Stabilized Numerical Methods for Stochastic Differential Equations driven by Diffusion and Jump-Diffusion Processes

THÈSE N^{O} 6771 (2015)

PRÉSENTÉE LE 22 OCTOBRE 2015
À LA FACULTÉ DES SCIENCES DE BASE
CHAIRE D'ANALYSE NUMÉRIQUE ET MATHÉMATIQUES COMPUTATIONNELLES
PROGRAMME DOCTORAL EN MATHÉMATIQUES

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Adrian BLUMENTHAL

acceptée sur proposition du jury:

Prof. T. Mountford, président du jury
Prof. A. Abdulle, directeur de thèse
Prof. E. Buckwar, rapporteuse
Prof. G. Lord, rapporteur
Prof. M. Picasso, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2015

Ambition, Passion,
Commitment and Dedication.

To my beloved family.

Acknowledgements

There are many people that I would like to say thank you for making this thesis possible.

First of all, I would to thank my family. Thank you so much Mum, Dad and Tamara for supporting me over the years and for always being there for me! I am also very grateful to all my other relatives. You are all a great inspiration for me.

Next, I would like to express my enormous gratitude to my thesis advisor Prof. Assyr Abdulle for his great guidance and for allowing me to pursue my research interests in stochastic calculus and numerical analysis. I appreciated the rich discussions over the years very much and it was a great pleasure to collaborate with you!

Then, I would like to thank all the members of the jury, who accepted to read my work and gave me valuable comments. Thank you Prof. Thomas Mountford for presiding the jury and thank you Prof. Eveyln Buckwar, Prof. Gabriel Lord and Prof. Marco Picasso for being part of the jury of my PhD examination.

Further, I am grateful to many people at the EPFL. Thank you Martin for being an amazing office mate and especially for being a good friend! Thank you Virginie, Annick, Ariane, Corinne, Catherine and Marie for your great support and the enjoyable discussions during my years at EPFL! Many thanks also to my colleagues Ondrej, Orane, Timothée, Yun, Andrea, Patrick, Antti, Gilles, Kostas and all the other people from MATHICSE. It has been a great pleasure to work with you all!

I would also like to make use of this opportunity to thank all of my friends. Thank you for being such great friends to me! I appreciate it very much.

Finally, I would like to thank everyone that has inspired me over the last few years!

Lausanne, August 2015 Adrian Blumenthal

Abstract

Stochastic models that account for sudden, unforeseeable events play a crucial role in many different fields such as finance, economics, biology, chemistry, physics and so on. That kind of stochastic problems can be modeled by stochastic differential equations driven by jump-diffusion processes. In addition, there are situations, where a stochastic model is based on stochastic differential equations with multiple scales. Such stochastic problems are called stiff and lead for classical explicit integrators such as the Euler-Maruyama method to time stepsize restrictions due to stability issues. This opens the door for stabilized explicit numerical methods to efficiently tackle such situations.

In this thesis we introduce first a stabilized multilevel Monte Carlo method for stiff stochastic differential equations. Using S-ROCK methods we show that this approach is very efficient for stochastic problems with multiple scales, but also for nonstiff problems with a significant noise part. Further, we present an improved version of the stabilized multilevel Monte Carlo method by considering S-ROCK methods with a higher weak order of convergence.

Then we extend the S-ROCK methods to jump-diffusion processes. We study in detail the strong order of convergence of the newly introduced methods and we discuss the corresponding mean square stability domains.

In the next part we present the multilevel Monte Carlo method for jump-diffusion processes. We state and prove a theorem that indicates the computational cost required to achieve a certain mean square accuracy. In the numerical section we compare the multilevel Monte Carlo approach to two variance reduction techniques, the antithetic and the control variates. We also show how the S-ROCK method for jump-diffusion processes, introduced in this thesis, can be used to create a stabilized multilevel Monte Carlo method for jump-diffusions that handles stiffness and considers the inclusion of jumps at the same time.

Finally, we propose in this thesis a variable time stepping algorithm that uses S-ROCK methods to approximate weak solutions of stiff stochastic differential equations. A rigorous analytical study is carried out to derive a computable leading term of the time discretization error and an adaptive algorithm is suggested that adapts the time grid and adjusts the number of stages of the S-ROCK method simultaneously.

Keywords: *Stochastic Differential Equations, Diffusion Processes, Jump-Diffusion Processes, Monte Carlo Method, Variance Reduction Techniques, Multilevel Monte Carlo Method, Stiffness, Stability, S-ROCK Methods, Variable Time Stepping.*

Résumé

Des modèles stochastiques qui prennent en compte des événements soudains ou imprévisibles gagnent en importance dans différentes branches comme, par exemple, en finance, en économie, en biologie, en chimie ou encore en physique. Ces problèmes stochastiques peuvent être modélisés par des équations différentielles stochastiques avec des sauts ou par des équations différentielles stochastiques multi-échelles. Ces problèmes sont appelés raides et leur résolution par des méthodes explicites classiques, comme le schéma d'Euler-Maruyama, nécéssite une réduction du pas de temps à cause des problèmes liés à la stabilité. Une solution est représentée par des méthodes explicites stabilisées.

Premièrement, dans cette thèse, une nouvelle méthode multilevel Monte Carlo stabilisée est introduite pour des équations différentielles stochastiques raides. En utilisant des schémas S-ROCK il est montré que cette approche est très efficace pour des problèmes raides, ainsi que pour des problèmes ayant un bruit important mais qui ne sont pas forcément raides. Pour conclure cette première partie, une amélioration des méthodes précédentes est proposée en considérant des méthodes S-ROCK avec un ordre de convergence faible élevé.

Deuxièmement, les schémas S-ROCK sont élargis pour des processus stochastiques de diffusion avec sauts. Une analyse complète de la convergence forte et de la stabilité en moyenne quadratique est présentée. Des expériences numériques vérifient les résultats théoriques obtenus précédemment.

Troisièmement, une méthode multilevel Monte Carlo pour des équations différentielles stochastiques avec sauts est présentée. Un théorème donnant le coût computationnel nécessaire pour atteindre une certaine précision est prouvé. Dans la partie numérique la méthode multilevel Monte Carlo pour des processus de diffusion avec sauts est comparée à deux méthodes de réduction de variance. Il est ensuite montré comment le schéma S-ROCK pour des équations différentielles stochastiques avec sauts peut être utilisé pour établir une nouvelle méthode multilevel Monte Carlo stabilisée pour des processus de diffusion avec sauts.

Finalement, un algorithme adaptatif, utilisant les méthodes S-ROCK pour approcher les solutions faibles des équations différentielles stochastiques raides, est proposé. L'algorithme ajuste le pas de temps et le nombre d'étages du schéma S-ROCK simultanément.

Mots-clefs : *Équations différentielles stochastiques, processus de diffusion, processus de diffusion avec sauts, méthode Monte Carlo, méthodes de réduction de variance, méthode multilevel Monte Carlo, raideur, stabilité, méthodes S-ROCK, pas de temps adaptatif.*

Contents

Acknowledgements **i**

Abstract **iii**

1 Introduction **1**
- 1.1 Main Contributions 4

2 Stochastical and Numerical Background **7**
- 2.1 Stochastic Differential Equations driven by Jump-Diffusion Processes 7
  - 2.1.1 Diffusion Processes 8
  - 2.1.2 Jump-Diffusion Processes 11
- 2.2 Numerical Schemes 23
  - 2.2.1 Strong and Weak Convergence 24
  - 2.2.2 Stability 25
  - 2.2.3 Euler-Maruyama Method 27
  - 2.2.4 S-ROCK Method 28
- 2.3 Monte Carlo Techniques 29
  - 2.3.1 Monte Carlo Method 30
  - 2.3.2 Variance Reduction Techniques 33
  - 2.3.3 Multilevel Monte Carlo Method 35

3 Stabilized Multilevel Monte Carlo Method for Stiff Stochastic Differential Equations **41**
- 3.1 Introduction 41
- 3.2 Preliminaries 43
  - 3.2.1 Numerical Schemes 44
  - 3.2.2 Stability of Numerical Methods 46
- 3.3 Multilevel Monte Carlo Method for Stiff SDEs 48
  - 3.3.1 Standard Multilevel Monte Carlo Method 48
  - 3.3.2 Stabilized Multilevel Monte Carlo Method 52
- 3.4 Improved Stabilized Multilevel Monte Carlo Method for Stiff SDEs 55
- 3.5 Numerical Examples 57
  - 3.5.1 Linear Stochastic Differential Equation 57
  - 3.5.2 Nonlinear Stochastic Differential Equation 59
  - 3.5.3 Space-discretized Stochastic Parital Differential Equation 62

3.5.4 Comparison Improved Stabilized MLMC vs Stabilized MLMC vs Standard MLMC 64
3.6 Conclusion 67

4 S-ROCK Methods for Jump-Diffusion Processes 69
4.1 Introduction 69
4.2 Preliminaries 70
4.2.1 Numerical Schemes 71
4.3 Strong Convergence 73
4.3.1 Strong Convergence of S-ROCK1-JD 74
4.3.2 Strong Convergence of PIROCK-JD 87
4.4 Mean Square Stability 91
4.4.1 Mean Square Stability Domain of S-ROCK1-JD 92
4.4.2 Mean Square Stability Domain of PIROCK-JD 93
4.4.3 Illustration of the Stability Regions of S-ROCK1-JD 94
4.5 Numerical Experiments 102
4.5.1 Numerical Study of the Strong Convergence 102
4.5.2 Comparison S-ROCK1-JD and Euler-Maruyama for Jump-Diffusions 106
4.6 Conclusion 110

5 Multilevel Monte Carlo Method for Stochastic Differential Equations driven by Jump-Diffusion Processes 113
5.1 Introduction 113
5.2 Preliminaries 115
5.2.1 Numerical Schemes 116
5.3 Multilevel Monte Carlo Method for Jump-Diffusion Processes 118
5.3.1 Monte Carlo Method for Jump-Diffusions 118
5.3.2 Multilevel Monte Carlo Method for Jump-Diffusions 120
5.3.3 Complexity Theorem for Jump-Diffusions 125
5.4 Numerical Examples 133
5.4.1 The Merton and the Kou model 134
5.4.2 Two Variance Reduction Techniques 135
5.4.3 Numerical Results 136
5.5 Stabilized Multilevel Monte Carlo Method for Jump-Diffusion Processes 139
5.5.1 Stabilized Monte Carlo Method for Jump-Diffusions 140
5.5.2 Stabilized Multilevel Monte Carlo Method for Jump-Diffusions 141
5.6 Conclusion 143

6 S-ROCK Method with Variable Time Stepping for Stiff Stochastic Differential Equations 145
6.1 Introduction 145
6.2 Preliminaries 146
6.2.1 Numerical Schemes 146

6.2.2 Mean Square Stability 148
6.3 Variable Time Stepping S-ROCK Method Using A Posteriori Error Control 150
6.3.1 Derivation of a Computable Leading Term of the Time Error 151
6.3.2 Adaptive Algorithm 163
6.4 Numerical Experiments 166
6.4.1 Nonstiff Stochastic Differential Equation 166
6.4.2 Stiff Stochastic Differential Equation 168
6.5 Conclusion 171

7 Conclusion and Outlook 173
7.1 Conclusion 173
7.2 Outlook 174

Bibliography 183

Curriculum Vitae 185

1 Introduction

There is a large amount of scenarios in nature, in science, in engineering, in industry, in the daily life and so on that can be studied and analyzed using mathematical models. Many problems can be modeled by so-called differential equations. These are equations that connect certain functions (e.g. some physical quantity) with their derivatives, i.e. for instance their variation over time. In biology there is for instance a population growth model, that shows how a population can evolve over time. In physics the Kepler problem (a particular case of the two-body problem) models the interaction of two bodies. Chemical reactions in chemistry can also be described by differential equations. Applications modeled by differential equations are deterministic. But what happens if there is some component of the model that cannot entirely or partly be determined? What if there is some external source that cannot be quantified but that affects the underlying model? What if the problem one wants to model is random? In all that kind of situation ordinary differential equations are normally not enough. To include randomness in the model one can add some noise component which leads to stochastic differential equations.

Mathematical models based on stochastic differential equations can be used in many different areas such as finance (to model for instance the price of a share), in biology (e.g. the modeling of the population of genes), in chemistry (for instance to model the Michaelis-Menten system) and so on. It is also common to take a deterministic model and to add a stochastic component to make it more realistic. For some models based on stochastic differential equations it is possible to derive an analytical solution, as this is for instance the case for the Black-Scholes model in finance. However, there are many more sophisticated stochastic models, that do not admit an analytical expression of the solution. This calls for numerical methods that can be used to approximate the solution of that kind of stochastic models. There are various numerical schemes for stochastic differential equations with different properties. Often one is interested in the convergence (behavior when the the time endpoint is fixed and the time stepsize tends to zero) and in the stability (long-term behavior of the numerical scheme for a fixed time stepsize) of the numerical method. Depending on the underlying problem one chooses a suitable numerical integrator to find an appropriate solution.

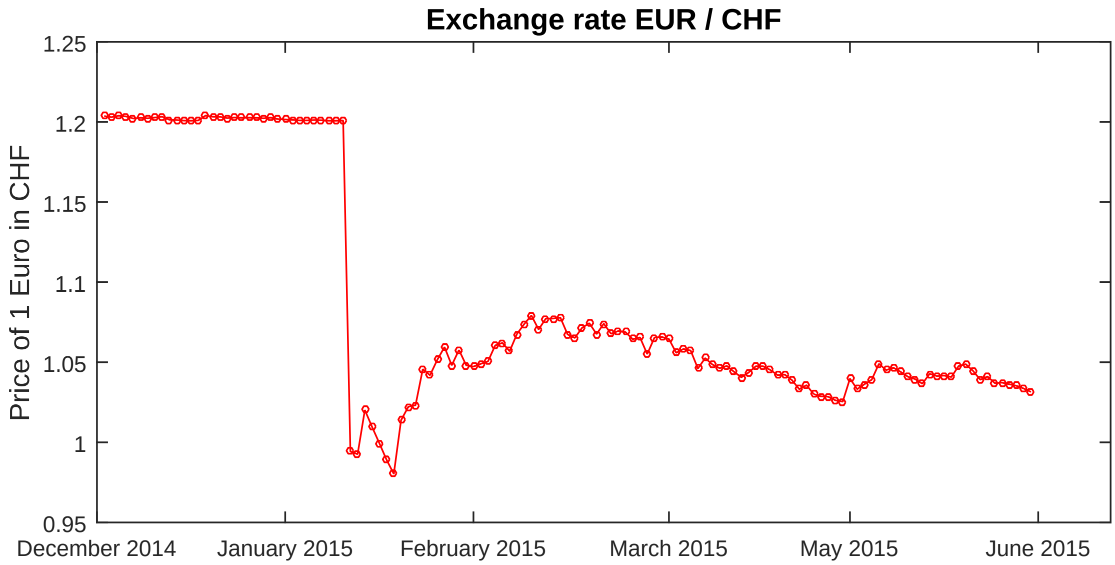

Figure 1.1: Behavior of the exchange rate of EUR/CHF from December 1st, 2014 to June 1st, 2015 (*data source: yahoo finance*).

We look now at an example from finance. In Figure 1.1 the behavior of the exchange rate EUR/CHF between start of December 2014 and the start of June 2015 is illustrated. It looks like most of the problem can be modeled by a stochastic differential equations, which is driven by a continuous diffusion process. However, we discover that there is one part, a singularity, that most likely cannot be modeled by a continuous process. We give now first an explanation of the behavior of the exchange rate observed in January 2015. In 2011 the Swiss National Bank introduced an exchange rate peg. In fact, since the people in charge at the Swiss National Bank considered the Swiss franc to be too strong (a strong Swiss franc is not good for the Swiss export industry, which is an important part of the Swiss economy), the Swiss National Bank insisted to keep the price for 1 Euro at least at 1.20 CHF. However, on January 15th the Swiss National Bank announced, without initial warning, that they no longer peg the Euro to the Swiss franc. As it can be seen in Figure 1.1 this led to a huge drop of the exchange rate over one day only. In fact, on January 15th the price for 1 Euro was 1.2009 CHF. Already the next day the price dropped to only 0.9943 CHF which corresponds to a drop of more than 17% over a single day. To capture this big variation over a very short time period we have to add jumps to our modeling procedure.

There are the so-called stochastic differential equations driven by jump-diffusion processes that can account for sudden events that cannot be predicted and that have a huge impact over a short time interval (like the one of Figure 1.1). In finance, for instance, such a behavior can frequently be observed and is related e.g. to the announcement of important news of big companies or governments, to environmental effects such as hurricanes and tsunamis. Also terrorist attacks or political incidents can be the reason of a jump in the underlying problem. Applications based on stochastic differential equations with jumps can be found in many different fields and they are on the rise. As for the stochastic differential equations driven by diffusion processes, it is often not possible to find an exact solution of a stochastic problem and one has to consider numerical schemes. In this thesis we present different efficient numerical techniques that can be used for problems based on jump-diffusion processes.

Another issue that we address in this thesis is the solution of problems related to stochastic differential equations with multiple scales. Figure 1.2, which shows the numerical solution of the heat equation with multiplicative noise, illustrates such a case. The heat equation is characterized by a stochastic partial differential equation. Fixing a spatial discretization and using the method of lines, a (possibly large) system of stochastic differential equations is obtained. Due to stability issues this stochastic problem is subject to some stepsize restriction for classical numerical integrators such as the Euler-Maruyama method. We call such problems stiff. If the stability constraint is not met, the successful application of the numerical scheme cannot be guaranteed. In this thesis we study explicit stochastic orthogonal Runge-Kutta Chebyshev methods that have an extended stability domain, and thus, are very efficient if it comes to stiffness. The approximation of the heat equation in Figure 1.2 has been produced using a stabilized explicit numerical integrator. Depending on the stiffness of the problem, and thus, on the time stepsize the same is difficult to realize with a classical explicit numerical scheme.

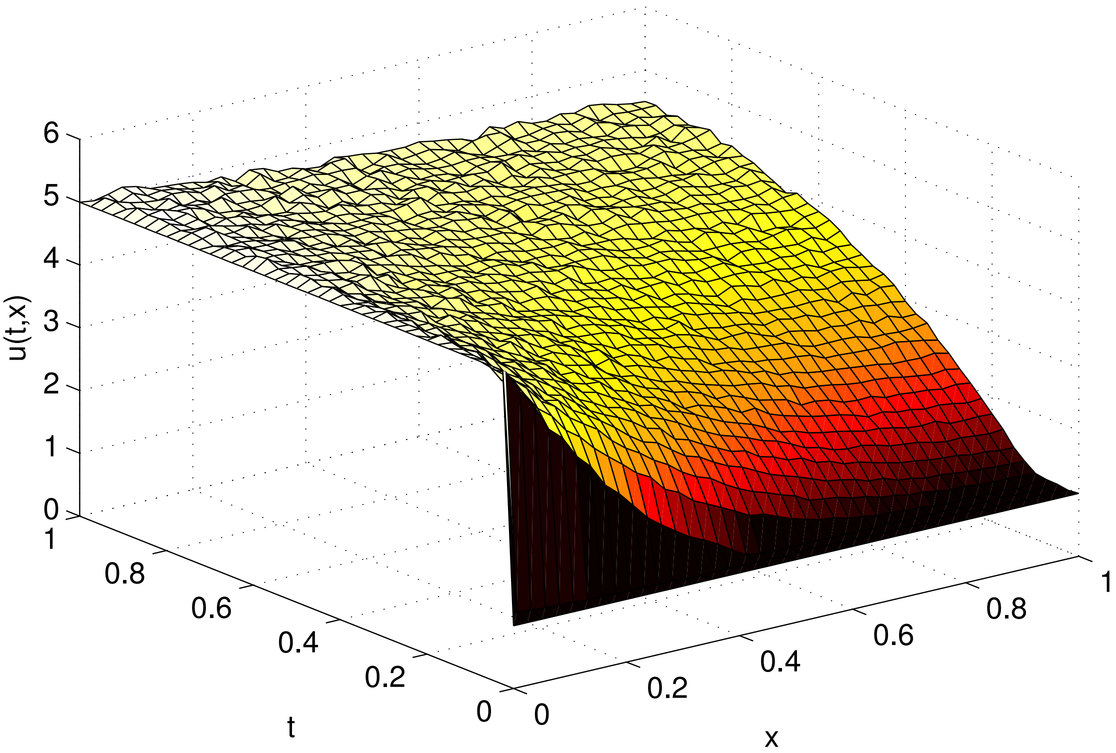

Figure 1.2: Numerical approximation of the heat equation with multiplicative noise using a space discretization of $\Delta x = 1/40$ and a time discretization of $\Delta t = 1/40$.

Stiff stochastic problems arise in many different areas such as financial engineering, biology, chemistry and so. It is possible to tackle stiffness by using drift-implicit integrators, however, this is not always straightforward. For instance for the example presented above, to solve the large system of stochastic differential equations with a implicit solvers results in solving a large linear system at each step, which can be computationally very expensive. In addition, the implementation of drift-implicit solvers can be quite tricky for some problems. In this thesis we pursue another strategy. We present various efficient numerical techniques based on explicit stabilized numerical integrators that can be used to approximate solutions of mean square stable stochastic differential equations with multiple scales.

1.1 Main Contributions

Here, we briefly present the main contributions of this thesis. Note that in each chapter (apart from Chapter 2 that recalls some numerical and stochastical notions) we mention what work has been done. The main aim of this thesis is to provide efficient stabilized numerical techniques to solve stochastic problems. This thesis addresses in particular two issues.

Stiffness. First, there is the issue of stiffness. To find the solution of the expectation of a functional depending on some stochastical process, multilevel Monte Carlo (MLMC) methods appear to be very efficient and lead to an improvement compared to the standard Monte Carlo techniques (see e.g. [46]). However, in Chapter 3 of this thesis we show that for stiff stochastic differential equations (SDE) due to stability issues the Euler-Maruyama approach cannot use all the levels, and thus, its performance deteriorates. We present a stabilized multilevel Monte Carlo method that is based on stabilized numerical integrators, the so-called S-ROCK methods (see [12, 10, 15]), which enables us to use all the levels of the multilevel Monte Carlo approach, and thus, this method prevails over the standard MLMC method. Another finding of this thesis is that even for nonstiff problems that have a significant noise component the newly introduced stabilized MLMC method performs better. In Chapter 3 we suggest also an improved stabilized MLMC method, which improves the performance even further.

Furthermore, in Chapter 4 we take the existing S-ROCK methods and we extend two versions of these schemes [12, 14] to account for jumps (hence, this is also part of the second issue). We study rigorously the strong convergence of the new S-ROCK methods for stochastic differential equations driven by jump-diffusion processes. The mean square stability is also analyzed in detail and the stability domains are characterized.

Finally, by using S-ROCK methods we extend in Chapter 6 a variable time stepping algorithm for the weak solution of SDEs [96], so that it can also deal with stiffness. We suggest an adaptive algorithm that adjusts the time grid and the number of stages of the S-ROCK integrator (to account for the stiffness) simultaneously.

Jump-Diffusions. Second, there is the issue of including jumps when modeling stochastic problems by stochastic differential equations. As we have seen above, in Chapter 4 we extend the S-ROCK method to jump-diffusions.

Moreover, in Chapter 5 we present the multilevel Monte Carlo method that can be used for stochastic differential equations driven by jump-diffusion processes. We propose how to deal with the jump terms and we state and prove a complexity theorem, that indicates how much computational cost is required to achieve a certain mean square accuracy. In the numerical part the MLMC method for jump-diffusions is compared to two variance reduction techniques, namely the antithetic variates and the control variates.

In Chapter 5 we show also how the results from that chapter can be combined with the findings from Chapter 4 to create a stabilized multilevel Monte Carlo method for jump-diffusion processes. This covers again the two main issues of this thesis, the stiffness and jump-diffusions.

Note that in each chapter various numerical experiments are carried out to corroborate the theoretical findings. In Chapter 7 we recapitulate the main findings of this thesis and we give an outlook how future research could look like.

2 Stochastical and Numerical Background

In this section we present a few topics from stochastic calculus and from numerical analysis. The background delivered here represents the backbone of the following sections. First, stochastic differential equations driven by either diffusion processes or jump-diffusion processes are discussed. We show why adding jumps to diffusion processes can be essential for many applications in finance, chemistry, physics, biology, and so on. Next, we introduce the numerical approximation methods used most in this thesis, namely the Euler-Maruyama method and the S-ROCK methods. We also present some of its properties such as the strong and weak convergence as well as the stability. Finally, we look at the so-called Monte Carlo techniques. The Monte Carlo method is studied and we show why it can be useful to improve the Monte Carlo approach by using variance reduction techniques such as antithetic variates or control variates. Another particular way to improve the performance of Monte Carlo is using the multilevel Monte Carlo method, which we will introduce in detail.

2.1 Stochastic Differential Equations driven by Jump-Diffusion Processes

Many phenomena in nature, in science, in economics etc. can be modeled by stochastic differential equations, the so-called SDEs. These are differential equations where one has added a stochastic noise. The reasons to include a stochastic term in the model characterization are various. When in a model not all involved quantities can be specified, when an external source has an impact on the model or when there is some randomness in the model (see for instance [81, 103, 49]); these are all situations where it is reasonable to consider a model based on stochastic differential equations. In what follows we introduce first SDEs driven by diffusion processes and then we look at SDEs where we have added jumps.

Remark 2.1.1. *Throughout this thesis, unless stated otherwise, we work with Itô integrals. There is another type of stochastic integral, the so-called Stratonovich integral, which follows the standard rules of calculus. However, there are many problems in finance, chemistry, biology etc. that are modeled in Itô form, and thus, we prefer to apply the Itô setting in this thesis. Note* *that using a simple transformation one can change any Itô integral into a Stratonovich integral and vice versa (see for instance [66]).*

2.1.1 Diffusion Processes

Before we explain what it is meant by a stochastic differential equation, we need to describe what one understands as random noise. A common way to include randomness in a model is to use a Brownian motion (see e.g. [95]), which is a stochastic process with certain properties. The Scottish botanist Robert Brown observed in 1827 that the movement of particles of pollen grains of a plant in water is random (see [24]). The first to characterize this random behavior of the particles in mathematical terms was at the beginning of the 20th century Louis Bachelier, a French doctoral student of mathematics. Bachelier introduced the definition of a Brownian motion to model the price of shares and options at the stock market (see [19]). In what follows we consider, unless stated otherwise, a probability space $(\Omega, \mathscr{F}, \mathbb{P})$, where Ω is a sample space, $\mathscr{F}$ a sigma-algebra and $\mathbb{P}$ a probability measure (see [95]).

Definition 2.1.2 (Brownian motion)**.** *A Brownian motion is a stochastic process* $(W(t))_{t\geq 0}$ *that satisfies:*

(i) Stationay distribution: *For any* $0 \leq s < t$

$$W(t) - W(s) \sim \mathcal{N}\left(0, \sigma^2 (t-s)\right),$$

i.e. the increment $W(t) - W(s)$ *is normally distributed with mean* $\mathbb{E}[W(t) - W(s)] = 0$ *and variance* $Var(W(t) - W(s)) = \sigma^2 (t-s)$.

(ii) Independent increments: *For any* $0 \leq s_1 < t_1 \leq s_2 < t_2$ *the two increments* $W(t_1) - W(s_1)$ *and* $W(t_2) - W(s_2)$ *are independent.*

(iii) Continuity: *For all* $\omega \in \Omega$ *except for a null set the sample path* $t \mapsto W(\omega, t)$ *is continuous.*

Note that if $\sigma^2 = 1$ *and* $W(0) = 0$*, then one speaks of a standard Brownian motion, which is also known in the literature as Wiener process (see e.g. [54, 94, 104]).*

Figure 2.1 shows an example of a sample path of a standard Brownian motion over the time interval $[0, 1]$.

We have now all the ingredients to introduce the notation of a stochastic differential equation. Let $(X(t))_{t\in[0,T]}$ be a stochastic process. Suppose this stochastic process is characterized by the following stochastic differential equation

$$\begin{cases} \mathrm{d}X(t) &= f(X(t))\,\mathrm{d}t + \sum_{r=1}^{m} g^r(X(t))\,\mathrm{d}W^r(t), \ 0 < t \leq T, \\ X(0) &= X_0, \end{cases} \tag{2.1}$$

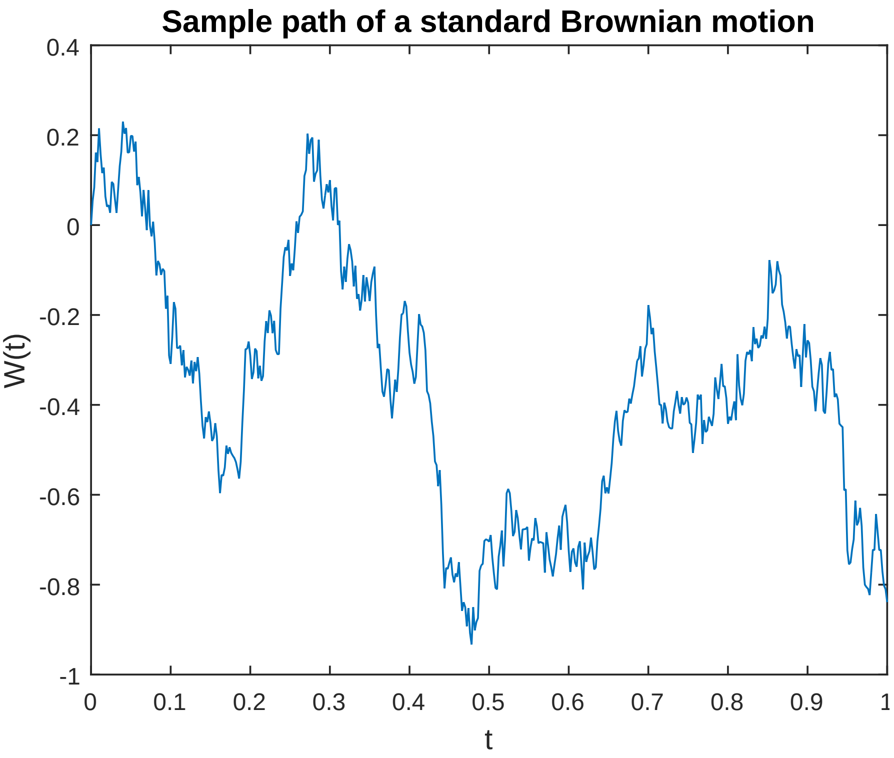

Figure 2.1: Simulation of a standard Brownian motion over the time interval $[0,1]$.

where $X(t) \in \mathbb{R}^d$, $f:\mathbb{R}^d \to \mathbb{R}^d$ and $g^r:\mathbb{R}^d \to \mathbb{R}^d$ for all $r = 1,2,\dots,m$. The initial value of the stochastic process is given by X_0. The processes $(W^r(t))_{t\in[0,T]}$ $(r = 1,2,\dots,m)$ are m independent standard Brownian motions. The function f describes the drift and the functions g^r the diffusion.

Remark 2.1.3. *Throughout this thesis we work, unless stated otherwise, with autonomous stochastic differential equations, i.e. the drift and the diffusion terms do not depend explicitly on the time variable. Note that any non-autonomous SDE can be modified to an autonomous one by a simple transformation (see e.g. [52]).*

We emphasize that (2.1) is simply a notation, which in facts means that

$$X(t) = X(0) + \int_0^t f(X(s))\,\mathrm{d}s + \sum_{r=1}^{m} \int_0^t g^r(X(s))\,\mathrm{d}W^r(s),$$

where $\int_0^t f(X(s))\,\mathrm{d}s$ is an integral in the Lebesgue sense and where $\int_0^t g^r(X(s))\,\mathrm{d}W^r(s)$ are stochastic integrals with respect to a Brownian motion (see [95]). In what follows we assume the standard Lipschitz and linear growth conditions on f and g^r such that the existence and uniqueness of a solution is ensured (see [66]).

A model that is used a lot in this thesis is the so-called Black-Scholes model, which was first mentioned by Samuelson in 1955 and it became popular in the financial world due to the paper of Fischer Black and Myron Scholes (see [22]), which appeared in 1973.

Definition 2.1.4 (Black-Scholes model)**.** *The Black-Scholes model is characterized by the stochastic differential equation*

$$\begin{cases} \mathrm{d}X(t) &= \mu X(t)\mathrm{d}t + \sigma X(t)\mathrm{d}W(t), \;\; 0 < t \leq T, \\ X(0) &= X_0, \end{cases} \tag{2.2}$$

where $X(t) \in \mathbb{R}$, $(W(t))_{t\in[0,T]}$ *is a standard Brownian motion and* X_0 *a given initial condition.*

One can derive an exact solution of the Black-Scholes model.

Proposition 2.1.5. *The Black-Scholes model specified by* (2.2) *admits the exact solution*

$$X(t) = X_0 \exp\left\{\left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W(t)\right\}.$$

Proof. Consider the logarithm of $X(t)$, i.e. $\ln(X(t))$. Applying Itô's lemma (see [70]) yields

$$\begin{aligned} \mathrm{d}\ln(X(t)) &= \mu X(t)\tfrac{1}{X(t)}\mathrm{d}t + \tfrac{\sigma^2 X(t)^2}{2}\left(-\tfrac{1}{X(t)^2}\right)\mathrm{d}t + \sigma X(t)\tfrac{1}{X(t)}\mathrm{d}W(t) \\ &= \left(\mu - \tfrac{\sigma^2}{2}\right)\mathrm{d}t + \sigma\mathrm{d}W(t). \end{aligned}$$

Integrating on both sides over the interval $[0,t]$ results in

$$\ln(X(t)) - \ln(X(0)) = \left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W(t).$$

Taking the expectation and rearranging the terms we obtain the desired result

$$X(t) = X_0 \exp\left\{\left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W(t)\right\}.$$

□

In Figure 2.2 we illustrate a sample path of the solution of the Black-Scholes model with time endpoint $T = 1$, initial condition $X_0 = 1$, drift parameter $\mu = 0.05$ and diffusion parameter $\sigma = 0.2$.

The Black-Scholes model is frequently used in finance to price financial derivatives, such as options and futures (see for instance [49, 70, 87]). Note that the drift coefficient often represents the risk-free interest rate of the underlying risk-free asset and the diffusion is given by the volatility of the problem. There are many more sophisticated models in finance, but the Black-Scholes model remains popular due to its simplicity (see [61]). As we have seen in Proposition 2.1.5 the Black-Scholes model admits an exact solution and it is therefore also very

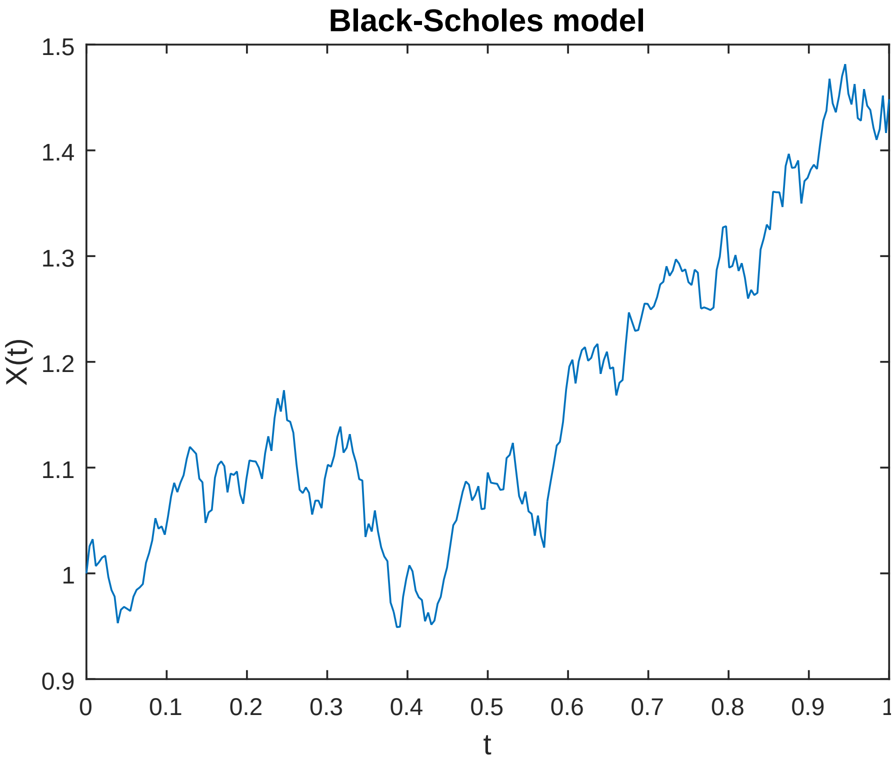

Figure 2.2: Simulation of the solution of the Black-Scholes model over the time interval $[0,1]$ with drift $\mu = 0.05$, diffusion $\sigma = 0.2$ and initial condition $X_0 = 1$.

useful if one has to test a numerical approach. Note that there are many more models based on stochastic differential equations driven by diffusion processes, but they do not admit an analytical solution. Hence numerical methods are required to find an adequate approximation. More on this in Section 2.2.

Models based on stochastic differential equations driven by diffusion processes are useful when one has to model continuous sample paths, such as for instance in option pricing where one assumes a complete market. However, often the data available is only in empirical form. Papers like (see [18]) suggest that diffusion processes are not sufficient to properly model many problems finance. In the next section we introduce jump-diffusion processes which can incorporate discontinuities by the means of so-called jumps.

2.1.2 Jump-Diffusion Processes

In this section we discuss stochastic differential equations driven by jump-diffusion processes. First, we motivate why including jumps leads often to more realistic models. Then we properly define what is meant by a jump in mathematical terms. Finally, by presenting Poisson and compound Poisson processes we introduce jump-diffusion processes and the associated stochastic differential equations. We conclude this section by introducing two jump-diffusion models, namely the Merton model and the Kou model, which are used throughout this thesis.

In many areas such as finance (see e.g. [18, 32, 26]), economics (see e.g. [80]), chemistry (see e.g. [48]), biology (see e.g. [103]), and so on it is necessary to take into account sudden, unforeseeable events that can lead to important variations over a very short time period. Environmental effects such as natural catastrophes like earthquakes (e.g. Haiti), hurricanes (e.g. Katrina) or tsunamis (e.g. Fukushima) can have a huge impact on the stock markets. The same is true for political incidents such as terrorist attacks (e.g. 9/11), the European dept crisis (e.g. Greece) and for changes of the policies and the regulations for banks (e.g. the Basel accords). Also the announcement of important news by governments or by influential companies can lead to a sudden drop or increase of the value of financial instruments. Models based solely on diffusion processes can often not account for this kind of events, whereas models that include jumps can.

Another drawback of models with diffusion processes is that one can use risk-free option pricing to perfectly hedge options and therefore there is no risk left. This is due to the fact that diffusion processes lead to continuous sample paths (see [98]). In reality it is not possible to hedge a financial product without taking any risk. Jumps in a model lead to discontinuous sample paths and a risk-free hedging is no longer possible. Hence models driven by jump-diffusion processes seem to be more realistic for these situations than the ones driven by diffusion processes.

