi\log^{\cdot} + \mathrm{ntlog}^{\cdot})$, and a bounded memory requirement that grows linearly with n. In these expressions, t designates the number of trees, ϕ is the sub-sampling size, and n is the number of records.

5.4.2.4 Processing Typographical Errors

The correction of typographical errors (TPOs) is a very important issue in the preparation of data processing. Spell checking stands as the crucial process of identifying and suggesting corrections for words that are inaccurately spelled. Essentially, classic spell checkers serve as computational tools leveraging a dictionary of words to execute this task. The efficacy of such a checker relies on the extent of its dictionary. A larger repository of words typically results in an increased ability to detect errors. However, due to their reliance on standard dictionaries, they are inadequate in capturing mistakes such as proper nouns, specialized terms pertinent to particular domains, acronyms, and other specialized terminologies. These types of words are called non-words in the literature (Lee et al., 2020). They represent a notable contrast to conventional words, commonly referred to as real words. This is well explained in Bassil and Alwani (2012). As we have already mentioned in Section 5.1, on one hand, much of the literature requires the use of external dictionaries, and on the

other hand, many algorithms in the literature use ML/DL techniques to handle domain knowledge, particularly to correct non-words. We introduce in what follows a procedure leveraging ML, efficient for both real and non-words without any use of domain knowledge. Let us start by introducing key concepts that are pivotal for our algorithm.

Damerau-Levenshtein Distance

Consider two distinct character strings A and B and let us consider the task of finding the minimum number of operations required to transform B into A.

When only substitutions are allowed, the minimum number of operations is given by the Hamming distance (Robinson, 2003), or when only transpositions are allowed, the Jaro distance (Jaro, 1989). The Levenshtein distance (Levenshtein, 1966) is the length of the shortest sequence when substitutions, insertions, and deletions are allowed. This distance is used in the longest common subsequence problem (Maier, 1978). In this chapter, we will use the Damerau-Levenshtein distance (Levenshtein, 1966; Damerau, 1964), which is optimal when four operations are possible (substitution, insertion, deletion, and transposition of two consecutive characters). The Damerau-Levenshtein distance (DLD) for the letters number i and j of words A and B is recursively obtained through $d(.,.)$ below (Boytsov, 2011):

$$d_{A,B}(i,j) = \min \begin{cases} 0 & \text{if } i = j = 0 \\ d_{A,B}(i-1,j) + 1 & \text{if } i > 0 \\ d_{A,B}(i,j-1) + 1 & \text{if } j > 0 \\ d_{A,B}(i-1,j-1) + 1_{(A_i \neq B_j)} & \text{if } i,j > 0 \\ d_{A,B}(i-2,j-2) + 1 & \text{if } i,j > 1 \text{ and } A_i = B_{j-1} \text{ and } A_{i-1} = B_j. \end{cases} \tag{5.5}$$

For example, since it only takes one transposition to go from *France* to *Fracne*:

$$DLD(France,\ Fracne) = 1.$$

In practice, we will not use the DLD itself, but rather a score that increases the closer two words are to each other. Indeed, a distance of 2 between two 4-letters words, for example, and a distance of 2 between two 15-letters words should not be interpreted in the same way. The Damerau-Levenshtein Score (DLS) is given as a function of the DLD between A and B, $DLD(A,B)$ and the maximum possible distance between two words of the lengths of A and B, noted $DMAX_{A,\,B}$:

$$DLS(A,B) = \frac{DMAX_{A,\,B} - DLD(A,B)}{DMAX_{A,\,B}}. \tag{5.6}$$

Considering the previous example, note that the maximal Damerau-Levenshtein distance between two words of length 6 is 6. Therefore, the score between *France* and *Fracne* is

given by:

$$DLS(France,\ Fracne) = \frac{6-1}{6} \approx 0.8334.$$

When considering now, two words of length 3, for example *Pau* and *Pou*, with still a DLD of 1, the score is lower than for the previous example:

$$DLS(Pau,\ Pou) = \frac{3-1}{3} \approx 0.667.$$

Detecting typographical errors

To detect all typographical errors in a database field, the ideal solution would be to calculate the DLS of each entry with all the others. This would give us a matrix of DLSs, and we would then group together the entries with low DLSs, as this would indicate that they were spelled very similarly and were therefore the same word spelled in different ways. To find the true value and thus correct the TPOs, it would be logical to take the entry that has been used the most often. Unfortunately, this ideal solution is not feasible due to its computational cost. Indeed, even when only considering the step involved in building the DLS matrix, it would require $O(n^2)$ Damerau-Levenshtein function calls, which is not feasible for a table with hundreds of thousands or even millions of entries. Nevertheless, we have decided to keep the essence of this method but attempt to reduce the size of the problem. To achieve this, we will sort the list of observations alphabetically before calculating the consecutive DLSs. The idea is that by sorting them in alphabetical order, we will increase our chances of grouping similar entries together and forming part of the aforementioned groups.

To identify the groups, we examine the curve of consecutive DLSs. As soon as a jump is observed on the DLS curve, we have a priori reached a different word. The threshold beyond which we admit to having changed words is set at 0.7. Figure 5.4.8 illustrates the curve of DLSs of words alphabetically ordered as well as the threshold. The words correspond to the entries of the field *State*.