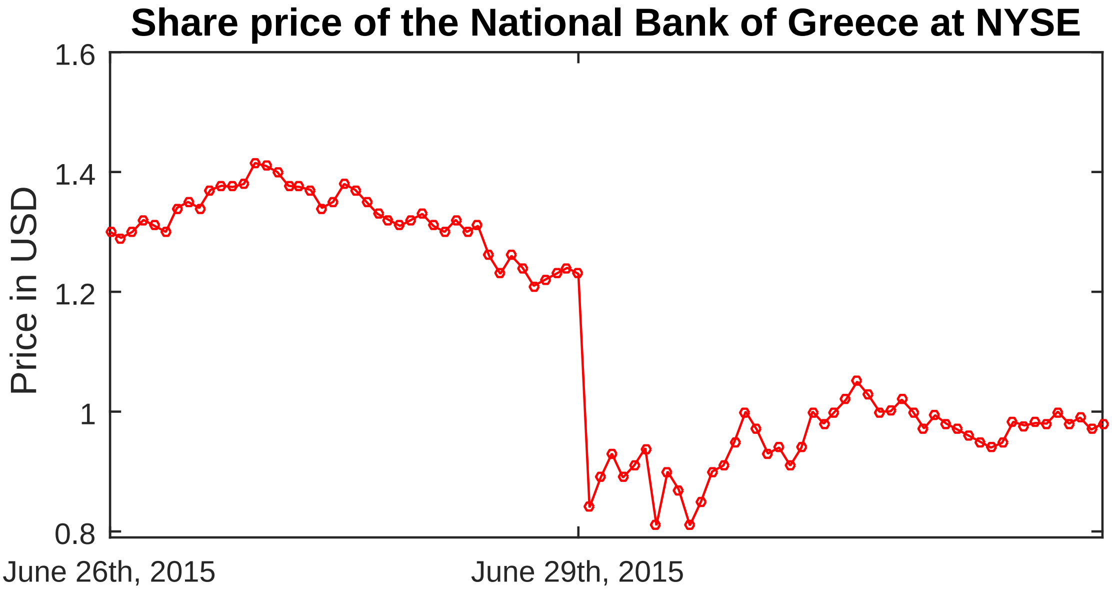

Figure 2.3: Behavior of the share price (in USD) of the National Bank of Greece at the New York stock exchange on June 26th, 2015 and June 29th, 2015 (*data source: yahoo finance*).

In the introduction we have already seen that the decision of the Swiss National Bank to no longer pegging the Euro to the Swiss franc has led to a jump of the exchange rate. We present now another scenario, where one can observe a jump. Figure 2.3 shows the behavior of the share price of the National Bank of Greece at the New York stock exchange during Friday June 26th, 2015 and Monday June 29th, 2015. During the weekend of June 27th and 28th the Greek government has announced that they will hold a referendum about the new debt deal with the international creditors. To avoid that panic about the financial situation spreads in the country, the government decided that the Greek banks remain closed during the entire week of June 29th, 2015. In addition, the cash withdrawals at the banks have been limited. In particular the National Bank of Greece was affected, and thus, the value of the bank dropped by a huge amount over a very short time period as one can see in Figure 2.3. To capture such events in a mathematical approach one has to include jumps in the underlying model.

We continue now by introducing the mathematical meaning of a jump and by describing jump processes as well as jump-diffusion processes.

Jump Processes

Here, we define first what a jump is and then we discuss Poisson and compound Poisson processes, two pure jump processes. To define in mathematical terms a jump, we have to introduce the notion of càdlàg functions.

Definition 2.1.6 (Càdlàg function). *Let T be the time endpoint and consider the real-valued function $f : [0,T] \to \mathbb{R}$. The function f is called a càdlàg function if for any $t \in [0,T]$ the left-hand side and the right-hand side limits exist, i.e.*

$$\begin{aligned} f(t-) &= \lim_{s \overset{<}{\to} t} f(s) < +\infty \\ f(t+) &= \lim_{s \overset{>}{\to} t} f(s) < +\infty, \end{aligned}$$

and if in addition f is right-continuous, i.e.

$$f(t+) = f(t).$$

Remark 2.1.7. *The term càdlàg refers to the French acronym* continu à droite, limite à gauche. *Some references also call càdlàg functions* rcll functions, *which stands for right-continuous with left limits (see [32]).*

Note that any continuous function is a càdlàg function. However, a càdlàg function can also include discontinuities, which we call jumps.

Definition 2.1.8 (Jump). *Let $f : [0,T] \to \mathbb{R}$ be a càdlàg function. A jump at time t is a discontinuity of f at t, i.e. $\left|f(t) - f(t-)\right| > 0$.*

Since we have now properly defined a jump, we discuss the first jump process, the Poisson process.

Definition 2.1.9 (Poisson process). *Let* $(\tau_i)_{i\geq 1}$ *be a sequence of independent exponential random variables with parameter* λ *and let* $T_n = \sum_{i=1}^{n} \tau_i$ *for any* $n \in \mathbb{N}$. *Moreover, let*

$$N(t) = \sum_{n\geq 1} 1_{\{t\geq T_n\}} \quad \forall t \geq 0.$$

Then the stochastic process $(N(t))_{t\geq 0}$ *is called a Poisson process with intensity* λ.

The definition of the Poisson process can be interpreted in the following way. The interarrival times of the jumps, i.e. the time between two jumps, are given by the random variables τ_i. The time of jump number n is described by T_n and $N(t)$ specifies the number of jumps occured in the time interval $[0, t]$. Hence, the increment $N(t) - N(s)$, with $0 \leq s < t$, gives the number of jumps in the time interval $]s, t]$.

The Poisson processes satisfy certain properties. Let $(N(t))_{t\geq 0}$ be a Poisson process with intensity λ. Among the most important properties are the following (for more information about Poisson processes see e.g. [95, 32]):

(i) *Distribution:* For any $t \geq 0$, the random variable $N(t)$ is characterized by a Poisson distribution with intensity λt.

(ii) *Stationary and independent increments:* Let $0 \leq t_1 < t_2 \leq t_3 < t_4$. Then the distribution of the increment $N(t_2) - N(t_1)$ is stationary, i.e. it only depends on $t_2 - t_1$ and not on t_1. Furthermore the increments $N(t_2) - N(t_1)$ and $N(t_4) - N(t_3)$ are independent.

(iii) *Càdlàg function:* Every sample path of Poisson process is a càdlàg function, i.e. for all $\omega \in \Omega$ the sample path $t \mapsto N(\omega, t)$ is continuous on the right-hand side and has left-hand and right-hand side limits.

(iv) *Continuity:* For any $\omega \in \Omega$, the sample paths of the Poisson process are almost surely continuous, since the discontinuities (described as jumps) form a set of measure zero.

In Figure 2.4 a sample path of a Poisson distribution with intensity $\lambda = 3$ is shown over a time interval $[0, 1]$. One can observe the properties of the Poisson process described above. In this illustration one counts three jumps. Note that at every jump the function value increases by one, i.e. the jumps are all of size one. This leads also to the restriction that every Poisson process is increasing and cannot decrease. Since not every event in the global economy, in nature or in science has the same impact, and thus, the same jumps, we need to consider a generalized concept of jump processes. To do so, one can take into account the next type of jump processes, the so-called compound Poisson processes.

Definition 2.1.10 (Compound Poisson process). *Consider a sequence of independent identically distributed random variables that are distributed according to a certain probability*

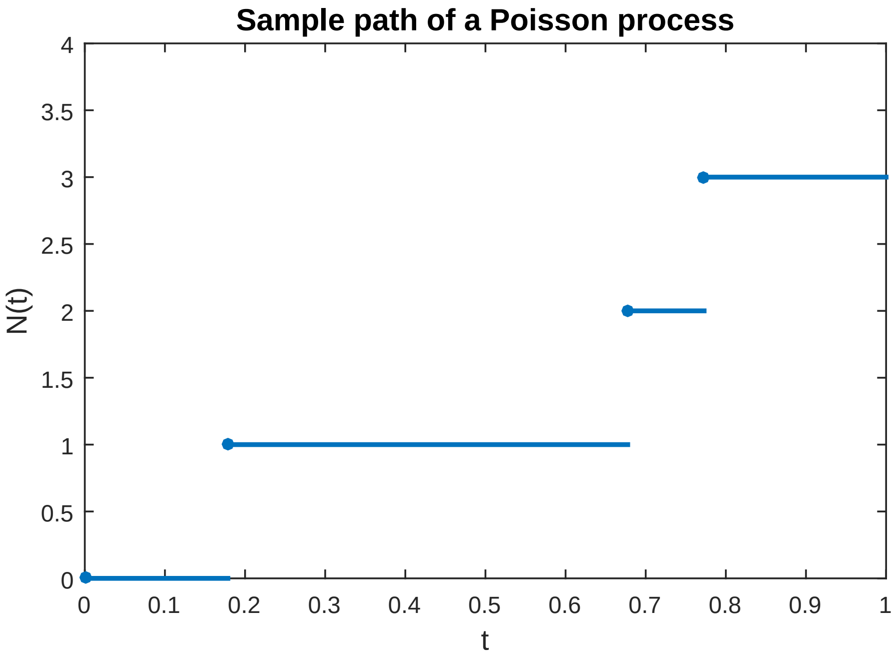

Figure 2.4: Simulation of a Poisson process with intensity $\lambda = 3$ over the time interval $[0,1]$.

distribution q. Furthermore let $(N(t))_{t\geq 0}$ be a Poisson process with intensity λ, that is independent from the random variables $(Y_i)_{i\geq 1}$. Then a compound Poisson process $(Q(t))_{t\geq 0}$ with intensity λ and jump size distribution q is defined by

$$Q(t) := \sum_{i=1}^{N(t)} Y_i \quad \forall t \geq 0.$$

The interpretation of the compound Poisson process is similar to the one of a Poisson process. The interarrival times, that indicate the times of the jumps, are exponentially distributed with rate λ. The random variable $N(t)$ specifies the number of jumps in the time interval $[0,t]$. The jump sizes are distributed according to the probability distribution q. The jump sizes between 0 and t are added up and stored in $Q(t)$. As for the Poisson process we list now a few important properties of the compound Poisson process. More information to compound Poisson processes can be found for instance in [95, 32].

Let $(Q(t))_{t\geq 0}$ be a compound Poisson process with jump intensity λ and jump size distribution q. Then the following properties hold:

(i) *Stationary and independent increments:* Let $0 \leq t_1 < t_2 \leq t_3 < t_4$. Then the distribution of the compound Poisson increment $Q(t_2) - Q(t_1)$ is stationary, i.e. it only depends on $t_2 - t_1$ and not t_1. Moreover, the increments $Q(t_2) - Q(t_1)$ and $Q(t_4) - Q(t_3)$ are independent.

(ii) *Càdlàg function:* Let $t \mapsto Q(\omega, t)$ with $\omega \in \Omega$ be a sample path of the compound Poisson process. Then this path is a càdlàg function.

(iii) *Continuity:* Every sample path of the compound Poisson process is almost surely continuous with the discontinuities (in the form of the jumps) representing a set of measure zero.

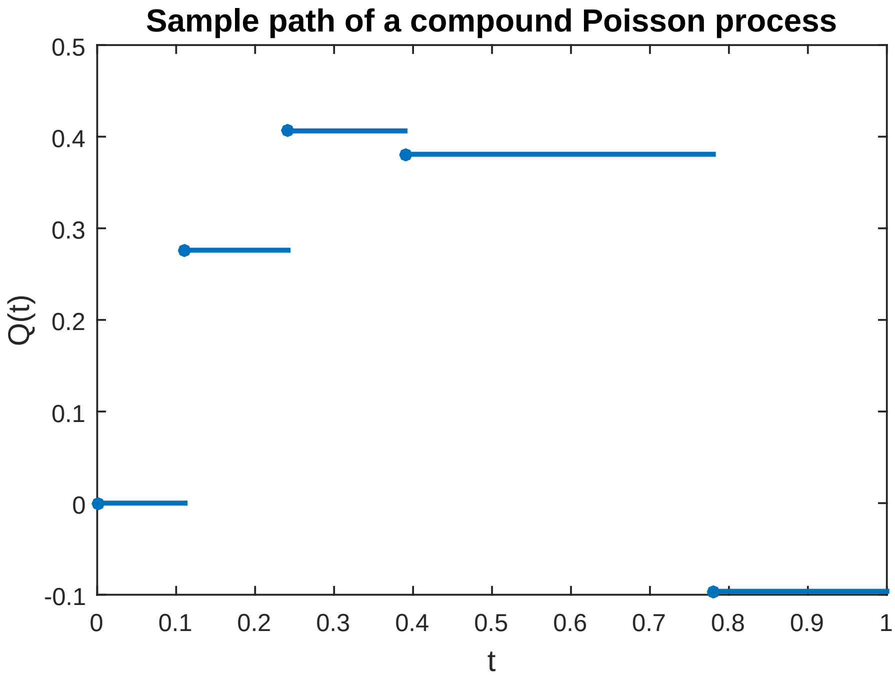

Figure 2.5: Simulation of a compound Poisson process with intensity $\lambda = 3$ and with normally distributed jump sizes with mean 0 and standard deviation $1/\lambda$ over the time interval $[0,1]$.

Figure 2.5 shows an example of a compound Poisson process with jump intensity $\lambda = 3$ and with jump sizes that are distributed according to a normal distribution with zero mean and standard deviation $1/\lambda$. One can observe that compared to the Poisson process (see Figure 2.4) the jump size is no longer fixed and the stochastic process does not necessarily increase with every jump. In fact, here we have four jumps in the time interval $[0,1]$ with the first two jumps being positive, whereas the last two are negative.

With the Brownian motion and the compound Poisson process properly defined, we have now all the ingredients to construct a jump-diffusion process.

Jump-Diffusion Processes

Here, we combine the concepts of diffusion and jump processes to create the so-called jump-diffusion processes. These stochastic processes mainly evolve like a diffusion process such as the Brownian motion. Then one adds to this process a finite number of jumps through the means of a jump process, for instance adding a Poisson or a compound Poisson process.

The resulting process incorporates the best features of both types of stochastic processes and yields a way to realistically model many problems in nature, science and so on.

Definition 2.1.11 (Jump-Diffusion process)**.** *Let* $(W(t))_{t\geq 0}$ *be a Brownian motion (see Definition 2.1.2) and let* $(Q(t))_{t\geq 0}$ *be a compound Poisson process with jump intensity* λ *and with a jump size distribution* q *(see Definition 2.1.10). Moreover let* $b \in \mathbb{R}$ *be a real number denoting the drift coefficient and let* σ *be the standard deviation of the diffusion term. Then the stochastic process* $(X(t))_{t\geq 0}$ *characterized by*

$$X(t) = bt + \sigma W(t) + Q(t)$$

is a jump-diffusion process with a diffusion component driven by a Brownian motion with drift b *and a jump component driven by a compound Poisson process.*

Note that if one considers the compound Poisson process defined as in Definition 2.1.10, i.e. $Q(t) = \sum_{i=1}^{N(t)} Y_i$, then one can rewrite the jump-diffusion process $(X(t))_{t\geq 0}$ as

$$X(t) = bt + \sigma W(t) + \sum_{i=1}^{N(t)} Y_i.$$

Between jumps the processes is characterized by a Brownian motion with drift b. The number of jumps between the start time and the time endpoint t is given by a Poisson process $N(t)$ with intensity λ. The size of jump number i is given by Y_i, which is distributed according to the distribution q. As for the other types of stochastic processes we list now the most important properties of jump-diffusion processes (for more detailed information about jump-diffusion process, see e.g. [16, 32, 95]). Let $X(t)$ be a jump-diffusion process as defined in Definition 2.1.11. Then this process has the following features:

(i) *Stationary and independent increments:* Let $0 \leq t_1 < t_2 \leq t_3 < t_4$. Then the jump-diffusion increment $X(t_2) - X(t_1)$ is stationary, i.e. it only depends on $t_2 - t_1$ and not on t_1. Furthermore, the increments $X(t_2) - X(t_1)$ and $X(t_4) - X(t_3)$ are independent.

(ii) *Càdlàg function:* Let $\omega \in \Omega$. Then the sample path $t \mapsto X(\omega, t)$ is a càdlàg function.

(iii) *Continuity:* All sample paths $t \mapsto X(\omega, t)$ (with $\omega \in \Omega$) are almost surely continuous. The discontinuities (called jumps) are of measure zero.

More information to compound Poisson processes can be found for instance in [95, 32].

In Figure 2.6 a jump-diffusion process for the time interval $[0,1]$ with a diffusion component driven by a Brownian motion with zero drift and with diffusion volatility $\sigma = 0.2$ and a jump component driven by a compound Poisson process with jump intensity $\lambda = 3$ and normal jump size distribution with zero mean and standard deviation $1/\lambda$ is illustrated. In this example

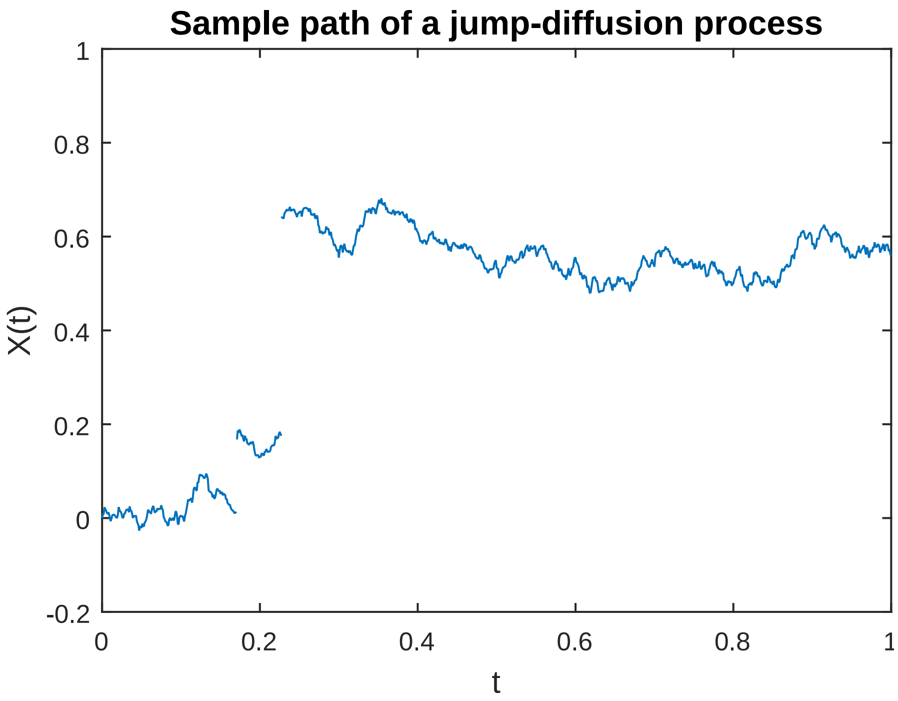

Figure 2.6: Simulation of a jump-diffusion process with intensity $\lambda = 3$ over the time interval $[0, 1]$.

there are two positive jumps both of different sizes. In between the jumps the process is a Brownian motion.

Before we construct stochastic differential equations driven by jump-diffusion processes, we briefly mention a concept, which generalizes all the different types of stochastic processes that we have seen here so far.

Beyond Jump-Diffusion Processes

As we have seen in the above sections, diffusion processes (such as the Brownian motion), jump processes (e.g. Poisson and compound Poisson processes) and jump-diffusion processes (constructed by adding a jump process to a diffusion process) all share some important properties such as stationary and independent increments, sample paths that are càdlàg functions and that are almost surely continuous. This is not just a coincidence. In fact all these processes are special cases of the so-called Lévy processes.

Definition 2.1.12 (Lévy process)**.** *Let* $(X(t))_{t \geq 0}$ *be a stochastic process such that*

(i) Initialisation: *The initial value of the stochastic process is zero, i.e.* $X(0) = 0$.

(ii) Càdlàg function: *Let* $\omega \in \Omega$*. Every sample path given by* $t \mapsto X(\omega, t)$ *is a càdlàg function.*

(iii) Stationary and independent increments: *Let* $0 \leq t_1 < t_2 \leq t_3 < t_4$*. Then the increment*

$X(t_2) - X(t_1)$ is stationary, i.e. it only depends on $t_2 - t_1$ and not on t_1. Furthermore the two stochastic increments $X(t_2) - X(t_1)$ and $X(t_4) - X(t_3)$ are independent.

(iv) Stochastic continuity: *Let $0 \leq t_1 < t_2$. For all $\varepsilon > 0$ the stochastic process satisfies*

$$\mathbb{P}\left(|X(t_2) - X(t_1)| \geq \varepsilon\right) \to 0 \quad \textit{as } t_2 \textit{ tends to } t_1.$$

General Lévy processes are beyond the scope of this thesis and we focus in the following on stochastic processes described in the previous sections. We would like to give just one remark about Lévy processes.

Remark 2.1.13. *The jump processes and the jump-diffusion processes we have seen so far have only a finite number of jumps in a finite time interval. Another special class of Lévy processes is to use processes with infinite activity, the so-called infinite activity Lévy processes. These processes are characterized by an infinite number of jumps in any time interval. Models based on infinite activity Lévy processes are usually rich enough so that there is no need to add a diffusion process such as the Brownian motion. To decide whether to model with jump-diffusion processes or with infinite activity Lévy processes depends usually on the particular problem and on modelling convenience (see e.g. [16, 64]).*

Stochastic Differential Equations driven by Jump-Diffusions

In this section we introduce stochastic differential equations that are driven by jump-diffusion processes. This enables us to model stochastic problems that can account for sudden, unforeseen events, that have a huge impact on the stochastic process over a very short time period.

Let $(X(t))_{t\geq 0}$ be a stochastic process that is characterized by the stochastic differential equation

$$\begin{cases} \mathrm{d}X(t) &= f(X(t-))\,\mathrm{d}t + \sum_{r=1}^{m_1} g_1^r(X(t-))\,\mathrm{d}W^r(t) + \sum_{r=1}^{m_2} g_2^r(X(t-))\,\mathrm{d}J^r(t), \quad 0 < t \leq T, \\ X(0) &= X_0, \end{cases} \tag{2.3}$$

where $X(t-)$ is the left-hand side limit of $X(t)$, $f : \mathbb{R}^d \to \mathbb{R}^d$ the drift function, $g_1^r : \mathbb{R}^d \to \mathbb{R}^d$ for $r = 1, 2, \ldots, m_1$ the diffusion functions and $g_2^r : \mathbb{R}^d \to \mathbb{R}^d$ for $r = 1, 2, \ldots, m_2$ the jump functions. The processes $(W^r)_{t\in[0,T]}$ with $r = 1, 2, \ldots, m_1$ are independent one-dimensional Brownian motions and $(J^r)_{t\in[0,T]}$ with $r = 1, 2, \ldots, m_2$ are independent jump processes, where $J^r(t) = \sum_{i=1}^{N^r(t)} (V_i - 1)$ with $N^r(t)$ resulting from a Poisson process with intensity λ. In the jump term of (2.3) $\mathrm{d}J^r(t)$ represents the jump of the process $(J^r)_{t\in[0,T]}$ at time t. The jump sizes are described by the random variables V_i that are distributed according to a specific distribution. The initial condition of the SDE driven by a jump-diffusion process is given by $X(0) = X_0$. For the same reasons as for stochastic differential equations based solely on diffusion processes we focus in the following on autonomous SDEs (see Remark 2.1.3).

Two jump-diffusion models that are widely used are the Merton model and the Kou model. Before we look at these models in particular, we study a specific class of stochastic differential equations driven by jump-diffusion processes that generalizes the two models. Let $(X(t))_{t\in[0,T]}$ be a stochastic process described by the SDE

$$\begin{cases} \mathrm{d}X(t) &= \mu X(t-)\mathrm{d}t + \sigma X(t-)\mathrm{d}W(t) + X(t-)\mathrm{d}J(t), \; 0 < t \le T, \\ X(0) &= X_0, \end{cases} \tag{2.4}$$

where $(W(t))_{t\in[0,T]}$ is a Brownian motion and $(J(t))_{t\in[0,T]}$ a jump process with $J(t) = \sum_{i}^{N(t)} (V_i - 1)$, where $N(t)$ is a Poisson process with intensity λ and where V_i is distributed according to a specific distribution.

Proposition 2.1.14. *The exact solution of the stochastic differential equation driven by jump-diffusions characterized by* (2.4) *is given by*

$$X(t) = X_0 \exp\left\{\left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W(t)\right\} \prod_{i=1}^{N(t)} V_i.$$

Proof. Let $(X(t))_{t\in[0,T]}$ be a stochastic process defined by (2.4) and let $\Delta X(t) := X(t) - X(t-)$. Consider the natural logarithm of $X(t)$, i.e. $\ln(X(t))$. Applying the Itô lemma for jump-diffusion processes (see e.g. [32]) yields

$$\begin{aligned} \mathrm{d}\ln(X(t)) &= \mu X(t)\tfrac{1}{X(t)}\mathrm{d}t + \tfrac{\sigma^2 X(t)^2}{2}\left(-\tfrac{1}{X(t)^2}\right)\mathrm{d}t \\ &\quad + \sigma X(t)\tfrac{1}{X(t)}\mathrm{d}W(t) + \left[\ln(X(t-) + \Delta X(t)) - \ln(X(t-))\right]. \end{aligned}$$

Taking into account the properties of the logarithm, the last term on the right-hand side can be expressed as

$$\begin{aligned} \ln(X(t-) + \Delta X(t)) - \ln(X(t-)) &= \ln\left(\tfrac{X(t-)+\Delta X(t)}{X(t-)}\right) \\ &= \ln\left(1 + \tfrac{\Delta X(t)}{X(t-)}\right) \\ &= \ln\left(1 + \tfrac{X(t-)(V_i - 1)}{X(t-)}\right) \\ &= \ln(1 + (V_i - 1)) \\ &= \ln(V_i). \end{aligned}$$

Hence, we obtain

$$\mathrm{d}\ln(X(t)) = \mu\mathrm{d}t - \frac{\sigma^2}{2}\mathrm{d}t + \sigma\mathrm{d}W(t) + \ln(V_i),$$

which can be rewritten as

$$\mathrm{d}\ln(X(t)) = \left(\mu - \frac{\sigma^2}{2}\right)\mathrm{d}t + \sigma \mathrm{d}W(t) + \ln(V_i).$$

Taking the integral on both sides over the interval between zero and t results in

$$\ln(X(t)) - \ln(X(0)) = \left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W(t) + \sum_{i=1}^{N(t)} \ln(V_i).$$

Rearranging the terms and applying the exponential on both yields, we obtain

$$X(t) = X_0 \exp\left\{\left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W(t)\right\}\prod_{i=1}^{N(t)} V_i,$$

which is the desired result. □

Remark 2.1.15. *Observe that the Black-Scholes model* (2.1.4) *is a particular form of the stochastic process defined by* (2.4)*. In fact, it can be obtained by setting the jump terms equal to zero. Therefore, the exact solution of the Black-Scholes model can also be obtained from* (2.1.14) *by removing the jumps.*

The Merton Model

Historically the first model to include jumps in the financial world was the Merton model, which was introduced in 1976 by Robert C. Merton in [80].

Definition 2.1.16 (Merton model)**.** *The Merton model is a stochastic process* $(X(t))_{t\in[0,T]}$ *characterized by the stochastic differential equation driven by jump-diffusions* (2.4)*, where the jump sizes are distributed according to a log-normal distribution with parameters* η *and* ν*, i.e.*

$$\log(V_i) \overset{\text{iid}}{\sim} \mathcal{N}\left(\eta, \nu^2\right).$$

The Merton model admits an exact solution derived in Proposition 2.1.14. The advantages of the Merton model over the Black-Scholes model, besides being a jump-diffusion model instead of a diffusion one, are that the Merton model leads to heavier tails of the distribution on the left-hand side as well as on the right-hand side [61]. Empirical data shows that the Black-Scholes model yields tails that are not significant enough (see for instance [32]). Figure 2.7 shows the illustration of the solution of the Merton model over the time interval $[0,1]$. As parameter we have chosen for the drift $\mu = 0.05$, for the diffusion $\sigma = 0.2$ and as initial condition $X(0) = 1$. The parameters related to the jump terms are the jump intensity $\lambda = 3$, the mean of the log-normal distribution $\eta = 0$ and the standard deviation $\nu = 0.03$. In our example we can observe three different jumps.

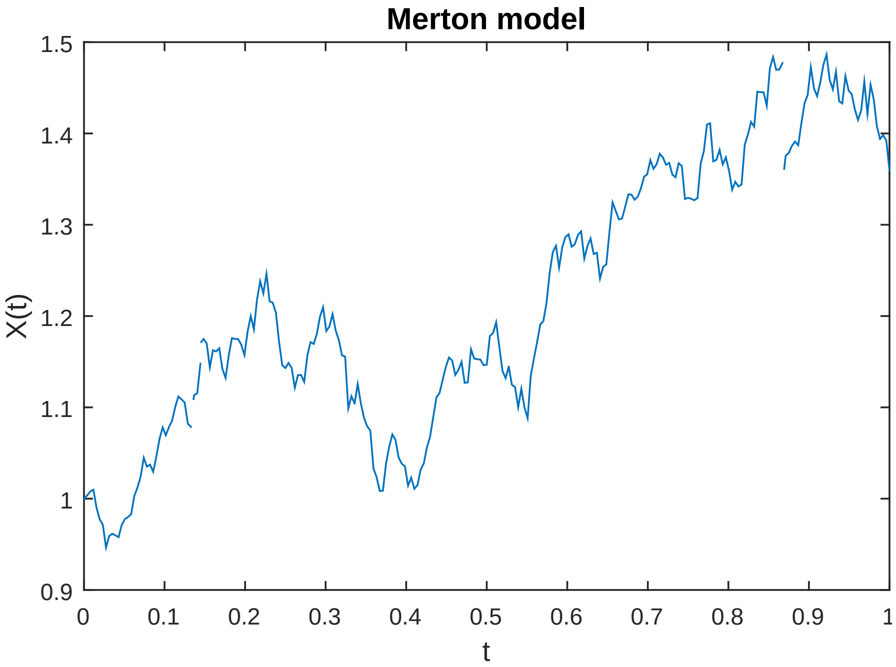

Figure 2.7: Simulation of the solution of the Merton model over the time interval $[0,1]$ with initial condition $X_0 = 1$, drift $\mu = 0.05$, diffusion $\sigma = 0.2$, jump intensity $\lambda = 3$, log-normal distribution of the jump sizes with $\eta = 0$ and $\nu = 0.03$.

The Kou Model

Another jump-diffusion model is given by the Kou model, which was introduced in [67] in 2002.

Definition 2.1.17 (Kou model)**.** *The Kou model is a stochastic process* $(X(t))_{t\in[0,T]}$ *defined by the stochastic differential equation driven by jump-diffusion processes* (2.4)*, where the jump sizes are double exponentially distributed, i.e.*

$$\log(V_i) \overset{\text{iid}}{\sim} \mathcal{K}\left(\eta_1, \eta_2, p\right)$$

with $\mathcal{K}$ *an asymmetric exponential distribution whose density function is given by*

$$f_{\mathcal{K}}(x) = p\eta_1 e^{-\eta_1 x} 1_{\{x\geq 0\}} + \left(1-p\right)\eta_2 e^{\eta_2 x} 1_{\{x<0\}}$$

with $x \in \mathbb{R}$*,* $\eta_1 > 1$*,* $\eta_2 > 0$ *and* $p \in [0,1]$*.*

Remark 2.1.18. *The parameters of the Kou model can be interpreted in the following way. The probability of an upward jump is specified by the parameter* p*, whereas the parameters* η_1 *and* η_2 *characterize the decay of the tails of the distribution of positive and negative jumps.*

Like the Merton model, the Kou model also has an exact solution, which is derived in Proposition 2.1.14. The Kou model is also useful, when one wants to derive an analytical solution to many different problems in derivative pricing. A particular advantage of the Kou model over the Merton model is that analytical solutions can also be derived for path-dependent options (see e.g. [67]).

When we compare the Kou model to the Black-Scholes model, empirical studies show that the Kou model leads to an improvement of the Black-Scholes model in two areas (see [67]). First, using the Kou model, the distribution of the returns is characterized by a higher peak than the normal distribution, heavier tails and a skew to the left. This is the so-called leptokurtic character which arises with the Kou model. Second, the implied volatility can be better reproduced using the Kou model than the Black-Scholes model (see [67, 61]).

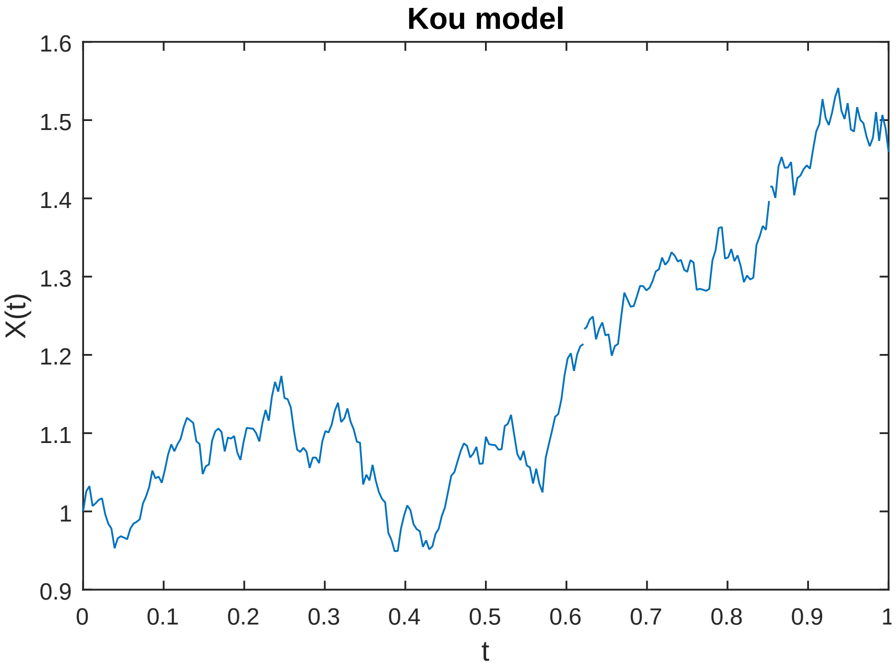

Figure 2.8: Simulation of the solution of the Kou model over the time interval $[0,1]$ with initial condition $X_0 = 1$, drift $\mu = 0.05$, diffusion $\sigma = 0.2$, jump intensity $\lambda = 3$, Kou parameters $p = 3$, $\eta_1 = 50$ and $\eta_2 = 25$.

In Figure 2.8 the simulation of the solution over the time interval $[0,1]$ of the Kou model is illustrated, where the parameters are $\mu = 0.05$ (drift), $\sigma = 0.2$ (diffusion), $\lambda = 3$ (jump intensity) and $p = 3$, $\eta_1 = 50$, $\eta_2 = 25$ (Kou parameters). The initial condition is chosen such that $X(0) = 1$. The simulation suggests that for this example there are two jumps that occur between $t = 0$ and $t = 1$.

2.2 Numerical Schemes

The previous Section 2.1 shows that many phenomena in nature, in science, in economics etc. can be modeled by stochastic differential equations driven by either diffusion or jump-diffusion processes. We have also presented a few particular models such as the Black-Scholes model, a diffusion model (see Definition 2.1.4), or the Merton and the Kou model, two jump-diffusion models (see Definition 2.1.16 and Definition 2.1.17). These three models all admit an exact solution as we have shown in Proposition 2.1.5 and Proposition 2.1.14. Note that this is an exception though and there are many more sophisticated models that do not have an analytical solution. Since one cannot derive an exact solution for many stochastic problems, we use numerical methods to approximate the solution.

In the following we present first the numerical discretization schemes that we frequently use in this thesis. This is followed by studying two important concepts of numerical methods, the convergence, where we distinguish between strong and weak convergence and the stability, where we focus on the mean square stability.

In this section we consider only diffusion models characterized by the SDE (2.1) that we recall here

$$\begin{cases} \mathrm{d}X(t) &= f(X(t))\,\mathrm{d}t + \sum\limits_{r=1}^{m} g^r(X(t))\,\mathrm{d}W^r(t), \;\; 0 < t \le T, \\ X(0) &= X_0. \end{cases}$$

Note that it is straightforward to extent what follows to jump-diffusion models. For simplicity we do not mention them any more for the rest of this section. However, we come back to the definition and properties of numerical schemes for jump-diffusion processes in the other chapters where this kind of methods are required.

In the following we assume standard Lipschitz and linear growth conditions on the drift and the diffusion so that the existence of a strong solution of the above stochastic differential equations is guaranteed (see e.g. [66]).

To approximate numerically the solution of (2.1) we take into account the discrete map

$$X_{n+1} = \Psi(X_n, h, \xi_n) \tag{2.5}$$

with $\Psi(\cdot, h, \xi_n) : \mathbb{R}^d \to \mathbb{R}^d$ for $n \ge 0$, h denoting the time stepsize and ξ_n a random vector. We define now two concepts, the convergence and the stability of a numerical scheme.

2.2.1 Strong and Weak Convergence

The first concept that we look at is the convergence. To do so, we fix a time endpoint T and we let h tend to zero. Since we study the convergence of numerical methods for stochastic differential equations, random variables are involved and we have to specify which measure of convergence one applies. That is why we distinguish in the following between strong and weak convergence.

The strong order of convergence indicates the rate at which the mean square of the error between exact and approximate solution tends to zero as the time stepsize h goes to zero. The strong convergence is important if one is interested in path-wise approximations (see [66]).

Definition 2.2.1 (Strong convergence)**.** *The numerical approximation* (2.5)*, starting from the exact initial value X_0 is said to have strong order of convergence r_s if*

$$\exists C \in \mathbb{R}_+ \quad \textit{such that} \quad \max_{0\leq n\leq T/h} \left(\mathbb{E}\left[|X_n - X(\tau_n)|^2\right]\right)^{1/2} \leq Ch^{r_s},$$

where $\tau_n = nh \in [0, T]$ with h small enough and where C is a constant independent of h.

Remark 2.2.2. *It is worth stressing that the constant C is independent of h but might depend on the time endpoint T, the stochastic process $X(t)$ and the numerical scheme.*

The other type of convergence is the weak order of convergence, which is useful if one deals with functionals of the stochastic process $(X(t))_{t\in[0,T]}$. The weak order of convergence measures in particular the rate of decay of the error of the moments of the stochastic process (see [66]).

Definition 2.2.3 (Weak convergence)**.** *The numercial method* (2.5)*, starting from the exact initial value X_0 is said to have weak order of convergence r_w if for all functions*
$\phi : \mathbb{R}^d \to \mathbb{R} \in C_P^{2(\gamma+1)}\left(\mathbb{R}^d, \mathbb{R}\right)$

$$\exists C \in \mathbb{R}_+ \quad \textit{such that} \quad \left|\mathbb{E}\left[\phi(X_n)\right] - \mathbb{E}\left[\phi(X(\tau_n))\right]\right| \leq Ch^{r_w},$$

for any $\tau_n = nh \in [0, T]$ fixed and h small enough.

Remark 2.2.4. *The constant C does not depend on the time stepsize h, but it can depend on T, $X(t)$, ϕ and the numerical scheme. Note also that here $C_P^{2(\gamma+1)}\left(\mathbb{R}^d, \mathbb{R}\right)$ denotes the space of γ times continuously differentiable functions $\mathbb{R}^d \to \mathbb{R}$ with all partial derivatives with polynomial growth.*

2.2.2 Stability

Another important concept is the stability of a numerical method which analyzes the long-term behavior of the solution (i.e. T tends to infinity) for a fixed time stepsize. There are different ways to look at the stability of the solution of a stochastic process. Two common concepts are the mean square stability, which measures the stability of moments, and the asymptotic stability, which measures the stability of a sample path on the whole (see e.g. [17, 55]). For the linear test problem that we use in the following, the mean square stability implies the asymptotic stability (see for instance [57]). In this thesis we focus on the mean square stability. The following is partly taken from [4].