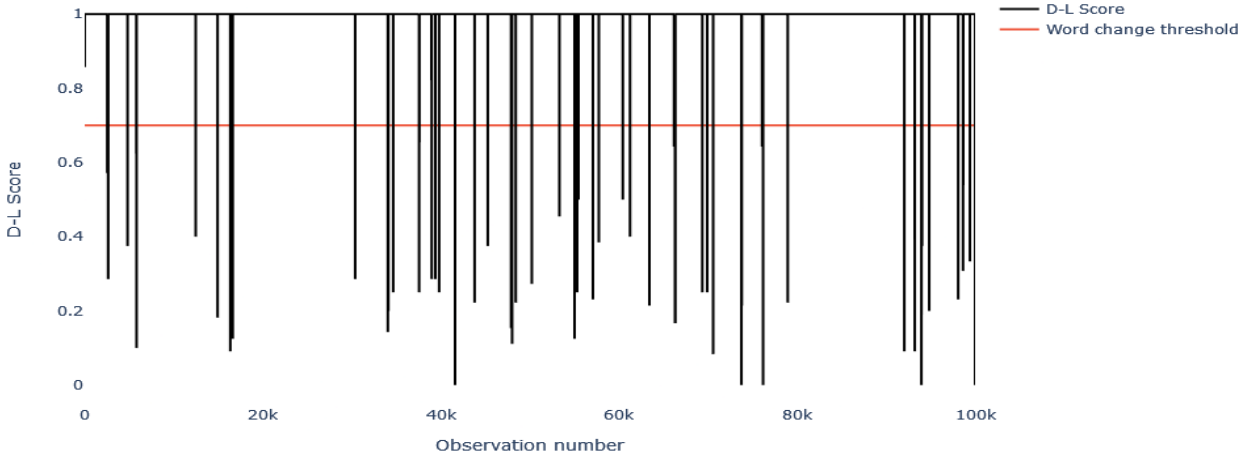

Figure 5.4.8: Damerau-Levenshtein score jumps for variable *State*

Having identified all the groups based on the DLS jumps, we can now represent each of these groups by their *dominant* element (the one with the most occurrences in the dataset). It is then necessary to retrieve any elements of a group we may have missed. For example, *France* and *Grance* are very close in DLS terms, but when sorted alphabetically, they will not be in the same group. This is the step where we use ML, specifically unsupervised learning via clustering techniques. The clustering algorithm seeks to group together all words with similar DLS vectors (a row of the DLS matrix).

The algorithm we decided to use is hierarchical agglomerative clustering (Müllner, 2011). In a different context from that of this document, where the use of a priori knowledge of the database could be exploited, the user could communicate the correct value (which could correspond, for instance, to the number of states where sales have been concluded). However, in our context, we are obliged to determine the number of clusters. To do this, we use the gap statistic method (Tibshirani et al., 2001), which proceeds by comparing the intra-cluster dispersion with that which would have been obtained in an empty set.

At this point, all similar elements have been grouped together. However, an additional step is necessary: correcting wrongfully grouped elements. While this situation is unlikely, it can occur when dealing with long and similar words. For example, *North Dakota* and *South Dakota* share ten out of twelve characters, resulting in a DLS of 0.8334, which exceeds the 0.7 threshold.

The introduced solution is quite straightforward. Within each group, we identify elements that differ from the dominant element (the one with the most occurrences) but still have high frequencies. These entries are candidates for being incorrectly attributed to a group. Depending on the nature of the field being analyzed, one of two solutions will be applied. If the words being tested are dictionary entries, a simple test is to determine if the elements differing from the dominant element with high occurrences are valid dictionary entries. If they are, they are identified as valid entries, and a flag (typically a *warning* in programming languages) is raised to prompt the user to check theses specific corrections after the whole process. On the other hand, if the words are not dictionary entries but non-words, no correction is made due to insufficient information to confidently discard the previously rigorously obtained result. Again, a flag is raised for the user to review.

Remarks

- It is important to note that this process, especially checking if the words are real or non-words, is not expensive since we only need to consider the unique dominant entries, whose number is significantly lower than the number of observations.
- The aim of this work is to construct a fully automated process to measure and enhance the quality of a data set. Flags are raised solely to inform the user of corrections that were or were not conducted in cases of potential ambiguity. No action is required from the user, ensuring that the process remains fully automated.
- It is important to underline the significance of the choice of the 0.7 threshold to

detect DLS jumps. This is a hyper-parameter of the algorithm. The higher the threshold, the fewer the number of mistakenly grouped words, but as a trade-off, the number of elements to include in the clustering analysis increases. On the other hand, the lower the threshold, the more mistakenly grouped words, and therefore, more corrections are required after clustering. To strike the right balance between accuracy and computational performance, it is necessary to carefully choose the threshold. A higher or lower threshold could yield better or worse results depending on the data set. Practitioners should note that it is better to use a higher threshold (typically close to 0.7) and recover words that were not initially grouped through clustering rather than grouping words incorrectly from the start.

The TPO detection algorithm described can be summarized as follows:

1. The list of words requiring TPO detection is sorted alphabetically, and the consecutive DLSs are calculated.
2. Using DLS jumps, we distinguish the initial similarity groups. Each group contains all the elements from the current peak (included) to the next (excluded). Each group is then represented by its dominant element (the element with the highest frequency).
3. Dominant elements that seem to represent the same word are regrouped. This is achieved by calculating the matrix of DLSs on all dominant elements, then clustering this matrix to regroup similar words. Dominant elements that are in the same clusters are considered to represent the same word. Their initial groups are merged. The new dominant element is the one among them with the highest occurrence.
4. A final verification is made to ensure that no words are wrongfully flagged as a TPO. In each cluster, we identify non-dominant elements with high frequencies (for instance, members of a cluster that have more than half the frequency of the dominant element), these have potentially been wrongfully identified as typographical errors.
   (a) If the field being analyzed contains real words that are listed in the dictionary, we check whether the potentially wrongfully identified words are valid dictionary entries, if so, they are considered as valid words and not TPOs.
   (b) Otherwise, we do not make any modifications to the previously constructed groups.
5. At the end of the process, we have as many groups as valid words (real words or non-words, depending on the field content). Each group is represented by one and unique dominant element, and each element in the group written differently will be corrected to the dominant element.

Complexity analysis

Let us analyze the time and space complexity of this algorithm. We recall that n is the number of records in the dataset, thus it is the number of words in a given field. Let us define g as the number of dominant elements, $g << n$. Let l_{avg} be the average length of the words being corrected. l_{avg} is a negligible value compared to n.

- **Time complexity**

- **Sorting the list of words alphabetically** takes $O(n \log n)$ time.
- The DLS calculation between two words has an average complexity of $O(l_{\text{avg}}^2)$. **Calculating DLS between consecutive words** in the sorted list has an average complexity of $O(nl_{\text{avg}}^2)$. We can reasonably state that $l_{\text{avg}}^2 << n$. Which leads to a time complexity of $O(n)$ for this step.
- **Distinguishing initial similarity groups and determining dominant elements** involves scanning through the list of words once to form groups and determine the dominant element within each group. This is $O(n)$ since it involves a linear pass through the list.
- To **regroup dominant elements by clustering DLS matrix**, we start by constructing the DLS matrix for g dominant element which involves $g(g-1)/2$ DLS calculations, each taking $O(l_{\text{avg}}^2)$, resulting in $O(g^2 l_{\text{avg}}^2)$. Here we can not make the assumption $l_{\text{avg}}^2 << g^2$. Clustering the matrix using hierarchical clustering has a complexity of $O(g^3)$.
- The last step of the algorithm consists in **verifying flagged TPOs**. It requires identifying non-dominant elements with high frequencies by scanning through the whole list, resulting in a complexity of $O(n)$. Then, checking against a dictionary (assuming the dictionary check is $O(1)$) involves scanning each flagged word, taking $O(n)$ in the worst case.
- **Constructing the final groups and correcting words**: This step involves scanning through the list and making necessary corrections and therefore has a time complexity of $O(n)$.