Definition 2.2.5 (Mean square stability of a stochastic process)**.** *A stochastic process $(X(t))_{t\geq 0}$ is said to be mean square stable if and only if*

$$\lim_{t\to\infty} \mathbb{E}\left[X(t)^2\right] = 0.$$

To carry out a stability analysis, the one-dimensional scalar linear SDE (in fact it is also know as the Black-Scholes model) specified through

$$\begin{cases} \mathrm{d}X(t) = \mu X(t)\mathrm{d}t + \sigma X(t)\mathrm{d}W(t), \ 0 \leq t, \mu \in \mathbb{C}, \sigma \in \mathbb{C}, \\ X(0) = 1, \end{cases} \tag{2.6}$$

is widely used in the literature as test problem (see e.g. [66]). As we have seen in Proposition 2.1.5 this stochastic differential equation admits an exact solution given by

$$X(t) = X_0 \exp\left\{\left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W(t)\right\}$$

and it can be shown that the exact solution is mean square stable if and only if

$$\mathscr{R}\{\mu\} + \frac{1}{2}|\sigma|^2 < 0,$$

where $\mathscr{R}\{\cdot\}$ denotes the real part of the complex number. It follows that the stability domain of the test problem (2.6) is given by

$$\mathscr{S}_{exact} := \left\{(\mu, \sigma) \in \mathbb{C}^2 \mid \mathscr{R}\{\mu\} + \frac{1}{2}|\sigma|^2 < 0\right\}. \tag{2.7}$$

To avoid stability issues, i.e. restrictions on the stepsize, the aim of numerical methods is to cover as much as possible of this stability domain.

Definition 2.2.6 (Mean square stability of a numerical method)**.** *A numerical method* (2.5) *is said to be mean square stable if and only if*

$$\lim_{n\to\infty} \mathbb{E}\left[X_n^2\right] = 0.$$

Some comments on the linear scalar test equation. We note that the justification of the test equation (2.7) is delicate for multi-dimensional systems. Indeed the extension of the stability analysis of numerical methods for SDEs already for multidimensional linear systems $\mathrm{d}X = AX\mathrm{d}t + \sum_{r=1}^{m} B^r X\mathrm{d}W^r(t)$, where A, B^r are $d \times d$ matrices and $\mathrm{d}W^r$ are independent one-dimensional Wiener processes is difficult in general as such systems cannot be simultaneously diagonalized if A and B^r, $r = 1, 2, \ldots, m$ do not commute. Attempts to study numerical stability on linear systems have been carried out in [93, 88] but these studies do not allow for an easy characterization of stability criterion. Another attempt to generalize the linear test equation has been proposed in [27] using the theory of stochastic stabilization and destabilization [75]. Two sets of test equations with $d = m = 2$ and $d = m = 3$ have been considered. It turns out that the stability behavior for the Euler-Maruyama method (or its generalization obtained by using the θ method for the drift term) applied to these more general test equations is essentially captured by the the linear test equation (2.6). Finally, we mention that for non normal drift (2.6) can indeed fail to characterize the stability property (at least in the pre-asymptotic regime) of numerical methods [60, 27]. This is already the case in the deterministic setting for the test equation $y' = \lambda y$ (see [53]).

2.2.3 Euler-Maruyama Method

Here, we present now a first method to numerically approximate the solutions of a SDE. The simplest method to approximate solutions to (2.1) is a generalization of the Euler method for ordinary differential equations, the Euler-Maruyama method. This numerical scheme was introduced in 1955 by Gisiro Maruyama [78].

Definition 2.2.7 (Euler-Maruyama method)**.** *Let h be a uniform time stepsize. Then the Euler-Maruyama method for* (2.1) *is defined by*

$$X_{n+1} = X_n + hf(X_n) + \sum_{r=1}^{m} g^r(X_n)\,\Delta W_{n,r}, \tag{2.8}$$

where $\Delta W_{n,r} \sim \mathcal{N}(0,h), r = 1,2,\ldots,m$ *are independent Wiener increments.*

It can be shown that the Euler-Maruyama method is of strong order of convergence 1/2 and of weak order of convergence 1 (see for instance [66]). Furthermore, for the Euler-Maruyama scheme we have a stability domain given by

$$\mathcal{S}_{EM} := \left\{(p,q) \in \mathbb{C}^2 \mid |1+p|^2 + q^2 < 1\right\}, \tag{2.9}$$

where $(p,q) = (h\mu, \sqrt{h}|\sigma|)$ (see [66]).

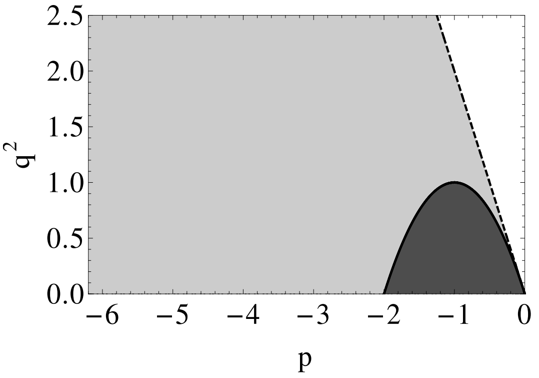

Figure 2.9: Stability region (dark gray) for the Euler-Maruyama method. The dashed line delimits the stability region (light gray) of the test problem (2.6).

Figure 2.9 illustrates the stability region (we call it region instead of domain if we consider the parameters to be real instead of complex) of the Euler-Maruyama method. One observes that the Euler-Maruyama only covers a small portion of the true stability region. Therefore, for stiff problems there might be a severe time stepsize restriction due to stability issues. One way to improve the stability properties is to use S-ROCK methods that we introduce in the next section.

2.2.4 S-ROCK Method

The so-called S-ROCK methods have first been introduced for Stratonovich stochastic differential equations in [9, 10] and they have been extended to Itô SDEs in [12]. Here, we will focus on the latter. S-ROCK methods are numerical integrators that offer an extended stability domain while remaining explicit. There are different types of S-ROCK methods (mainly depending on the weak and strong order), but since they all are built up the same way (there is a deterministic stabilization procedure and a stochastic finishing procedure, which guarantees that the desired accuracy is achieved), we present in this section only S-ROCK methods of weak order 1 and strong order 1/2. We call this type of S-ROCK methods in the following S-ROCK1. S-ROCK2 (weak order 2, strong order 1/2) and other schemes are used in other chapters of this thesis and they are presented in the corresponding sections.

Here we define the S-ROCK1 method with s stages and of weak order 1 and strong order $\frac{1}{2}$ (see [12]).

Definition 2.2.8 (S-ROCK1 method)**.** *Let h be a uniform time stepsize. The s-stage S-ROCK1 method for* (2.1) *is defined for all $s \geq 2$ as follows:*

$$\begin{aligned}
K_0 &= X_n \\
K_1 &= X_n + h\frac{\omega_1}{\omega_0} f(K_0) \\
K_i &= 2h\omega_1 \frac{T_{i-1}(\omega_0)}{T_i(\omega_0)} f(K_{i-1}) + 2\omega_0 \frac{T_{i-1}(\omega_0)}{T_i(\omega_0)} K_{i-1} - \frac{T_{i-2}(\omega_0)}{T_i(\omega_0)} K_{i-2}, \ i = 2,3,\dots,s-1, \\
K_s &= 2h\omega_1 \frac{T_{s-1}(\omega_0)}{T_s(\omega_0)} f(K_{s-1}) + 2\omega_0 \frac{T_{s-1}(\omega_0)}{T_s(\omega_0)} K_{s-1} - \frac{T_{s-2}(\omega_0)}{T_s(\omega_0)} K_{s-2} + \sum_{r=1}^{m} g^r(K_{s-1}) \Delta W_{n+1,r},
\end{aligned}$$

where $\omega_0 = 1 + \frac{\eta}{s^2}$, $\omega_1 = \frac{T_s(\omega_0)}{T_s'(\omega_0)}$ and $\Delta W_{n+1,r} = W^r(\tau_{n+1}) - W^r(\tau_n)$ and we set $X_{n+1} = K_s$.

Note that $(T_i(x))_{i\geq 0}$ are the orthogonal Chebyshev polynomials, which are recursively given by

$$T_0(x) = 1, \ T_1(x) = x, \ T_i(x) = 2xT_{i-1}(x) - T_{i-2}(x) \text{ for } i \geq 2, \ x \in \mathbb{R}.$$

The parameter η is known as the *damping* parameter and is used to enlarge the width of the stability domain in the direction of the noise. The value of η can be chosen to optimize the stability (in the mean square sense) of the method (see [12]). We note that in the absence of noise, the S-ROCK method coincides with the Chebyshev method introduced in [101]. Further, we note that for $s = 1$ one obtains the Euler-Maruyama method (2.8) so that the s-stage S-ROCK1 method is defined for all $s \geq 1$.

To define the stability domain of S-ROCK methods we first consider

$$\mathscr{S}_{SDE,a} = \left\{ (p,q) \in [-a,0] \times \mathbb{R} \mid |q| \leq \sqrt{-2p} \right\},$$

a *portion* of the true stability region. Furthermore, we define

$$a^* = \sup \left\{ a > 0 \mid \mathscr{S}_{SDE,a} \subset \mathscr{S}_{num} \right\},$$

where $\mathscr{S}_{num}$ denotes the stability domain of the numerical approximation scheme (see [12]). In [12] it is shown that S-ROCK methods have stability domains with large a^* and that the growth of the portion of the true stability region increases as $a_s^* \approx c_{SR1} s^2$ with $c_{SR1} \geq 0.33$, where s is the number of stages of the S-ROCK method (see also Section 3.2.2).

Remark 2.2.9. *A crucial property for the S-ROCK methods is that a_s^* grows quadratically with the stage number s, whereas the number of function evaluations only increases linearly with s.*

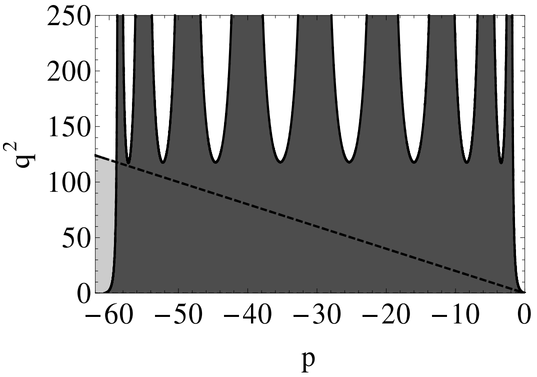

Figure 2.10: Stability regions (dark gray) for the S-ROCK1 method with stage number $s = 10$ and damping parameter $\eta = 5.9$. The dashed line delimits the stability region (light gray) of the test problem (2.7).

Figure 2.10 illustrates for instance the stability region (dark gray) of the S-ROCK1 method with $s = 10$ stages and damping parameter $\eta = 5.9$. The interior part (light gray) of the dashed lines represents the stability region of the true solution. We see that the S-ROCK1 methods cover a significantly larger region than the EM method (see Figure 2.9). By varying s, any portion of the true stability region can be covered [12].

2.3 Monte Carlo Techniques

In many areas, be it in economics (e.g. [61]), in physics (e.g. [81]), in chemistry (e.g. [48]), in biology (e.g. [103]) and so on, there are problems that can be modeled by using stochastic differential equations. Many times one is not only interested in the exact solution to the problem described by the SDE, but its properties such as its mean or its variance. More generally speaking it is interesting to compute the expectation of a functional of a stochastic process. In mathematical terms this means that one would like to compute

$$\mathbb{E}\left[\phi(X(T))\right],$$

where T is some fixed time endpoint, $(X(t))_{t\in[0,T]}$ is a stochastic process with $X(t)\in\mathbb{R}^d$ and $\phi:\mathbb{R}^d\to\mathbb{R}^d$ is some functional.

Now there are problems, where this kind of quantities can be computed analytically. Take for instance the scenario of pricing European options. The underlying share price can e.g. be modeled by the Black-Scholes model. To determine the price of a European option based on this share price, one has to compute the expectation of a functional as described above. It can be shown that there can be derived an analytical expression for the option price (see e.g. [70]). This is one particular situation where this works very well. However, there are many different problem settings where one cannot necessarily derive an exact solution. This is for instance the case when more sophisticated stochastic models are used (e.g. the general Black-Scholes model with stochastic volatility [61]) or when the functional is more complex (as this is the case for instance for path-dependent options like Asian or barrier options [36]). Therefore one needs a numerical approach to find an approximation to the solution. Basically there are two different approaches to tackle this kind of situations. Either one reformulates the problem as partial differential equations using the forward Kolmogorov equation and then solves the resulting PDEs using, for instance, finite differences, or one uses Monte Carlo techniques. In this thesis we focus on the latter, which has the advantage that it can easily be implemented and that it is not affected by the dimension of the problem. However, Monte Carlo techniques can be computationally expensive, and thus, slow, so one is usually interested in speeding them up.

This section is structured as follows. We give first the formal description of the Monte Carlo approach and we discuss how to measure the efficiency of simulation estimators. Then we briefly discuss some well-known variance reduction techniques (two of them have been implemented in this thesis) and then we present the multilevel Monte Carlo methods, which are quite popular when it comes to speeding up simulations and which can be used for many different kind of problems.

2.3.1 Monte Carlo Method

Here, we present how Monte Carlo is used to approximate the expectation of functionals depending on a random variable. Let X be a random variable with probability distribution F, $\phi:\mathbb{R}^d\to\mathbb{R}^d$ some functional and suppose that we are interested in approximating

$$\mathbb{E}\left[\phi(X)\right]. \tag{2.10}$$

The estimator of (2.10) using Monte Carlo simulations can be obtained as follows:

1. Draw N independent realizations from F, the distribution of X to get
$$X_1, X_2, \ldots, X_N.$$

2. Apply the functional ϕ to every realization so that we obtain
$$\phi(X_1), \phi(X_2), \ldots, \phi(X_N).$$

3. Estimate the expectation of $\phi(X)$ by taking the sample average
$$\mathbb{E}\left[\phi(X)\right] \approx \frac{1}{N}\sum_{i=1}^{N}\phi(X_i).$$

We give now an example which shows how Monte Carlo simulations can be used to get a numerical approximation of an integral (see for instance [45]). Let $f : \mathbb{R} \to \mathbb{R}$ be some function. Furthermore, assume that there are two real numbers a and b such that $b > a$. Suppose that one would like to compute numerically the integral of f over the time interval $[a, b]$, i.e. we try to estimate
$$\int_a^b f(x)\mathrm{d}x.$$
The first step is to rewrite the integral as an expectation, i.e.
$$\int_a^b f(x)\mathrm{d}x = (b-a)\int_{-\infty}^{+\infty} f(x)\frac{1}{(b-a)}1_{[a,b]}\mathrm{d}x = (b-a)\mathbb{E}\left[f(x)\right]$$
with the underlying law being the uniform distribution over the interval $[a, b]$. The second and final step consists in applying Monte Carlo to obtain an estimate of the expectation. In particular, one samples independently N numbers $x_1, x_2, \ldots, x_N$ from a uniform distribution over $[a, b]$. This yields as estimator for the integral
$$\int_a^b f(x)\mathrm{d}x \approx (b-a)\frac{1}{N}\sum_{i=1}^{N} f(x_i).$$

There exist many other approximation techniques to numerically integrate integrals, especially in dimension one (see e.g. [69]). However, one of the huge benefits of using Monte Carlo simulations is that the convergence rate is independent of the dimension of the problem. The speed of convergence does not depend on the dimension of the integral but only on its number of simulations N (see [69]). A typical rate of convergence of Monte Carlo methods based on N simulations is $N^{-1/2}$ (see e.g. [45]). There are many books and papers that deal with Monte Carlo techniques, we name here a few [41, 45, 89]. To be able to compare Monte Carlo estimators to other type of estimators we have to fix some criterion that enables us to judge which estimator is more efficient. We do this in the next section.

Efficiency of Simulation Estimators

In this section we present a way how one can compare the efficiency of simulation estimators. This is helpful if one has to decide which estimator to pick. The following is mainly based on [49]. To measure the efficiency of estimators we look at three quantities, which are the computing time, the bias and the variance of the estimator. We pursue now by putting all these quantities in relation to each other.

Let X be the quantity that we would like to estimate. First suppose that we deal with estimators $\hat{X}_N$ that are defined by

$$\hat{X}_N := \frac{1}{N}\sum_{i=1}^{N} X_i,$$

where $X_1, X_2, \ldots, X_N$ are independent and identically distributed with mean $\mathbb{E}[X_i] = X$ and variance $\mathrm{Var}(X_i) = \sigma^2$, where σ is some finite real number. Observe that these estimators are unbiased, i.e. using the linearity of the expectation we get

$$\mathbb{E}\left[\hat{X}_N\right] = \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N} X_i\right] = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[X_i] = X$$

(see [31]). Furthermore, taking into account the independence of the simulations $X_1, X_2, \ldots, X_N$ we obtain for the variance

$$\mathrm{Var}\left(\hat{X}_N\right) = \mathrm{Var}\left(\frac{1}{N}\sum_{i=1}^{N} X_i\right) = \frac{1}{N^2}\sum_{i=1}^{N}\mathrm{Var}(X_i) = \frac{\sigma^2}{N}.$$

Hence, the central limit theorem yields

$$\frac{\hat{X}_N - \mathbb{E}\left[\hat{X}_N\right]}{\sqrt{\mathrm{Var}\left(\hat{X}_N\right)}} = \frac{\frac{1}{N}\sum\limits_{i=1}^{N} X_i - X}{\frac{\sigma}{\sqrt{N}}} \to Z \sim \mathcal{N}(0,1) \text{ as } N \to \infty. \tag{2.11}$$

And thus, for N sufficiently large, we have

$$\hat{X}_N - X \approx \mathcal{N}\left(0, \frac{\sigma^2}{N}\right).$$

Therefore, everything but the variance being equal, estimators with a lower variance are more efficient. Note that in the case where the variance σ^2 is unknown, it is possible to to replace it by a consistent estimator such as, for instance, $\hat{\sigma}^2 = \frac{1}{N-1}\sum\limits_{i=1}^{N}\left(X_i - \hat{X}_N\right)^2$ (see e.g. [49]).

So far we have not considered the computing time, but we will do this now to account for the computational effort required to produce an estimator. Let b be the computational budget and let τ be the computing time required to generate a replication of X_i. Using only relative values in the following and assuming that b and τ share the same unit, we do not need to indicate the computing time. The total number of replications that one can generate is given by $N = \left\lfloor \frac{b}{\tau} \right\rfloor$. Observe that as the budget b tends to infinity, $\frac{N}{b} \to \frac{1}{\tau}$, and thus, by (2.11)

$$\sqrt{b}\left(\hat{X}_{\lfloor \frac{b}{\tau} \rfloor} - X\right) \to \mathcal{N}\left(0, \sigma^2 \tau\right).$$

This can be interpreted as follows. For unbiased estimators, to find out which one of two estimators performs better, one can try to figure out for which one the value of the product of variance and the computing time for a single replication is lower. This approach can be generalized if we consider that for each replication X_i there is a corresponding computing time τ_i. Supposing that the couples (X_i, τ_i) for $i = 1, 2, \ldots, N$ are independent and identically distributed, one has to compare the quantity $\sigma^2 \mathbb{E}[\tau]$ of the different estimators (see e.g. [49]). If desired this approach can even further be generalized (for more details see [51]).

The last relation that we build up now is the one between bias and variance. In the following we suppose that there is a fixed computational budget. Until now we have considered only unbiased estimators, which is important to make sure that the estimator converges to the right value. In small samples bias is usually acceptable under the condition that the bias can be reduced as much as required by increasing the computational budget (see [49]). Increasing the computational effort of a replication reduces the bias, however, at the same time this leads to a larger variance since the number of simulations decreases. Therefore, there is a trade-off between bias and variance which can be measured, for instance, by the mean square error (MSE) that is defined by

$$\mathrm{MSE}\left(\hat{X}_N\right) = \mathbb{E}\left[\left(\hat{X}_N - X\right)^2\right].$$

By adding and subtracting the term $\mathbb{E}\left[\hat{X}_N\right]$ and by using the properties of the expectation, it is straightforward to show that

$$\mathrm{MSE}\left(\hat{X}_N\right) = \mathrm{Var}\left(\hat{X}_N\right) + \left(\mathrm{bias}\left(\hat{X}_N\right)\right)^2.$$

Note that, to compare two estimators in this thesis, we usually put into relation the mean square error and the computational cost. The MSE measures the precision of the estimator and the computational cost counts the number of function evaluations, and thus, is an indicator of the computing time. By fixing a desired mean square error one can look at the corresponding computational cost and then pick the estimator with a lower one.

Implementing small bias is often easier than small variance (see e.g. [49]). Hence, it is worth studying methods that try to decrease the variance of the standard Monte Carlo approach. We discuss this in the next section.

2.3.2 Variance Reduction Techniques

Monte Carlo techniques are simple and work very well even in higher dimension. One drawback of Monte Carlo methods though is that they are computationally expensive, and thus, these methods can be very slow (see Section 2.3.1). Hence, it is useful to look for approaches that can speed up the simulation procedure. One way to improve the performance is to reduce the computing time of a problem by lowering the variance, which leads to the so-called variance reduction techniques. There are many different variance reduction techniques and they are difficult to compare, because they usually depend on the underlying problem. It is not obvious to generally say which variance reduction techniques is the most efficient one. One has to study the problem and then to decide which approaches might be the one suited best. In the following we briefly state the most common variance reduction techniques and we give their advantages and drawbacks. In this thesis we have applied two of the techniques, namely the antithetic variates and the control variates (see Section 5.4.2). The following is based on [49].

Antithetic variates The idea of approach is to consider pairs of replications that are in negative correlation to each other. This leads to a variance reduction, and thus, improves the efficiency of the method. One big advantage of this method is that it is straightforward to apply, but one has to remain cautious, because it is also possible that the antithetic variates technique increases the variance of the estimator for certain problems.

Conditional Monte Carlo This variance reduction technique replaces the expectation by the conditional expectation. This has the advantage that parts of the integration can be computed analytically and only the remaining parts have to be estimated by Monte Carlo simulations. This procedure is also known as Rao-Blackwellization.

Control variates For this technique new estimators with a reduced variance are produced by using known quantities, the so-called control variates. As control variate one uses often a random variable with a known mean. The more this random variable is correlated to the one to be estimated, the better results can be expected. The control variates approach is easy to implement and guarantees that the variance does not increase.

Importance sampling By changing the probability measure to give important outcomes more weight, this variance reduction approach aims to increase the efficiency. If used properly this technique can be very strong and lead to a huge improvement of the performance. However, it is also possible that this approach increases the variance and finding an adequate probability change can be quite challenging.

Matching underlying assets The matching underlying assets approach comprises different methods that all intend to get specific sample means that coincide with the value they would reach after carrying out infinitely many simulations. To this category we could, among others, the moment matching approach realized by adjusting the path or the weighted Monte Carlo method.

Stratified sampling The idea of this variance reduction technique is to divide the sample space into specific subsets (the so-called strata) and then to sample from each strata. Creating the strata it is important that the union of all the strata cover the entire sample space and that the different strata are disjoint. This is an approach that can be quite powerful, but it requires a higher effort and the additional knowledge for the correct implementation.

Latin hypercube sampling This is an extension of the previous variance reduction technique to higher dimensional problems. This method is usually more effective, but also more complex than the stratified sampling approach. Note that stratified sampling in higher dimensions is often infeasible.

In the next section we present another way to speed up Monte Carlo methods, the so-called multilevel Monte Carlo method. A nice feature of the multilevel approach is that we can combine it with variance reduction techniques to obtain even more powerful methods. In addition, a huge advantage of the multilevel Monte Carlo method over most variance reduction techniques is that the multilevel Monte Carlo approach does not change the underlying problem, whereas many variance reduction techniques do.

2.3.3 Multilevel Monte Carlo Method

In this section we present the multilevel Monte Carlo (MLMC) method, which was first introduced by Michael B. Giles in [46]. The method has its origins in a paper of Ahmed Kebaier, who published in [65] a new variance reduction technique. In fact, Kebaier introduced a new control variates approach based on statistical Romberg extrapolation in which he applies the Monte Carlo method twice for two different time stepsizes. Then the two approximations are combined by taking many simulations of the coarse time grid and only a few from the fine time grid. Fixing a mean square error of $\mathcal{O}\left(\varepsilon^2\right)$ for some $\varepsilon > 0$ Kebaier's method has a computational cost of $\mathcal{O}\left(\varepsilon^{-2.5}\right)$, which is an improvement over the computational cost of the standard Monte Carlo approach $\mathcal{O}\left(\varepsilon^{-3}\right)$. Inspired by this approach Giles extended this idea from two levels to several levels. The strategy remains the same, for simulations based on a small time stepsize take just a few, because they are computationally expensive. For simulations with a large time stepsize, which are computationally cheap, take many. Finding the right balance between number of simulations and time stepsize, the multilevel Monte Carlo method improves the efficiency of Monte Carlo techniques significantly. In fact, to reach a mean square accuracy of $\mathcal{O}\left(\varepsilon^2\right)$, it can be shown that the MLMC method reduces the computational cost from $\mathcal{O}\left(\varepsilon^{-3}\right)$ for standard Monte Carlo to $\mathcal{O}\left(\varepsilon^{-2}\left(\log\varepsilon\right)^2\right)$.

Let $(X(t))_{t\in[0,T]}$ (with T a fixed time endpoint) be a stochastic process based on the following SDE

$$\begin{cases} \mathrm{d}X(t) &= f(X(t))\,\mathrm{d}t + \sum_{r=1}^{m} g^r(X(t))\,\mathrm{d}W^r(t), \quad 0 < t \le T, \\ X(0) &= X_0, \end{cases}$$

where we have used the same notation as in (2.1). Furthermore, let $\phi : \mathbb{R}^d \to \mathbb{R}$ be a Lipschitz continuous function. Suppose that one would like to estimate the expectation

$$\mathbb{E}\left[\phi\left(X(T)\right)\right] =: E.$$

In what follows we use the Euler-Maruyama method (2.8) as numerical integrator. In other chapters of this thesis we have used different integrators, such as for instance the regular and jump-adapted Euler-Maruyama methods in the jump-diffusion case (see Section 5.3) and various S-ROCK methods (see Section 3.3). Before we show how the multilevel Monte Carlo approach works, we present how one can approximate E by using standard Monte Carlo.

The Standard Monte Carlo Method

The Monte Carlo estimator of E is given by

$$\mathbb{E}\left[\phi\left(X(T)\right)\right] \approx \frac{1}{\tilde{N}}\sum_{i=1}^{\tilde{N}} \phi\left(X_{T/h}^{(i)}\right) =: E_{MC}$$

a sample average over $\tilde{N}$ independent simulations, where h is the uniform time stepsize of the Euler-Maruyama method. By applying the Monte Carlo approach, two different types of errors arise. First, there is the error due to the numerical discretization of the solution of $X(t)$, which leads to a bias. In fact, approximating $X(T)$ using Euler-Maruyama with time stepsize h yields

$$X(T) \approx X_{T/h}.$$

Applying the functional and taking the expectation on both sides results in

$$\mathbb{E}\left[\phi\left(X(T)\right)\right] \approx \mathbb{E}\left[\phi\left(X_{T/h}\right)\right].$$

When we compute the bias of the estimator E_{MC} we obtain

$$\begin{aligned} \operatorname{bias}\left(E_{MC}\right) &= \mathbb{E}\left[E_{MC}\right] - \mathbb{E}\left[\phi\left(X(T)\right)\right] \\ &= \mathbb{E}\left[\frac{1}{\tilde{N}}\sum_{i=1}^{\tilde{N}} \phi\left(X_{T/h}^{(i)}\right)\right] - \mathbb{E}\left[\phi\left(X(T)\right)\right] \\ &= \mathbb{E}\left[\phi\left(X_{T/h}\right)\right] - \mathbb{E}\left[\phi\left(X(T)\right)\right] \\ &= \mathcal{O}(h), \end{aligned} \tag{2.12}$$

where we have used the linearity of the expectation and that the samples $X_{T/h}^{(i)}$ are identically distributed. The last equality results from the fact that we apply a numerical integrator of weak order of convergence 1.

Second, since the expectation is approximated by a sample average there is a statistical error. In fact, due to the strong law of large numbers and the central limit theorem there is almost surely no bias. However, this approximation introduces a certain variance, that depends on the number of sample simulations $\tilde{N}$. The variance of the Monte Carlo estimator E_{MC} can be expressed as follows:

$$\operatorname{Var}(E_{MC}) = \operatorname{Var}\left(\mathbb{E}\left[\frac{1}{\tilde{N}}\sum_{i=1}^{\tilde{N}} \phi\left(X_{T/h}^{(i)}\right)\right]\right) = \frac{1}{\tilde{N}^2}\sum_{i=1}^{\tilde{N}} \operatorname{Var}\left(\phi\left(X_{T/h}^{(i)}\right)\right) = \frac{\operatorname{Var}\left(\phi\left(X_{T/h}\right)\right)}{\tilde{N}} = \mathcal{O}\left(\frac{1}{\tilde{N}}\right), \tag{2.13}$$

where we have used the fact that the samples $X_{T/h}^{(i)}$ are independent and identically distributed and that we apply a numerical integrator with strong order of convergence 1/2.

As we have seen earlier in Section 2.3.1, a good way to measure the efficiency of an estimator is using the mean square error. Assume now that a certain mean square accuracy

$$\operatorname{MSE}(E_{MC}) = \mathcal{O}\left(\varepsilon^2\right)$$

is desired with $\varepsilon > 0$. Since the mean square error can be split into variance and bias to the square, i.e.

$$\operatorname{MSE}(E_{MC}) = \operatorname{Var}(E_{MC}) + \left(\operatorname{bias}(E_{MC})\right)^2,$$

and using the results (2.13) and (2.12) we obtain

$$\tilde{N} = \mathcal{O}\left(\varepsilon^{-2}\right) \quad \text{and} \quad h = \mathcal{O}(\varepsilon).$$

Measuring the computational cost (or also called computational complexity) by the number of function evaluations of a numerical discretization per sample path times the total number of sample paths, we get for the standard Monte Carlo method

$$\operatorname{Cost}(E_{MC}) = \tilde{N}\frac{T}{h} = \mathcal{O}\left(\varepsilon^{-3}\right).$$

The Multilevel Monte Carlo Method

Here, we present now the multilevel Monte Carlo method. Let L be a positive integer representing the total number of levels. As refinement factor we fix here a particular case by taking $k = 2$. A more general description of the MLMC method with refinement factor k is given in Section 3.3.1. The MLMC method with L levels and refinement factor $k = 2$ uses the following sequence of nested time stepsize:

$$h_\ell = \frac{T}{2^\ell} \quad \text{with} \quad \ell = 0, 1, \ldots, L.$$

In fact, that means passing from level ℓ to level $\ell + 1$ each time step from level ℓ is divided by two. Note that the number of steps per level ℓ is given by $M_\ell := 2^\ell$. Figure 2.11 shows the idea of the MLMC approach. The plot shows the solution to the Black-Scholes model (see 2.2) with drift coefficient $\mu = 0.05$, diffusion coefficient $\sigma = 0.2$ and time endpoint $T = 1$.

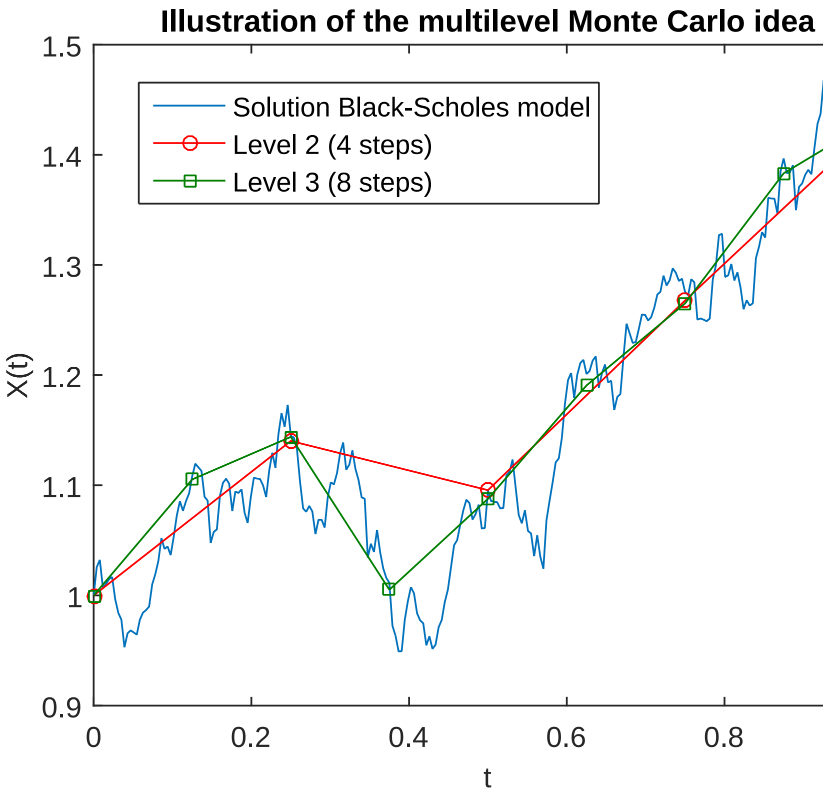

Figure 2.11: Solution of the Black-Scholes model with drift $\mu = 0.05$ and diffusion $\sigma = 0.2$ over the time interval $[0,1]$. In addition, there are two Euler-Maruyama approximations, one with 4 steps (level 2) and one with 8 steps (level 3).

Furthermore, Figure 2.11 illustrates the levels 2 and 3 of the MLMC approach, i.e. the Euler-Maruyama approximation with time stepsize $h_2 = 1/4$ and time stepsize $h_3 = 1/8$, respectively. It is important to point out that for both levels the same Brownian path has been used.

Let $\phi_\ell := \phi\left(X_{M_\ell}\right) \approx \phi(X(T))$ be the Euler-Maruyama approximation with time stepsize h_ℓ of $\phi(X(T))$. Using the telescopic sum we can write

$$\phi_L = \sum_{\ell=0}^{L} \left(\phi_\ell - \phi_{\ell-1}\right),$$

where $\phi_{-1} \equiv 0$. The multilevel Monte Carlo estimator is given by

$$E^* := \sum_{\ell=0}^{L} E_\ell^* \quad \text{with} \quad E_\ell^* := \frac{1}{N_\ell} \sum_{i=1}^{N_\ell} \left(\phi_\ell^{(i)} - \phi_{\ell-1}^{(i)}\right),$$

which is a sample average over N_ℓ independent realizations. It is important to stress that both estimates, $\phi_\ell^{(i)}$ and $\phi_{\ell-1}^{(i)}$ are produced using the same Brownian motion path.

As for the standard Monte Carlo method, we use the mean square error as measure of the accuracy of the MLMC estimator. The mean square error can be divided two components, the variance and the bias to the square, i.e.

$$\operatorname{MSE}\left(E^*\right) = \operatorname{Var}\left(E^*\right) + \left(\operatorname{bias}\left(E^*\right)\right)^2.$$

Observe that the mean of the MLMC estimator is given by

$$\mathbb{E}\left[E^*\right] = \sum_{\ell=0}^{L} \mathbb{E}\left[\frac{1}{N_\ell}\sum_{i=1}^{N_\ell}\left(\phi_\ell^{(i)} - \phi_{\ell-1}^{(i)}\right)\right] = \sum_{\ell=0}^{L} \mathbb{E}\left[\phi_\ell - \phi_{\ell-1}\right] = \mathbb{E}\left[\phi_L\right],$$

where we have used the properties of the expectation and the fact that the $\phi_\ell^{(i)}$ are identically distributed. Hence, we obtain for the bias

$$\operatorname{bias}\left(E^*\right) = \mathbb{E}\left[E^*\right] - E = \mathbb{E}\left[\phi_L\right] - E.$$

Using the weak order of convergence 1 of the Euler-Maruyama method yields

$$\operatorname{bias}\left(E^*\right) = \mathcal{O}\left(2^{-L}\right). \tag{2.14}$$

Next, we derive a similar result for the variance of the MLMC estimator. Note that using the Cauchy-Schwarz inequality we have

$$\operatorname{Var}\left(\phi_\ell - \phi_{\ell-1}\right) \leq \left(\operatorname{Var}\left(\phi_\ell - E\right)^{1/2} + \operatorname{Var}\left(\phi_{\ell-1} - E\right)^{1/2}\right)^2.$$

Furthermore, it holds that

$$\operatorname{Var}\left(\phi_\ell - E\right) \leq \mathbb{E}\left[\left(\phi_\ell - E\right)^2\right] = \mathbb{E}\left[\left(\phi\left(X_{M_\ell}\right) - \phi\left(X(T)\right)\right)^2\right] \leq C2^{-\ell},$$

where C is some constant resulting from the Lipschitz continuity. We have also used that the Euler-Maruyama method is of strong order of convergence $1/2$. Therefore, we obtain for the variance

$$\operatorname{Var}\left(E^*\right) = \sum_{\ell=0}^{L} \frac{\operatorname{Var}\left(\phi_\ell - \phi_{\ell-1}\right)}{N_\ell} = C\sum_{\ell=0}^{L}\frac{2^{-\ell}}{N_\ell}. \tag{2.15}$$

As for the MC approach, we assume that a mean square precision of $\operatorname{MSE}(E^*) = \mathcal{O}\left(\varepsilon^2\right)$ is desired. Using the results for the bias (2.14) and for the variance (2.15) yields that $\varepsilon = 2^{-L}$, i.e. $L = -\frac{\log\varepsilon}{\log 2}$, and the number of simulations per level ℓ has to be chosen such that $N_\ell = 2^{2L}2^{-\ell}L$.

Finally, for the computational complexity we obtain

$$\operatorname{Cost}\left(E^*\right) = \sum_{\ell=0}^{L} N_\ell M_\ell = \sum_{\ell=0}^{L} 2^{2L}2^{-\ell}L2^{\ell} = 2^{2L}L(L+1) = \mathcal{O}\left(\varepsilon^{-2}\left(\log\varepsilon\right)^2\right),$$

which is a significant reduction of the computational cost compared to the standard Monte Carlo method for a same mean square accuracy of $\mathcal{O}\left(\varepsilon^2\right)$.

3 Stabilized Multilevel Monte Carlo Method for Stiff Stochastic Differential Equations

In this chapter we present a multilevel Monte Carlo method for mean square stable stochastic differential equations with multiple scales. The MLMC approach based on the classical explicit numerical integrators deteriorates for such stiff problems. In fact, due to the time stepsize restriction of such numerical methods, not all levels of the MLMC method can be exploited. We introduce here multilevel Monte Carlo method that is based on an explicit stabilized numerical method, the S-ROCK1 method. By balancing the stabilization procedure and by considering at the same time the hierarchical sampling strategy of multilevel Monte Carlo approach, we obtain a method whose computational cost is significantly reduced compared to the standard MLMC method for stiff systems. We also show that for nonstiff problems our stabilized MLMC method can outperform the MLMC method based on the classical numerical methods. A big advantage of the stabilized approach is that the method remains fully explicit and easy to implement. Last but not least we also show how the stabilized multilevel Monte Carlo method can be further improved by using a higher weak order scheme on the finest time grid (in our case we use the S-ROCK2 method). Various numerical experiments illustrate the theoretical findings.