Let us now aggregate all complexities. Based on the analysis conducted for each of the previous steps, the overall complexity is $O\left(n \log n + n + g^2 l_{\text{avg}}^2 + g^3\right)$, which is equivalent to $O\left(n \log n + g^2 l_{\text{avg}}^2 + g^3\right)$. This complexity analysis again demonstrates how beneficial it is, from a computational point of view, to reduce the clustering process dimension by sorting the elements and only taking dominant elements. When g is very small, the overall complexity becomes $O(n \log n)$ whereas the algorithm without the reduction through g^3 would have an overall complexity of $O(n^3)$. This low complexity, justifies the very good computational performance obtained as we will show in Section 5.5.

- **Space complexity** The space complexity, is mainly dominated by two storage needs. First, we need to store all the words being analyzed and their consecutive DLSs. This has a space complexity of $O(n)$. Then we need to store the dominat elements matrix which has a space complexity of $O(g^2)$. Therefore the overall space complexity is $O(n + g^2)$. Therefore, when $g <^2< n$, the space complexity is in $O(n)$.
  The space complexity is mainly dominated by two storage needs. First, we need to store all the words being analyzed and their consecutive DLSs. This has a space complexity of $O(n)$. Then we need to store the dominant elements DLS matrix, which has a space complexity of $O(g^2)$. Therefore, the overall space complexity is $O(n + g^2)$. Consequently, when $g <^2< n$, the space complexity is $O(n)$.

5.4.2.5 Processing logic errors

Logic errors in a dataset are the most delicate to identify and rectify. Unlike typographical errors or certain outliers, these errors do not readily reveal themselves, even through observation of the dataset. Logic errors arise when the data does not adhere to the expected logical relationships between variables, making them challenging to detect. To address this issue, we propose an algorithm that, despite its simplicity, effectively identifies these elusive errors. Our method involves the use of data mining techniques to uncover relationships between various variables in the dataset. By identifying and establishing the strongest relationships as true, we can identify observations that fail to conform to these logical connections. Such non-conforming observations are then flagged as inconsistent with the logical structure of the dataset. This approach is used in the work of (A.M.Rajeswari et al., 2014) and (Karthikeyan and Vembandasamy, 2015a), where the authors used similar techniques not to specifically detect logical errors but to identify outliers[7]. This method ensures that the integrity of the dataset is maintained by validating that the data adheres to the underlying logical relationships expected within the dataset. As we had already mentioned in Section 5.4.2.2, unjustified missing text values will also be flagged in this process.

To detect relationships between variables, we decided to use the association rule mining algorithm, Apriori (Agrawal et al., 1994). The algorithm is often used in market basket analysis and purchase recommendation systems. Like the Isolation Forest used for outliers, this algorithm seems less known, so we describe its functioning here:

Apriori algorithm

Let us start by recalling the main constituents of the Apriori algorithm:

- **Itemset**: An itemset is a collection of one or more items. A k-itemset is an itemset containing k items.
- **Support**: The support is the frequency of a particular itemset. If an itemset I appears in s out of n records, then the support of I is calculated as s/n. The minimum support threshold is a user-defined threshold that an itemset's support must meet or exceed to be considered frequent. Itemsets with support below this threshold are discarded from further consideration.
- **Association Rule**: An association rule is an implication of the form $I_1 \rightarrow I_2$, where I_1 and I_2 are itemsets. The rule suggests that records containing I_1 are likely to also contain I_2.
- **Confidence**: Confidence is a measure of the reliability of an association rule. For a rule $I_1 \rightarrow I_2$, the confidence is given by $\text{confidence}(I_1 \rightarrow I_2) = \text{support}(I_1 \cup I_2)/\text{support}(I_1)$. The minimum confidence threshold is a user-defined threshold that an association rule's confidence must meet or exceed to be considered strong.

[7] In these references, outliers do not designate statistical outliers like in ths work. Instead, they indicate abnormal behavior of a whole record.

The steps of the Apriori algorithm with constituents are as follows:

1. **Generate candidate itemsets**: The algorithm starts with all individual items in the database, generating candidate 1-itemsets. These are evaluated to determine their support.
2. **Filter candidates**: For each candidate itemset, the algorithm calculates its support. Only those itemsets whose support meets or exceeds the minimum support threshold are considered frequent itemsets. Let us designate by F_k, the set of k-itemsets kept after the filter.
3. **Generate frequent itemsets**: Using the frequent k-itemsets F_k, the algorithm generates candidate $(k+1)$-itemsets by joining F_k with itself. It then filters the candidates that do not meet the support threshold to form the frequent $(k+1)$-itemsets F_{k+1}.
4. **Repeat**: This process is repeated iteratively, generating candidate and frequent itemsets of increasing size until no new frequent itemsets are found.
5. **Generate association rules**: From the collection of all frequent itemsets, the algorithm generates possible association rules. For each frequent itemset, it generates all possible rules and calculates their confidence. Only the rules that meet the minimum confidence threshold are considered strong association rules.

Naturally, the algorithm in its initial version takes a long time to run as it has a high time complexity. For instance, for each k, the algorithm generates $\binom{n}{k}$ candidates. For that reason, we decided to make a change in the functioning of the algorithm. In the original version of the algorithm, the user can adjust the minimum support, as well as the confidence level of a rule. The higher these two values are, the fewer the number of rules that will be searched. We add another parameter that can be adjusted, the maximum length of k-itemsets. This gives us the option of limiting the number of elements that can actually define a rule. This affects the number of repetitions conducted in step 4 as well as the sizes of the data being generated and analyzed in steps 1 to 3 of the description above. This change saves a considerable amount of time. Beyond the computational pragmatism of this change, this modification is also supported by the fact that above a certain size of k-itemsets, the rule becomes unreadable, and we lose the explainability that is at the core of our concerns in this chapter.

In our research, we employed the Apriori algorithm with the following parametrization: the minimum support threshold is set at 0.33% and the confidence level of 99%. The combination of a low minimum support threshold of 0.33% and a high confidence interval of 99% implies that the Apriori algorithm will focus on identifying rare but highly reliable associations within the dataset. This configuration is particularly useful in scenarios where discovering infrequent yet strongly indicative patterns is crucial, such as errors in a data set. By setting a low support threshold, the algorithm ensures that even the quiet uncommon itemsets are considered, while the high confidence requirement ensures that only the most reliable and significant associations are retained. This approach allows for a comprehensive analysis, albeit at the cost of increased computational resources due to the larger number

of candidate itemsets. The maximum length of k-itemsets is limited at 3, reducing the impact of the low minimum support threshold. This process yields a set of rules, which are subsequently filtered to retain only those with confidence levels below 100%. In this way, we identify those that have high confidence levels but which in rare cases fail: they are our potential logic errors. For each of these identified rules, all the composing observations (which are a subset of a given record) are marked as invalid due to logic errors. Finally, corrections can be made based on the expected rules that were violated despite the high confidence in those rules.

Complexity analysis
To our knowledge, the literature does not contain research that specifically addresses the issue of the complexity of the Apriori algorithm. Here, we attempt a brief complexity analysis.

- **Time Complexity**
  The time complexity of the Apriori algorithm is influenced by the number of records in the database (n), the average transaction length (a), and the number of frequent k-itemsets generated, C_k.
  - For each level k, **the algorithm generates candidate itemsets** of length $k+1$ from frequent itemsets of length k. In the worst case, each pair of frequent k-itemsets is considered (has a support above the threshold). The join operation is combinatorial. The complexity of generating candidate itemsets for each level can therefore be approximated as $O(C_k^2)$ in the worst case.
  - For each candidate k-itemset, the algorithm needs to **scan the entire database to count its support**. Scanning the database n times, takes $O(naC_k)$ for each level k.

  Therefore, the overall time complexity can be expressed as: $O\left(\sum_{k=1}^{a}\left(C_k^2 + naC_k\right)\right)$. Note that in our case, a is bounded to the maximal length of k-itemsets that we set at 3.
- **Space Complexity**
  The space complexity of the Apriori algorithm is mainly influenced by the number of candidate itemsets stored at each level. At each level k, the algorithm stores candidate k-itemsets and their support counts. The average space required for this is $O(aC_k)$ for each level. Therefore, the overall space complexity can hence be approximated as: $O\left(\sum_{k=1}^{a} aC_k\right)$.

5.5 Results

We recall that for the quality enhancement phase, we have at our disposal (again, without using them) the exact positions of the invalid entries (see Table 5.3.2). This allows us to compute the exact accuracies we have achieved. However, for the pre-quality enhancement phase, specifically for the primary key, this is not the case.

5.5.1 Pre-Quality enhancement Results

5.5.1.1 Identification of primary keys

We applied the steps described in Subsection 5.4.1.1.

- Step 1 of the procedure consisted in identifying, columns with low rates of missing values. This narrowed down the selection to only the twelve out of fifty-three fields that are listed in table 5.5.1.
- Then, in step 2, identifying fields with specific expressions in their names (*ID*, *CODE*, or *KEY*) would leave us with three fields. Therefore, we need to proceed to step 2.b, which involves finding combinations that allow to identify unique values.
- In step 2.b, it is quickly determined that for the subset $K = (SalesID, ModelID)$, only about 0.1% of value rows are duplicated. Consequently, this subset is identified as the primary key and will be utilized for the subsequent steps of the framework.

Field	Variable Type	Proportion of missing values (%)
auctioneerID	INT64	0.000
Enclosure	STR	0.025
fiBaseModel	STR	0.000
fiModelDesc	STR	0.000
fiProductClassDesc	STR	0.000
ModelID	INT64	0.000
ProductGroupDesc	STR	0.000
SalesID	FLOAT64	0.047
saledate	STR	0.000
state	STR	0.000
YearMade	INT64	0.000
datasource	INT64	0.000

Table 5.5.1: Candidates for the primary key. The listed fields have less than 5% of missing values, the fields in red are potential candidates due to their names and are the first tested in the uniqueness analysis, the fields shaded in blue are the ones finally chosen as the primary key.

The whole process is executed under 1 second. This efficient execution can be attributed to the fact that we initiated step 2.b with attributes identified during the pattern recognition step. Also note that in a dataset, primary keys are not necessarily a unique subset.

5.5.1.2 Processes mapped to specific fields

After applying the filters introduced in Subsection 5.4.1.2, each of the algorithms of the framework will be applied to the following fields.

- **Redundancies**
  The redundancy analysis will be conducted on the attributes that comprise the primary key, namely *SalesID* and *ModelID*.
- **Absences**
  All fields will be screened for missing values.
- **Statistical outliers**
  The analysis of outliers will only be applied to three fields: *SalePrice*, *YearMade*, and *MachineHoursCurrentMeter*.
- **Typographical errors**
  After applying the filter, only three out of fifty-three fields will be analyzed for typographical errors: *state*, *Enclosure*, and *ProductGroupDesc*.
- **Logic errors**
  Finally, even after the filtering process, not as many fields are excluded from the analysis compared to the other steps of the framework. Fourteen fields remain: *Enclosure, Ripper, fiBaseModel, Drive_System, state, Transmission, ProductGroupDesc, fiProductClassDesc, Pad_Type, fiSecondaryDesc, saledate, ProductSize, Hydraulics*, and *fiModelDesc*.

5.5.2 Quality enhancement results

Overall, the framework we introduce produces reasonably good results. The difficulty of our work lies in finding the right balance between performance and explainability. Any invalid data identified is fully explainable and is natively corrected. The accuracy for statistical outliers and logic errors, could be improved with reasonable ease if we were less exigent regarding the explainability of the results and the algorithms.

In terms of computer performance[8], with the exception of the logic errors detection algorithm, all the algorithms run in a matter of seconds. Handling duplicates takes 1 second, missing values are dealt within less than 1 second, outliers are identified and imputed in 3 seconds, and typographical errors are handled in 33 seconds. The logic error identification took 45 minutes.

Redundancies and Absences

100% of duplicates and 100% of missing values were successfully identified. This task does not represent a challenge per se. However, it is interesting to note that the simple rule of thumb used to designate unjustified missing numerical values (95% threshold) is efficient.

Statistical outliers and Logic Errors

Only 51% of statistical outliers were identified. As we mentioned in Section 5.1, detecting statistical outliers while maintaining maximal explainability presents several challenges. The solution we introduced in Subsection 5.4.2.3 allows for fully explainable identification.

[8]The configuration of the computer used is rather moderate: Intel® Core™ i5-7200U with 2.5 GHz base frequency

For any flagged variable, it is possible to explain exactly why the variable was considered an outlier. The variable is either too extreme in a non-extreme distribution, or it is isolated from other extreme values in a distribution that has valid extreme values. The issue with this approach is that it does not consider the surrounding variables. For instance, when analyzing statistical outliers in a dataset with fields representing prices and products, it is very challenging to do so without considering the specific products. There are many algorithms that could effectively consider all the surrounding information (typically an Isolation Forest in high dimensions), but they would not be able to flag the specific field with an issue (therefore failing point 1.(c) of our definition of explainability and interpretability) or would somehow not be able to guarantee explainability and interpretability. A solution could be to handle outliers as logic errors. This solution was considered but was discarded due to the heavy computational cost it would imply. Indeed, it would mechanically increase the number of fields to consider for logic errors, and it would require transforming numerical values into categorical ones (for instance, prices from 1 to 100, then 100 to 500, and so on instead of using numerical values).