The following is mainly taken from the scientific papers [4] and [5].

3.1 Introduction

For computing expectations of functionals depending on a stochastic process, Monte Carlo (MC) methods are an essential tool. In the context of stochastic differential equations (SDEs), sample paths of the solution are computed by a numerical integrator and the MC approach consists in approximating the expected value of a given functional of the solution by the average of the computed samples. Bias and statistical errors are introduced in such an approximation procedure. The bias of the method is related to the weak order of convergence of the considered numerical integrator, while the statistical error scales as the inverse of the square root of the number of samples and involves the variance of the process. This statistical error is a computational burden for many applications and many strategies to reduce the computational cost of MC method have been proposed. We mention the variance reduction techniques such as as estimators based on control variates or antithetic variates (see e.g. [49]).

A recent approach, originating with Heinrich [56] in the context of numerical quadrature, proposed by Giles [46] for SDEs is the so-called multilevel Monte Carlo (MLMC) method that allows to significantly speed up the classical MC method thanks to hierarchical sampling. The main idea of MLMC methods is to apply the MC method for a nested sequence of stepsizes while balancing the number of samples according to the stepsize. Precisely, consider the square root of the mean square error as a measure of the accuracy, and e.g. the Euler-Maruyama (EM) method [78] as the basic numerical integrator. Then, the computational cost of $\mathcal{O}(\varepsilon^{-3})$ for the MC method is reduced to $\mathcal{O}(\varepsilon^{-2}(\log(\varepsilon))^2)$ for the MLMC method to compute the expectation of functionals with an accuracy of $\mathcal{O}(\varepsilon)$.

However, this computational saving is obtained assuming that the coarsest levels of the MLMC method are *accessible*. But it is well known [57] that for classes of problems, e.g. stochastic partial differential equations discretized by the method of lines, stability issues with standard explicit methods can prevent to take coarse stepsizes. Indeed, the wide range of scales present in the SDEs forces the numerical integrator to resolve the fastest scale leading to a possible severe stepsize reduction. In this chapter we consider mean square stable stiff systems of SDEs for which standard explicit integrators, e.g. the well-known Euler-Maruyama (EM) method, face a severe stepsize restriction [57, 92]. Such problems and related computational issues arise in the modeling of many problems in biology, chemistry, physics or economics [30, 91, 39, 86]. We call these systems stiff. Another finding of this chapter is that even for SDEs usually characterized as *nonstiff* but with significant noise, stepsize restriction prevents to use the EM method for all levels of the MLMC. We emphasize that in the SDE context there exists, besides mean square stable problems, various other classes of interesting problems with multiple scales that need other numerical treatments [11, 102].

To the best of the author's knowledge the issue of applying MLMC method for stiff SDEs has not been addressed in the literature. We note however that related work extending the MLMC method for problems with multiple scales in space has recently been proposed in [3] in the context of numerical homogenization of stochastic elliptic multiscale PDEs.

One possible strategy to extend MLMC method for stiff problems is to use drift-implicit numerical methods with favorable mean square stability. When applicable, such methods are a good alternative to the stabilized method proposed in this chapter. We note however that for large problems, originating for example from a spatially discretized stochastic partial differential equations with stiff nonlinear terms (e.g. reaction terms), solving the full problem with an implicit method is sometimes very difficult if not impossible. For such problems, decoupling the diffusion operator (solved explicitly with a stabilized method [13, 2]) from the reaction terms (stiff problems of small dimension at each spatial node solved implicitly) using a splitting method [37] or a partitioned method [14] is a very efficient strategy. In both approaches [14, 37] the use of *explicit stabilized methods* is essential. Furthermore, the computational complexity for drift-implicit numerical methods is somehow less transparent than in the original MLMC method based on explicit integrators as the number of iterations to solve the nonlinear systems has to be accounted for. Precisely balancing the accuracy of the linear solver (e.g. iteration of an (inexact) Newton method) with the level of the MLMC method is a nontrivial task. We refer to [40] for such adaptive inexact Newton methods in the context of deterministic PDEs.

Here we explore another avenue and propose to stabilize the EM method to allow to access the coarse stepsizes of the MLMC method. In turn, our method is as easy to implement as the EM method and switching from such a code to a stabilized one as proposed here for a MLMC implementation is straightforward. For the stabilization procedure, we resort to the S-ROCK methods of weak order one, a class of explicit Chebyshev methods recently introduced for stiff stochastic problems [10, 12] and extended for higher weak order in [15]. The stabilized MLMC methods remain fully explicit, as easy to implement as the original MLMC methods based on the EM method but much more efficient as shown in Section 3.3.2. The S-ROCK methods consist in a family of numerical methods indexed by their stage number. This number can in turn vary to accommodate the required stability requirement. If only one stage is used, the S-ROCK method coincides with the EM method and for nonstiff problems we recover the classical MLMC method. Moreover we would like to point out that in [62] the divergence of the MLMC method using Euler-Maruyama for nonlinear SDEs is discussed.

This chapter is organized as follows. In Section 3.2 the numerical methods used in this chapter to approximate stochastic differential equations are described, the order of convergence and the stability are recalled. In Section 3.3 we discuss the issues faced by the standard MLMC approach in presence of stiffness. We then introduce our stabilized multilevel MC method and discuss its complexity. Using a numerical integrator with a higher weak order, we show in Section 3.4 how the stabilized MLMC method can be further improved. Numerical experiments on a one-dimensional linear SDE, a two-dimensional nonlinear SDE and a large system of SDEs originating from a SPDE are studied in Section 3.5 to illustrate the performance of our new MLMC method. In addition the improved stabilized multilevel Monte Carlo method is compared to the stabilized MLMC method and the standard MLMC method.

3.2 Preliminaries

In the following we consider stochastic processes $(X(t))_{t\in[0,T]}$ on the bounded interval $[0,T]$ described by the stochastic differential equation

$$\left\{\begin{array}{lcl} \mathrm{d}X(t) & = & f(X(t))\mathrm{d}t + \sum\limits_{r=1}^{m} g^r(X(t))\mathrm{d}W_r(t), \quad 0\le t\le T, \\ X(0) & = & X_0, \end{array}\right. \tag{3.1}$$

where $X(t)$ is a $\mathbb{R}^d$-valued random variable, $f : \mathbb{R}^d \to \mathbb{R}^d$ is the drift term, $g^r : \mathbb{R}^d \to \mathbb{R}^d$ with $r = 1,2,\dots,m$ are the diffusion terms and $W_r(t)$ with $r = 1,2,\dots,m$ are independent one-dimensional standard Brownian motions. For simplicity autonomous drift and diffusion functions are considered, but emphasize that a general SDE can always be transformed in such autonomous form. We assume standard Lipschitz and linear growth conditions on the drift and diffusion functions to ensure the existence of a strong solution of the SDE (3.1) (see [81, 66, 17]).

3.2.1 Numerical Schemes

To approximate numerically the solution of (3.1) we consider the discrete map

$$X_{n+1} = \Psi(X_n, h, \xi_n), \tag{3.2}$$

where $\Psi(\cdot, h, \xi_n) : \mathbb{R}^d \to \mathbb{R}^d$, $X_n \in \mathbb{R}^d$ for $n \geq 0$, h denotes the stepsize, and ξ_n denotes a random vector. We briefly recall two concepts of accuracy and stability for the numerical integration of SDEs, which we have presented in more details in Section 2.2. Suppose there is a numerical approximation as defined in (3.2). Consider any $\tau_n = nh \in [0,T]$ for h sufficiently small. The numerical approximation is said to be of strong order of convergence r_s if

$$\max_{0\leq n\leq T/h} \left(\mathbb{E}\left[|X_n - X(\tau_n)|^2\right]\right)^{1/2} \leq Ch^{r_s}$$

for a constant C (independent of h) (see Definition 2.2.1). It is said to be of weak order r_w if for any function $\phi \in C_P^{2(\gamma+1)}(\mathbb{R}^d, \mathbb{R})$ (with $C_P^{2(\gamma+1)}$ denoting the space of $2(\gamma+1)$ times continuously differentiable functions with all partial derivatives bounded by a term of order $1 + |x|^{2u}$ with $u \in \mathbb{N}$ (polynomial growth)) there exists a constant C (independent of h) such that

$$|\mathbb{E}[\phi(X_n)] - \mathbb{E}[\phi(X(\tau_n))]| \leq Ch^{r_w}$$

(see Definition 2.2.3).

Euler-Maruyama Method

The simplest method to approximate solutions to (3.1) is a generalization of the Euler method for ordinary differential equation (ODEs), the Euler-Maruyama method. Taking a uniform stepsize h, the method is defined by

$$X_{n+1} = X_n + hf(X_n) + \sum_{r=1}^{m} g^r(X_n)\Delta W_{n,r}, \tag{3.3}$$

where $\Delta W_{n,r} \sim \mathcal{N}(0,h),\ r = 1,2,\dots m$ are independent Wiener increments. This method has strong order $\frac{1}{2}$ and weak order 1 in general for a system of Itô SDEs [78]. As we will see in Section 3.2.2, the method (3.3) requires a stepsize restriction when applied to stiff stochastic problems.

S-ROCK1 and S-ROCK2 Methods

Stabilized explicit numerical integrators, that are efficient for stiff problems, are given by the so-called S-ROCK methods. S-ROCK methods are explicit orthogonal Runge-Kutta Chebyshev methods with an extended mean square stability domain (see Section 3.2.2). These methods have first been introduced for Stratonovich stochastic differential equations in [9, 10] and they have been extended to Itô SDEs in [12]. Here we will focus on the latter. In this chapters we use two different S-ROCK methods.

First, we consider the s-stage Itô S-ROCK method of weak order 1 and strong order $\frac{1}{2}$ (see [12]), the so-called S-ROCK1 method (see Definition 2.2.8). We recall its definition here. For all integer $s \geq 2$ we define the s-stage S-ROCK1 method as follows:

$$\begin{aligned}
K_0 &= X_{n-1} \\
K_1 &= X_{n-1} + h\tfrac{\omega_1}{\omega_0} f(K_0) \\
K_i &= 2h\omega_1 \tfrac{T_{i-1}(\omega_0)}{T_i(\omega_0)} f(K_{i-1}) + 2\omega_0 \tfrac{T_{i-1}(\omega_0)}{T_i(\omega_0)} K_{i-1} - \tfrac{T_{i-2}(\omega_0)}{T_i(\omega_0)} K_{i-2}, \; i = 2,3,\ldots,s-1, \\
K_s &= 2h\omega_1 \tfrac{T_{s-1}(\omega_0)}{T_s(\omega_0)} f(K_{s-1}) + 2\omega_0 \tfrac{T_{s-1}(\omega_0)}{T_s(\omega_0)} K_{s-1} - \tfrac{T_{s-2}(\omega_0)}{T_s(\omega_0)} K_{s-2} + \sum_{r=1}^{m} g^r(K_{s-1}) \Delta W_{n,r},
\end{aligned}$$

where $\omega_0 = 1 + \frac{\eta}{s^2}$, $\omega_1 = \frac{T_s(\omega_0)}{T_s'(\omega_0)}$ and $\Delta W_{n,r} = W_r(\tau_n) - W_r(\tau_{n-1})$ and we set $X_n = K_s$. Recall that the $(T_i(x))_{i\geq 0}$ are the orthogonal Chebyshev polynomials and that we call η the damping parameter, which can be used to adjust the stability domain. For $s = 1$ we take the Euler-Maruyama method (3.3), and thus, the s-stage Itô S-ROCK method well defined for any $s \geq 1$.

Second, we consider a S-ROCK method, which is as S-ROCK1 of strong order of convergence 1/2, but which is characterized by a weak order of convergence 2. We call this method in the following S-ROCK2. This numerical integrator was first introduced in [15]. Similar to S-ROCK1 this scheme uses a stabilization procedure (in this case ROCK2 [13]) on the first $s-2$ stages and then a finishing procedure on the last two stages.

Definition 3.2.1 (S-ROCK2 method)**.** *Let h be a uniform time stepsize. The s-stage S-ROCK2 method for* (3.1) *is defined for all $s \geq 2$ as follows:*

$$
\begin{aligned}
K_0 &= X_n \\
K_1 &= X_n + \mu_1 \alpha h f(K_0) \\
K_i &= \mu_i \alpha h f(K_{i-1}) - \nu_i K_{i-1} - \kappa_i K_{i-2}, \ i = 2, 3, \ldots, s, \\
K_{s-1}^* &= K_{s-2} + 2\tau_\alpha h f(K_{s-2}) + \sqrt{h} \sum_{r=1}^{m} g^r(K_s)\psi_r, \\
X_{n+1} &= K_{s-2} + \left(2\sigma_\alpha - \tfrac{1}{2}\right) h f(K_{s-2}) + \tfrac{1}{2} h f\left(K_{s-1}^*\right) \\
&\quad + \tfrac{1}{2} \sum_{r=1}^{m} \left(g^r\left(K_s + \sum_{q=1}^{m} g^q(K_s) J_{q,r}\right) - g^r\left(K_s - \sum_{q=1}^{m} g^q(K_s) J_{q,r}\right)\right) \\
&\quad + \tfrac{h}{2} \sum_{r=1}^{m} \left(g^r\left(K_{s-1} + \sqrt{\frac{h}{2}} \sum_{q=1}^{m} g^q(K_s)\xi_q\right) + g^r\left(K_{s-1} - \sqrt{\frac{h}{2}} \sum_{q=1}^{m} g^q(K_s)\xi_q\right)\right),
\end{aligned}
$$

where $\alpha, \sigma_\alpha, \tau_\alpha, \mu_i, \nu_i, \kappa_i, J_{q,r}$ and ξ_q are as defined in [15].

3.2.2 Stability of Numerical Methods

The efficiency of an approximation does not only depend on the order of convergence but also on its stability that is essential to correctly capture the long-time behavior of the exact solution. The stability of numerical methods has been studied in detail in Section 2.2.2. Recall that we consider the test problem

$$
\begin{cases}
\mathrm{d}X(t) = \lambda X(t)\mathrm{d}t + \mu X(t)\mathrm{d}W(t), & 0 \leq t, \lambda \in \mathbb{C}, \mu \in \mathbb{C}, \\
X(0) = 1,
\end{cases}
\tag{3.4}
$$

and that the stability domain of the test problem is given by

$$
\mathcal{S}_{exact} = \left\{ (\lambda, \mu) \in \mathbb{C}^2 \mid \mathcal{R}\{\lambda\} + \frac{1}{2}|\mu|^2 < 0 \right\}
\tag{3.5}
$$

(see (2.7)).

We also recall that the stability domain of the Euler-Maruyama method is characterized by

$$
\mathcal{S}_{EM} = \left\{ (p, q) \in \mathbb{C}^2 \mid |1 + p|^2 + q^2 < 1 \right\},
\tag{3.6}
$$

where $(p, q) = (h\lambda, \sqrt{h}|\mu|)$ (see [66]). Choosing $(\lambda, \mu) \in \mathbb{R}^2$ such that the linear SDE (3.4) is mean square stable, it can be shown that the stepsize h of the EM method has to satisfy

$$\rho_{EM} h := \frac{|\lambda|^2}{2|\lambda| - |\mu|^2} h < 1 \;\Leftrightarrow h < \frac{1}{\rho_{EM}} \tag{3.7}$$

to guarantee stability of the numerical scheme. In particular, for $\mu = 0$ (deterministic case), $\rho_{EM} = \frac{|\lambda|}{2}$ and $\rho_{EM} \to \infty$ for $|\mu|^2 \to 2|\lambda|$.

While remaining explicit the two S-ROCK methods have an extended stability domain, which can be characterized as follows. Recall that

$$\mathscr{S}_{SDE,a} = \left\{(p,q) \in [-a,0] \times \mathbb{R} \mid |q| \leq \sqrt{-2p}\right\}$$

represents a *portion* of $\mathscr{S}_{exact}$ and

$$a^* = \sup\left\{a > 0 \mid \mathscr{S}_{SDE,a} \subset \mathscr{S}_{num}\right\}$$

with $\mathscr{S}_{num}$ denoting the stability domain of the numerical method. It can be shown that for S-ROCK1 and S-ROCK2 $a_s^* = c_{SR1}(s)s^2$ and $a_s^* = c_{SR2}(s)(s+2)^2$, respectively. As s increases the constants $c_{SR1}(s)$ and $c_{SR2}(s)$ quickly reach a value independent of the stage number that can be estimated numerically as $c_{SR1} = 0.33$ (S-ROCK1) and $c_{SR2} = 0.42$ (S-ROCK2) [9, 12, 15]. We also recall that a crucial property of S-ROCK methods is that a_s^* grows quadratically with s, but the computational cost, measured by the number of function evaluations, only increases linearly with s. Similar to (3.7), for $(\lambda,\mu) \in \mathbb{R}^2$ such that the test problem (3.4) is stable, by choosing $a^* = |\lambda|$ and the stage number s such that

$$c_{SR1} s^2 := a^* h \quad \text{with } 0.33 \leq c_{SR1} \leq 1.01 \tag{3.8}$$

the S-ROCK methods are mean square stable for any stepsize h. It is worth noting that condition (3.8) is independent of the diffusion term μ, and we will define $\rho_{SR} := |\lambda|$.

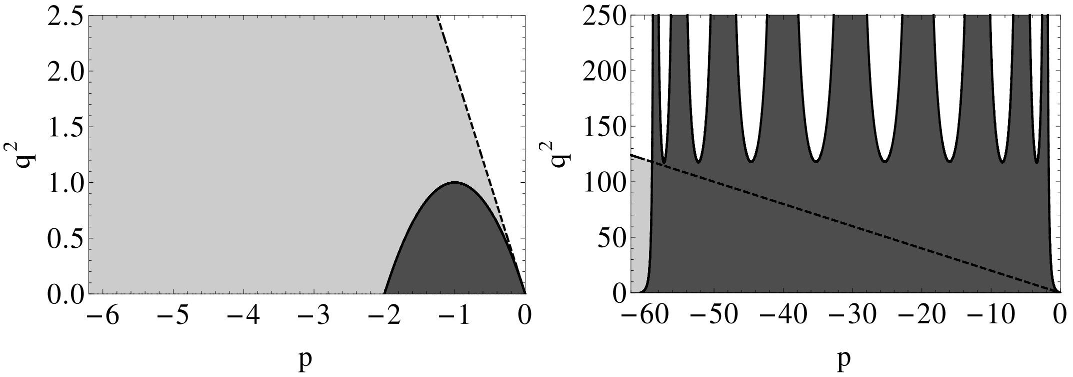

Figure 3.1: Stability regions (dark gray) for the Euler-Maruyama method (left-hand side) and the S-ROCK1 method with stage number $s = 10$ and damping parameter $\eta = 5.9$ (right-hand side). The dashed line delimits the stability region (light gray) of the test problem (3.5).

In Figure 3.1 we recall the stability domains of the Euler-Maruyama method and the S-ROCK1 method with $s = 10$ stages and damping parameter $\eta = 5.9$. For more details see Figure 2.9 and Figure 2.10.

3.3 Multilevel Monte Carlo Method for Stiff SDEs

The idea of the multilevel Monte Carlo method [46] is to apply the Monte Carlo method for several nested levels of stepsizes and to compute different numbers of paths on each level, from a few paths when the stepsize is small to many paths when the stepsize is large. By choosing the right balance between the stepsizes and the number of simulated trajectories at each level it is possible to reduce the computational complexity compared to that of the standard Monte Carlo method for a given mean square accuracy.

In the following we use the terms *computational cost* and *computational complexity* synonymously to represent the work of a numerical method defined as the number of function evaluations of a numerical discretization per sample path times the total number of sample paths. This measure of the complexity of numerical algorithms will be used when we compare the performance of various methods.

In this section we discuss the multilevel Monte Carlo method for stiff stochastic differential equations. In the following we first briefly recall the standard MLMC method and show why stability issues restrict this approach for stiff problems. We then present a stabilized multilevel Monte Carlo method using the S-ROCK method.

3.3.1 Standard Multilevel Monte Carlo Method

Here we present briefly the standard multilevel Monte Carlo approach introduced in [46], which we have also discussed as particular case in Section 2.3.3 to illustrate the MLMC idea. The version of the MLMC method that we give here uses a general refinement factor, i.e. we no longer assume that passing from a level to next one higher up the time intervals are split in two, they are split into k parts with k some integer larger than two.

Consider the diffusion process $(X(t))_{t\in[0,T]}$ (with T a fixed positive number) solution of the SDE (3.1) and a Lipschitz continuous function $\phi : \mathbb{R}^d \to \mathbb{R}$. Our aim is to estimate the expectation $\mathbb{E}\left[\phi(X(T))\right]$, which we denote by E, from many realizations of the numerical solution of (3.1). Let an integer $k \geq 2$ be the refinement factor and let an integer L be the total number of levels. The nested stepsizes of the multilevel Monte Carlo method are given by

$$h_l = \frac{T}{M_l}, \quad l = 0, 1, \ldots, L, \tag{3.9}$$

where $M_l = k^l$ indicates the number of time steps in the discretization over the time interval $[0, T]$ at level l. Let $\phi_l := \phi(X_{M_l}) \approx \phi(X(T))$ be an approximation of $\phi(X(T))$ using a numerical scheme with M_l discretization steps of size h_l. Applying the telescopic sum yields

$$\phi_L = \sum_{l=0}^{L} \left(\phi_l - \phi_{l-1}\right) \quad \text{with } \phi_{-1} \equiv 0.$$

The multilevel Monte Carlo estimator is defined by

$$E^* := \sum_{l=0}^{L} E_l^* \quad \text{with } E_l^* := \frac{1}{N_l} \sum_{i=1}^{N_l} \left(\phi_l^{(i)} - \phi_{l-1}^{(i)}\right)$$

a sample average over N_l independent samples. We emphasize that the estimates $\phi_l^{(i)}$ and $\phi_{l-1}^{(i)}$ are based on the same diffusion path, i.e. the same Brownian motion path. The mean square error, a measure of accuracy for estimators (see e.g. [49]), of E^* can be decomposed as

$$\begin{aligned} \mathrm{MSE}\left(E^*\right) = \mathbb{E}\left[\left(E^* - E\right)^2\right] &= \mathbb{E}\left[\left(E^* - \mathbb{E}\left[E^*\right]\right)^2\right] + \left(\mathbb{E}\left[E^*\right] - E\right)^2 \\ &= \mathrm{Var}\left(E^*\right) + \left(\mathrm{bias}\left(E^*\right)\right)^2. \end{aligned} \tag{3.10}$$

Since the estimates $\phi_l^{(i)}$ are identically distributed, the following holds

$$\mathbb{E}\left[E^*\right] = \sum_{l=0}^{L} \mathbb{E}\left[\frac{1}{N_l} \sum_{i=1}^{N_l} \left(\phi_l^{(i)} - \phi_{l-1}^{(i)}\right)\right] = \sum_{l=0}^{L} \mathbb{E}\left[\phi_l - \phi_{l-1}\right] = \mathbb{E}\left[\phi_L\right].$$

Using this equality and considering a numerical integrator with weak order of convergence 1 yields

$$\mathrm{bias}\left(E^*\right) = \mathbb{E}\left[E^*\right] - E = \mathbb{E}\left[\phi_L\right] - E = \mathcal{O}\left(k^{-L}\right). \tag{3.11}$$

Moreover, we observe that by the Cauchy-Schwarz inequality

$$\mathrm{Var}\left(\phi_l - \phi_{l-1}\right) \leq \left(\mathrm{Var}\left(\phi_l - E\right)^{1/2} + \mathrm{Var}\left(\phi_{l-1} - E\right)^{1/2}\right)^2 \tag{3.12}$$

and since ϕ is Lipschitz continuous and a strong order $\frac{1}{2}$ is assumed

$$\mathrm{Var}\left(\phi_l - E\right) \leq \mathbb{E}\left[\left(\phi_l - E\right)^2\right] = \mathbb{E}\left[\left(\phi\left(X_{M_l}\right) - \phi\left(X(T)\right)\right)^2\right] \leq C k^{-l}. \tag{3.13}$$

Therefore, using (3.12) and (3.13) we obtain

$$\mathrm{Var}\left(E^*\right) = \sum_{l=0}^{L} \frac{\mathrm{Var}\left(\phi_l - \phi_{l-1}\right)}{N_l} = C \sum_{l=0}^{L} \frac{k^{-l}}{N_l}, \tag{3.14}$$

where C is a positive constant. Assuming now a mean square accuracy of $\mathrm{MSE}\left(E^*\right) = \mathcal{O}\left(\varepsilon^2\right)$ and considering (3.10) and (3.11) yields $\varepsilon = k^{-L}$.

Inspired by (3.14) the number of simulations per level l is chosen such that $N_l = k^{2L} k^{-l} L$, which guarantees that $\mathrm{Var}\left(E^*\right) = \mathcal{O}\left(\varepsilon^2\right)$ as L tends to infinity. It is straightforward to show that the corresponding computational complexity of E^* is given by

$$\operatorname{Cost}\left(E^*\right) = \sum_{l=0}^{L} N_l M_l (1+m) = \mathcal{O}\left(\varepsilon^{-2}(\log(\varepsilon))^2\right),$$

which is a significant improvement over the standard Monte Carlo method with a computational complexity of $\mathcal{O}\left(\varepsilon^{-3}\right)$. However, one has to be careful when applying the standard MLMC approach for stiff systems as we show in the next section. Indeed, stability of the numerical method used in the standard multilevel Monte Carlo approach is assumed to ensure that all levels of the method are accessible [46]. This will not be the case for stiff problems as will be discussed in the next section.

Multilevel Monte Carlo Method for Stiff SDEs using Euler-Maruyama

Assume a mean square stable problem for which a standard numerical method is only mean square stable for a stepsize smaller than a certain threshold. In such a case the multilevel Monte Carlo method cannot be applied at the levels whose stepsize is larger than this threshold. Inspired by the mean square stable one-dimensional scalar linear SDE (3.4) the following stability constraint is assumed:

$$k^{-l_{EM}} \rho \le 1, \tag{3.15}$$

where l_{EM} corresponds to the largest possible stepsize $h_{l_{EM}}$ such that the Euler-Maruyama method is stable for a given stiffness parameter denoted by ρ.

Remark 3.3.1. *For example, for the test problem* (3.4), $\rho = \rho_{EM} = \frac{|\lambda|^2}{2|\lambda| - |\mu|^2}$, *and thus,* $l_{EM} = \frac{\log\left(\frac{|\lambda|^2}{2|\lambda|-|\mu|^2}\right)}{\log(k)}$ *(see* (3.7)*). For* $l < l_{EM}$ *the EM cannot be applied as the integration is unstable. We emphasize that large* l_{EM} *can arise in situations usually characterized as nonstiff, i.e. when* $|\lambda|$ *is small but* $|\mu|$ *close to* $\sqrt{2|\lambda|}$ *(see Figure 3.4).*

Remark 3.3.2. *Note that in* (3.15) *we assume a relatively small value of* T *and we willingly ignore* T*. For a large value of* T *the following results remain valid by replacing the stiffness parameter* ρ *by the product* $T\rho$*. Hence increasing* T *has the same effect as increasing the stiffness.*

Suppose a mean square accuracy of $k^{-2L} = \varepsilon^2$ is desired. We distinguish two cases.

(a) No MLMC: $l_{EM} > L$

If l_{EM} is larger than L, then all the stepsize h_l (with $l \in \{0, 1, \dots, L\}$) are too large to account for stability. Thus, the multilevel Monte Carlo approach cannot be applied and standard Monte Carlo has to be used instead with $M_{l_{EM}}$ time steps. Therefore, in this case a mean square accuracy of $\mathcal{O}\left(\varepsilon_{MC}^2\right)$ with $\varepsilon_{MC} = k^{-l_{EM}}$ is achieved and a computational cost of $\mathcal{O}\left(\varepsilon_{MC}^{-3}\right)$ is necessary. We emphasize that $\varepsilon_{MC} = k^{-l_{EM}}$ is smaller than the required accuracy $\varepsilon = k^{-L}$ and in turn $\mathcal{O}\left(\varepsilon_{MC}^{-3}\right)$ is larger than $\mathcal{O}\left(\varepsilon^{-3}\right)$.

(b) MLMC: $0 < l_{EM} \leq L$

If l_{EM} lies between 0 and L, only the levels $l_{EM}, l_{EM}+1, \ldots, L$ satisfy the stability constraint (3.15), and thus, the multilevel Monte Carlo estimator using the Euler-Maruyama scheme is defined by

$$\widetilde{E} := \sum_{l=l_{EM}}^{L} \widetilde{E}_l \quad \text{with } \widetilde{E}_l := \frac{1}{N_l} \sum_{i=1}^{N_l} \left(\phi_l^{(i)} - \phi_{l-1}^{(i)}\right)$$

a sample average over N_l independent samples, where $\phi_{l_{EM}-1} \equiv 0$. As in (3.10) the mean square error of $\widetilde{E}$ can be divided into bias and variance:

$$\operatorname{MSE}\left(\widetilde{E}\right) = \operatorname{Var}\left(\widetilde{E}\right) + \left(\operatorname{bias}\left(\widetilde{E}\right)\right)^2.$$

Taking into account the weak order of convergence 1 of the Euler-Maruyama scheme (see Section 3.2), the bias of $\widetilde{E}$ is of order k^{-L}, i.e. $\operatorname{bias}\left(\widetilde{E}\right) = \mathcal{O}\left(k^{-L}\right)$. Using the independence of the samples, ϕ being Lipschitz continuous and strong order of convergence $\frac{1}{2}$ of the Euler-Maruyama method, the variance of $\widetilde{E}$ satisfies

$$\operatorname{Var}\left(\widetilde{E}\right) = \sum_{l=l_{EM}+1}^{L} \frac{\operatorname{Var}\left(\phi_l - \phi_{l-1}\right)}{N_l} + \frac{\operatorname{Var}\left(\phi_{l_{EM}}\right)}{N_{l_{EM}}} = C \sum_{l=l_{EM}+1}^{L} \frac{k^{-l}}{N_l} + \frac{\operatorname{Var}\left(\phi_{l_{EM}}\right)}{N_{l_{EM}}}, \tag{3.16}$$

where C is a positive constant. Recall that a mean square accuracy of $k^{-2L} = \varepsilon^2$ is wanted. Inspired by (3.16), the number of simulations per level is chosen such that

$$N_l = \begin{cases} k^{2L} k^{-l} \left(L - (l_{EM}+1)\right) & \text{if } l \in \{l_{EM}+1, l_{EM}+2, \ldots, L\}, \\ k^{2L} & \text{if } l = l_{EM}. \end{cases} \tag{3.17}$$

Hence, for the variance of our estimator $\operatorname{Var}\left(\widetilde{E}\right) = \mathcal{O}\left(\varepsilon^2\right)$ holds and the mean square error is indeed $\operatorname{MSE}\left(\widetilde{E}\right) = \mathcal{O}\left(\varepsilon^2\right)$. We compute now the computational complexity that is necessary to achieve such a mean square accuracy. Taking the choice of N_l in (3.17) into consideration, we obtain a computational complexity of

$$\begin{aligned}
\mathrm{Cost}\left(\widetilde{E}\right) &= \sum_{l=l_{EM}+1}^{L} N_l M_l (1+m) + N_{l_{EM}} M_{l_{EM}} (1+m) \\
&= \sum_{l=l_{EM}+1}^{L} k^{2L-l} \left(L-(l_{EM}+1)\right) k^{l} (1+m) + k^{2L} k^{l_{EM}} (1+m) \\
&= k^{2L}(1+m)\left[\left(L-(l_{EM}+1)\right)\left(L-l_{EM}\right)+k^{l_{EM}}\right] \\
&= \varepsilon^{-2}(1+m)\left[\left(\frac{\log\left(\varepsilon^{l_{EM}/L}\right)-\log(\varepsilon)}{\log(k)}-1\right)\left(\frac{\log\left(\varepsilon^{l_{EM}/L}\right)-\log(\varepsilon)}{\log(k)}\right)+\varepsilon^{-l_{EM}/L}\right]. \\
&\leq \varepsilon^{-2}(1+m)\left[C\left(\frac{\log(\varepsilon)}{\log(k)}\right)^2+\varepsilon^{-l_{EM}/L}\right] = \mathcal{O}\left(\varepsilon^{-2}\left(\left(\log(\varepsilon)\right)^2+\varepsilon^{-l_{EM}/L}\right)\right),
\end{aligned}$$

where C is a positive constant.

Remark 3.3.3. *Note that as l_{EM} tends to L, the computational cost of $\widetilde{E}$ tends to $\mathcal{O}\left(\varepsilon^{-3}\right)$, the computational cost of the standard Monte Carlo approach. If l_{EM} tends to zero, the computational cost tends to $\mathcal{O}\left(\varepsilon^{-2}\left(\log(\varepsilon)\right)^2\right)$, which is the computational cost of the multilevel Monte Carlo method for nonstiff SDEs. Indeed in that case, there is no stepsize restriction for the EM method.*

3.3.2 Stabilized Multilevel Monte Carlo Method

We describe now a stabilized multilevel Monte Carlo method, which enables us to use all the levels of the MLMC approach even in presence of stiffness. As numerical integrator we use the S-ROCK1 method presented in Section 3.2. The following stability constraint is taken into account: (for $s_l \geq 2$)

$$\frac{k^{-l}\rho}{c_{SR1} s_l^2} \leq 1, \tag{3.18}$$

where the stiffness parameter ρ and c_{SR1} are two positive constants. For the test problem (3.4), $\rho = \rho_{SR} = |\lambda|$. In other words, the number of stages at level l satisfies $s_l \geq \max\left(\sqrt{\frac{\rho}{c_{SR1}}} k^{-l/2}, 2\right)$.

For the same reasons as in Remark 3.3.2 a $\mathcal{O}(1)$ value for T is assumed.

Remark 3.3.4. *The value of c_{SR1} depends on s_l, but it can be estimated numerically for any $s_l \geq 2$ (see Abdulle and Li [12]). It lies between* 0.33 *and* 1.01.

Using the same notation as above, the stabilized multilevel Monte Carlo estimator is given by

$$\widehat{E} := \sum_{l=0}^{L} \widehat{E}_l \quad \text{with } \widehat{E}_l := \frac{1}{N_l}\sum_{i=1}^{N_l}\left(\phi_l^{(i)} - \phi_{l-1}^{(i)}\right)$$

a sample average over N_l independent samples. Again we emphasize that the estimates $\phi_l^{(i)}$ and $\phi_{l-1}^{(i)}$ are based on the same Brownian motion path. The mean square error of the stabilized estimator $\widehat{E}$ can be decomposed as in (3.10)

$$\mathrm{MSE}\left(\widehat{E}\right) \quad = \quad \mathrm{Var}\left(\widehat{E}\right) + \left(\mathrm{bias}\left(\widehat{E}\right)\right)^2. \tag{3.19}$$

By the weak order of convergence 1 of the S-ROCK1 scheme (see Section 3.2), for the bias the following holds:

$$\mathrm{bias}\left(\widehat{E}\right) = \mathbb{E}\left[\widehat{E}\right] - E = \mathbb{E}\left[\phi_L\right] - E = \mathcal{O}\left(k^{-L}\right).$$

For the variance of $\widehat{E}$ we obtain

$$\mathrm{Var}\left(\widehat{E}\right) = \sum_{l=0}^{L} \frac{\mathrm{Var}\left(\phi_l - \phi_{l-1}\right)}{N_l} = C \sum_{l=0}^{L} \frac{k^{-l}}{N_l} \tag{3.20}$$

with C a positive constant. To establish this we have used the independence of the samples, ϕ being Lipschitz continuous and strong order of convergence $\frac{1}{2}$ of the S-ROCK1 method (see Section 3.2). Suppose now that a mean square accuracy of $\mathrm{MSE}\left(\widehat{E}\right) = \mathcal{O}\left(\varepsilon^2\right)$ with $\varepsilon = k^{-L}$ is desired. Inspired by (3.20), we set the number of simulations per level l to $N_l = k^{2L}k^{-l}L$ such that $\mathrm{Var}\left(\widehat{E}\right) = Ck^{-2L}\left(1 + \frac{1}{L}\right)$, and thus, $\mathrm{MSE}\left(\widehat{E}\right) = \mathcal{O}\left(\varepsilon^2\right)$. The computational complexity to achieve such a mean square accuracy is given by

$$\begin{aligned}
\mathrm{Cost}\left(\widehat{E}\right) \quad &= \quad \sum_{l=0}^{L} N_l M_l (s_l + m) = k^{2L} L \left(\sqrt{\frac{\rho}{c_{SR1}}} \sum_{l=0}^{L} k^{-l/2} + m(L+1) \right) \\
&= \quad k^{2L} L \left(\sqrt{\tfrac{\rho}{c_{SR1}}} \tfrac{\sqrt{k} - k^{-L/2}}{\sqrt{k}-1} + m(L+1) \right) \\
&= \quad \varepsilon^{-2} \left(-\tfrac{\log(\varepsilon)}{\log(k)} \right) \left(\sqrt{\tfrac{\rho}{c_{SR1}}} \tfrac{\sqrt{k} - \varepsilon^{1/2}}{\sqrt{k}-1} + m \left(-\tfrac{\log(\varepsilon)}{\log(k)} + 1 \right) \right) \\
&\leq \quad C\varepsilon^{-2} \left(\left|\log(\varepsilon)\right| \sqrt{\rho} + m(\log(\varepsilon))^2 \right) = \mathcal{O}\left(\varepsilon^{-2} \left(\log(\varepsilon)\right)^2 \left(1 + \tfrac{\sqrt{\rho}}{|\log(\varepsilon)|} \right) \right),
\end{aligned} \tag{3.21}$$

where C is a positive constant. Note that we recover the result for nonstiff problems up to a factor $\sqrt{\rho}$. It is also worth noting that using MLMC with Euler-Maruyama for stiff SDEs only standard Monte Carlo can be applied in the case $l_{EM} > L$, see Section 3.3.1. The resulting computational complexity is given by $\mathcal{O}\left(\varepsilon_{MC}^{-3}\right)$. Taking into account that $\varepsilon = k^{-L} > k^{-l_{EM}} = \varepsilon_{MC}$, one observes that the computational cost for stabilized MLMC is significantly smaller.

Remark 3.3.5. *If $l_{EM} \leq L$, then the Euler-Maruyama method can be applied from level l_{EM} up to level L. The variance* (3.20) *can be decomposed as*

$$Var\left(\widehat{E}\right) = \sum_{l=0}^{l_{EM}-1} \frac{Var\left(\phi_l - \phi_{l-1}\right)}{N_l} + \sum_{l=l_{EM}}^{L} \frac{Var\left(\phi_l - \phi_{l-1}\right)}{N_l}.$$

Using the strong convergence of order 1/2 of the numerical schemes yields

$$Var(\widehat{E}) = C\left(\sum_{l=0}^{l_{EM}-1} \frac{k^{-l}}{N_l} + \sum_{l=l_{EM}}^{L} \frac{k^{-l}}{N_l}\right).$$

Inspired by this decomposition, the number of simulations per level is chosen according to $N_l = k^{2L}k^{-l}(l_{EM}-1)$ *for* $l \in \{0,1,\ldots,l_{EM}-1\}$ *and* $N_l = k^{2L}k^{-l}(L-l_{EM})$ *for* $l \in \{l_{EM}, l_{EM}+1,\ldots,L\}$ *such that* $MSE(\widehat{E}) = \mathcal{O}\left(\varepsilon^2\right)$ *with* $\varepsilon = k^{-L}$. *The resulting computational cost is similar.*

Stabilized Multilevel Monte Carlo versus Stabilized Single-Level Monte Carlo

In the previous section we have seen that the multilevel Monte Carlo method with S-ROCK1 as numerical integrator requires a computational cost of $\text{Cost}(\widehat{E})$, as specified in (3.21), to achieve a mean square accuracy of $\text{MSE}\left(\widehat{E}\right) = \mathcal{O}\left(\varepsilon^2\right)$.