Only 35% of logic errors are identified. This result is attributable to the other constraint set in this work, which is not using domain knowledge. Let us recall the logic errors in the dataset introduced in Table 5.3.2. We had *Wrong category*, *Incoherent machine / Drive system / Product group description*, and *YearMade* > *saledate*. Apart from the last case, all these types of errors could be directly identified using domain knowledge. Specifically, in our case, by using the specific and unique mapping that describes the relationship from one machine (*MachineID*) to its class (*ProductClassDesc*) with all specifications in-between (*Forks*, *Transmissions*, ...). A solution to keep the framework domain knowledge-free while being more accurate would be to add a third step in the pre-quality enhancement phase that would try to establish dependency relationships between each field of the dataset and then focus an intensive Apriori analysis between these variables without limiting the maximal length of k-itemsets. To determine the dependency relationships, solutions that could be explored include, but are not limited to, random forest feature importance (Breiman, 2001) and ANOVA analysis (St et al., 1989). Additionally, it would be highly beneficial to transform numerical variables into meaningful categorical variables to include them in the logic error detections. As stated above, this would also be advantageous for the detection of statistical outliers. Deepening the research in the mentioned directions would allow for more accurate detection and correction of logic errors while ensuring explainability and interpretability. However, this would come at a hefty computational price. It would be advisable to conduct these evolutions along with parallelized implementation of the algorithms.

Typographical errors

100% of typographical errors are identified. This is a very good result as we have successfully identified all typographical errors, including entry errors as well as case errors. We have an algorithm that combines clustering and Damerau-Levenshtein distance in an interesting yet not complicated way, achieving remarkable accuracy in detecting typographical errors.

The algorithm is made feasible by the simple trick of sorting the words to be corrected alphabetically, which considerably reduces processing (by achieving a very good time complexity, close to $O(n \log n)$, as shown in Subsection 5.4.2.4). For example, for the variable *state*, we start with 102 unique observations among the 100,000 initial words. We end up with 63 groups resulting from the DLS jumps, which corresponds to a reduction of around 60% in the matrix size of DLSs. Another very good result obtained is that no false positives are produced by the algorithm.

5.6 Conclusion

We have introduced a framework that measures the quality of a dataset without any use of domain knowledge, while maintaining the explainability and interpretability of our results. Our framework is indeed able to identify and correct errors in a dataset without using any information beyond the provided dataset, while being able to fully justify why each flagged observation is considered invalid and propose corrections. The framework handles missing values, duplicates, outliers, typographical errors, and logic errors. After thoroughly reviewing the literature and analyzing our framework in detail, we can see that our work distinguishes itself from the literature by adhering to two constraints (guaranteeing explainability and interpretability and being domain knowledge-free), by the simplicity of the algorithms used, and by the way the algorithms combine machine learning, statistics, and some common-sense tricks.

The analysis of the results we have conducted allows us to draw two main conclusions. First, the concept of analyzing the quality of a dataset without any information about the data will often come at the expense of computing performance. For instance, the need to start the framework with a pre-quality enhancement phase to search for information that would have been obtained in a classical situation is an additional computational cost. Another illustration can be found in the logic error detection algorithm, where domain knowledge would expedite the association rule mining algorithm as the related targets would be manually selected, thus allowing for targeted mining of strong relationships. This would also considerably increase the accuracy of the algorithm. Secondly, the aim of being entirely explainable and interpretable will often reduce the accuracy of the algorithms used. Indeed, this aim systematically involves stepping away from powerful tools, typically some deep learning (DL) and machine learning (ML) algorithms. A good illustration is our choice of using the isolation forest algorithm only in one dimension.

Even with these strict constraints, we have also shown that it is entirely possible to obtain very good results. Indeed, apart from outliers and logic errors, all other types of errors have been fully identified and corrected. Typically, the algorithm to identify and correct typographical errors performs very well with a very good complexity. Another conclusion that can be drawn from this exploratory work is that, in a setup constrained as ours, it is necessary to have well-defined pre-analysis steps that will target the analysis and reduce the processing time.

To extend the work we have conducted, many interesting analyses could be undertaken. First, we could consider, as already mentioned, analyzing in more depth the relationships between the attributes. This would not only help reduce computational costs but also provide a better understanding of the database, allowing for analysis beyond this framework. Additionally, it could be beneficial to consider confidence levels on the corrections made. For instance, when handling outliers, we could run the algorithm multiple times with different parameters, especially α_s, α_k, and γ. Based on the different results, we could explore the possibility of computing a confidence level on the identifications made.

Bibliography

Abramowitz, M., Stegun, I. A., and Romer, R. H. (1988). Handbook of mathematical functions with formulas, graphs, and mathematical tables.

Agrawal, R., Srikant, R., et al. (1994). Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499. Santiago.

Akinci, O. and Queralto, A. (2022). Credit spreads, financial crises, and macroprudential policy. *American Economic Journal: Macroeconomics*, 14(2):469–507.

Albawi, S., Mohammed, T. A., and Al-Zawi, S. (2017). Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)*, pages 1–6. Ieee.

Alfonsi, A. (2005). On the discretization schemes for the cir (and bessel squared) processes.

Alfonsi, A. (2013). Strong order one convergence of a drift implicit Euler scheme: application to the CIR process. *Statist. Probab. Lett.*, 83(2):602–607.

Alfonsi, A. et al. (2015a). *Affine diffusions and related processes: simulation, theory and applications*. Springer.

Alfonsi, A. et al. (2015b). *Affine diffusions and related processes: simulation, theory and applications*, volume 6. Springer.

A.M.Rajeswari, M.Sridevi, and C.Deisy (2014). Outliers detection on educational data using fuzzy association rule mining.

Avramov, D., Jostova, G., and Philipov, A. (2007). Understanding changes in corporate credit spreads. *Financial Analysts Journal*, 63(2):90–105.

Aydilek, I. and Arslan, A. (2013). A hybrid method for imputation of missing values using optimized fuzzy c-means with support vector regression and a genetic algorithm. *Inf. Sci.*, 233:25–35.

Barker, R., Dickinson, A. S., and Lipton, A. (2016). Simulation in the real world. *Available at SSRN 2831726*.

Barnett, V. and Lewis, T. (1994). *Outliers in Statistical Data*, volume 3. Wiley, New York, NY, USA.

Bassil, Y. and Alwani, M. (2012). Context-sensitive spelling correction using google web 1t 5-gram information. *arXiv preprint arXiv:1204.5852*.