Using the same numerical method and the same mean square accuracy, the standard Monte Carlo method satisfies $M_L = \mathcal{O}\left(\varepsilon^{-1}\right), N_L = \mathcal{O}\left(\varepsilon^{-2}\right)$ and $s_L = \mathcal{O}\left(\sqrt{\varepsilon\rho}\right)$ (due to the stability criterion (3.18)), and thus, the computational cost required is given by

$$\text{Cost}\left(\widehat{E}_{MC}\right) = M_L N_L (s_L + m) = \mathcal{O}\left(\varepsilon^{-5/2}\sqrt{\rho} + \varepsilon^{-3}\right). \tag{3.22}$$

Remark 3.3.6. *In applications* ε *corresponds to the user's desired accuracy. As in the multilevel construction, Monte Carlo using S-ROCK1 can be applied for any* ε*, whereas Monte Carlo using Euler-Maruyama can be subject to stepsize restriction, and thus, one is forced to choose an* $\bar{\varepsilon}$ *which is significantly smaller than the user's desired accuracy* ε.

Figure 3.2 compares the computational cost of the stabilized MLMC method and the standard MC method using S-ROCK1 as a basic integrator against the finest stepsize h_L for $k = 2$, $m = 1$ and different values of the stiffness parameter ρ with $\rho \in \{1, 1000\}$. Recall that $\varepsilon = k^{-L} = h_L$, and thus, as h_L decreases the accuracy increases. One observes that for any stiffness ρ, as h_L decreases the stabilized multilevel Monte Carlo method prevails over the Monte Carlo method based on S-ROCK1. For instance, in Fig. 3.2 (b) for $\rho = 1000$, at $h_L = 2^{-20}$ the computational cost of Monte Carlo is about 10^3 times larger than the computational cost of multilevel Monte Carlo.

Note that as the stiffness ρ increases, the number of stages per level s_l increases, and thus, the computational complexity. Since the standard Monte Carlo method only uses s_L stages, whereas the MLMC method uses at each level l s_l stages, the number of function evaluations for standard MC is smaller than for MLMC for small values of L. However, as L increases, the MLMC approach significantly reduces the computational cost compared to the MC approach.

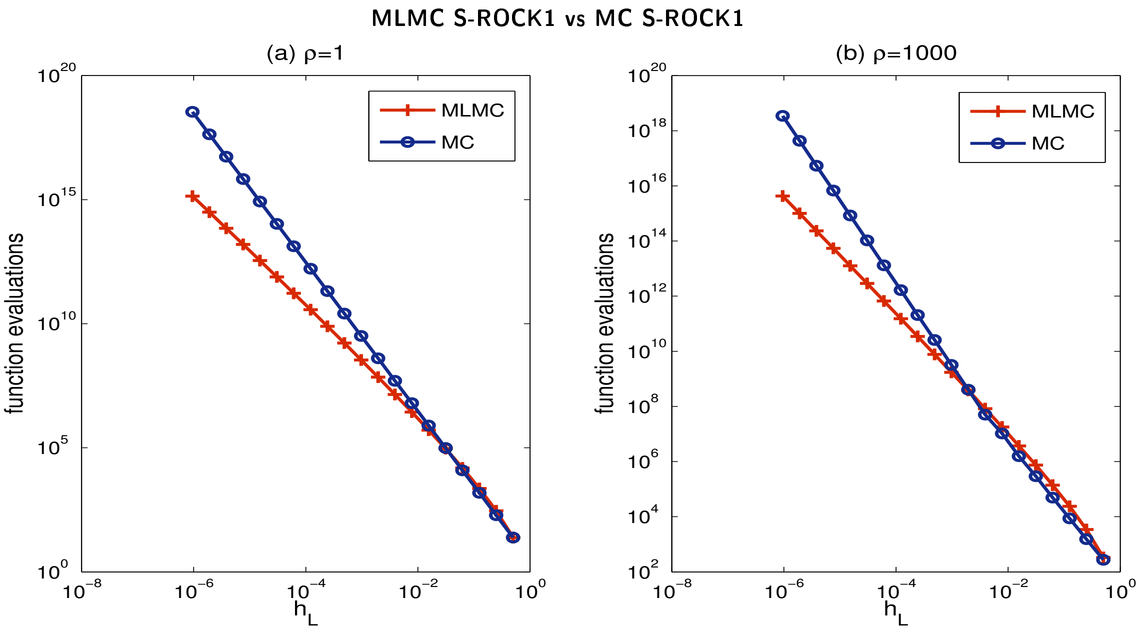

Figure 3.2: Computational cost of the stabilized multilevel Monte Carlo and the standard Monte Carlo method (using S-ROCK1), respectively, against the finest stepsize h_L for different values of the stiffness parameter ρ.

3.4 Improved Stabilized Multilevel Monte Carlo Method for Stiff SDEs

In this section we describe how the stabilized multilevel Monte Carlo method presented in Section 3.3.2 can further be improved. As mentioned in Section 3.2.1, the Euler-Maruyama method as well as the S-ROCK1 method are both of weak order 1 and strong order 1/2. The idea is to use a numerical integrator of higher weak order for the finest time grid (see [34]), in our case S-ROCK2 [15] with weak order 2, which leads to a reduction of the bias. In fact, due to the telescopic sum representation of the multilevel estimator, only the estimator based on the smallest time stepsize (which uses S-ROCK2) appears in the bias. A smaller bias yields a reduction of the total number of levels, and thus, a reduced computational cost without decreasing the accuracy. Note that in the following we focus on problems that are either stiff or nonstiff but with significant noise. Problems with no stability issues can be treated in a similar way.

Recall the sequence of nested stepsizes (3.9). For $l = 0, 1, \ldots, L-1$ we denote by ϕ_l the approximation of $\phi(X(T))$ using S-ROCK1 with time stepsize h_l. The approximation of $\phi(X(T))$ using S-ROCK2 on the finest time grid which is based on h_L is indicated by ϕ_L. The improved stabilized multilevel Monte Carlo estimator is defined by

$$\widetilde{E} := \sum_{l=0}^{L} \frac{1}{N_l} \sum_{i=1}^{N_l} \left(\phi_l^{(i)} - \phi_{l-1}^{(i)} \right) \quad \text{with } \phi_{-1} \equiv 0, \tag{3.23}$$

a sum of sample averages over N_l independent and identically distributed samples. Note that $\phi_l^{(i)}$ and $\phi_{l-1}^{(i)}$ are based on the same Wiener path. The accuracy of the estimator $\widetilde{E}$ can be measured, e.g. by the mean square error (see e.g. [49]), which can be split into bias and variance as follows:

$$\mathrm{MSE}\left(\widetilde{E}\right) = \mathbb{E}\left[\left(\widetilde{E}-E\right)^2\right] = \mathrm{Var}\left(\widetilde{E}\right) + \left(\mathrm{bias}\left(\widetilde{E}\right)\right)^2.$$

Using the properties of the expectation we obtain

$$\mathbb{E}\left[\widetilde{E}\right] = \sum_{l=0}^{L}\left(\mathbb{E}\left[\phi_l\right] - \mathbb{E}\left[\phi_{l-1}\right]\right) = \mathbb{E}\left[\sum_{l=0}^{L}\left(\phi_l - \phi_{l-1}\right)\right] = \mathbb{E}\left[\phi_L\right].$$

Hence the bias satisfies

$$\mathrm{bias}\left(\widetilde{E}\right) = \mathbb{E}\left[\widetilde{E}\right] - E = \mathbb{E}\left[\phi_L\right] - E = \mathcal{O}\left(k^{-2L}\right) \tag{3.24}$$

since the S-ROCK2 method, on which ϕ_L is based, is of weak order 2. Furthermore, for the variance we use the Cauchy-Schwarz inequality to obtain

$$\mathrm{Var}\left(\phi_l - \phi_{l-1}\right) \leq \left(\mathrm{Var}\left(\phi_l - E\right)^{1/2} + \mathrm{Var}\left(\phi_{l-1} - E\right)^{1/2}\right)^2.$$

Both numerical integrators, S-ROCK1 and S-ROCK2, are of strong order 1/2 and ϕ is Lipschitz continuous by assumption. Thus

$$\mathrm{Var}\left(\phi_l - E\right) \leq \mathbb{E}\left[\left(\phi_l - E\right)^2\right] \leq \mathbb{E}\left[\left(\phi\left(X_{M_l}\right) - \phi\left(X(T)\right)\right)^2\right] \leq Ck^{-l}$$

and therefore

$$\mathrm{Var}\left(\widetilde{E}\right) = \sum_{l=0}^{L}\frac{\mathrm{Var}\left(\phi_l - \phi_{l-1}\right)}{N_l} \leq C\left(\sum_{l=0}^{L-1}\frac{k^{-l}}{N_l} + \frac{k^{-L}}{N_L}\right), \tag{3.25}$$

where C is a positive constant.

Assume now a mean square precision of $\mathrm{MSE}\left(\widetilde{E}\right) = \mathcal{O}\left(\varepsilon^2\right)$ is desired for some $\varepsilon > 0$. Considering (3.24) we obtain a total number of levels $L = -\frac{1}{2}\frac{\log(\varepsilon)}{\log(k)}$ (or equivalently $\varepsilon = k^{-2L}$). Inspired by (3.25) the number of simulations per level l is set to $N_l = k^{-l}k^{4L}(L-1)$ for $l = 0, 1, \ldots, L-1$ and $N_L = k^{-L}k^{4L}$, which yields $\mathrm{Var}\left(\widetilde{E}\right) \leq Ck^{-4L}\left(2 + \frac{1}{L-1}\right) = \mathcal{O}\left(\varepsilon^2\right)$.

As mentioned above the stability constraint of S-ROCK1 is given by $\frac{k^{-l}\rho}{c_{SR1}s_l^2} \leq 1$. In a similar way one can define a stability criterion for S-ROCK2 $\frac{k^{-L}\rho}{c_{SR2}(s_L+2)^2} \leq 1$ with $s_L \geq 2$ and with c_{SR2} as defined above.

Theorem 3.4.1. *Let* $\widetilde{\mathrm{E}}$ *be the improved stabilized MLMC estimator introduced in* (3.23). *For a desired mean square accuracy of* $MSE\left(\widetilde{E}\right) = \mathcal{O}\left(\varepsilon^2\right)$ *the computational cost of* $\widetilde{\mathrm{E}}$ *is given by*

$$Cost\left(\widetilde{E}\right) = \frac{1}{4}\varepsilon^{-2}\left(\frac{\log(\varepsilon)}{\log(k)}\right)^2\widetilde{\alpha},$$

where $\widetilde{\alpha} = \left(m\frac{L-1}{L} + \frac{1}{L}\left(\frac{\sqrt{k}}{\sqrt{k}-1}\right)\sqrt{\frac{\rho}{c_{SR1}}}\right) - \left(d_1\frac{\varepsilon^{1/4}\sqrt{\rho}}{L} + d_2\frac{(\sqrt{\rho}-d_3)}{L^2}\right)$ *with* d_1, d_2, d_3 *some positive con-*

stants.

Proof. For the computational cost of $\widetilde{E}$ we obtain $\mathrm{Cost}\left(\widetilde{E}\right)$

$$\begin{aligned}
&= \sum_{l=0}^{L-1} N_l M_l \left(s_l + m\right) + N_L M_L \left(s_L + 8 + 2m\right) \\
&= \sum_{l=0}^{L-1} k^{4L}(L-1)\left(\sqrt{\frac{\rho}{c_{SR1}}}k^{-l/2} + m\right) + k^{4L}\left(\sqrt{\frac{\rho}{c_{SR2}}}k^{-L/2} + 6 + 2m\right) \\
&= k^{4L}(L-1)\left(\sqrt{\frac{\rho}{c_{SR1}}}\frac{\sqrt{k}-k^{-L/2+1/2}}{\sqrt{k}-1} + mL\right) + k^{4L}\left(\sqrt{\frac{\rho}{c_{SR2}}}k^{-L/2} + 6 + 2m\right).
\end{aligned}$$

Using $\varepsilon = k^{-2L}$ and rearranging terms yields $\mathrm{Cost}\left(\widetilde{E}\right) = \frac{1}{4}\varepsilon^{-2}\left(\frac{\log(\varepsilon)}{\log(k)}\right)^2 \widetilde{\alpha}$ with $\widetilde{\alpha}$ as defined above. ☐

In comparison, for a same mean square accuracy, the cost of the stabilized MLMC estimator $\widehat{E}$ of [4] is given by $\mathrm{Cost}\left(\widehat{E}\right) = \varepsilon^{-2}\left(\frac{\log(\varepsilon)}{\log(k)}\right)^2 \widehat{\alpha}$, where $\widehat{\alpha} = \left(m\frac{L+1/2}{L} + \frac{1}{2L}\left(\frac{\sqrt{k}}{\sqrt{k}-1}\right)\sqrt{\frac{\rho}{c_{SR1}}}\right) - d_4\frac{\varepsilon^{1/2}\sqrt{\rho}}{L}$ with d_4 a positive constant.

Asymptotically we observe that for both estimators

$$\mathrm{Cost}\left(\widetilde{E}\right) = \mathrm{Cost}\left(\widehat{E}\right) = \mathcal{O}\left(\varepsilon^{-2}\left(\log(\varepsilon)\right)^2\left(1 + \frac{\sqrt{\rho}}{|\log(\varepsilon)|}\right)\right),$$

however with a smaller constant prefactor for $\widetilde{E}$ allowing for a cost reduction by a factor roughly between 0.25 (nonstiff problems but significant noise) and 0.5 (stiff problems). This can be seen by comparing $\widetilde{\alpha}$ and $\widehat{\alpha}$.

3.5 Numerical Examples

In this section we study the multilevel Monte Carlo method for stiff stochastic differential equations numerically. Comparisons of the MLMC method for SDEs using S-ROCK1 and Euler-Maruyama, respectively, are carried out first on a one-dimensional linear SDE, followed by a two-dimensional nonlinear SDE and finally on a stochastic partial differential equation. The last numerical experiment is based again on a two-dimensional nonlinear SDE and the improved stabilized multilevel Monte Carlo method is compared to the stabilized MLMC method and the standard MLMC approach. In the following we use a refinement factor of $k = 2$.

3.5.1 Linear Stochastic Differential Equation

The first problem taken into account is the scalar linear test problem (3.4) with $t \in [0,1]$. To test numerically how well the stabilized MLMC method using S-ROCK1 performs compared to the MLMC method using Euler-Maruyama, we count the number of function evaluations (adding the number of drift and diffusion evaluations) using a total number of levels L, where $L \in \{1, 2, \ldots, 15\}$.

We consider two scenarios:

- a setting usually considered as nonstiff where $\lambda = -1$ and $\mu = \sqrt{-2\lambda - \delta}$ with $0 < \delta \leq 2$. In this scenario, as $\delta \to 0$ the parameters of the exact SDE are approaching the boundary of the exact stability domain. Then the gap between the EM mean square stability domain and the boundary of the true stability domain (see Figure 3.3 left) triggers an increasingly severe stepsize restriction. In turn only limited levels of the MLMC are accessible. In contrast the stabilized MLMC is always applicable and the mean square stability region for large value of $|\mu|$ is much larger and moreover not vanishing with increasing value of $|\mu|$ belonging to the true mean square stability region (see Figure 3.3 right).
- a setting considered as stiff with $\lambda \in \{-1, -100, -10000\}$ and $\mu = \sqrt{|\lambda|}$ where as expected, a decreasing number of levels are accessible for the MLMC based on the EM method in contrast to the stabilized MLMC.

In Figure 3.4 (a)-(c) we report the results for the first scenario and monitor the number of function evaluations required for the stabilized MLMC method and the MLMC method using EM, respectively, against the finest stepsize h_L. The diffusion is chosen such that $\delta = 0.1$ (Fig. 3.4 (a)), $\delta = 0.01$ (Fig. 3.4 (b)) and $\delta = 0.0001$ (Fig. 3.4 (c)). As expected, the stabilized MLMC method prevails over the MLMC method using Euler-Maruyama. The latter is subject to a stepsize restriction which becomes more severe for decreasing δ.

In Figure 3.4 (d)-(f) we report the results for the second scenario comparing the the stabilized MLMC method and the MLMC method using EM We consider a varying drift term with $\lambda \in \{-1, -100, -10000\}$ and a diffusion term given by $\mu = |\lambda|^{1/2}$. In all cases the parameters (λ, μ) lie in the stability region of the test problem (3.5). One observes that as $|\lambda|$ increases, the EM approach can only be used from a certain stepsize on, whereas the S-ROCK1 approach can be used for any stepsize.

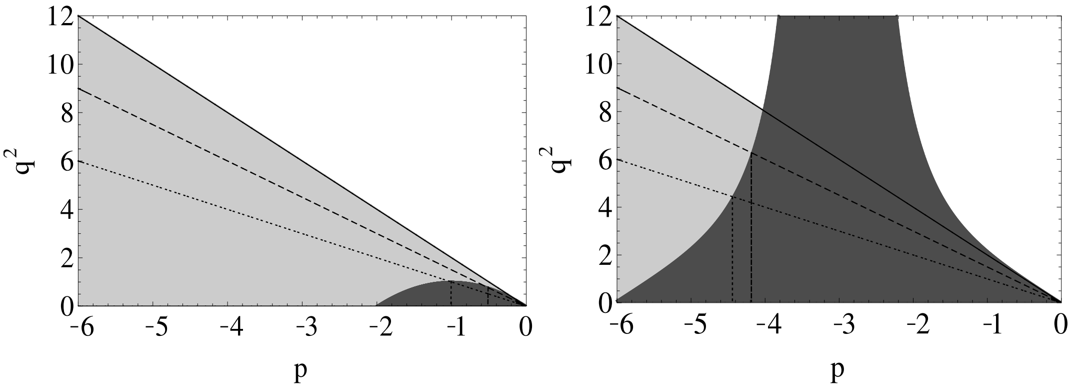

Figure 3.3: One-dimensional linear SDE: Stability regions of the test problem (3.4) (light gray), the EM method (dark gray, left-hand side) and the S-ROCK1 method with $s = 2$ (dark gray, right-hand side). Straight lines of slope -1.5 (dashed) and -1 (dotted), respectively.

3.5.2 Nonlinear Stochastic Differential Equation

The second stiff numerical experiment that we consider here is a two-dimensional noncommutative stiff SDE given by

$$\begin{cases} \mathrm{d}\begin{pmatrix} X_1(t) \\ X_2(t) \end{pmatrix} = \begin{pmatrix} \alpha(X_2(t)-1) - \lambda_1 X_1(t)(1-X_1(t)) \\ -\lambda_2 X_2(t)(1-X_2(t)) \end{pmatrix} \mathrm{d}t \\ \qquad + \begin{pmatrix} -\mu_1 X_1(t)(1-X_1(t)) \\ -\mu_2 X_2(t)(1-X_2(t)) \end{pmatrix} \mathrm{d}W_1(t) \\ \qquad + \begin{pmatrix} -\mu_2(1-X_1(t)) \\ 0 \end{pmatrix} \mathrm{d}W_2(t), \quad 0 \leq t \leq T, \\ \begin{pmatrix} X_1(0) \\ X_2(0) \end{pmatrix} \text{given}, \end{cases} \tag{3.26}$$

where $(W_1(t))_{t\in[0,T]}$ and $(W_2(t))_{t\in[0,T]}$ represent two independent standard Brownian motions. This model is inspired by the one-dimensional population dynamic model (see [84]). One can observe that $(X_1(t), X_2(t)) = (1,1)\ \forall t \in [0,T]$ represents a stationary solution of (3.26). We carry out a similar numerical experiment as in Section 3.5.1 by comparing the MLMC method using S-ROCK1 and Euler-Maruyama, respectively. As parameter we choose $T = 1, L \in$

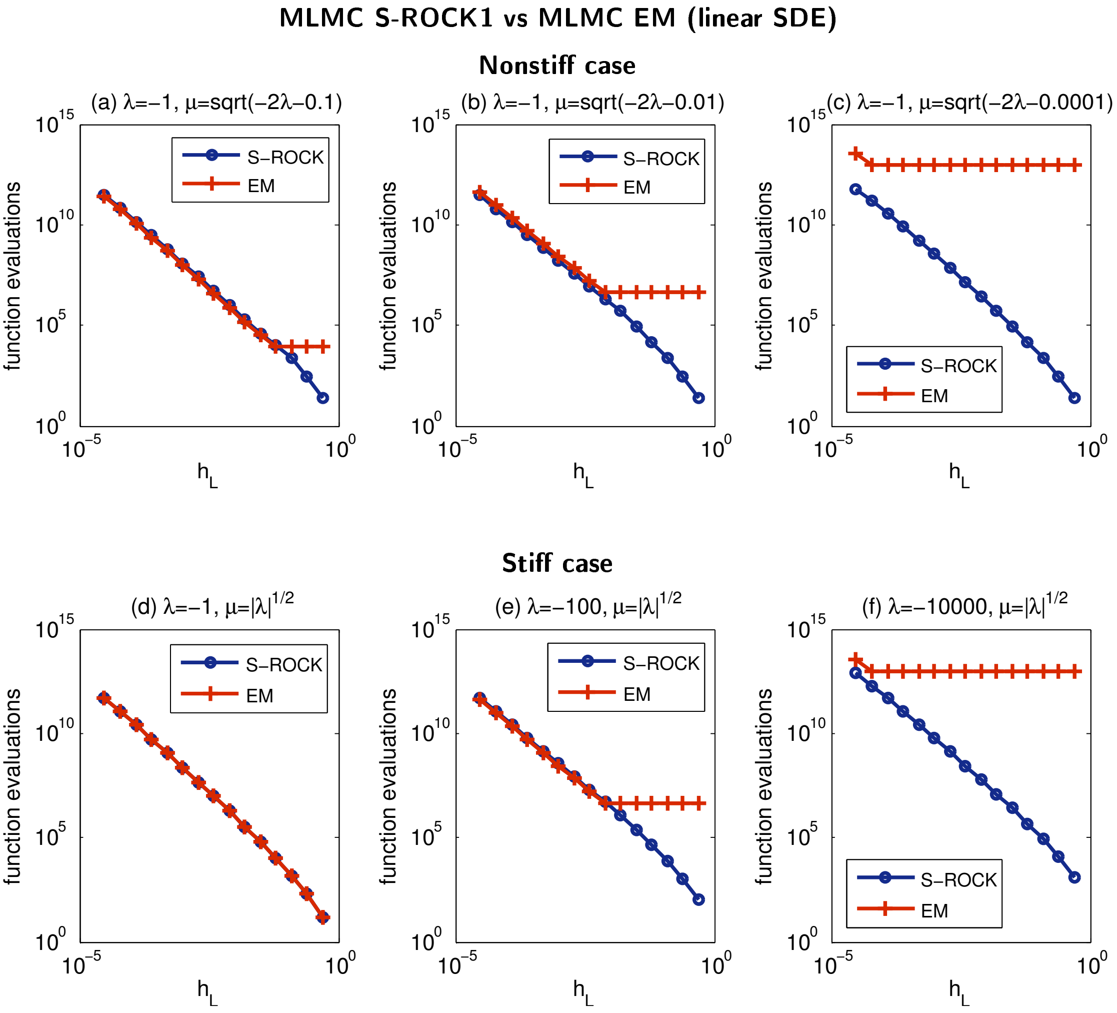

Figure 3.4: One-dimensional linear SDE: Function evaluations against finest stepsize h_L comparing the MLMC method using S-ROCK1 with the MLMC method using Euler-Maruyama.

$\{1,2,\ldots,10\}, \alpha = 2, \lambda_2 = -1, \mu_2 = 0.5$ with (λ_1, μ_1) the same as (λ, μ) in the previous section. As initial condition we pick $(X_1(0), X_2(0)) = (0.95, 0.95)$. Note that the two sets of parameters (λ_1, μ_1) and (λ_2, μ_2) both lie in the stability domain with (λ_1, μ_1) governing the stiffness of the SDE.

Figure 3.5 illustrates the number of function evaluations against the finest stepsize for the two-dimensional nonlinear noncommutative SDE given in (3.26). The results are similar to the ones of the scalar linear SDE. Note that stability of the approximations has been checked by looking at the second moment at the time end point $T = 1$. The S-ROCK1 approach can be applied under any choice of the finest stepsize h_L, whereas the Euler-Maruyama approach has some severe stepsize restrictions. Again this corroborates our theoretical findings and illustrates the significant improvement of the stabilized MLMC over the standard MLMC method.

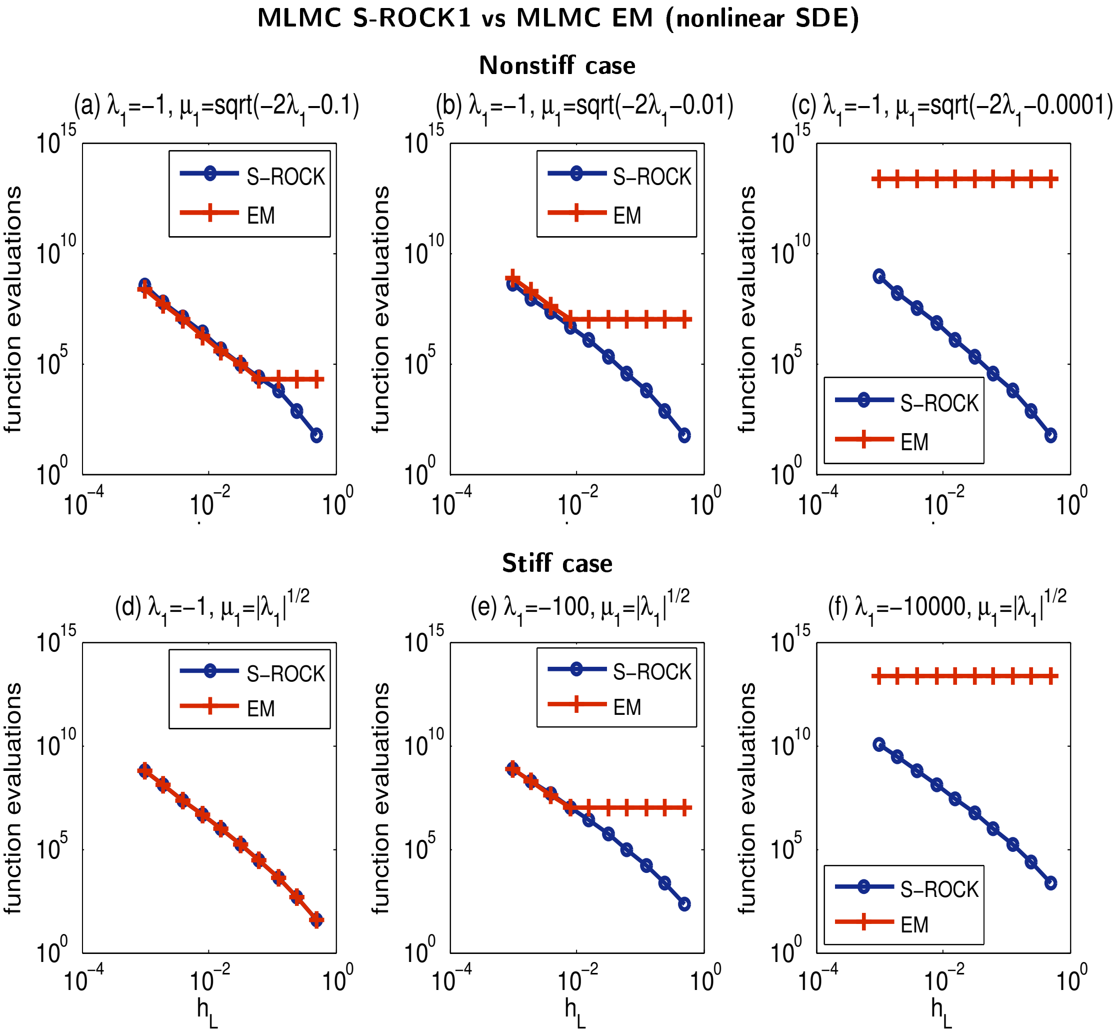

Figure 3.5: Two-dimensional nonlinear noncommutative stiff SDE: Function evaluations against finest stepsize h_L comparing the MLMC method using S-ROCK1 with the MLMC method using Euler-Maruyama for different values of λ_1 and μ_1.

We next study the error behavior of the multilevel Monte Carlo method using S-ROCK1 applied to the two-dimensional noncommutative nonlinear SDE (3.26). We focus again on the second moment of the stochastic process $\begin{pmatrix} X_1(t) \\ X_2(t) \end{pmatrix}$ at the time end point $T = 1$. Since an exact solution of the second moment is not known, a reference solution is computed using standard Monte Carlo with Euler-Maruyama and a stepsize of $h = 2^{-12}$. In total 2^{24} Monte Carlo simulations are carried out. Figure 3.6 illustrates an approximation of the root mean square error of the second moment of $X_1(t)$ and $X_2(t)$, respectively, at $t = 1$ against the finest stepsize h_L, approximating the expectation by taking a sample average (over 10 samples). We take into account a nonstiff problem with $(\lambda_1 = -1, \mu_1 = \sqrt{-2\lambda_1 - 0.01})$ (see Fig. 3.6(a)) and a stiff problem with $(\lambda_1 = -100, \mu_1 = |\lambda_1|^{1/2})$ (see Fig. 3.6(b)). The other parameters have been

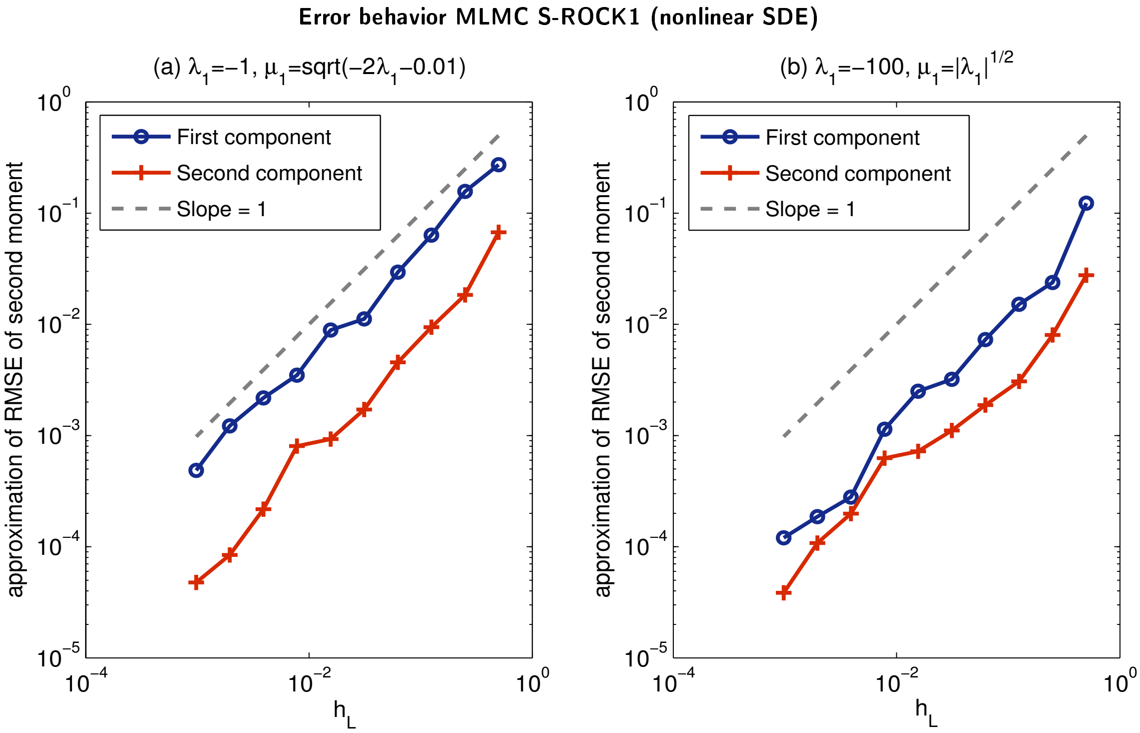

Figure 3.6: Two-dimensional nonlinear noncommutative stiff SDE: Error behavior of the MLMC method using S-ROCK1 applied to (3.26) in a nonstiff (a) and a stiff (b) context. The first and second component correspond to the RMSE of the second moment of $X_1(1)$ and $X_2(1)$, respectively.

chosen as above. One observes that in both cases, stiff or nonstiff, the behavior of the RMSE is as expected roughly linear and of slope 1.

3.5.3 Space-discretized Stochastic Parital Differential Equation

The last problem we consider is a stochastic partial differential equation (SPDE) obtained by adding multiplicative noise to the heat equation. The SPDE is specified through

$$\begin{cases} \frac{\partial u(t,x)}{\partial t} = \frac{\partial^2 u(t,x)}{\partial x^2} + \sigma u(t,x)\dot{W}(t,x), \ (t,x) \in [0,T]\times[0,1], \\ u(0,x) = 1, \ x \in [0,1], \\ u(t,0) = 5, \ t \in [0,T], \ \frac{\partial u(t,1)}{\partial x} = 0, \ t \in [0,T], \end{cases} \tag{3.27}$$

where $\dot{W}$ is a space-time white noise and σ a noise parameter (see e.g. [72]). Discretizing in space by using the method of lines yields

$$\mathrm{d}u_i = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2}\mathrm{d}t + \sigma\frac{u_i}{\sqrt{\Delta x}}\mathrm{d}W_i, \ i = 1,2,\ldots,\frac{1}{\Delta x} = M,$$

where $u_i \approx u(t, x_i)$ with $x_i = i\Delta x$. By the boundary conditions we have $u_0 = 5$ and $u_{M+1} = u_M$. Note that $W_1, W_2, \ldots, W_M$ are M independent standard Brownian motions and Itô noise has been considered. In the following we use $T = 1$ and $\sigma = 10^{-2}$. Figure 3.7 shows one trajectory of the heat equation with noise (3.27) using a space stepsize of $\Delta x = 1/40$ and a time stepsize of $h = 1/40$.

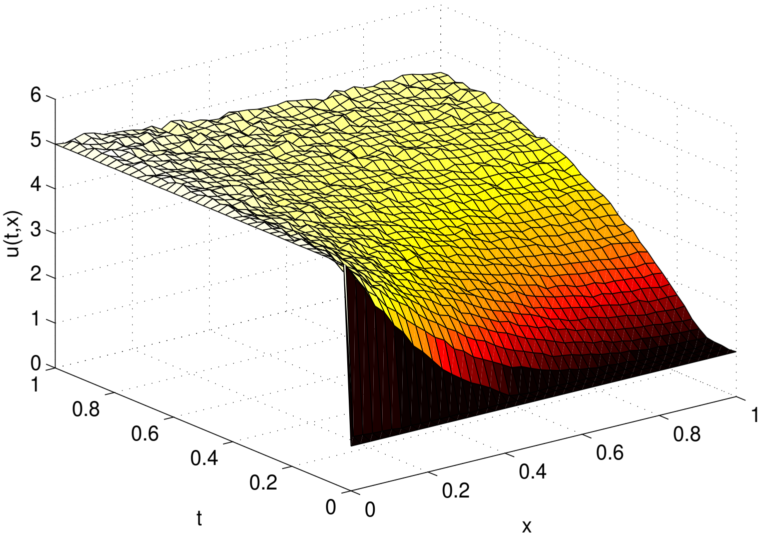

Figure 3.7: Stochastic partial differential equation: Numerical approximation of the heat equation with multiplicative noise (3.27) using $\Delta x = 1/40$ and $h = 1/40$.

Remark 3.5.1. *In the following we only vary the time stepsize with the level in the multilevel Monte Carlo construction and consider a fixed spatial discretization. Note that for some applications (especially for multi-dimensional problems) the spatial meshing is not trivial and the flexibility to adapt the spatial mesh is limited (see e.g. [14, 37]). We however mention MLMC approaches for SPDEs, where both the time and the space discretizations are adapted [20, 47]. These are certainly interesting approaches when applicable but will not be pursued here.*

Figure 3.8 illustrates the number of function evaluations of the stabilized multilevel Monte Carlo method and the standard Monte Carlo method using S-ROCK1 as numerical integrator. The space stepsize is set to $\Delta x = 1/40$ and the finest time stepsize varies between 2^{-7} and 2^{-20}. It can be observed that for small time stepsizes h_L the stabilized MLMC method reduces the computational cost significantly compared to the standard MC method.

In Figure 3.9 the mean of $u(t, x)$ is approximated at $t = 1$ using stabilized multilevel Monte Carlo. The finest time stepsize is chosen as $h_L = 2^{-10}$. For the space discretization $\Delta x = 1/40$ (see Fig. 3.9(a)) and $\Delta x = 1/80$ (see Fig. 3.9(b)) are used, respectively. In addition, on each plot a single trajectory of $u(t, x)$ at $t = 1$ using S-ROCK1 and $h = 2^{-10}$ is added. Note that an approximation of the mean of $u(t, x)$ using the standard Monte Carlo method with Euler-Maruyama would require in the case $\Delta x = 1/40$ a time stepsize smaller than 3.1×10^{-4} and a computational cost of approximately 2.6×10^{12} function evaluations. In the case $\Delta x = 1/80$, the

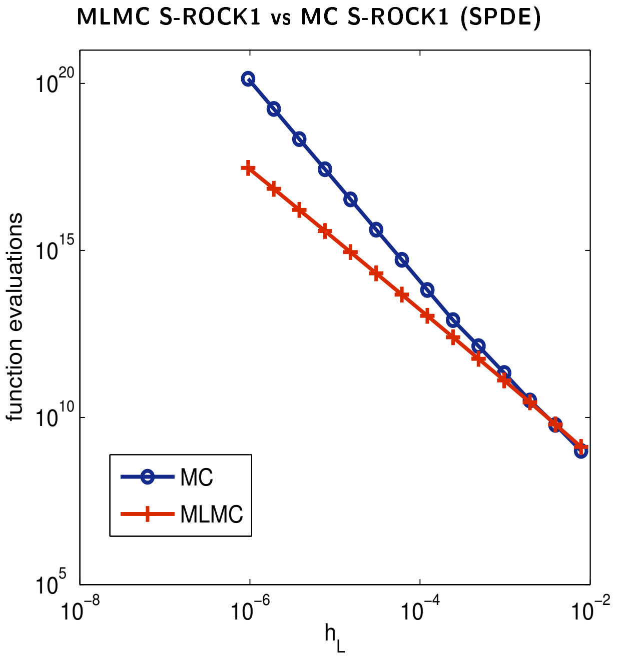

Figure 3.8: Stochastic partial differential equation: Number of function evaluations for the stabilized multilevel Monte Carlo method and the stabilized single-level Monte Carlo method, respectively, using $\Delta x = 1/40$ and $h_L \in \{2^{-7}, 2^{-8}, \ldots, 2^{-20}\}$.

time stepsize would have to be smaller than 7.8×10^{-5} and the corresponding computational cost would be about 3.4×10^{14} function evaluations.

Figure 3.10 illustrates an approximation of the second moment of $u(t,x)$ at $t = 1$ using the stabilized multilevel Monte Carlo method with finest time stepsize $h_L = 2^{-10}$ and space discretization $\Delta x = 1/40$ and $\Delta x = 1/80$, respectively. The number of simulations per level varies from $N_0 = 10 \cdot 2^{20}$ for the coarsest grid to $N_L = 10 \cdot 2^{10}$ for the finest grid. Furthermore, a dotted line represents a single trajectory of the approximation of $u(1,x_i)^2$ using S-ROCK1 with $h = 2^{-10}$.

In Figure 3.11 approximations of $\mathbb{E}\left[u(t,x_i)u(t,x_j)\right]$ at $t = 1$ with $i, j \in \{0, 1, \ldots, 1/\Delta x\}$ are shown. As approximation procedure the stabilized MLMC method with $L = 10$ and space discretization $\Delta x = 1/40$ and $\Delta x = 1/80$, respectively, is used. Single trajectories of S-ROCK1 approximations of $u(t,x_i)u(t,x_j)$ at $t = 1$ are illustrated in Figure 3.12.

3.5.4 Comparison Improved Stabilized MLMC vs Stabilized MLMC vs Standard MLMC

In this section we compare the improved stabilized multilevel Monte Carlo method with the stabilized MLMC method and the standard MLMC method. As numerical experiment we use

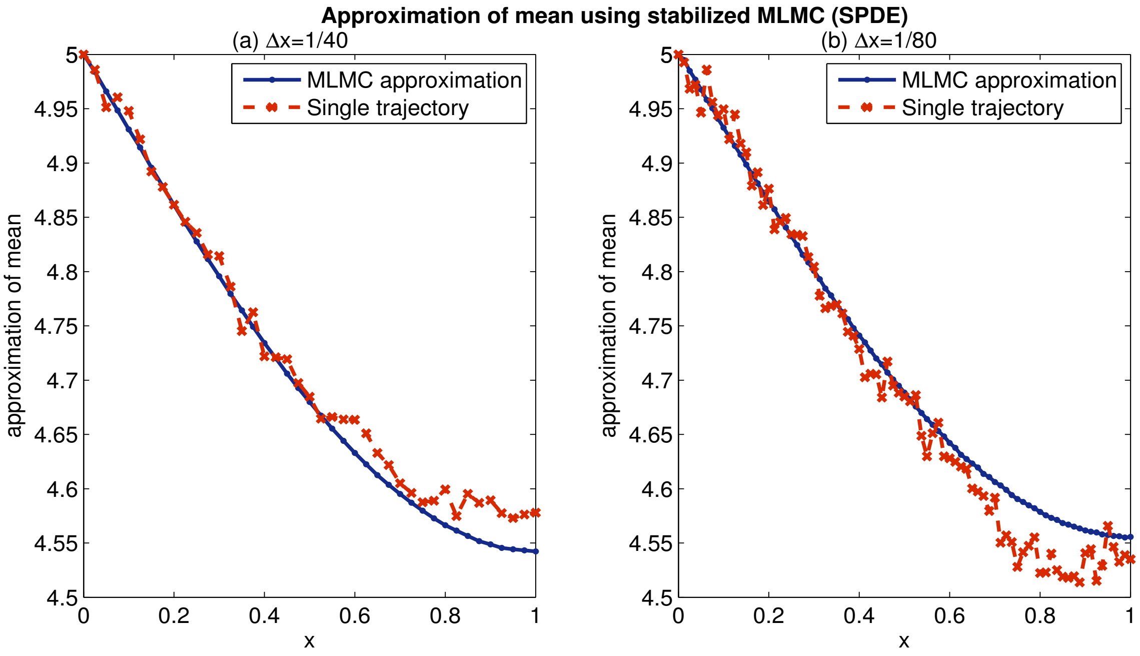

Figure 3.9: Stochastic partial differential equation: Approximation of $\mathbb{E}[u(1,x_i)]$, the mean of $u(t,x)$ at $t=1$, using stabilized MLMC with $L=10$, $\Delta x = 1/40$ (see (a)) and $\Delta x = 1/80$ (see (b)), respectively. The dotted lines represent a single trajectory at $t=1$ using S-ROCK1 with $h=2^{-10}$.

the two-dimensional nonlinear noncommutative SDE (3.26), which we can rewrite as

$$\mathrm{d}\begin{pmatrix} X_1(t) \\ X_2(t) \end{pmatrix} = \begin{pmatrix} \alpha a_2(t) - \lambda_1 b_1(t) \\ -\lambda_2 b_2(t) \end{pmatrix} \mathrm{d}t + \begin{pmatrix} -\mu_1 b_1(t) & \mu_2 a_1(t) \\ -\mu_2 b_2(t) & 0 \end{pmatrix} \begin{pmatrix} \mathrm{d}W_1(t) \\ \mathrm{d}W_2(t) \end{pmatrix}$$

for $0 \le t \le T$, where $a_i(t) = X_i(t) - 1$ and $b_i(t) = X_i(t)(1 - X_i(t))$ for $i \in \{1,2\}$. The initial condition is given by $(X_1(0), X_2(0)) = (0.95, 0.95)$ and $(W_1(t))_{t\in[0,1]}$ and $(W_2(t))_{t\in[0,1]}$ are two independent Wiener processes. We consider two different scenarios. First a stiff problem with drift term $\lambda_1 \in \{-1, -100, -10000\}$ and noise term $\mu_1 = \sqrt{|\lambda_1|}$. And then a nonstiff problem with no small noise by fixing $\lambda_1 = -1$ and varying $\mu_1 = \sqrt{-2\lambda_1 - \delta}$ with $\delta \in \{10^{-1}, 10^{-2}, 10^{-4}\}$. In addition we pick $\alpha = 2$, $\lambda_2 = -1$, $\mu_2 = 0.5$, $k = 2$. As root mean square accuracy we choose k^{-2L} with $L \in \{1, 2, \ldots, 5\}$. Stability is guaranteed by assessing the second moment at the time end point. In Figure 3.13 we compare the number of function evaluations (by counting the drift and diffusion evaluations) of the improved stabilized (using S-ROCK1 and S-ROCK2), the stabilized (using S-ROCK1) and the standard (using EM) MLMC method. As expected the improved stabilized approach yields a cost reduction over the other two methods (see also Table 3.1).

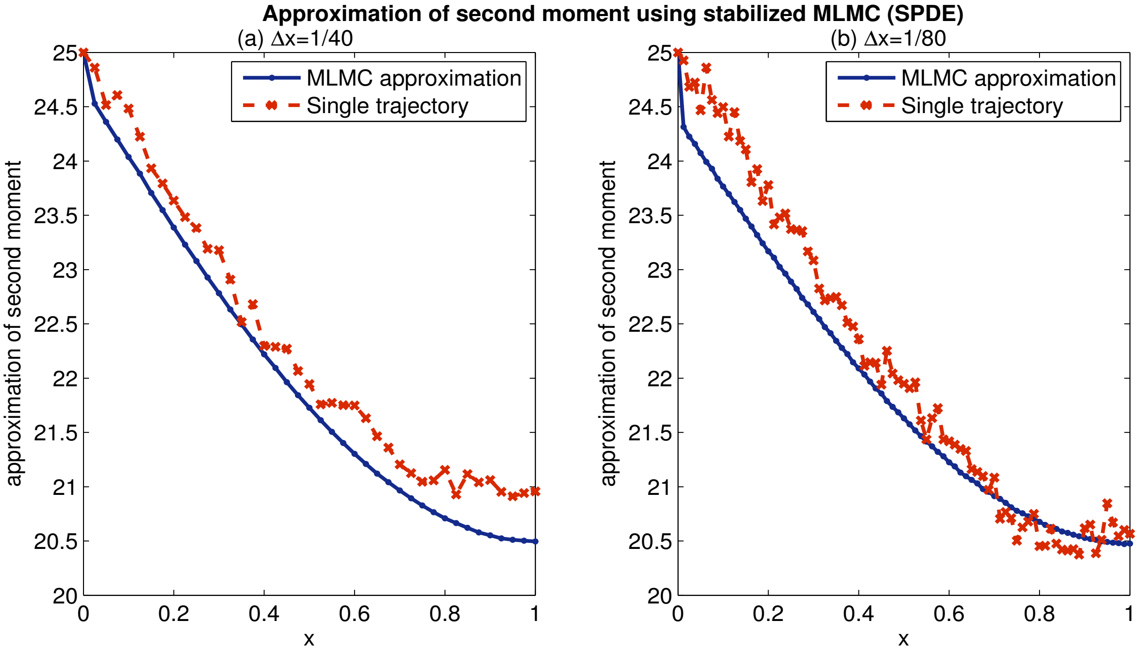

Figure 3.10: Stochastic partial differential equation: Approximation of $\mathbb{E}\left[u(1,x_i)^2\right]$, the second moment of $u(t,x)$ at $t=1$, using stabilized MLMC with $L=10$, $\Delta x=1/40$ (see (a)) and $\Delta x=1/80$ (see (b)), respectively. The dotted lines represent a single trajectory at $t=1$ using S-ROCK1 with $h=2^{-10}$.

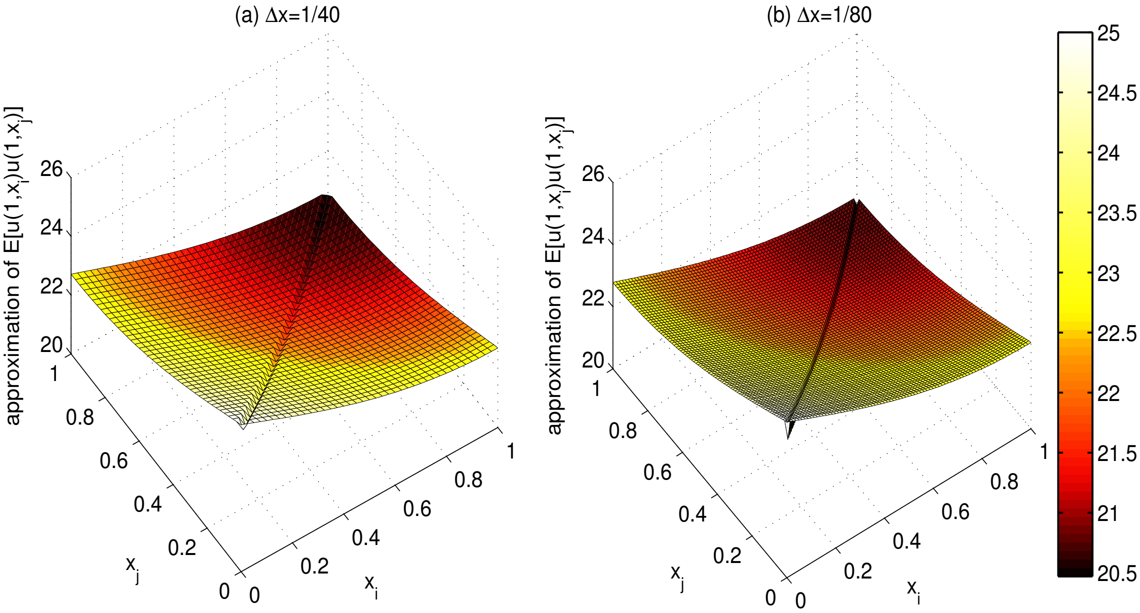

Figure 3.11: Stochastic partial differential equation: Approximation of $\mathbb{E}\left[u(t,x_i)u(t,x_j)\right]$ at $t=1$ using stabilized MLMC with $L=10$, $\Delta x=1/40$ (see (a)) and $\Delta x=1/80$ (see (b)).

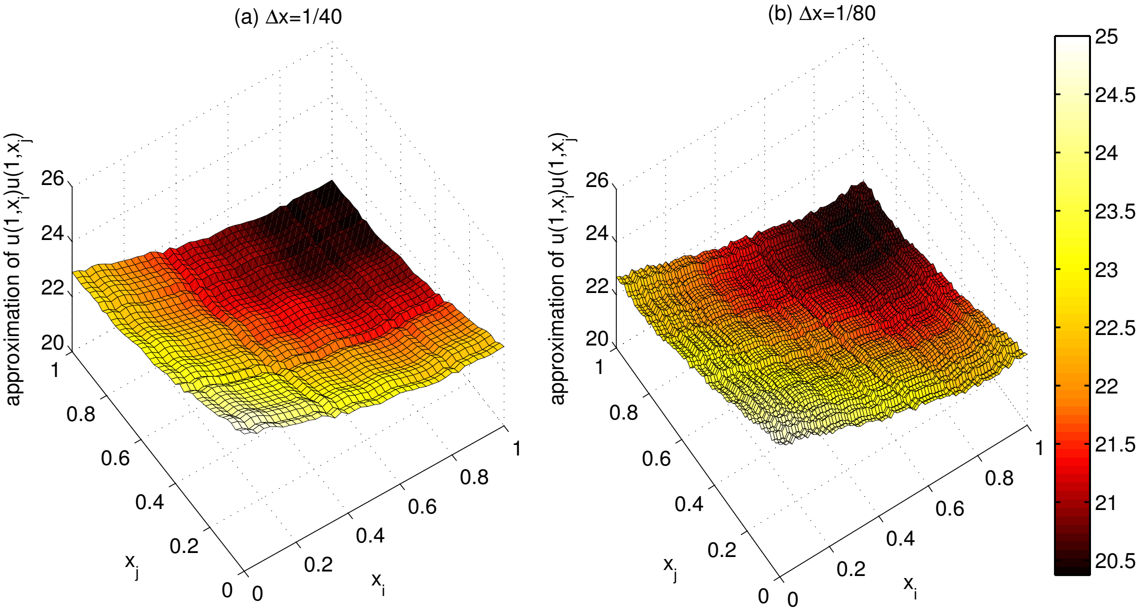

Figure 3.12: Stochastic partial differential equation: Single trajectory of $u(t,x_i)u(t,x_j)$ at $t=1$ using S-ROCK1 with $h=2^{-10}$, $\Delta x=1/40$ (see (a)) and $\Delta x=1/80$ (see (b)).

Table 3.1: Number of function evaluations of the improved stabilized MLMC (using S-ROCK1 and S-ROCK2), the stabilized MLMC (using S-ROCK1) and standard MLMC (using EM) for different values of the root mean square error. As parameters we take $\lambda_1=-1, \mu_1=\sqrt{-2\lambda_1-0.01}$ (b) and $\lambda_1=-100, \mu_1=\sqrt{|\lambda_1|}$ (e).

	precision	2^{-2}	2^{-4}	2^{-6}	2^{-8}	2^{-10}
(b)	imp.stab.MLMC	64	4352	184320	5.70×10^6	14.99×10^7
	stab.MLMC	672	35840	1204224	19.92×10^6	37.12×10^7
	MLMC	10.49×10^6	10.49×10^6	10.49×10^6	42.27×10^6	70.25×10^7
(e)	imp.stab.MLMC	256	16896	614400	16.91×10^6	39.53×10^7
	stab.MLMC	2272	95232	2629632	42.73×10^6	73.61×10^7
	MLMC	10.49×10^6	10.49×10^6	10.49×10^6	42.27×10^6	70.25×10^7

3.6 Conclusion

We have presented a new stabilized multilevel Monte Carlo method for mean square stable SDEs with multiple scales. We have shown that the standard MLMC method fails to achieve the optimal computational complexity $\mathcal{O}(\varepsilon^{-2}(\log(\varepsilon))^2)$ to compute the expectation of functionals with an accuracy of $\mathcal{O}(\varepsilon)$ as some or all the sequence of stepsizes needed in the MLMC method are not accessible due to stepsize restriction. In the worst case, only a standard Monte Carlo method can be used and the computational complexity can deteriorate to $\mathcal{O}(\varepsilon_{MC}^{-3})$, where

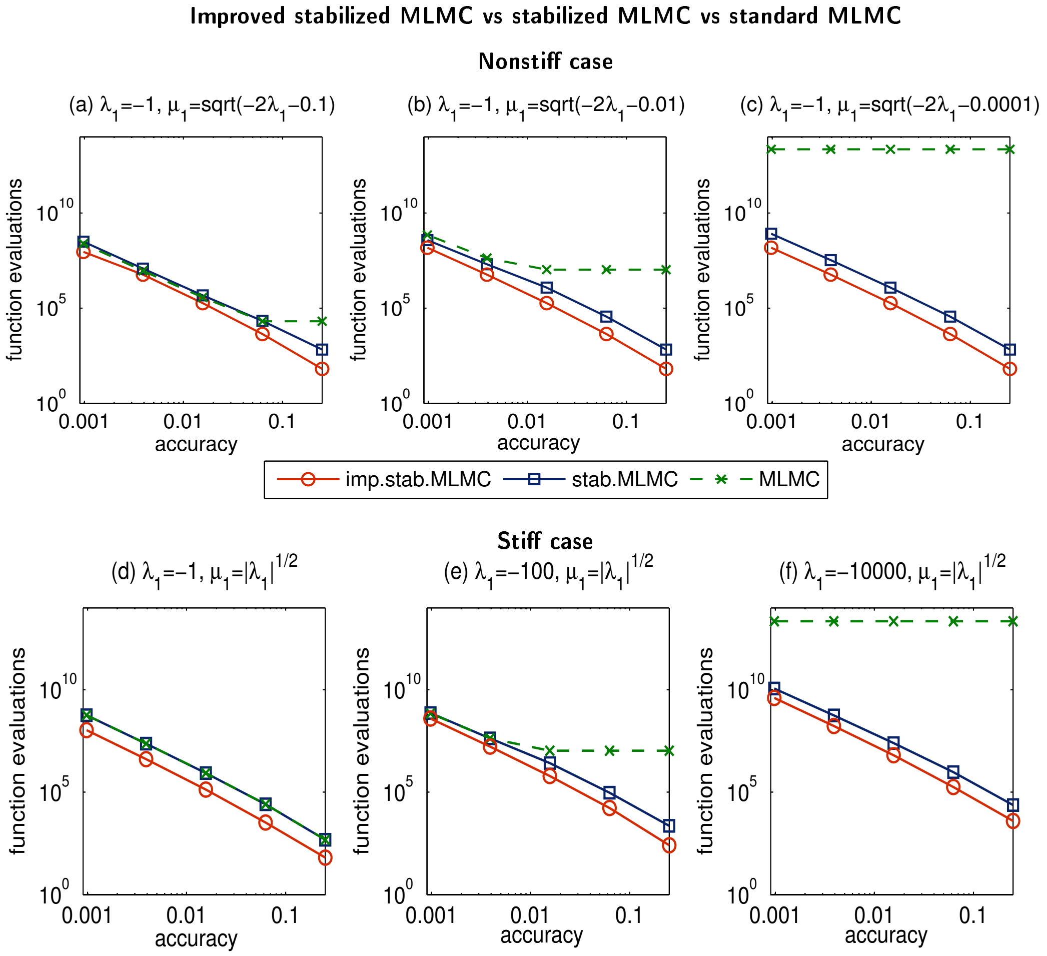

Figure 3.13: Function evaluations against root mean square accuracy comparing the improved stabilized MLMC method using S-ROCK1 and S-ROCK2 with the stabilized (S-ROCK1) and the standard (EM) MLMC method.

ε_{MC} is smaller than ε, the desired accuracy. We have then shown that using the S-ROCK1 methods, a family of stabilized methods based on the Euler-Maruyama scheme, it is possible to define a stabilized MLMC method that is applicable for stiff mean square stable problems. By an optimal choice of the stabilization procedure, varying from the coarse to the fine MLMC levels, we showed that it is possible to recover the optimal complexity of the MLMC for nonstiff problems up to a factor involving the square root of a quantity called the stiffness parameter. Even though our stability analysis relies on the usual linear scalar SDE used to characterize mean square stability of numerical integrators, we have shown through numerical experiments on multidimensional nonlinear noncommutative stiff SDEs and on a system of SDEs obtained from a space-discretized SPDE that our new stabilized MLMC method is efficient also for more general problems. Furthermore, by using a higher weak order scheme (namely S-ROCK2) we have presented an improved stabilized MLMC method and we have compared it numerically to the stabilized MLMC method and the standard MLMC method.

4 S-ROCK Methods for Jump-Diffusion Processes

In this chapter we introduce an explicit stabilized numerical integrator based on orthogonal Chebyshev polynomials that can be used to approximate the solution of stochastic problems characterized by stochastic differential equations driven by jump-diffusion processes. In particular we present two new numerical schemes, the S-ROCK1-JD method and the PIROCK-JD method. Both discretization methods are an extension of existing methods for diffusions to jump-diffusions. We study rigorously the strong order of convergence of the two methods, which we prove to be equal to $1/2$. In this chapter we also analyze the mean square stability of the newly defined methods and we specify their stability domains. Finally, we carry out several numerical experiments to numerically visualize the theoretical findings.

4.1 Introduction

In many different fields the number of stochastic problems that are modeled by jump-diffusion processes is increasing. In finance (see e.g. [32, 26, 54]), in biology (see e.g. [103, 54]), in chemistry (see e.g. [48]), in medicine (see e.g. [54]), and so on models based on jump-diffusions are very useful to capture sudden, unforeseeable events, that can lead to a huge variation of the underlying stochastic process over a very small time period. Mathematical models based on diffusions are not able to account for such events, and thus, an extension to models driven by jump-diffusion processes is essential (see for instance [80, 18, 98]). As for diffusion models, often there does not exist an analytical solution of the jump-diffusion models. Hence, one requires numerical integrators to provide an approximate solution. There are various scientific papers and books that propose and analyze numerical integrators that can solve jump-diffusion processes (see for instance [26, 59, 73, 50, 32, 49, 54]).

Discretizing stochastic partial differential equations, which possibly include a jump term, by using the method of lines, it is possible that stepsize restrictions arise due to stability issues. These restrictions can force explicit numerical integrators into using potentially a very small time stepsize, which can make the numerical approximation very expensive in terms of function evaluations. In this chapter we consider stiff mean square stable systems for which standard numerical integrator, such as the Euler-Maruyama method (see e.g. [59, 92]), face severe time stepsize restrictions.

We provide in this chapter stabilized explicit numerical integrators that approximate stiff stochastic differential equations driven by jump-diffusion processes efficiently. The methods we propose remain fully explicit and they are simple to implement. The idea to extend the S-ROCK methods to stochastic differential equations driven by jump-diffusion processes first came up in [21]. Here, we give a rigorous study of the strong convergence of the suggested numerical integrators. In addition, we study the mean square stability of the numerical methods and we carry out various numerical experiments. To the best of the author's knowledge such stabilized explicit numerical integrators for stiff stochastic differential equations driven by jump-diffusions do not exist in the literature. We note though that drift-implicit numerical integrators have been suggested for jump-diffusion models (see for instance [59]). These methods are a good alternative when they can be applied. However, for instance for stochastic problems modeled by stochastic partial differential equations with stiff nonlinear terms, it can be really difficult or even impossible to solve such large systems with an implicit integrator.

This chapter is organized as follows. First, we specify in Section 4.2 which kind of stochastic differential equations we deal with and we define the numerical integrators that we use in this chapter. In Section 4.3 we prove the strong order of convergence of the numerical integrators. This is followed by the study of the mean square stability of the two numerical methods in Section 4.4. Finally, we provide in Section 4.5 various numerical experiments to illustrate the theoretical findings of the previous sections.

The following is based on a scientific paper that is in preparation [6].

4.2 Preliminaries

In this chapter we consider stochastic differential equation driven by a jump-diffusion process, which are characterized by

$$\left\{\begin{array}{rcl} \mathrm{d}X(t) &=& f(X(t-))\mathrm{d}t + \sum_{r=1}^{m_1} g_1^r(X(t-))\mathrm{d}W_t^r + \sum_{r=1}^{m_2} g_2^r(X(t-))\mathrm{d}N_t^r, \quad 0 \le t \le T, \\ X(0) &=& X_0\,, \end{array}\right. \tag{4.1}$$

where $X(t)$ is a $\mathbb{R}^d$-valued random variable, $X(t-) = \lim_{s \nearrow t} X(s)$, $f : \mathbb{R}^d \to \mathbb{R}^d$ is the drift term, $g_1^r : \mathbb{R}^d \to \mathbb{R}^d$ with $r = 1, 2, \dots, m$ are the diffusion terms, $g_2^r : \mathbb{R}^d \to \mathbb{R}^d$ with $r = 1, 2, \dots, m$ are the jump terms, $(W^r(t))_{t\in[0,T]}$ with $r = 1, 2, \dots, m$ are independent one-dimensional Wiener processes and $(N^r(t))_{t\in[0,T]}$ with $r = 1, 2, \dots, m$ are independent Poisson processes with jump intensity ρ. Without loss of generality we assume in this chapter autonomous stochastic differential equations. Using a simple transformation any non-autonomous SDE can be changed into an autonomous one (see Remark 2.1.3).

To get a numerical approximation of the solution of (4.1), we take into account a discrete map

$$X_{n+1} = \Psi\left(X_n, \Delta t, \xi_n\right), \tag{4.2}$$

where $\Psi\left(\cdot, \Delta t, \xi_n\right) : \mathbb{R}^d \to \mathbb{R}^d$, $X_n \in \mathbb{R}^d$ with $n \geq 0$, Δt the time stepsize of the numerical scheme and ξ_n a random vector. We use two concepts in this chapter, namely the accuracy measured by the mean square stability and the convergence of the numerical integrators. They are introduced in detail in Section 2.2 and we briefly recall them here. When dealing with stochastic differential equations, there are two different types of convergence. In this chapter we study the strong convergence, and thus, we recall its definition here. Starting from the initial condition X_0 of (4.1), a numerical scheme (4.2) is said to be of strong order of convergence r_s if

$$\exists C \in \mathbb{R}_+ \quad \text{such that} \quad \max_{0 \leq n \leq T/h} \left(\mathbb{E}\left[|X_n - X\left(\tau_n\right)|^2\right]\right)^{1/2} \leq C \Delta t^{r_s}$$

with $\tau_n = n\Delta t \in [0, T]$ and Δt sufficiently small. Note that C is a constant that is independent of the time stepsize Δt.

The next concept of SDEs that we study in this chapter is the mean square stability of a numerical method. The mean square stability is interesting if one is fixing a time stepsize Δt and then looks at the long-time behavior of the exact and the approximate solution. In fact, a stochastic process $(X(t))_{t \geq 0}$ is said to be mean square stable if

$$\lim_{t \to \infty} \mathbb{E}\left[X(t)^2\right] = 0.$$

By using a specific test problem, see Section 4.4, we can find the true stability domain of the test problem. To avoid a time stepsize restriction, which especially arises for stiff problems, it is useful to have a numerical integrator that covers as much as possible of this true stability domain. A numerical method is said to be mean square stable if

$$\lim_{n \to \infty} \mathbb{E}\left[X_n^2\right] = 0.$$

Details of the test problem, the stability domains of the test problem and the numerical schemes are given in Section 4.4.

4.2.1 Numerical Schemes

In this section we briefly define the numerical integrators that we use in this chapter. Since we are dealing with jump-diffusion processes, one has to distinguish between a regular time grid and a jump-adapted time grid. The difference is that the regular time grid is based on a uniform time stepsize, whereas in the jump-adapted time grid the jump times are added to the regular time steps. To define the numerical integrators, we consider a time grid

$$\tau_{\Delta t}^N := \{t_0, t_1, \ldots, t_N\}, \tag{4.3}$$

which is, depending on the situation, the time grid of the regular or the jump-adapted numerical scheme with an underlying uniform stepsize of Δt.

S-ROCK1-JD

The first scheme that we introduce here is the S-ROCK1-JD method, which we define as follows.

Definition 4.2.1 (S-ROCK1-JD method)**.** *Consider the time grid* (4.3)*. The S-ROCK1-JD method with s stages (with $s \geq 2$) for* (4.1) *is defined by*

$$\begin{aligned}
K_0 &= Y_n, \\
K_1 &= Y_n + \Delta t_{n+1} \tfrac{\omega_1}{\omega_0} f\left(K_0\right), \\
K_j &= 2\Delta t_{n+1}\omega_1 \tfrac{T_{j-1}(\omega_0)}{T_j(\omega_0)} f\left(K_{j-1}\right) + 2\omega_0 \tfrac{T_{j-1}(\omega_0)}{T_j(\omega_0)} K_{j-1} - \tfrac{T_{j-2}(\omega_0)}{T_j(\omega_0)} K_{j-2}, \;\; j = 2,3,\dots,s, \\
Y_{n+1} &= K_s + \sum_{r=1}^{m_1} g_1^r\left(K_s\right)\Delta W_{n+1}^r + \sum_{r=1}^{m_2} g_2^r\left(K_s\right)\Delta N_{n+1}^r,
\end{aligned} \tag{4.4}$$

where $\Delta t_{n+1} = t_{n+1} - t_n$, $\omega_0 = 1 + \frac{\eta}{s^2}$, $\omega_1 = \frac{T_s(\omega_0)}{T_s'(\omega_0)}$, $\Delta W_{n+1}^r = W^r\left(t_{n+1}\right) - W^r\left(t_n\right)$ *and* $\Delta N_{n+1}^r = N^r\left(t_{n+1}\right) - N^r\left(t_n\right)$.

Note that $\left(T_j(x)\right)_{j\geq 0}$ represent the orthogonal Chebyshev polynomials and η is a damping parameter, which allows to adjust the stability domain of the numerical integrator (see [12]).

PIROCK-JD

The second numerical scheme that we define here is the so-called PIROCK-JD method. It is based on the PIROCK method introduced in [14] to which we add a jump component. The PIROCK-JD method (with no reaction and no advection) that we use in this chapter is defined as follows.

Definition 4.2.2 (PIROCK-JD method)**.** *Consider the time grid* (4.3)*. The PIROCK-JD method with s stages for* (4.1) *is defined by*

$$\begin{aligned}
K_0 &= Y_n, \\
K_1 &= Y_n + \alpha\mu_1 \Delta t_{n+1} f(K_0), \\
K_j &= \alpha\mu_j \Delta t_{n+1} f\left(K_{j-1}\right) - \nu_j K_{j-1} - \kappa_j K_{j-2}, \; j = 2,3,\dots,s \\
K^*_{s-1} &= K_{s-2} + \sigma_\alpha \Delta t_{n+1} f(K_{s-2}), \\
K^*_s &= K^*_{s-1} + \sigma_\alpha \Delta t_{n+1} f\left(K^*_{s-1}\right), \\
K^*_{s+1} &= K_s + \beta \Delta t_{n+1} f(K_s) \\
Y_{n+1} &= K^*_s - \sigma_\alpha \left(1 - \tfrac{\tau_\alpha}{\sigma^2_\alpha}\right) \Delta t_{n+1} \left(f\left(K^*_{s-1}\right) - f(K_{s-2})\right) \\
&\quad + \sum_{r=1}^{m_1} g^r_1 \left(K^*_{s+1}\right) \Delta W^r_{n+1} + \sum_{r=1}^{m_2} g^r_2 \left(K^*_{s+1}\right) \Delta N^r_{n+1},
\end{aligned} \tag{4.5}$$

where $\Delta t_{n+1} = t_{n+1} - t_n$, $\Delta W^r_{n+1} = W^r(t_{n+1}) - W^r(t_n)$ *and* $\Delta N^r_{n+1} = N^r(t_{n+1}) - N^r(t_n)$. *The parameters* $\alpha = 1, \beta = 1 - 2\alpha P'_s(0)$ *and* $\mu_j, \nu_j, \kappa_j, \tau_\alpha, \sigma_\alpha$ *are as defined in [14].*

Euler-Maruyama-JD Method

Here, we give briefly the definition of the Euler-Maruyama method for stochastic differential equations driven by jump-diffusions (called the Euler-Maruyama-JD method in the following) that has been discussed e.g. in [26, 59]. As the two previous schemes, the Euler-Maruyama method is defined as well on a regular as on a jump-adapted time grid. In Section 4.5.2 we compare numerically the Euler-Maruyama method to the S-ROCK1-JD method.

Definition 4.2.3 (Euler-Maruyama-JD method)**.** *Consider the time grid* (4.3)*. The Euler-Maruyama-JD method for* (4.1) *is defined by*

$$Y_{n+1} = Y_n + f(Y_n)\Delta t_{n+1} + \sum_{r=1}^{m_1} g^r_1(Y_n)\Delta W^r_{n+1} + \sum_{r=1}^{m_2} g^r_2(Y_n)\Delta N^r_{n+1},$$

where $\Delta t_{n+1} = t_{n+1} - t_n$, $\Delta W^r_{n+1} = W^r(t_{n+1}) - W^r(t_n)$ *and* $\Delta N^r_{n+1} = N^r(t_{n+1}) - N^r(t_n)$.

4.3 Strong Convergence

In this section we derive the strong order of convergence of the S-ROCK1-JD method and of the PIROCK-JD method defined in Definition 4.2.1 and Definition 4.2.2, respectively. We show that both numerical integrators are of strong order of convergence 1/2. The following is inspired by [59], which proves in particular the strong order of convergence of the Euler-Maruyama method for jump-diffusion processes. We start by proving the strong order of convergence of the S-ROCK1-JD method.

4.3.1 Strong Convergence of S-ROCK1-JD

In the following we consider a stochastic differential equation driven by a jump-diffusion process of the form (4.1). As numerical integrator we take into account the S-ROCK1 method for jump-diffusions with s stages defined in (4.4), where we suppose that we use a time stepsize of Δt. Note that in the jump-adapted case this represents the largest time stepsize of the time grid and in the regular case it is the uniform time stepsize.

We state now a few assumptions, that we use in the following. We indicate them with (A$\cdot$), where $\cdot$ represents a specific number. First of all, we assume that the functions f, g_1^r, g_2^r are Lipschitz continuous (A1), i.e.

$$\begin{array}{rcl} \left|f(x)-f(y)\right|^2 & \leq & K|x-y|^2, \\ \left|g_1^r(x)-g_1^r(y)\right|^2 & \leq & K_1|x-y|^2, \\ \left|g_2^r(x)-g_2^r(y)\right|^2 & \leq & K_2|x-y|^2, \end{array}$$

for all x, y and for all r with K, K_1 and K_2 some positive constants. It follows that these functions also allow a linear growth bound, namely

$$\begin{array}{rcl} \left|f(x)\right|^2 & \leq & L\left(1+|x|^2\right), \\ \left|g_1^r(x)\right|^2 & \leq & L_1\left(1+|x|^2\right), \\ \left|g_2^r(x)\right|^2 & \leq & L_2\left(1+|x|^2\right), \end{array}$$

where L, L_1 and L_2 are some positive constants. Furthermore, we assume that the second moment of the initial condition is finite, i.e. $\mathbb{E}\left[|X_0|^2\right] < +\infty$ and that X_0 is independent of the Brownian processes $\left(W_t^r\right)_{t\in[0,T]}$ and the Poisson processes $\left(N_t^r\right)_{t\in[0,T]}$ (A2). This guarantees the existence and uniqueness of a solution to (4.1) (see e.g. [26]).

We show now that the S-ROCK1-JD method (4.4) has a strong order of convergence 1/2. This requires a few lemmas and results in a theorem, which proves the convergence. But first observe that we can rewrite (4.4) as

$$Y_{n+1} = Y_n + \alpha\left(K_s, Y_n\right)\Delta t + \sum_{r=1}^{m_1} g_1^r\left(K_s\right)\Delta W_{n+1}^r + \sum_{r=1}^{m_2} g_2^r\left(K_s\right)\Delta N_{n+1}^r,$$

where

$$\alpha\left(K_s, Y_n\right) = \frac{K_s - Y_n}{\Delta t}. \tag{4.6}$$

It is straightforward to show by recurrence that $\alpha\left(K_s, Y_n\right) = \sum_{j=0}^{s-1} \lambda_{sj} f\left(K_j\right)$, and thus, the function α is also Lipschitz continuous and has a linear growth bound. For a solely theoretical purpose we can extend the discrete numerical solution to continuous time. In fact, we define

$$Z_1(t) = \sum_k Y_k 1_{[k\Delta t,(k+1)\Delta t[}(t),$$

and thus, we can write the numerical solution in continuous time as

$$\begin{aligned} Y(t) &= Y_0 + \int_0^t \alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right) \mathrm{d}s \\ &\quad + \sum_{r=1}^{m_1} \int_0^t g_1^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) \mathrm{d}W_s^r + \sum_{r=1}^{m_2} \int_0^t g_2^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) \mathrm{d}N_s^r, \end{aligned} \tag{4.7}$$

where we have used that $K_s = Y_n + \Delta t \sum_{j=0}^{s-1} \lambda_{sj} f\left(K_j\right) = Y_n + \mathcal{O}(\Delta t)$. Proposition 4.3.1 gives another result, which we use to prove the strong convergence.

Proposition 4.3.1. *Suppose that the assumption (A1) holds. Consider the function α defined in* (4.6). *Then α satisfies the following upper bound:*

$$|\alpha(K_s, Y_n)|^2 \leq Ls\left(\sum_{j=0}^{s-1} \left|\lambda_{sj}\right|^2\right)\left(1 + 2|Y_n|^2 + \mathcal{O}\left(\Delta t^2\right)\right).$$

Proof. Taking into account the definition of α we get

$$|\alpha(K_s, Y_n)|^2 = \left|\frac{K_s - Y_n}{\Delta t}\right|^2 = \left|\sum_{j=0}^{s-1} \lambda_{sj} f\left(K_j\right)\right|^2.$$

Using the Cauchy-Schwarz inequality, the linear growth bound for f and the fact that $K_j = Y_n + \mathcal{O}(\Delta t)$ we obtain the desired upper bound

$$\begin{aligned} |\alpha(K_s, Y_n)|^2 &= \left|\sum_{j=0}^{s-1} \lambda_{sj} f\left(K_j\right)\right|^2 \\ &\leq \sum_{j=0}^{s-1} \left|\lambda_{sj}\right|^2 \sum_{j=0}^{s-1} \left|f\left(K_j\right)\right|^2 \\ &\leq \sum_{j=0}^{s-1} \left|\lambda_{sj}\right|^2 \sum_{j=0}^{s-1} L\left(1 + \left|K_j\right|^2\right) \\ &\leq \sum_{j=0}^{s-1} \left|\lambda_{sj}\right|^2 \sum_{j=0}^{s-1} L\left(1 + 2|Y_n|^2 + \mathcal{O}\left(\Delta t^2\right)\right) \\ &\leq Ls\left(\sum_{j=0}^{s-1} \left|\lambda_{sj}\right|^2\right)\left(1 + 2|Y_n|^2 + \mathcal{O}\left(\Delta t^2\right)\right). \end{aligned}$$

□

We state and prove now three lemmas before we tackle the theorem.