Bateman, H. (1953). *Higher transcendental functions [volumes i-iii]*, volume 1. McGRAW-HILL book company.

Baum, C. F., Schäfer, D., and Stephan, A. (2016). Credit rating agency downgrades and the eurozone sovereign debt crises. *Journal of Financial Stability*, 24:117–131.

BCBS and BIS (2011). *Basel III: A Global Regulatory Framework for More Resilient Banks and Banking Systems*. Bank for International Settlements.

Ben Alaya, M. and Kebaier, A. (2012a). Parameter estimation for the square-root diffusions: ergodic and nonergodic cases. *Stochastic Models*, 28(4):609–634.

Ben Alaya, M. and Kebaier, A. (2012b). Parameter estimation for the square-root diffusions: ergodic and nonergodic cases. *Stoch. Models*, 28(4):609–634.

Ben Alaya, M. and Kebaier, A. (2013). Asymptotic behavior of the maximum likelihood estimator for ergodic and nonergodic square-root diffusions. *Stochastic Analysis and Applications*, 31(4):552–573.

Berninger, C. and Pfeiffer, J. (2021). The gauss2++ model: a comparison of different measure change specifications for a consistent risk neutral and real world calibration. *European Actuarial Journal*, 11(2):677–705.

Bielecki, T. R., Jeanblanc, M., and Rutkowski, M. (2011). Hedging of a credit default swaption in the cir default intensity model. *Finance and Stochastics*, 15(3):541–572.

Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities. *Journal of political economy*, 81(3):637–654.

Bloch, D. A. (2019). Neural networks based dynamic implied volatility surface. *Available at SSRN 3492662*.

Boytsov, L. (2011). Indexing methods for approximate dictionary searching. *Journal of Experimental Algorithmics*.

Brace, A., Gatarek, D., and Musiela, M. (1997). The market model of interest rate dynamics. *Mathematical Finance*, 7(2):127–155.

Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32.

Brigo, D. and Alfonsi, A. (2005). Credit default swap calibration and derivatives pricing with the ssrd stochastic intensity model. *Finance and stochastics*, 9(1):29–42.

Brigo, D. and Mercurio, F. (2006). *Interest rate models-theory and practice: with smile, inflation and credit*, volume 2. Springer.

Bruti-Liberati, N., Nikitopoulos-Sklibosios, C., and Platen, E. (2010). Real-world jump-diffusion term structure models. *Quantitative Finance*, 10(1):23–37.

Büchel, P., Kratochwil, M., Nagl, M., and Rösch, D. (2022). Deep calibration of financial models: turning theory into practice. *Review of Derivatives Research*, pages 1–28.

Buehler, H., Gonon, L., Teichmann, J., and Wood, B. (2019). Deep hedging. *Quantitative Finance*, 19(8):1271–1291.

Çelik, M., Dadaşer-Çelik, F., and Dokuz, A. Ş. (2011). Anomaly detection in temperature data using dbscan algorithm. In *2011 international symposium on innovations in intelligent systems and applications*, pages 91–95. IEEE.

Chan, K. C., Karolyi, G. A., Longstaff, F. A., and Sanders, A. B. (1992). An empirical comparison of alternative models of the short-term interest rate. *Journal of Finance*, 47(3):1209–1227.

Chen, L., Pelger, M., and Zhu, J. (2020). Deep learning in asset pricing. *Available at SSRN 3350138*.

Chen, Y.-T., Lee, C.-F., and Sheu, Y.-C. (2009). An integral-equation approach for defaultable bond prices with application to credit spreads. *Journal of Applied Probability*, 46(1):99–118.

Chiarella, C., Fanelli, V., and Musti, S. (2011). Modelling the evolution of credit spreads using the cox process within the hjm framework: A cds option pricing model. *European Journal of Operational Research*, 208(2):95–108.

Cline, W. (2023). Fighting the pandemic inflation surge of 2021-2022. *Economics International Inc., Working Paper*, pages 23–1.

Corrales, D. C., Ledezma, A., and Corrales, J. C. (2018). From theory to practice: A data quality framework for classification tasks. *Symmetry*, 10(7):248.

Cousineau, D. and Chartier, S. (2010). Outliers detection and treatment: A review. *International Journal of Psychological Research*, 3(1):58–67.

Covitz, D. and Downing, C. (2007). Liquidity or credit risk? the determinants of very short-term corporate yield spreads. *The Journal of Finance*, 62(5):2303–2328.

Cox, J. C., Ingersoll, J. E., and Ross, S. A. (1985a). A theory of the term structure of interest rates. *Econometrica*, 53(2):385–407.

Cox, J. C., Ingersoll Jr, J. E., and Ross, S. A. (1985b). A theory of the term structure of interest rates. *Econometrica*, 53(2):385–408.

Cox, J. C., Ingersoll Jr, J. E., and Ross, S. A. (2005). A theory of the term structure of interest rates. In *Theory of valuation*, pages 129–164. World Scientific.

Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., and Bharath, A. A. (2018). Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65.

Dai, Q. and Singleton, K. J. (2000). Specification analysis of affine term structure models. *The journal of finance*, 55(5):1943–1978.

Dai, W., Yoshigoe, K., and Parsley, W. (2018). Improving data quality through deep learning and statistical models. In *Information Technology-New Generations: 14th International Conference on Information Technology*, pages 515–522. Springer.

Damerau, F. (1964). A technique for computer detection and correction of spelling errors. *Commun ACM*, 7(3):171–176.

Davies, A. (2008). Credit spread determinants: An 85 year perspective. *Journal of Financial Markets*, 11(2):180–197.

De Santis, R. (2016). Credit spreads, economic activity and fragmentation. Working Paper Series 1930, European Central Bank.

Derouich, M. B. and Kebaier, A. (2022). The interpolated drift implicit euler scheme multilevel monte carlo method for pricing barrier options and applications to the cir and cev models.

Despotovic, M., Nedic, V., Despotovic, D., and Cvetanovic, S. (2016). Evaluation of empirical models for predicting monthly mean horizontal diffuse solar radiation. *Renewable and Sustainable Energy Reviews*, 56:246–260.

Doshi-Velez, F. and Kim, B. (2018). Considerations for evaluation and generalization in interpretable machine learning. *Explainable and interpretable models in computer vision and machine learning*, pages 3–17.

Duffee, G. R. (2002). Term premia and interest rate forecasts in affine models. *The Journal of Finance*, 57(1):405–443.

Duffie, D. and Singleton, K. J. (1999). Modeling term structures of defaultable bonds. *The Review of Financial Studies*, 12(4):687–720.