Lemma 4.3.2. *Suppose that the assumptions (A1) and (A2) hold. Then there exists* $\Delta t^* > 0$ *such that for all* $0 < \Delta t \le \Delta t^*$

$$\mathbb{E}\left[|Y_k|^2\right] \le C_1\left(1+\mathbb{E}\left[|X(0)|^2\right]\right), \quad \forall\ k\Delta t \le T.$$

Proof. By extension to continuous time we have

$$\begin{aligned} Y_{k+1} &= Y_0 + \int_0^{(k+1)\Delta t} \alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right) \mathrm{d}s \\ &\quad + \sum_{r=1}^{m_1} \int_0^{(k+1)\Delta t} g_1^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) \mathrm{d}W_s^r \\ &\quad + \sum_{r=1}^{m_2} \int_0^{(k+1)\Delta t} g_2^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) \mathrm{d}N_s^r. \end{aligned}$$

Hence, for $(k+1)\Delta t \le T$,

$$\begin{aligned} \mathbb{E}\left[|Y_{k+1}|^2\right] &\le 4\mathbb{E}\left[|Y_0|^2\right] + 4\mathbb{E}\left[\left|\int_0^{(k+1)\Delta t} \alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right) \mathrm{d}s\right|^2\right] \\ &\quad + 4m_1 \sum_{r=1}^{m_1} \mathbb{E}\left[\left|\int_0^{(k+1)\Delta t} g_1^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) \mathrm{d}W_s^r\right|^2\right] \\ &\quad + 4m_2 \sum_{r=1}^{m_2} \mathbb{E}\left[\left|\int_0^{(k+1)\Delta t} g_2^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) \mathrm{d}N_s^r\right|^2\right]. \end{aligned}$$

We derive now for each of the last three expressions an upper bound.

(i) The first upper bound controls the deterministic integral. In fact, using the Cauchy-Schwarz inequality, Proposition 4.3.1 and Fubini's theorem we get

$$\begin{aligned} & \mathbb{E}\left[\left|\int_0^{(k+1)\Delta t} \alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right) \mathrm{d}s\right|^2\right] \\ \le\ & T\mathbb{E}\left[\int_0^{(k+1)\Delta t} \left|\alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right)\right|^2 \mathrm{d}s\right] \\ \le\ & T\mathbb{E}\left[\int_0^{(k+1)\Delta t} Ls\left(\sum_{j=0}^{s-1} \left|\lambda_{sj}\right|^2\right)\left(1 + 2|Z_1(s)|^2 + \mathcal{O}\left(\Delta t^2\right)\right) \mathrm{d}s\right] \\ \le\ & Ls\left(\sum_{j=0}^{s-1} \left|\lambda_{sj}\right|^2\right) T\left(T\left(1 + \mathcal{O}\left(\Delta t^2\right)\right) + 2\int_0^{(k+1)\Delta t} \mathbb{E}\left[|Z_1(s)|^2\right] \mathrm{d}s\right) \\ =\ & Ls\left(\sum_{j=0}^{s-1} \left|\lambda_{sj}\right|^2\right) T^2\left(1 + \mathcal{O}\left(\Delta t^2\right)\right) + 2Ls\left(\sum_{j=0}^{s-1} \left|\lambda_{sj}\right|^2\right) T\Delta t \sum_0^k \mathbb{E}\left[|Y_i|^2\right]. \end{aligned}$$

(ii) The next upper bound that we derive deals with the stochastic integral with respect to a Brownian motion. Taking into account the Itô isometry and the linear growth bound yields

$$
\begin{aligned}
& \mathbb{E}\left[\left|\int_0^{(k+1)\Delta t} g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}W_s^r\right|^2\right] \\
= \;& \int_0^{(k+1)\Delta t}\mathbb{E}\left[\left|g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\right|^2\right]\mathrm{d}s \\
= \;& \Delta t\sum_{j=0}^{k}\mathbb{E}\left[\left|Y_j+\mathcal{O}(\Delta t)\right|^2\right] \\
\leq \;& \Delta t L\sum_{j=0}^{k}\mathbb{E}\left[\left(1+\left|Y_j+\mathcal{O}(\Delta t)\right|^2\right)\right] \\
\leq \;& TL\left(1+\mathcal{O}\left(\Delta t^2\right)\right)+2\Delta t L\sum_{j=0}^{k}\mathbb{E}\left[\left|Y_j\right|^2\right].
\end{aligned}
$$

(iii) Note that $(N^r)_{t\in[0,T]}$ are Poisson processes with intensity ρ. Hence, the processes defined by $\left(\widehat{N}^r\right)_{t\in[0,T]}$ with $\widehat{N}^r(t) := N^r(t)-\rho t$ are the associated compensated Poisson processes, which are martingales, and thus, the martingale isometry holds for these processes (see e.g. [54]). For the stochastic integral with respect to the jump processes we can derive the following upper bound. Using the martingale isometry, the Cauchy-Schwarz inequality, Fubini's theorem and the linear growth bound of g_2^r we get

$$
\begin{aligned}
& \mathbb{E}\left[\left|\int_0^{(k+1)\Delta t} g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}N_s^r\right|^2\right] \\
= \;& \mathbb{E}\left[\left|\int_0^{(k+1)\Delta t} g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}\widehat{N}_s^r+\rho\int_0^{(k+1)\Delta t} g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}s\right|^2\right] \\
\leq \;& 2\mathbb{E}\left[\left|\int_0^{(k+1)\Delta t} g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}\widehat{N}_s^r\right|^2\right] \\
& +2\rho^2\mathbb{E}\left[\left|\int_0^{(k+1)\Delta t} g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}s\right|^2\right] \\
\leq \;& 2\rho\int_0^{(k+1)\Delta t}\mathbb{E}\left[\left|g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\right|^2\right]\mathrm{d}s \\
& +2\rho^2 T\int_0^{(k+1)\Delta t}\mathbb{E}\left[\left|g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\right|^2\right]\mathrm{d}s \\
= \;& \left(2\rho+2\rho^2T\right)\Delta t\sum_{j=0}^{k}\mathbb{E}\left[\left|g_2^r\left(Y_j+\mathcal{O}(\Delta t)\right)\right|^2\right] \\
\leq \;& \left(2\rho+2\rho^2T\right)\Delta t L\sum_{j=0}^{k}\mathbb{E}\left[1+\left|Y_j+\mathcal{O}(\Delta t)\right|^2\right] \\
\leq \;& \left(2\rho+2\rho^2T\right)TL\left(1+\mathcal{O}\left(\Delta t^2\right)\right)+2\Delta t L\sum_{j=0}^{k}\mathbb{E}\left[\left|Y_j\right|^2\right] \\
= \;& 2\rho T\left(1+\rho T\right)L\left(1+\mathcal{O}\left(\Delta t^2\right)\right)+4\rho\left(1+\rho T\right)\Delta t L\sum_{j=0}^{k}\mathbb{E}\left[\left|Y_j\right|^2\right].
\end{aligned}
$$

Taking all upper bounds computed in $(i)-(iii)$, we obtain

$$\begin{aligned}
&\mathbb{E}\left[|Y_{k+1}|^2\right]\\
\leq\ & 4\mathbb{E}\left[|Y_0|^2\right]\\
&+4\left(Ls\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)T^2\left(1+\mathcal{O}\left(\Delta t^2\right)\right)+2Ls\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)T\Delta t\sum_0^k\mathbb{E}\left[|Y_i|^2\right]\right)\\
&+4m_1\sum_{r=1}^{m_1}\left(TL\left(1+\mathcal{O}\left(\Delta t^2\right)\right)+2\Delta tL\sum_{j=0}^{k}\mathbb{E}\left[\left|Y_j\right|^2\right]\right)\\
&+4m_2\sum_{r=1}^{m_2}\left(2\rho T\left(1+\rho T\right)L\left(1+\mathcal{O}\left(\Delta t^2\right)\right)+4\rho\left(1+\rho T\right)\Delta tL\sum_{j=0}^{k}\mathbb{E}\left[\left|Y_j\right|^2\right]\right)\\
=\ & 4\mathbb{E}\left[|Y_0|^2\right]+4TL\left(1+\mathcal{O}\left(\Delta t^2\right)\right)\left(s\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)T+m_1^2+2m_2^2\rho\left(1+\rho T\right)\right)\\
&+8\Delta tL\left(s\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)T+m_1^2+2m_2^2\rho\left(1+\rho T\right)\right)\sum_{j=0}^{k}\mathbb{E}\left[\left|Y_j\right|^2\right].
\end{aligned}$$

Note that $\left(1+\mathcal{O}\left(\Delta t^2\right)\right)$ is bounded for Δt sufficiently small. Applying the discrete Gronwall inequality (see e.g. [74]) yields the result of the lemma. □

Remark 4.3.3. *Observe that the stage number s appears in the final upper bound derived in the proof of Lemma 4.3.2 in combination with the expression $\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2$. Note that it can be shown that the coefficients λ_{sj} behave like $\frac{1}{s^2}$ as s increases, and thus, $s\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)$ remains bounded as s increases.*

Lemma 4.3.4. *Suppose that the assumptions (A1) and (A2) hold. Then there exists $\Delta t^* > 0$ such that for all $0 < \Delta t \leq \Delta t^*$*

$$\mathbb{E}\left[\sup_{t\in[0,T]}|Y(t)|^2\right]\leq C_2\left(1+\mathbb{E}\left[|X(0)|^2\right]\right).$$

Proof. Consider the discrete solution expressed in continuous time

$$\begin{aligned}
Y(t) \quad = \quad & Y_0+\int_0^t\alpha\left(Z_1(s)+\mathcal{O}(\Delta t),Z_1(s)\right)\mathrm{d}s\\
&+\sum_{r=1}^{m_1}\int_0^t g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}W_s^r+\sum_{r=1}^{m_2}\int_0^t g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}N_s^r.
\end{aligned}$$

Taking first the square of the norm, then the supremum over the time interval $[0, T]$ and finally the expectation yields

$$\begin{aligned}
\mathbb{E}\left[\sup_{t\in[0,T]}|Y(t)|^2\right] &\leq 4\mathbb{E}\left[\sup_{t\in[0,T]}|Y_0|^2\right] \\
&+4\mathbb{E}\left[\sup_{t\in[0,T]}\left|\int_0^t \alpha\left(Z_1(s)+\mathcal{O}(\Delta t), Z_1(s)\right)\mathrm{d}s\right|^2\right] \\
&+4m_1\sum_{r=1}^{m_1}\mathbb{E}\left[\sup_{t\in[0,T]}\left|\int_0^t g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}W_s^r\right|^2\right] \\
&+4m_2\sum_{r=1}^{m_2}\mathbb{E}\left[\sup_{t\in[0,T]}\left|\int_0^t g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}N_s^r\right|^2\right].
\end{aligned}$$

As in the previous lemma, we look separately for an upper bound for the last three expressions.

(i) We derive first an upper bound for the deterministic term. Observe that

$$\begin{aligned}
&\mathbb{E}\left[\sup_{t\in[0,T]}\left|\int_0^t \alpha\left(Z_1(s)+\mathcal{O}(\Delta t), Z_1(s)\right)\mathrm{d}s\right|^2\right] \\
\leq\ & \mathbb{E}\left[\sup_{t\in[0,T]}\int_0^t 1^2\mathrm{d}s\int_0^t\left|\alpha\left(Z_1(s)+\mathcal{O}(\Delta t), Z_1(s)\right)\right|^2\mathrm{d}s\right] \\
\leq\ & T\mathbb{E}\left[\int_0^T\left|\alpha\left(Z_1(s)+\mathcal{O}(\Delta t), Z_1(s)\right)\right|^2\mathrm{d}s\right] \\
\leq\ & T^2Ls\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)\left(1+\mathcal{O}\left(\Delta t^2\right)\right)+2TLs\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)\int_0^T\mathbb{E}\left[|Z_1(s)|^2\right]\mathrm{d}s,
\end{aligned}$$

where we have successively applied the Cauchy-Schwarz inequality, Proposition 4.3.1 and Fubini's theorem.

(ii) Next, we find an upper bound for the expression with the Brownian motion. Using Doob's inequality, the Itô isometry, the linear growth bound of g_1^r and Fubini's theorem we obtain

$$\begin{aligned}
&\mathbb{E}\left[\sup_{t\in[0,T]}\left|\int_0^t g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}W_s^r\right|^2\right] \\
\leq\ & 4\mathbb{E}\left[\left|\int_0^T g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}W_s^r\right|^2\right] \\
=\ & 4\mathbb{E}\left[\int_0^T\left|g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\right|^2\mathrm{d}s\right] \\
\leq\ & 4TL\left(1+\mathcal{O}\left(\Delta t^2\right)\right)+8L\int_0^T\mathbb{E}\left[|Z_1(s)|^2\right]\mathrm{d}s.
\end{aligned}$$

(iii) Finally, we are looking to control the expression corresponding to the Poisson process. Using the same trick with the compensated Poisson process as in (iii) of the proof of Lemma 4.3.2, Doob's inequality, the Cauchy-Schwarz inequality and the linear growth bound of g_2^r we get

$$\begin{aligned}
&\mathbb{E}\left[\sup_{t\in[0,T]}\left|\int_0^t g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}N_s^r\right|^2\right]\\
=\;&\mathbb{E}\left[\sup_{t\in[0,T]}\left|\int_0^t g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}\widehat{N}_s^r+\rho\int_0^t g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}s\right|^2\right]\\
\leq\;&2\mathbb{E}\left[\sup_{t\in[0,T]}\left|\int_0^t g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}\widehat{N}_s^r\right|^2\right]\\
&+2\rho^2\mathbb{E}\left[\sup_{t\in[0,T]}\left|\int_0^t g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}s\right|^2\right]\\
\leq\;&8\mathbb{E}\left[\sup_{t\in[0,T]}\left|\int_0^t g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}\widehat{N}_s^r\right|^2\right]\\
&+2\rho^2\mathbb{E}\left[\sup_{t\in[0,T]}\left|\int_0^t g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}s\right|^2\right]\\
\leq\;&8\rho\int_0^T\mathbb{E}\left[\left|g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\right|^2\right]\mathrm{d}s+2\rho^2T\mathbb{E}\left[\left|g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\right|^2\right]\mathrm{d}s\\
\leq\;&\left(8\rho L+2\rho^2TL\right)\left(T\left(1+\mathcal{O}\left(\Delta t^2\right)\right)+2\int_0^T\mathbb{E}\left[|Z_1(s)|^2\right]\mathrm{d}s\right).
\end{aligned}$$

Putting the results $(i)-(iii)$ together, we obtain

$$\begin{aligned}
&\mathbb{E}\left[\sup_{t\in[0,T]}|Y(t)|^2\right]\\
\leq\;&4\mathbb{E}\left[|Y_0|^2\right]+4T^2Ls\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)\left(1+\mathcal{O}\left(\Delta t^2\right)\right)\\
&+16m_1^2TL\left(1+\mathcal{O}\left(\Delta t^2\right)\right)+4m_2^2T\left(8\rho L+2\rho^2TL\right)\left(1+\mathcal{O}\left(\Delta t^2\right)\right)\\
&+\left(8TLs\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)+32m_1^2L+8m_2^2\left(8\rho L+2\rho^2TL\right)\right)\int_0^T\mathbb{E}\left[|Z_1(s)|^2\right]\mathrm{d}s.
\end{aligned}$$

Applying Lemma 4.3.2 yields the desired result. □

Lemma 4.3.5. *Assume that the assumptions (A1) and (A2) hold. Then there exists* $\Delta t^*>0$ *such that for all* $0<t\leq\Delta t^*$

$$\mathbb{E}\left[\sup_{t\in[0,T]}|Y(t)-Z_1(t)|^2\right]\leq C_3\Delta t\left(1+\mathbb{E}\left[|X(0)|^2\right]\right).$$

Proof. Let $t \in [k\Delta t, (k+1)\,\Delta t]$, a subinterval of $[0, T]$. By definition of the numerical scheme in continuous time we get

$$\begin{aligned} Y(t) - Z_1(t) = Y(t) - Y_k &= \int_{k\Delta t}^{t} \alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right) \mathrm{d}s \\ &\quad + \sum_{r=1}^{m_1} \int_{k\Delta t}^{t} g_1^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) \mathrm{d}W_s^r \\ &\quad + \sum_{r=1}^{m_2} \int_{k\Delta t}^{t} g_2^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) \mathrm{d}N_s^r. \end{aligned}$$

Taking the square of the norm it follows that

$$\begin{aligned} |Y(t) - Z_1(t)|^2 &\leq 3\left|\int_{k\Delta t}^{t} \alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right) \mathrm{d}s\right|^2 \\ &\quad + 3m_1 \sum_{r=1}^{m_1} \left|\int_{k\Delta t}^{t} g_1^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) \mathrm{d}W_s^r\right|^2 \\ &\quad + 3m_2 \sum_{r=1}^{m_2} \left|\int_{k\Delta t}^{t} g_2^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) \mathrm{d}N_s^r\right|^2. \end{aligned}$$

Note that this holds for any $t \in [k\Delta t, (k+1)\,\Delta t]$. Therefore, we can take the supremum and the expectation, so that we get

$$\begin{aligned} &\mathbb{E}\left[\sup_{t\in[0,T]} |Y(t) - Z_1(t)|^2\right] \\ \leq\;& 3\mathbb{E}\left[\max_{k=0,1,\dots,\frac{T}{\Delta t}-1}\left(\sup_{\tau\in[k\Delta t,(k+1)\Delta t]}\left|\int_{k\Delta t}^{\tau} \alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right) \mathrm{d}s\right|^2\right)\right] \\ &+3m_1 \sum_{r=1}^{m_1} \mathbb{E}\left[\max_{k=0,1,\dots,\frac{T}{\Delta t}-1}\left(\sup_{\tau\in[k\Delta t,(k+1)\Delta t]}\left|\int_{k\Delta t}^{\tau} g_1^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) \mathrm{d}W_s^r\right|^2\right)\right] \\ &+3m_2 \sum_{r=1}^{m_2} \mathbb{E}\left[\max_{k=0,1,\dots,\frac{T}{\Delta t}-1}\left(\sup_{\tau\in[k\Delta t,(k+1)\Delta t]}\left|\int_{k\Delta t}^{\tau} g_2^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) \mathrm{d}N_s^r\right|^2\right)\right]. \end{aligned}$$

As for Lemma 4.3.2 and Lemma 4.3.4, we derive for each of the last three expressions an upper bound, and then we put them together.

(i) Using the Cauchy-Schwarz inequality, Proposition 4.3.1 and Fubini's theorem for the deterministic part we obtain

$$\begin{aligned}
&\mathbb{E}\left[\max_{k=0,1,\ldots,\frac{T}{\Delta t}-1}\left(\sup_{\tau\in[k\Delta t,(k+1)\Delta t]}\left|\int_{k\Delta t}^{\tau}\alpha\left(Z_1(s)+\mathcal{O}(\Delta t),Z_1(s)\right)\mathrm{d}s\right|^2\right)\right]\\
\leq\ & \max_{k=0,1,\ldots,\frac{T}{\Delta t}-1}\mathbb{E}\left[\sup_{\tau\in[k\Delta t,(k+1)\Delta t]}\int_{k\Delta t}^{\tau}1^2\mathrm{d}s\int_{k\Delta t}^{\tau}\left|\alpha\left(Z_1(s)+\mathcal{O}(\Delta t),Z_1(s)\right)\right|^2\mathrm{d}s\right]\\
\leq\ & \max_{k=0,1,\ldots,\frac{T}{\Delta t}-1}\Delta t\,\mathbb{E}\left[\int_{k\Delta t}^{(k+1)\Delta t}\left|\alpha\left(Z_1(s)+\mathcal{O}(\Delta t),Z_1(s)\right)\right|^2\mathrm{d}s\right]\\
\leq\ & \max_{k=0,1,\ldots,\frac{T}{\Delta t}-1}\Delta t\left(\Delta t Ls\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)\left(1+\mathcal{O}\left(\Delta t^2\right)\right)\right.\\
&\left.+2Ls\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)\int_{k\Delta t}^{(k+1)\Delta t}\mathbb{E}\left[|Z_1(s)|^2\right]\mathrm{d}s\right).
\end{aligned}$$

(ii) Taking into account Doob's inequality, the Itô isometry, the linear growth bound of g_1^r and Fubini's theorem yields for the stochastic expression associated to the Brownian motion

$$\begin{aligned}
&\mathbb{E}\left[\max_{k=0,1,\ldots,\frac{T}{\Delta t}-1}\left(\sup_{\tau\in[k\Delta t,(k+1)\Delta t]}\left|\int_{k\Delta t}^{\tau}g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}W_s^r\right|^2\right)\right]\\
\leq\ & \max_{k=0,1,\ldots,\frac{T}{\Delta t}-1}4\mathbb{E}\left[\left|\int_{k\Delta t}^{(k+1)\Delta t}g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}W_s^r\right|^2\right]\\
=\ & \max_{k=0,1,\ldots,\frac{T}{\Delta t}-1}4\mathbb{E}\left[\int_{k\Delta t}^{(k+1)\Delta t}\left|g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\right|^2\mathrm{d}s\right]\\
\leq\ & \max_{k=0,1,\ldots,\frac{T}{\Delta t}-1}4L\left(\Delta t\left(1+\mathcal{O}\left(\Delta t^2\right)\right)+2\int_{k\Delta t}^{(k+1)\Delta t}\mathbb{E}\left[|Z_1(s)|^2\right]\mathrm{d}s\right).
\end{aligned}$$

In the last step we have used a similar argument to (ii) in the proof of Lemma 4.3.4.

(iii) For the term corresponding to the Poisson process we have

$$\begin{aligned}
&\mathbb{E}\left[\max_{k=0,1,\ldots,\frac{T}{\Delta t}-1}\left(\sup_{\tau\in[k\Delta t,(k+1)\Delta t]}\left|\int_{k\Delta t}^{\tau}g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}N_s^r\right|^2\right)\right]\\
\leq\ & \max_{k=0,1,\ldots,\frac{T}{\Delta t}-1}2\mathbb{E}\left[\sup_{\tau\in[k\Delta t,(k+1)\Delta t]}\left|\int_{k\Delta t}^{\tau}g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}\widehat{N}_s^r\right|^2\right]\\
&+\max_{k=0,1,\ldots,\frac{T}{\Delta t}-1}2\rho^2\mathbb{E}\left[\sup_{\tau\in[k\Delta t,(k+1)\Delta t]}\int_{k\Delta t}^{\tau}g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)\mathrm{d}s|^2\right]\\
\leq\ & \max_{k=0,1,\ldots,\frac{T}{\Delta t}-1}\left(2\rho L+2\rho^2\Delta t L\right)\left(\Delta t\left(1+\mathcal{O}\left(\Delta t^2\right)\right)+2\int_{k\Delta t}^{(k+1)\Delta t}\mathbb{E}\left[|Z_1(s)|^2\right]\mathrm{d}s\right),
\end{aligned}$$

where we have used the compensated Poisson process $\left(\widehat{N}_t^r\right)_{t\in[0,T]}$ with $\widehat{N}_t^r=N_t^r-\rho t$ and the bounds derived in (iii) of the proof of Lemma 4.3.4.

Considering the results $(i)-(iii)$, we get

$$\begin{aligned}
& \mathbb{E}\left[\sup_{t\in[0,T]}|Y(t)-Z_1(t)|^2\right] \\
\leq\ & \max_{k=0,1,\dots,\frac{T}{\Delta t}-1} 3\Delta t^2 Ls\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)\left(1+\mathcal{O}\left(\Delta t^2\right)\right) \\
& +12m_1^2 L\Delta t\left(1+\mathcal{O}\left(\Delta t^2\right)\right) \\
& +3m_2^2\left(2\rho L+2\rho^2\Delta tL\right)\Delta t\left(1+\mathcal{O}\left(\Delta t^2\right)\right) \\
& +\left(6Ls\left(\sum_{j=0}^{s-1}\left|\lambda_{sj}\right|^2\right)\int_{k\Delta t}^{(k+1)\Delta t}+24m_1^2L+6m_2^2\left(2\rho L+2\rho^2\Delta tL\right)\right) \\
& \cdot\int_{k\Delta t}^{(k+1)\Delta t}\mathbb{E}\left[|Z_1(s)|^2\right]\mathrm{d}s.
\end{aligned}$$

Observe that by Lemma 4.3.2

$$\int_{k\Delta t}^{(k+1)\Delta t}\mathbb{E}\left[|Z_1(s)|^2\right]\mathrm{d}s \leq \int_{k\Delta t}^{(k+1)\Delta t} C_1\left(1+\mathbb{E}\left[|X(0)|^2\right]\right)\mathrm{d}s = \Delta t C_1\left(1+\mathbb{E}\left[|X(0)|^2\right]\right),$$

and thus, there is no more dependance on k and the result follows. □

We have now everything at hand to prove the following theorem.

Theorem 4.3.6. *Let* $(X(t))_{t\in[0,T]}$ *be a stochastic process defined by* (4.1) *and let* $(Y(t))_{t\in[0,T]}$ *be the numerical approximation by the S-ROCK1-JD method* (4.4) *that we have extended to continuous time* (4.7). *Suppose that the assumptions (A1) and (A2) hold. Then there exists a* $\Delta t^* > 0$ *such that for all* $0 < \Delta t \leq \Delta t^*$

$$\mathbb{E}\left[\sup_{t\in[0,T]}|Y(t)-X(t)|^2\right] \leq C_5\Delta t\left(1+\mathbb{E}\left[|X(0)|^2\right]\right).$$

Proof. By the definition of $Y(t)$ and $X(t)$ we have

$$\begin{aligned}
Y(t)-X(t) \ =\ & \int_0^t\left(\alpha\left(Z_1(s)+\mathcal{O}(\Delta t),Z_1(s)\right)-f\left(X_{s-}\right)\right)\mathrm{d}s \\
& +\sum_{r=1}^{m_1}\int_0^t\left(g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)-g_1^r\left(X_{s-}\right)\right)\mathrm{d}W_s^r \\
& +\sum_{r=1}^{m_2}\int_0^t\left(g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)-g_2^r\left(X_{s-}\right)\right)\mathrm{d}N_s^r.
\end{aligned}$$

Let $0 \leq t_1 \leq T$ and take the norm, the supremum over $[0, t_1]$ and the expectation to get

$$\begin{aligned}
&\mathbb{E}\left[\sup_{t\in[0,t_1]} |Y(t) - X(t)|^2\right] \\
\leq\ & 3\mathbb{E}\left[\sup_{t\in[0,t_1]} \left|\int_0^t \left(\alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right) - f\left(X_{s-}\right)\right) \mathrm{d}s\right|^2\right] \\
& +3m_1 \sum_{r=1}^{m_1} \mathbb{E}\left[\sup_{t\in[0,t_1]} \left|\int_0^t \left(g_1^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) - g_1^r\left(X_{s-}\right)\right) \mathrm{d}W_s^r\right|^2\right] \\
& +3m_2 \sum_{r=1}^{m_2} \mathbb{E}\left[\sup_{t\in[0,t_1]} \left|\int_0^t \left(g_2^r\left(Z_1(s) + \mathcal{O}(\Delta t)\right) - g_2^r\left(X_{s-}\right)\right) \mathrm{d}N_s^r\right|^2\right].
\end{aligned}$$

We proceed now by investigating each of the three expressions on the right-hand side separately.

(i) Using the Cauchy-Schwarz inequality and Fubini's theorem a first upper bound is given by

$$\begin{aligned}
&\mathbb{E}\left[\sup_{t\in[0,t_1]} \left|\int_0^t \left(\alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right) - f\left(X_{s-}\right)\right) \mathrm{d}s\right|^2\right] \\
\leq\ & \mathbb{E}\left[\sup_{t\in[0,t_1]} \int_0^t 1^2 \mathrm{d}s \int_0^t \left|\alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right) - f\left(X_{s-}\right)\right|^2 \mathrm{d}s\right] \\
\leq\ & t_1 \int_0^{t_1} \mathbb{E}\left[\left|\alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right) - f\left(X_{s-}\right)\right|^2\right] \mathrm{d}s.
\end{aligned}$$

Observe that for the S-ROCK1-JD method we have

$$K_s = Y_n + \Delta t \omega_1 \frac{T_s'(\omega_0)}{T_s(\omega_0)} f(Y_n) + \mathcal{O}\left(\Delta t^2\right)$$

with $\omega_1 \frac{T_s'(\omega_0)}{T_s(\omega_0)} = 1$ by the choice of ω_1 (and since the method converges). Hence, we get

$$K_s = Y_n + \Delta t f(Y_n) + \mathcal{O}\left(\Delta t^2\right),$$

and thus, α can be expressed as

$$\alpha(K_s, Y_n) = f(Y_n) + \mathcal{O}(\Delta t).$$

It follows that

$$\begin{aligned}
&\left|\alpha\left(Z_1(s) + \mathcal{O}(\Delta t), Z_1(s)\right) - f\left(X_{s-}\right)\right|^2 = \left|f\left(Z_1(s)\right) + \mathcal{O}(\Delta t) - f\left(X_{s-}\right)\right|^2 \\
&\leq 2K\left|Z_1(s) - X_{s-}\right|^2 + \mathcal{O}\left(\Delta t^2\right),
\end{aligned}$$

where we have used that f is Lipschitz continuous. Therefore, putting the results together yields

$$\begin{aligned} &\mathbb{E}\left[\sup_{t\in[0,t_1]}\left|\int_0^t \left(\alpha\left(Z_1(s)+\mathcal{O}(\Delta t), Z_1(s)\right) - f(X_{s-})\right)\mathrm{d}s\right|^2\right] \\ \leq\ & t_1\int_0^{t_1}\mathbb{E}\left[2K|Z_1(s)-X_{s-}|^2+\mathcal{O}\left(\Delta t^2\right)\right]\mathrm{d}s \\ =\ & 2Kt_1\int_0^{t_1}\mathbb{E}\left[|Z_1(s)-X_{s-}|^2\right]\mathrm{d}s + t_1^2\mathcal{O}\left(\Delta t^2\right). \end{aligned}$$

(ii) For the stochastic part driven by the Brownian motion we use Doob's inequality, the Itô isometry, the Lipschitz continuity of g_1^r and Fubini's theorem to get

$$\begin{aligned} &\mathbb{E}\left[\sup_{t\in[0,t_1]}\left|\int_0^t \left(g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right) - g_1^r(X_{s-})\right)\mathrm{d}W_s^r\right|^2\right] \\ \leq\ & 4\mathbb{E}\left[\left|\int_0^{t_1} \left(g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right) - g_1^r(X_{s-})\right)\mathrm{d}W_s^r\right|^2\right] \\ =\ & 4\mathbb{E}\left[\int_0^{t_1}\left|g_1^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right) - g_1^r(X_{s-})\right|^2\mathrm{d}s\right] \\ \leq\ & 4\mathbb{E}\left[\int_0^{t_1}K_1|Z_1(s)+\mathcal{O}(\Delta t)-X_{s-}|^2\mathrm{d}s\right] \\ \leq\ & 4K_1\left(t_1\mathcal{O}\left(\Delta t^2\right)+2\int_0^{t_1}\mathbb{E}\left[|Z_1(s)-X_{s-}|^2\right]\mathrm{d}s\right). \end{aligned}$$

(iii) We consider the compensated Poisson process $\left(\widehat{N}_t^r\right)_{t\in[0,T]}$ associated to the Poisson process $\left(N_t^r\right)_{t\in[0,T]}$ by $\widehat{N}_t^r = N_t^r - \rho t$. Observe that

$$\begin{aligned} &\mathbb{E}\left[\sup_{t\in[0,t_1]}\left|\int_0^t \left(g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right) - g_2^r(X_{s-})\right)\mathrm{d}N_s^r\right|^2\right] \\ \leq\ & 2\mathbb{E}\left[\sup_{t\in[0,t_1]}\left|\int_0^t \left(g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right) - g_2^r(X_{s-})\right)\mathrm{d}\widehat{N}_s^r\right|^2\right] \\ & +2\rho^2\mathbb{E}\left[\sup_{t\in[0,t_1]}\left|\int_0^t \left(g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right) - g_2^r(X_{s-})\right)\mathrm{d}s\right|^2\right]. \end{aligned}$$

Using Doob's inequality and the martingale isometry we obtain

$$\begin{aligned} &2\mathbb{E}\left[\sup_{t\in[0,t_1]}\left|\int_0^t \left(g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right) - g_2^r(X_{s-})\right)\mathrm{d}\widehat{N}_s^r\right|^2\right] \\ \leq\ & 8\mathbb{E}\left[\left|\int_0^{t_1} \left(g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right) - g_2^r(X_{s-})\right)\mathrm{d}\widehat{N}_s^r\right|^2\right] \\ =\ & 8\rho\int_0^{t_1}\mathbb{E}\left[\left|g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right) - g_2^r(X_{s-})\right|^2\right]\mathrm{d}s \\ \leq\ & 8\rho K_2\left(t_1\mathcal{O}\left(\Delta t^2\right)+2\int_0^{t_1}\mathbb{E}\left[|Z_1(s)-X_{s-}|^2\right]\mathrm{d}s\right), \end{aligned}$$

where we have used for the last step that g_2^r is Lipschitz continuous and Fubini's theorem to swap the integral and the expectation.

Furthermore, the Cauchy-Schwary inequality combined with the Lipschitz continuity of

g_2^r and Fubini's theorem results in

$$\begin{aligned}
& 2\rho^2\mathbb{E}\left[\sup_{t\in[0,t_1]}\left|\int_0^t \left(g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)-g_2^r\left(X_{s-}\right)\right)\mathrm{d}s\right|^2\right]\\
\leq\ & 2\rho^2 t_1\mathbb{E}\left[\int_0^{t_1}\left|g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)-g_2^r\left(X_{s-}\right)\right|^2\mathrm{d}s\right]\\
\leq\ & 2\rho^2 t_1 K_2\left(t_1\mathcal{O}\left(\Delta t^2\right)+2\int_0^{t_1}\mathbb{E}\left[|Z_1(s)-X_{s-}|^2\right]\mathrm{d}s\right).
\end{aligned}$$

Merging all upper bounds yields for the Poisson expression

$$\begin{aligned}
& \mathbb{E}\left[\sup_{t\in[0,t_1]}\left|\int_0^t \left(g_2^r\left(Z_1(s)+\mathcal{O}(\Delta t)\right)-g_2^r\left(X_{s-}\right)\right)\mathrm{d}N_s^r\right|^2\right]\\
\leq\ & \left(8\rho+2\rho^2 t_1\right)K_2\left(t_1\mathcal{O}\left(\Delta t^2\right)+2\int_0^{t_1}\mathbb{E}\left[|Z_1(s)-X_{s-}|^2\right]\mathrm{d}s\right).
\end{aligned}$$

Taking the results $(i)-(iii)$ we get

$$\begin{aligned}
& \mathbb{E}\left[\sup_{t\in[0,t_1]}|Y(t)-X(t)|^2\right]\\
\leq\ & 3t_1^2\mathcal{O}\left(\Delta t^2\right)+12m_1^2K_1t_1\mathcal{O}\left(\Delta t^2\right)+3m_2^2\left(8\rho+2\rho^2t_1\right)K_2t_1\mathcal{O}\left(\Delta t^2\right)\\
& +\left(6Kt_1+24m_1^2K_1+6m_2^2\left(8\rho+2\rho^2t_1\right)K_2\right)\int_0^{t_1}\mathbb{E}\left[|Z_1(s)-X_{s-}|^2\right]\mathrm{d}s\\
=\ & B_1\mathcal{O}\left(\Delta t^2\right)+B_2\int_0^{t_1}\mathbb{E}\left[|Z_1(s)-X_{s-}|^2\right]\mathrm{d}s
\end{aligned}$$

with

$$B_1 := 3t_1^2+12m_1^2K_1t_1+3m_2^2\left(8\rho+2\rho^2t_1\right)K_2t_1$$

and

$$B_2 := 6Kt_1+24m_1^2K_1+6m_2^2\left(8\rho+2\rho^2t_1\right)K_2.$$

Now observe that

$$|Z_1(s)-X_{s-}|^2=|Z_1(s)-Y(s)+Y(s)-X_{s-}|^2\leq 2|Z_1(s)-Y(s)|^2+2|Y(s)-X_{s-}|^2,$$

and thus,

$$\begin{aligned}
& \mathbb{E}\left[\sup_{t\in[0,t_1]}|Y(t)-X(t)|^2\right]\\
\leq\ & B_1\mathcal{O}\left(\Delta t^2\right)+2B_2\left(\int_0^{t_1}\mathbb{E}\left[|Z_1(s)-Y(s)|^2\right]\mathrm{d}s+\int_0^{t_1}\mathbb{E}\left[|Y(s)-X_{s-}|^2\right]\mathrm{d}s\right)\\
\leq\ & B_1\mathcal{O}\left(\Delta t^2\right)+2B_2C_3t_1\Delta t\left(1+\mathbb{E}\left[|X(0)|^2\right]\right)+2B_2\int_0^{t_1}\mathbb{E}\left[|Y(s)-X_{s-}|^2\right]\mathrm{d}s\\
\leq\ & B_1\mathcal{O}\left(\Delta t^2\right)+2B_2C_3t_1\Delta t\left(1+\mathbb{E}\left[|X(0)|^2\right]\right)+2B_2\int_0^{t_1}\mathbb{E}\left[\sup_{t\in[0,s]}|Y(s)-X_{s-}|^2\right]\mathrm{d}s,
\end{aligned}$$

where we have used Lemma 4.3.5. By using the continuous Gronwall inequality (see e.g. [76, 85]) we find the result of the theorem. □

This proves that the S-ROCK1-JD method for jump-diffusion processes is of strong order of convergence 1/2.

4.3.2 Strong Convergence of PIROCK-JD

In the previous section we have shown that the S-ROCK1-JD method for jump-diffusion processes is of strong order of convergence 1/2. Here, we show now that this result also holds for the PIROCK-JD method, which consists of the PIROCK method introduced in [14] enriched with a Poisson noise that deals with jumps and is defined in Definition 4.2.2. We consider the numerical approximation (4.5) with a time stepsize Δt. As for the S-ROCK1-JD method, this time stepsize represents the uniform time stepsize in a regular time grid and the largest time stepsize in a jump-adapted time grid.

Observe that we can rewrite the PIROCK-JD method scheme as

$$Y_{n+1} = Y_n + \alpha\left(K_s^*, K_{s-1}^*, K_{s-2}, Y_n\right)\Delta t + \sum_{r=1}^{m_1} g_1^r\left(K_{s+1}^*\right)\Delta W_{n+1}^r + \sum_{r=1}^{m_2} g_2^r\left(K_{s+1}^*\right)\Delta N_{n+1}^r$$

with

$$\alpha\left(K_s^*, K_{s-1}^*, K_{s-2}, Y_n\right) = \frac{K_s^* - \sigma_\alpha\left(1 - \frac{\tau_\alpha}{\sigma_\alpha^2}\right)\Delta t\left(f\left(K_{s-1}^*\right) - f\left(K_{s-2}\right)\right) - Y_n}{\Delta t}.$$

We state and prove now two propositions, which are required so that the proof of strong convergence for S-ROCK1-JD in Section 4.3.1 also holds for PIROCK-JD.