El Karoui, N., Jeanblanc, M., and Jiao, Y. (2010). What happens after a default: the conditional density approach. *Stochastic processes and their applications*, 120(7):1011–1032.

Escalante, H. J. (2005). A comparison of outlier detection algorithms for machine learning. In *Proceedings of the International Conference on Communications in Computing*.

Escalante, H. J., Escalera, S., Guyon, I., Baró, X., Güçlütürk, Y., Güçlü, U., and van Gerven, M. (2018). *Explainable and Interpretable Models in Computer Vision and Machine Learning*. Springer.

Feldhütter, P. and Schaefer, S. M. (2018). The myth of the credit spread puzzle. *The Review of Financial Studies*, 31(8):2897–2942.

García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J. M., and Herrera, F. (2016). Big data preprocessing: methods and prospects. *Big data analytics*, 1:1–22.

Geman, H., El Karoui, N., and Rochet, J.-C. (1995). Changes of numeraire, changes of probability measure and option pricing. *Journal of Applied probability*, 32(2):443–458.

Giacometti, R. and Teocchi, M. (2005). On pricing of credit spread options. *European journal of operational research*, 163(1):52–64.

Gunning, R. C. and Rossi, H. (2022). *Analytic functions of several complex variables*, volume 368. American Mathematical Society.

Gurrieri, S., Nakabayashi, M., and Wong, T. (2009). Calibration methods of hull-white model. *Available at SSRN 1514192*.

Han, J., Kamber, M., and Pei, J. (2012). Data mining concepts and techniques third edition. *University of Illinois at Urbana-Champaign Micheline Kamber Jian Pei Simon Fraser University*.

Han, J., Pei, J., and Tong, H. (2022). *Data mining: concepts and techniques*. Morgan kaufmann.

Harrison, J. M. and Kreps, D. M. (1979). Martingales and arbitrage in multiperiod securities markets. *Journal of Economic theory*, 20(3):381–408.

Harrison, J. M. and Pliska, S. R. (1981). Martingales and stochastic integrals in the theory of continuous trading. *Stochastic processes and their applications*, 11(3):215–260.

Heaton, J., Polson, N. G., and Witte, J. H. (2016). Deep learning in finance. *arXiv preprint arXiv:1602.06561*.

Heaton, J. B., Polson, N. G., and Witte, J. H. (2017). Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(1):3–12.

Hernandez, A. (2016). Model calibration with neural networks. *Available at SSRN 2812140*.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Horvath, B., Muguruza, A., and Tomas, M. (2021). Deep learning volatility: a deep neural network perspective on pricing and calibration in (rough) volatility models. *Quantitative Finance*, 21(1):11–27.

Howell, D. (1998). *Statistical methods in human sciences*. Wadsworth, New York.

Huang, J., Chai, J., and Cho, S. (2020). Deep learning in finance and banking: A literature review and classification. *Frontiers of Business Research in China*, 14:1–24.

Huang, Y., Murphey, Y. L., and Ge, Y. (2013a). Automotive diagnosis typo correction using domain knowledge and machine learning. In *IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*.

Huang, Y., Murphey, Y. L., and Ge, Y. (2013b). Automotive diagnosis typo correction using domain knowledge and machine learning. In *2013 IEEE symposium on computational intelligence and data mining (CIDM)*, pages 267–274. IEEE.

Hull, J. and White, A. (1993). One-factor interest-rate models and the valuation of interest-rate derivative securities. *Journal of Financial and Quantitative Analysis*, 28(2):235–254.

Hull, J. and White, A. (1994). Numerical procedures for implementing term structure models ii: Two-factor models. *Journal of Derivatives*, 2(2):37–48.

Hull, J. and White, A. (2001). The general hull–white model and supercalibration. *Financial Analysts Journal*, 57(6):34–43.

Hull, J. C., Sokol, A., and White, A. (2014). Modeling the short rate: the real and risk-neutral worlds. *Rotman School of Management Working Paper*, (2403067).

Jacovides, A. (2008). *Forecasting Interest Rates from the Term Structure: Support Vector Machines Vs Neural Networks*. PhD thesis, Citeseer.

Jang, J. H., Yoon, J., Kim, J., Gu, J., and Kim, H. Y. (2021). Deepoption: A novel option pricing framework based on deep learning with fused distilled data from multiple parametric methods. *Information Fusion*, 70:43–59.

Jaro, M. (1989). Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. *J Am Stat Assoc*, 84(406):414–420.

Johnson, R. and Wichern, D. (2014). *Applied Multivariate Statistical Analysis*, volume 4. Hall, Upper Saddle River, NJ, USA.

Kachuee, M., Karkkainen, K., Goldstein, O., Darabi, S., and Sarrafzadeh, M. (2020). Generative imputation and stochastic prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3):1278–1288.

Kaelbling, L. P., Littman, M. L., and Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285.

Karami, A. and Johansson, R. (2014). Choosing dbscan parameters automatically using differential evolution. *International Journal of Computer Applications*, 91(7):1–11.

Karatzas, I. and Shreve, S. (1991). *Stochastic calculus and Brownian motion*. Springer-Verlag, New York.

Karthikeyan, T. and Vembandasamy, K. (2015a). A novel algorithm to diagnosis type ii diabetes mellitus based on association rule mining using mpso-lssvm with outlier detection method. *Indian Journal of Science and Technology*, 8(S8).

Karthikeyan, T. and Vembandasamy, K. (2015b). A novel algorithm to diagnosis type ii diabetes mellitus based on association rule mining using mpso-lssvm with outlier detection method. *Indian Journal of Science and Technology*, 8(S8):310–320.

Keil, F. C. (2006). Explanation and understanding. *Annu. Rev. Psychol.*, 57:227–254.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kladívko, K. (2007). Maximum likelihood estimation of the cox-ingersoll-ross process: the matlab implementation. *Technical Computing Prague*, 7(8).

Köhler, H., Link, S., and Zhou, X. (2015). Possible and certain sql key. *PVLDB*, 8(12):1978–1989.

Le, V. B. T., Link, S., and Memari, M. (2012). Schema- and data-driven discovery of sql keys. *Journal of Computing Science and Engineering*, 6(3):193–204.

Lee, J.-H., Kim, M., and Kwon, H.-C. (2020). Deep learning-based context-sensitive spelling typing error correction. *IEEE Access*, 8:152565–152578.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Sov Phys Dokl*, 10(8):707–710.

Leys, C., Ley, C., Klein, O., Bernard, P., and Licata, L. (2013). Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology*, 49(4):764–766.

Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2008). Isolation forest. In *2008, eighth ieee international conference on data mining*.

Liu, S., Borovykh, A., Grzelak, L. A., and Oosterlee, C. W. (2019). A neural network-based framework for financial model calibration. *Journal of Mathematics in Industry*, 9:1–28.

Lo, C. and Hui, C. (2000). Stress testing model of defaultable bond values. Technical report, Working paper.

Lopes, S. D. and Vázquez, C. (2018). Real-world scenarios with negative interest rates based on the libor market model. *Applied Mathematical Finance*, 25(5-6):466–482.

Madan, D. and Unal, H. (2000). A two-factor hazard rate model for pricing risky debt and the term structure of credit spreads. *Journal of Financial and Quantitative analysis*, 35(1):43–65.

Maier, D. (1978). The complexity of some problems on subsequences and supersequences. *J ACM*, 25(2):322–336.

Manzoni, K. (2002). Modeling credit spreads: An application to the sterling eurobond market. *International Review of Financial Analysis*, 11(2):183–218.

Marr, B. (2016). *Big data in practice: How 45 successful companies used big data analytics to deliver extraordinary results*. John Wiley & Sons.

Mbaye, C. and Vrins, F. (2018). A subordinated cir intensity model with application to wrong-way risk cva. *International Journal of Theoretical and Applied Finance*, 21(07):1850045.

Mbaye, C. and Vrins, F. (2022). Affine term structure models: A time-change approach with perfect fit to market curves. *Mathematical Finance*, 32(2):678–724.

McKinney, W. (2010). Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, volume 445, pages 51–56. Austin, TX.

Merton, R. C. (1974). On the pricing of corporate debt: The risk structure of interest rates. *The Journal of finance*, 29(2):449–470.

Miller, J. (1991). Reaction time analysis with outlier exclusion: Bias varies with sample size. *The Quarterly Journal of Experimental Psychology*, 43(4):907–912.

Morris, R. and Cherry, L. L. (1975). Computer detection of typographical errors. *IEEE Transactions on Professional Communication*, (1):54–56.

Moubayed, A., Injadat, M., Shami, A., and Lutfiyya, H. (2020). Dns typo-squatting domain detection: A data analytics & machine learning based approach.

Müllner, D. (2011). Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*.

Neysiani, B. S. and Babamir, S. M. (2019). Automatic interconnected lexical typo correction in bug reports of software triage systems.

Ng, A. et al. (2011). Sparse autoencoder. *CS294A Lecture notes*, 72(2011):1–19.

Norman, J. P. (2009). Real world interest rate modelling with the bgm model. *Available at SSRN 1480174*.

Oh, K. J. and Han, I. (2000). Using change-point detection to support artificial neural networks for interest rates forecasting. *Expert systems with applications*, 19(2):105–115.

Orlando, G., Mininni, R. M., and Bufalo, M. (2019). Interest rates calibration with a cir model. *The Journal of Risk Finance*.

Pal, S. K. and Mitra, S. (1992). Multilayer perceptron, fuzzy sets, classifiaction.

Pandya, U., Mistry, V., Rathwa, A., Kachroo, H., and Jivani, A. (2020). 2dbscan with local outlier detection. In *Ambient Communications and Computer Systems: RACCCS 2019*, pages 255–263. Springer.

Pironneau, O. (2019). Calibration of heston model with keras.

Pyle, D. (1999). *Data preparation for data mining*. Morgan Kaufmann.

Ras, G., van Gerven, M., and Haselager, P. (2018). Explanation methods in deep learning: Users, values, concerns and challenges.

Rezapour, M. (2019). Anomaly detection using unsupervised methods: credit card fraud case study. *International Journal of Advanced Computer Science and Applications*, 10(11).

Robinson, D. (2003). *An Introduction to Abstract Algebra*. Walter de Gruyter, Berlin.

Rosenblum, H., Strongin, S., et al. (1983). Interest rate volatility in historical perspective. *Economic Perspectives*, 7(1):10–19.

Russo, V. and Fabozzi, F. J. (2017). Calibrating short interest rate models in negative rate environments. *The Journal of Derivatives*, 24(4):80–92.

Russo, V., Giacometti, R., and Fabozzi, F. (2020). Closed-form solution for defaultable bond options under a two-factor gaussian model for risky rates modeling. *The Journal of Derivatives*, 27(4):54–69.

Russo, V. and Torri, G. (2019). Calibration of one-factor and two-factor hull–white models using swaptions. *Computational Management Science*, 16(1):275–295.

Sainani, K. L. (2015). Dealing with missing data. *Pm&r*, 7(9):990–994.

Savarimuthu, N. and Karesiddaiah, S. (2021). An unsupervised neural network approach for imputation of missing values in univariate time series data. *Concurrency and Computation: Practice and experience*, 33(9):e6156.

Schelter, S., Lange, D., Schmidt, P., Celikel, M., Biessmann, F., and Grafberger, A. (2018). Automating large-scale data quality verification. *Proceedings of the VLDB Endowment*, 11(12):1781–1794.

Schlenkrich, S. (2012). Efficient calibration of the hull white model. *Optimal Control Applications and Methods*, 33(3):352–362.

Schönbucher, P. (1999). A tree implementation of a credit spread model for credit derivatives. *Available at SSRN 240868*.

St, L., Wold, S., et al. (1989). Analysis of variance (anova). *Chemometrics and intelligent laboratory systems*, 6(4):259–272.

Thang, T. M. and Kim, J. (2011). The anomaly detection by using dbscan clustering with multiple parameters. In *2011 International Conference on Information Science and Applications*, pages 1–5. IEEE.

Tibshirani, R., Walther, G., and Hastie, T. (2001). Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423.

Vasicek, O. (1977a). An equilibrium characterization of the term structure. *Journal of financial economics*, 5(2):177–188.

Vasicek, O. A. (1977b). An equilibrium characterization of the term structure. *Journal of Financial Economics*, 5(2):177–188.

Vela, D. (2013). Forecasting latin-american yield curves: An artificial neural network approach. *Borradores de Economía; No. 761*.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors

(2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Widder, D. V. (2015). *Laplace transform (PMS-6)*, volume 61. Princeton university press.

Wright, J. H. (2006). The yield curve and predicting recessions.

Xu, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., Liu, Y., Zhao, Y., Pei, D., Feng, Y., et al. (2018). Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 world wide web conference*, pages 187–196.

Xu, R. and Li, S. (2009). A tree model for pricing credit spread options subject to equity, and market risk. In *2009 ISECS International Colloquium on Computing, Communication, Control, and Management*, volume 2, pages 46–50. IEEE.

Zaki, M. J. and Meira, W. (2014). *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press.

Zhou, P., Li, M., Huang, J., and Fang, H. (2014). Research on database schema comparison of relational databases and key-value stores. *Advanced Materials Research*, 1049-1050:1860–1863.