Proposition 4.3.7. *Consider the numerical method PIROCK-JD as defined in* (4.5). *Assume that f is sufficiently smooth. Then the following holds:*

$$\begin{array}{llll}
(a) & K_j & = & Y_n + \alpha P_j'(0)\Delta t f(Y_n) + \mathcal{O}\left(\Delta t^2\right), \ j = 0,1,\ldots,s, \\[2ex]
(b) & K_{s-1}^* & = & Y_n + \mathcal{O}(\Delta t), \\[2ex]
(c) & K_s^* & = & Y_n + \mathcal{O}(\Delta t), \\[2ex]
(d) & K_{s+1}^* & = & Y_n + \mathcal{O}(\Delta t), \\[2ex]
(e) & K_j & = & Y_n + \Delta t \sum_{i=0}^{j-1} \lambda_{ji} f(K_i), \ j = 0,1,\ldots,s.
\end{array}$$

Proof. (a) We use a proof by recurrence. Observe that the assertion holds for $K_0 = Y_n$ and $K_1 = Y_n + \alpha\mu_1\Delta t f(Y_n)$. Suppose now that the assertion is true for any $i \leq j-1$. Then we have

$$\begin{aligned}
K_j &= \alpha\mu_j\Delta t f\left(K_{j-1}\right) - \nu_j K_{j-1} - \kappa_j K_{j-2} \\
&= \alpha\mu_j\Delta t f\left(Y_n + \alpha P'_{j-1}(0)\Delta t f(Y_n) + \mathcal{O}\left(\Delta t^2\right)\right) \\
&\quad -\nu_j\left(Y_n + \alpha P'_{j-1}(0)\Delta t f(Y_n) + \mathcal{O}\left(\Delta t^2\right)\right) \\
&\quad -\kappa_j\left(Y_n + \alpha P'_{j-2}(0)\Delta t f(Y_n) + \mathcal{O}\left(\Delta t^2\right)\right) \\
&= \left(-\nu_j - \kappa_j\right) Y_n + \alpha\left(\mu_j - \nu_j P'_{j-1}(0) - \kappa_j P'_{j-2}(0)\right)\Delta t f(Y_n) + \mathcal{O}\left(\Delta t^2\right) \\
&= Y_n + \alpha P'_j(0)\Delta t f(Y_n) + \mathcal{O}\left(\Delta t^2\right),
\end{aligned}$$

where we have used $f\left(Y_n + \alpha P'_{j-1}(0)\Delta t f(Y_n) + \mathcal{O}\left(\Delta t^2\right)\right) = f(Y_n) + \mathcal{O}(\Delta t)$ (by Taylor expansion), $-\nu_j - \kappa_j = 1$ (by normalization of the orthogonal polynomials P_j) and $\mu_j - \nu_j P'_{j-1}(0) - \kappa_j P'_{j-2}(0) = P'_j(0)$.

(b) Using (a) and the Taylor expansion we obtain

$$\begin{aligned}
K^*_{s-1} &= K_{s-2} + \sigma_\alpha\Delta t f(K_{s-2}) \\
&= Y_n + \mathcal{O}(\Delta t) + \sigma_\alpha\Delta t f(Y_n + \mathcal{O}(\Delta t)) \\
&= Y_n + \mathcal{O}(\Delta t) + \sigma_\alpha\Delta t f(Y_n) + \mathcal{O}\left(\Delta t^2\right) = Y_n + \mathcal{O}(\Delta t).
\end{aligned}$$

(c) Applying the result (b) and using Taylor expansion we get

$$\begin{aligned}
K^*_s &= K^*_{s-1} + \sigma_\alpha\Delta t f\left(K^*_{s-1}\right) \\
&= Y_n + \mathcal{O}(\Delta t) + \sigma_\alpha\Delta t f(Y_n + \mathcal{O}(\Delta t)) \\
&= Y_n + \mathcal{O}(\Delta t) + \sigma_\alpha\Delta t f(Y_n) + \mathcal{O}\left(\Delta t^2\right) = Y_n + \mathcal{O}(\Delta t).
\end{aligned}$$

(d) The proof is similar to (a).

(e) We prove the result by recurrence. For $K_0 = Y_n$ and $K_1 = Y_n + \alpha\mu_1\Delta t f(K_0)$ the assertion holds. Suppose now that the assertion is also true for any $i \leq j-1$. It follows that

$$\begin{aligned}
K_j &= \alpha\mu_j \Delta t f\left(K_{j-1}\right) - \nu_j K_{j-1} - \kappa_j K_{j-2} \\
&= \alpha\mu_j \Delta t f\left(K_{j-1}\right) - \nu_j \left(Y_n + \Delta t \sum_{i=0}^{j-2} \lambda_{j-1 i} f\left(K_i\right) \right) \\
&\quad -\kappa_j \left(Y_n + \Delta t \sum_{i=0}^{j-3} \lambda_{j-2 i} f\left(K_i\right) \right) \\
&= (-\nu_j - \kappa_j) Y_n \\
&\quad +\Delta t \left(\alpha\mu_j f\left(K_{j-1}\right) + \sum_{i=0}^{j-3} (-\nu_j \lambda_{j-1 i} - \kappa_j \lambda_{j-2 i}) f(K_i) - \nu_j \lambda_{j-1 j-2} f\left(K_{j-2}\right) \right), \\
&= Y_n + \Delta t \sum_{i=0}^{j-1} \lambda_{j i} f\left(K_i\right),
\end{aligned}$$

where we used again the normalization of the polynomials P_j.

□

Proposition 4.3.8. *Consider the numerical scheme PIROCK-JD* (4.5)*. Assume that f is Lipschitz continuous.*

Then the following holds:

$$\begin{aligned}
&(a) \quad \alpha\left(K_s^*, K_{s-1}^*, K_{s-2}, Y_n\right) &&= f\left(Y_n\right) + \mathcal{O}\left(\Delta t\right) \\
&(b) \quad \left|\alpha\left(K_s^*, K_{s-1}^*, K_{s-2}, Y_n\right)\right|^2 &&\leq \widehat{C}\left(1 + 2\left|Y_n\right|^2 + \mathcal{O}\left(\Delta t^2\right)\right),
\end{aligned}$$

where

$$\widehat{C} := \left(6L(s-2) \left(\sum_{i=0}^{s-3} |\lambda_{s-2} i|^2 \right) + 12\sigma_\alpha^2 L + 8 \left(\sigma_\alpha \left(1 - \frac{\tau_\alpha}{\sigma_\alpha^2} \right) \right)^2 K \right) \left(1 + 2\left|Y_n\right|^2 + \mathcal{O}\left(\Delta t^2\right)\right).$$

Proof. (a) Since the numerical method converges it holds that $Y_{n+1} = Y_n + \Delta t f\left(Y_n\right) + \mathcal{O}\left(\Delta t^2\right)$. The result follows.

(b) By definition we have

$$\begin{aligned}
\alpha\left(K_s^*, K_{s-1}^*, K_{s-2}, Y_n\right) &= \frac{K_s^* - \sigma_\alpha\left(1 - \frac{\tau_\alpha}{\sigma_\alpha^2}\right)\Delta t\left(f\left(K_{s-1}^*\right) - f\left(K_{s-2}\right)\right) - Y_n}{\Delta t} \\
&= \frac{K_s^* - Y_n}{\Delta t} - \sigma_\alpha\left(1 - \frac{\tau_\alpha}{\sigma_\alpha^2}\right)\left(f\left(K_{s-1}^*\right) - f\left(K_{s-2}\right)\right).
\end{aligned}$$

It follows that

$$\begin{aligned}
&\left|\alpha\left(K_s^*, K_{s-1}^*, K_{s-2}, Y_n\right)\right|^2 \\
\leq\;& 2\left|\tfrac{K_s^*-Y_n}{\Delta t}\right|^2 + 2\left(\sigma_\alpha\left(1-\tfrac{\tau_\alpha}{\sigma_\alpha^2}\right)\right)^2 \left|f\left(K_{s-1}^*\right) - f\left(K_{s-2}\right)\right|^2 \\
\leq\;& 2\left|\tfrac{K_{s-1}^*-Y_n}{\Delta t} + \sigma_\alpha f\left(K_{s-1}^*\right)\right|^2 + 2\left(\sigma_\alpha\left(1-\tfrac{\tau_\alpha}{\sigma_\alpha^2}\right)\right)^2 K\left|K_{s-1}^* - K_{s-2}\right|^2 \\
\leq\;& 2\left|\tfrac{K_{s-2}-Y_n}{\Delta t} + \sigma_\alpha f\left(K_{s-1}^*\right) + \sigma_\alpha f\left(K_{s-2}\right)\right|^2 \\
&+2\left(\sigma_\alpha\left(1-\tfrac{\tau_\alpha}{\sigma_\alpha^2}\right)\right)^2 K\left(2\left|K_{s-1}^*\right|^2 + 2\left|K_{s-2}\right|^2\right) \\
\leq\;& 6\left|\tfrac{K_{s-2}-Y_n}{\Delta t}\right|^2 + 6\sigma_\alpha^2\left|f\left(K_{s-1}^*\right)\right|^2 + 6\sigma_\alpha^2\left|f\left(K_{s-2}\right)\right|^2 \\
&+16\left(\sigma_\alpha\left(1-\tfrac{\tau_\alpha}{\sigma_\alpha^2}\right)\right)^2 K\left(|Y_n|^2 + \mathcal{O}\left(\Delta t^2\right)\right),
\end{aligned}$$

where we have used that f is Lipschitz continuous, the definitions of K_s^* and K_{s-1}^* and Proposition 4.3.7 (a). Observe that $\left|\frac{K_{s-2}-Y_n}{\Delta t}\right|^2 = \left|\sum_{i=0}^{s-3} \lambda_{s-2} i f\left(K_i\right)\right|^2$ by Proposition 4.3.7 (e). Using Proposition 4.3.1 yields

$$\left|\frac{K_{s-2}-Y_n}{\Delta t}\right|^2 \leq L(s-2)\left(\sum_{i=0}^{s-3} |\lambda_{s-2} i|^2\right)\left(1+2|Y_n|^2+\mathcal{O}\left(\Delta t^2\right)\right).$$

Moreover, taking into account the linear growth bound of f and Proposition 4.3.7 (b) we obtain

$$\left|f\left(K_{s-1}^*\right)\right|^2 \leq L\left(1+\left|K_{s-1}^*\right|^2\right) \leq L\left(1+2|Y_n|^2+\mathcal{O}\left(\Delta t^2\right)\right).$$

The same holds for $\left|f\left(K_{s-2}\right)\right|^2$.

Putting all upper bounds together results in

$$\begin{aligned}
&\left|\alpha\left(K_s^*, K_{s-1}^*, K_{s-2}, Y_n\right)\right|^2 \\
\leq\;& \left(6L(s-2)\left(\sum_{i=0}^{s-3} |\lambda_{s-2} i|^2\right) + 12\sigma_\alpha^2 L\right)\left(1+2|Y_n|^2+\mathcal{O}\left(\Delta t^2\right)\right) \\
&+16\left(\sigma_\alpha\left(1-\tfrac{\tau_\alpha}{\sigma_\alpha^2}\right)\right)^2 K\left(|Y_n|^2+\mathcal{O}\left(\Delta t^2\right)\right) \\
\leq\;& \left(6L(s-2)\left(\sum_{i=0}^{s-3} |\lambda_{s-2} i|^2\right) + 12\sigma_\alpha^2 L + 8\left(\sigma_\alpha\left(1-\tfrac{\tau_\alpha}{\sigma_\alpha^2}\right)\right)^2 K\right) \\
&\cdot\left(1+2|Y_n|^2+\mathcal{O}\left(\Delta t^2\right)\right),
\end{aligned}$$

which concludes the proof.

□

Considering the results of Proposition 4.3.7 and Proposition 4.3.8 it is straightforward to adapt the proof of strong convergence for S-ROCK1-JD in Section 4.3.1 to the numerical method (4.5). Therefore, the PIROCK-JD method with no reaction and no advection term is also of strong order of convergence 1/2.

4.4 Mean Square Stability

In this section we study the mean square stability of S-ROCK1-JD and PIROCK-JD. Let the time stepsize Δt be fixed and let us look at the long-term behavior of the stochastic problem, i.e. how does the exact and the numerical solution to the problem behave as $t \to \infty$.

First of all, to study the mean square stability, we consider the linear test equation given by

$$\begin{cases} \mathrm{d}X(t) &= \mu X(t-)\mathrm{d}t + \sigma X(t-)\mathrm{d}W_t + \gamma X(t-)\mathrm{d}N_t, \;\; t>0, \\ \\ X(0) &= X_0, \end{cases} \tag{4.8}$$

with X_0 given ($X_0 \neq 0$ with probability one), $X(t) \in \mathbb{R}$, μ, σ and γ real constants. The exact solution to (4.8) is given by

$$X(t) = X_0 \exp\left\{\left(\mu - \frac{1}{2}\sigma^2\right) t + \sigma W(t)\right\} (1+\gamma)^{N(t)} \tag{4.9}$$

(see e.g. [50, 67]). In what follows we suppose that $\gamma \neq -1$, however the results remain true in the case $\gamma = -1$. Observe that

$$\begin{aligned} \mathbb{E}\left[(1+\gamma)^{2N(t)}\right] &= \mathbb{E}\left[\exp\left\{N(t)\ln(1+\gamma)^2\right\}\right] \\ &= \exp\left\{\rho t\left(\exp\left(\ln(1+\gamma)^2\right)-1\right)\right\} \\ &= \exp\left\{\rho t\left((1+\gamma)^2-1\right)\right\} = \exp\{\rho t\gamma(\gamma+2)\}, \end{aligned} \tag{4.10}$$

where we have used the moment generating function of a Poisson distribution. Putting $X(t)$ of (4.9) to the square and taking the expectation yields

$$\begin{aligned} \mathbb{E}\left[X(t)^2\right] &= \mathbb{E}\left[X_0^2\right]\exp\left\{2\left(\mu-\tfrac{1}{2}\sigma^2\right)t\right\}\mathbb{E}\left[\exp\{2\sigma W(t)\}\right]\mathbb{E}\left[(1+\gamma)^{2N(t)}\right] \\ &= \mathbb{E}\left[X_0^2\right]\exp\left\{2\left(\mu-\tfrac{1}{2}\sigma^2\right)t\right\}\exp\left\{2\sigma^2 t\right\}\exp\{\rho t\gamma(\gamma+2)\} \\ &= \mathbb{E}\left[X_0^2\right]\exp\left\{\left(2\mu+\sigma^2+\rho\gamma(\gamma+2)\right)t\right\}, \end{aligned}$$

where we have taken into account the moment generating function of a normal distribution and (4.10). Since $X_0 \neq 0$ with probability one, it follows that the exact solution of (4.8) is mean square stable if and only if

$$\lim_{t\to\infty} \mathbb{E}\left[X(t)^2\right] = 0 \Leftrightarrow 2\mu + \sigma^2 + \rho\gamma\left(\gamma+2\right) < 0. \tag{4.11}$$

We proceed now by computing the mean square stability domains of the numerical schemes and then we will compare them to the true mean square stability domain (4.11).

4.4.1 Mean Square Stability Domain of S-ROCK1-JD

In this section we derive the mean square stability domain of the S-ROCK1-JD method defined in (4.4). First of all, we apply the S-ROCK1-JD scheme for jump-diffusions to the linear test problem (4.8), which results in

$$Y_{n+1} = K_s + \sigma K_s \Delta W_{n+1} + \gamma K_s \Delta N_{n+1}$$

with

$$K_j = 2\Delta t\mu\omega_1 \frac{T_{j-1}(\omega_0)}{T_j(\omega_0)} K_{j-1} + 2\omega_0 \frac{T_{j-1}(\omega_0)}{T_j(\omega_0)} K_{j-1} - \frac{T_{j-2}(\omega_0)}{T_j(\omega_0)} K_{j-2}.$$

Proposition 4.4.1. *Using the S-ROCK1-JD method* (4.4) *applied to the linear test equation* (4.8) *leads to*

$$K_j = \frac{T_j\left(\omega_0 + \Delta t\mu\omega_1\right)}{T_j(\omega_0)} Y_n,$$

for all $j = 0, 1, \ldots, s$.

Proof. The result is straightforward using a proof by recurrence. □

Next by using Proposition 4.4.1 we obtain

$$\begin{aligned} Y_{n+1} &= \tfrac{T_s(\omega_0+\Delta t\mu\omega_1)}{T_s(\omega_0)} Y_n + \sigma \tfrac{T_s(\omega_0+\Delta t\mu\omega_1)}{T_s(\omega_0)} Y_n \Delta W_{n+1} + \gamma \tfrac{T_s(\omega_0+\Delta t\mu\omega_1)}{T_s(\omega_0)} Y_n \Delta N_{n+1} \\ &= \left(1 + \sigma\Delta W_{n+1} + \gamma \Delta N_{n+1}\right) \tfrac{T_s(\omega_0+\Delta t\mu\omega_1)}{T_s(\omega_0)} Y_n. \end{aligned}$$

Putting Y_{n+1} to the square and taking the expectation yields

$$\mathbb{E}\left[Y_{n+1}^2\right] = \frac{T_s^2\left(\omega_0 + \Delta t\mu\omega_1\right)}{T_s^2(\omega_0)} \mathbb{E}\left[\left(1 + \sigma\Delta W_{n+1} + \gamma\Delta N_{n+1}\right)^2\right] \mathbb{E}\left[Y_n^2\right],$$

where we have used the independence of the stochastic increments. Observe that

$$\begin{aligned} & \mathbb{E}\left[\left(1+\sigma\Delta W_{n+1}+\gamma\Delta N_{n+1}\right)^2\right] \\ = & \ \mathbb{E}\left[1+\sigma^2\Delta W_{n+1}^2+\gamma^2\Delta N_{n+1}^2+2\sigma\Delta W_{n+1}+2\gamma\Delta N_{n+1}+2\sigma\Delta W_{n+1}\gamma\Delta N_{n+1}\right] \\ = & \ 1+\sigma^2\mathbb{E}\left[\Delta W_{n+1}^2\right]+\gamma^2\mathbb{E}\left[\Delta N_{n+1}^2\right]+2\sigma\mathbb{E}\left[\Delta W_{n+1}\right]+2\gamma\mathbb{E}\left[\Delta N_{n+1}\right]+2\sigma\mathbb{E}\left[\Delta W_{n+1}\right]\gamma\mathbb{E}\left[\Delta N_{n+1}\right] \\ = & \ 1+\sigma^2\Delta t+\gamma^2\rho\Delta t\left(1+\rho\Delta t\right)+2\gamma\rho\Delta t \\ = & \ 1+\sigma^2\Delta t+\rho\Delta t\gamma\left(\gamma+2\right)+\gamma^2\rho^2\Delta t^2, \end{aligned}$$

where we have used the independence of the stochastic processes as well as the mean and the variance of a normal and a Poisson distribution, and thus,

$$\mathbb{E}\left[Y_{n+1}^2\right]=\frac{T_s^2\left(\omega_0+\Delta t\mu\omega_1\right)}{T_s^2\left(\omega_0\right)}\left(1+\sigma^2\Delta t+\rho\Delta t\gamma\left(\gamma+2\right)+\gamma^2\rho^2\Delta t^2\right)\mathbb{E}\left[Y_n^2\right].$$

Finally, the S-ROCK1-JD method is mean square stable if and only if

$$\lim_{t\to\infty}\mathbb{E}\left[Y_{n+1}^2\right]=0\Leftrightarrow\left|\frac{T_s^2\left(\omega_0+\Delta t\mu\omega_1\right)}{T_s^2\left(\omega_0\right)}\left(1+\sigma^2\Delta t+\rho\Delta t\gamma\left(\gamma+2\right)+\gamma^2\rho^2\Delta t^2\right)\right|<1. \tag{4.12}$$

4.4.2 Mean Square Stability Domain of PIROCK-JD

Here, we study the mean square stability of the PIROCK-JD method (4.5). Before we start developing the mean square stability domain of the PIROCK-JD scheme, we state a result for the internal stages.

Proposition 4.4.2. *Applying the PIROCK-JD scheme to the linear test problem* (4.8) *yields for the Runge-Kutta stages*

$$K_j=P_j\left(\alpha\Delta t\mu\right)Y_n=P_j\left(\alpha p\right)Y_n$$

for $j=0,1,\dots,s$ *and with* $p=\Delta t\mu$.

Proof. We prove the result by recurrence. For K_0 and K_1 the assertion holds. Suppose now the result is also true for any $i\leq j-1$. Then we have

$$\begin{aligned} K_j &= \alpha\mu_j\Delta t\mu K_{j-1}-\nu_jK_{j-1}-\kappa_jK_{j-2} \\ &= \left(\left(\mu_j\alpha\Delta t\mu-\nu_j\right)P_{j-1}\left(\alpha\Delta t\mu\right)-\kappa_jP_{j-2}\left(\alpha\Delta t\mu\right)\right)Y_n \\ &= P_j\left(\alpha\Delta t\mu\right)Y_n, \end{aligned}$$

where we have used the recurrence relation of the orthogonal polynomials (see [1]). □

Using Proposition 4.4.2 we get for the additional stages

$$\begin{aligned} K^*_{s-1} &= K_{s-2} + \sigma_\alpha \Delta t \mu K_{s-2} &&= \left(1+\sigma_\alpha p\right) P_{s-2}\left(\alpha p\right) Y_n, \\ K^*_{s} &= K^*_{s-1} + \sigma_\alpha \Delta t \mu K^*_{s-1} &&= \left(1+\sigma_\alpha p\right)^2 P_{s-2}\left(\alpha p\right) Y_n, \\ K^*_{s+1} &= K_{s} + \beta \Delta t \mu K_{s} &&= \left(1+\beta p\right) P_{s}\left(\alpha p\right) Y_n. \end{aligned}$$

Hence, we obtain

$$\begin{aligned} Y_{n+1} &= K^*_s - \sigma_\alpha \left(1 - \tfrac{\tau_\alpha}{\sigma_\alpha^2}\right) \Delta t \mu \left(K^*_{s-1} - K_{s-2}\right) + \sigma K^*_{s+1} \Delta W_{n+1} + \gamma K^*_{s+1} \Delta N_{n+1} \\ &= \left(1+\sigma_\alpha p\right)^2 P_{s-2}\left(\alpha p\right) Y_n - \sigma_\alpha \left(1 - \tfrac{\tau_\alpha}{\sigma_\alpha^2}\right) p \left(\left(1+\sigma_\alpha p\right) P_{s-2}\left(\alpha p\right) Y_n - P_{s-2}\left(\alpha p\right) Y_n\right) \\ &\quad + \sigma \left(1+\beta p\right) P_s\left(\alpha p\right) Y_n \Delta W_{n+1} + \gamma \left(1+\beta p\right) P_s\left(\alpha p\right) Y_n \Delta N_{n+1} \\ &= \left(1+2\sigma_\alpha p + \tau_\alpha p^2\right) P_{s-2}\left(\alpha p\right) Y_n \\ &\quad + \sqrt{\Delta t} \sigma \left(1+\beta p\right) P_s\left(\alpha p\right) Y_n \xi + \gamma \left(1+\beta p\right) P_s\left(\alpha p\right) Y_n \Delta N_{n+1}, \end{aligned}$$

where $\xi \sim N(0,1)$. For simplicity we can rewrite this as

$$Y_{n+1} = \left(A(p) + B(p) q \xi + \gamma C(p) \Delta N_{n+1}\right) Y_n,$$

where we use $q := \sqrt{\Delta t}\sigma$, $A(p) := \left(1+2\sigma_\alpha p + \tau_\alpha p^2\right) P_{s-2}\left(\alpha p\right)$, $B(p) := \left(1+\beta p\right) P_s\left(\alpha p\right)$ and $C(p) := B(p)$. Without loss of generality we assume now $Y_0 = 1$. Taking the expectation of Y_{n+1} to the power two and using the mean and the variance of the Gaussian and the Poisson distribution leads to

$$\begin{aligned} \mathbb{E}\left[Y^2_{n+1}\right] &= \mathbb{E}\left[Y^2_n\right] \Big(A(p)^2 + B(p)^2 q^2 \mathbb{E}\left[\xi^2\right] + \gamma^2 C(p)^2 \mathbb{E}\left[\Delta N^2_{n+1}\right] + 2A(p)B(p)q\mathbb{E}\left[\xi\right] \\ &\quad + 2A(p)C(p)\gamma\mathbb{E}\left[\Delta N_{n+1}\right] + 2B(p)\gamma C(p) q \mathbb{E}\left[\xi\right] \mathbb{E}\left[\Delta N_{n+1}\right]\Big) \\ &= \mathbb{E}\left[Y^2_n\right] \Big(A(p)^2 + B(p)^2 q^2 + \gamma^2 \left(\rho \Delta t + \left(\rho \Delta t\right)^2\right) C(p)^2 + 2A(p)C(p)\gamma\rho\Delta t\Big). \end{aligned}$$

Therefore, the PIROCK-JD method is mean square stable if and only if

$$\begin{aligned} & \lim_{t\to\infty} \mathbb{E}\left[Y^2_{n+1}\right] = 0 \\ \Leftrightarrow \quad & \left| A(p)^2 + B(p)^2 q^2 + \rho \Delta t \gamma^2 C(p)^2 + 2\rho\Delta t \gamma A(p) C(p) + \left(\rho\Delta t\right)^2 \gamma^2 C(p)^2 \right| < 1. \end{aligned}$$

4.4.3 Illustration of the Stability Regions of S-ROCK1-JD

In this section we illustrate the stability regions (i.e. we restrict the stability domain to real parameters) for the S-ROCK1-JD method. Recall that we consider the test problem (4.8) and that the stability region of the exact solution to the test problem is characterized by

$$2\mu + \sigma^2 + \rho\gamma\left(\gamma + 2\right) < 0$$

(see (4.11)). By multiplying on both sides by the uniform time stepsize Δt, we can rewrite this as

$$2\mu\Delta t + \sigma^2\Delta t + \rho\gamma\left(\gamma + 2\right)\Delta t < 0$$

or as

$$q^2 < -2p - r \tag{4.13}$$

with $p = \mu\Delta t$, $q = \sigma\sqrt{\Delta t}$ and $r = \rho\gamma\left(\gamma + 2\right)\Delta t$. Observe that one can reformulate the mean square stability condition of the S-ROCK1-JD integrator (4.12) as

$$\left|\frac{T_s^2\left(\omega_0 + p\omega_1\right)}{T_s^2\left(\omega_0\right)}\left(1 + q^2 + r_1 + r_2\right)\right| < 1,$$

where p and q are defined as above and $r_1 = \rho\gamma\left(\gamma + 2\right)\Delta t$ and $r_2 = \rho^2\gamma^2\Delta t^2$. This characterization is equivalent to

$$q^2 < \frac{T_s^2\left(\omega_0\right)}{T_s^2\left(\omega_0 + p\omega_1\right)} - 1 - r_1 - r_2. \tag{4.14}$$

Using (4.13) and (4.14), we illustrate now how much of the true stability region is covered by the stability region of the S-ROCK1-JD method. We distinguish two different scenarios, first we look at $r \geq 0$ and then we study the case where $r < 0$. For the first case we proceed as follows. We fix a stage number s and then we plot the stability domains for different values of r, respectively r_1. Note that to get the plots we have ignored r_2. Since r_2 consists only of positive terms, it is positive as well. Hence, neglecting the proportion between r_1 and r_2, adding r_2 has the same effect as increasing r_1.

Figure 4.1 shows the stability regions of the S-ROCK1-JD integrator using $s = 6$ stages and an adequate damping (see [12, 15]). The values of r we have chosen to plot are $r = 0$ (which corresponds to the case with no jumps), $r = 1$, $r = 10$ and $r = 20$. One observes that in all cases still a portion of the true stability region is covered by the stability region of the numerical scheme. As r increases, the region of the true stability region moves to the left and the stability region of the S-ROCK1-JD method is slightly reduced on the left-hand as well as on the right-hand side.

In Figure 4.2 the results for the S-ROCK1-JD method with $s = 13$ is shown. The same comments as for the previous case with 6 stages hold. In addition, one observes that as r increases, the stability region of the numerical scheme decreases in height. However, there is still a whole portion of the true stability domain covered.

Figure 4.3 shows the stability regions of the S-ROCK1-JD method with $s = 28$ stages. The same can be said about these regions as for the two previous figures. Note that in the last two plots

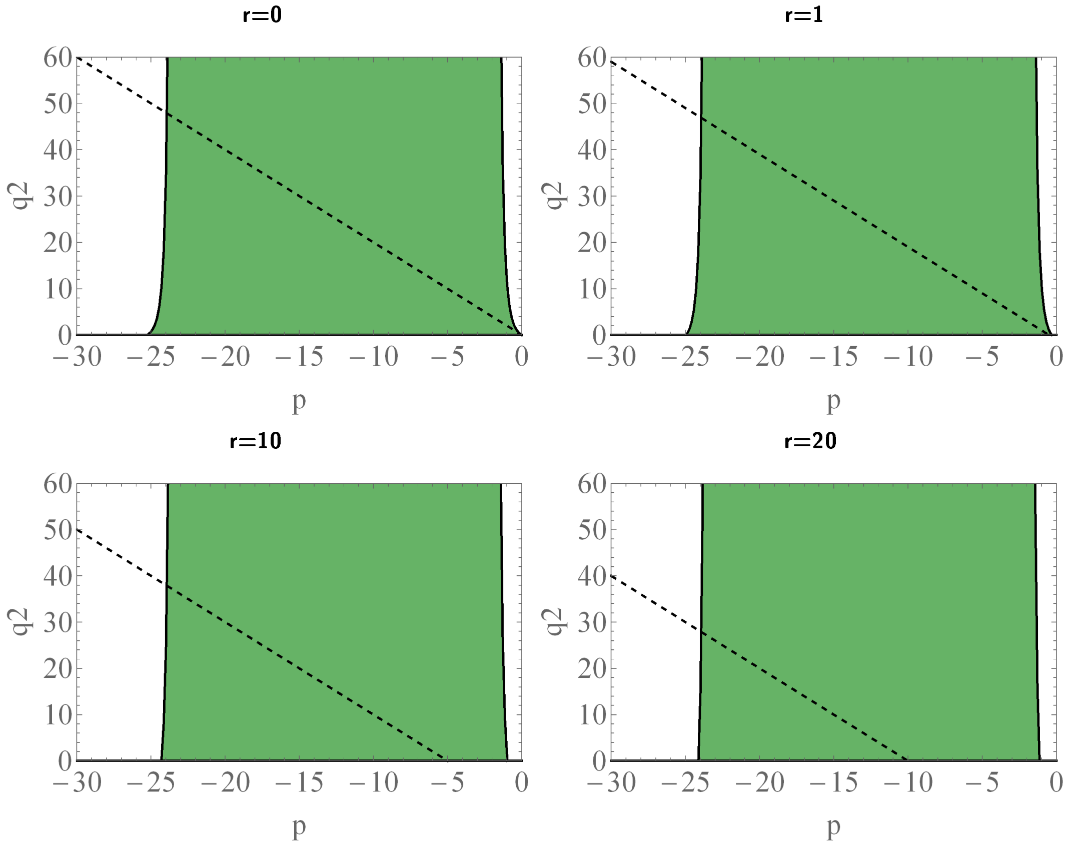

Figure 4.1: Stability regions of the S-ROCK1-JD method with $s = 6$ stages and for different values of $r \in \{0, 1, 10, 20\}$. The true stability region is indicated by the area below the dotted line.

($r = 10$ and $r = 20$) the stability region of the numerical integrator nearly cuts into the stability region of the true stability region of the test problem. Should this be the case, one can increase the damping accordingly which will increase the stability region in vertical direction. However, this also reduces the area covered by the stability domain in horizontal direction. Hence, if needed one can also increase the stage number s.

Note that a critical area for numerical stability is often the region around the origin. In Figure 4.4 we show that stability region of the numerical method covers everything of the true stability region near the origin. Similar plots can be obtained for the other cases.

We study now the second scenario. What happens if r is negative, i.e. $r < 0$? First of all, observe that we can express r_2 by r_1. In fact,

$$r_2 = \rho^2\gamma^2\Delta t^2 = \rho^2\gamma^2\Delta t^2 \frac{(\gamma+2)^2}{(\gamma+2)^2} = \frac{r_1^2}{(\gamma+2)^2}.$$

Without loss of generality we assume that $\gamma = -1$. Next, we fix a stage number s and an

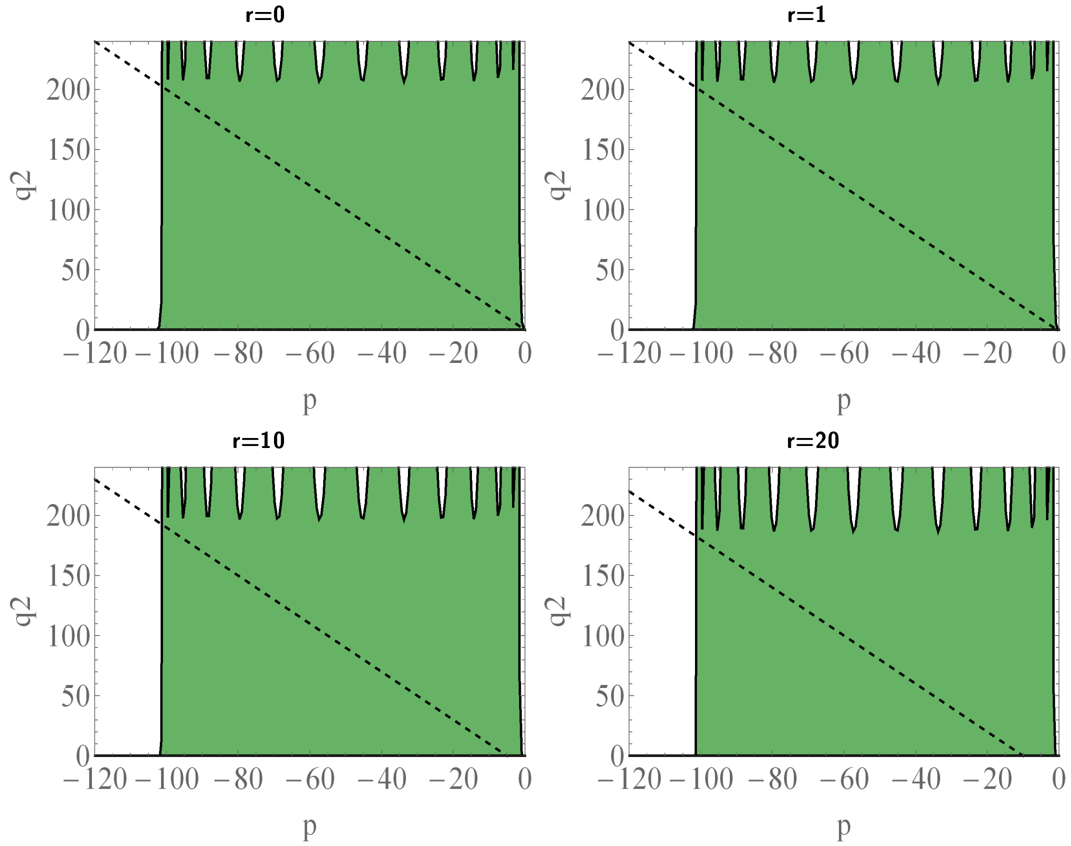

Figure 4.2: Stability regions of the S-ROCK1-JD method with $s = 13$ stages and for different values of $r \in \{0, 1, 10, 20\}$. The true stability region is indicated by the area below the dotted line.

adequate damping (see [12, 15]). Finally, we vary the value of r, respectively r_1. Here we consider $r \in \{-1, -0.5, -0.1\}$.

Figure 4.5 shows the stability regions of the S-ROCK1-JD scheme with $s = 6$ stages. A region of special interest is the one around the origin, that is why we provide for each stability region a zoom of the region near zero. One can observe that for $r = -1$ there is a whole part of the true stability region that is not covered by the stability region of the numerical method. However, by decreasing r in its absolute value, one notices that more and more of the true stability domain is covered around the origin.

Figure 4.6 and Figure 4.7 lead to the same conclusion for the case of $s = 13$ and $s = 28$ as for the case of $s = 6$. Since r depends on ρ and γ, which are usually fixed and given by the problem, one can only change the value of r by adjusting the time stepsize Δt. We state and prove now a proposition that helps us to choose an appropriate time stepsize.

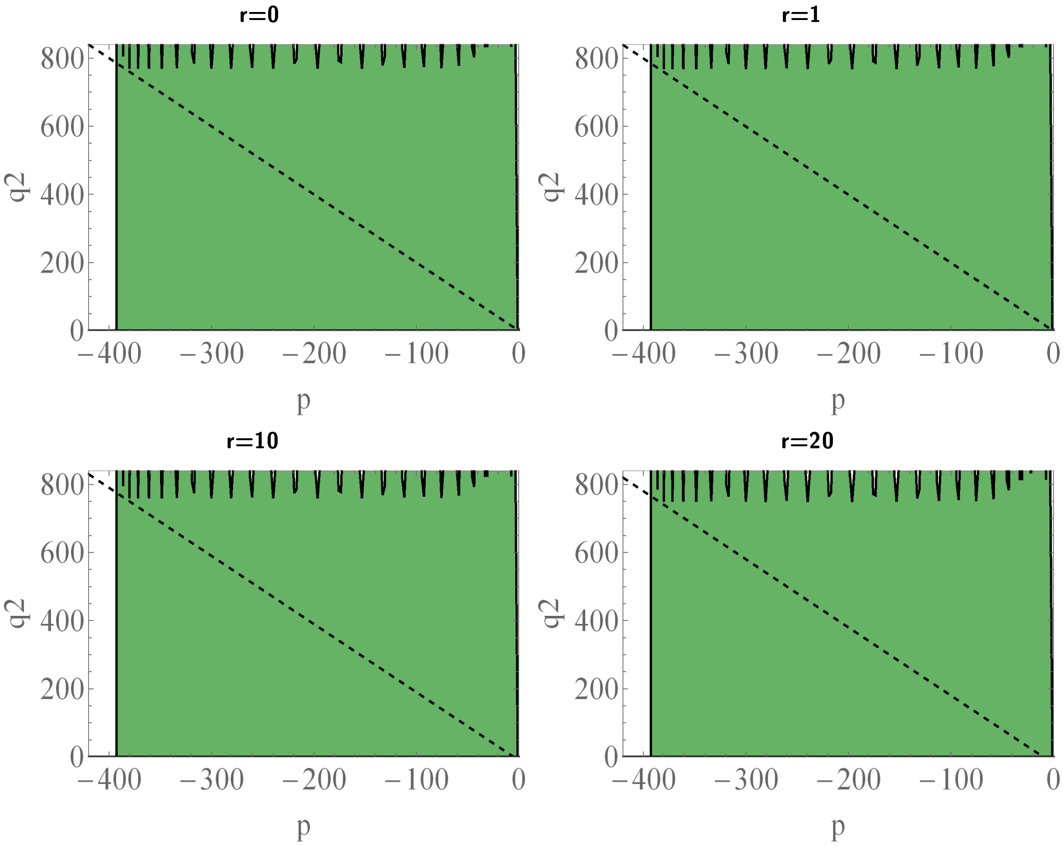

Figure 4.3: Stability regions of the S-ROCK1-JD method with $s = 28$ stages and for different values of $r \in \{0, 1, 10, 20\}$. The true stability region is indicated by the area below the dotted line.

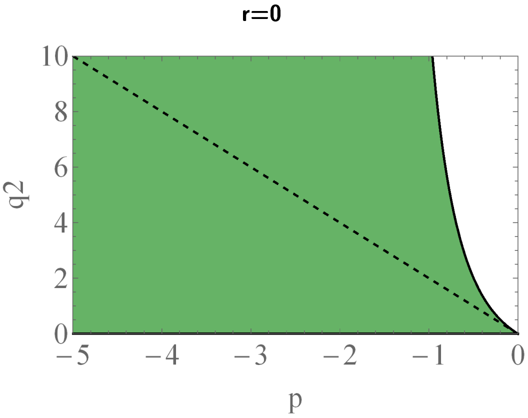

Figure 4.4: Zoom of the stability region of the S-ROCK1-JD method with $s = 6$ stages around the origin. The true stability region is represented by the area below the dotted line.

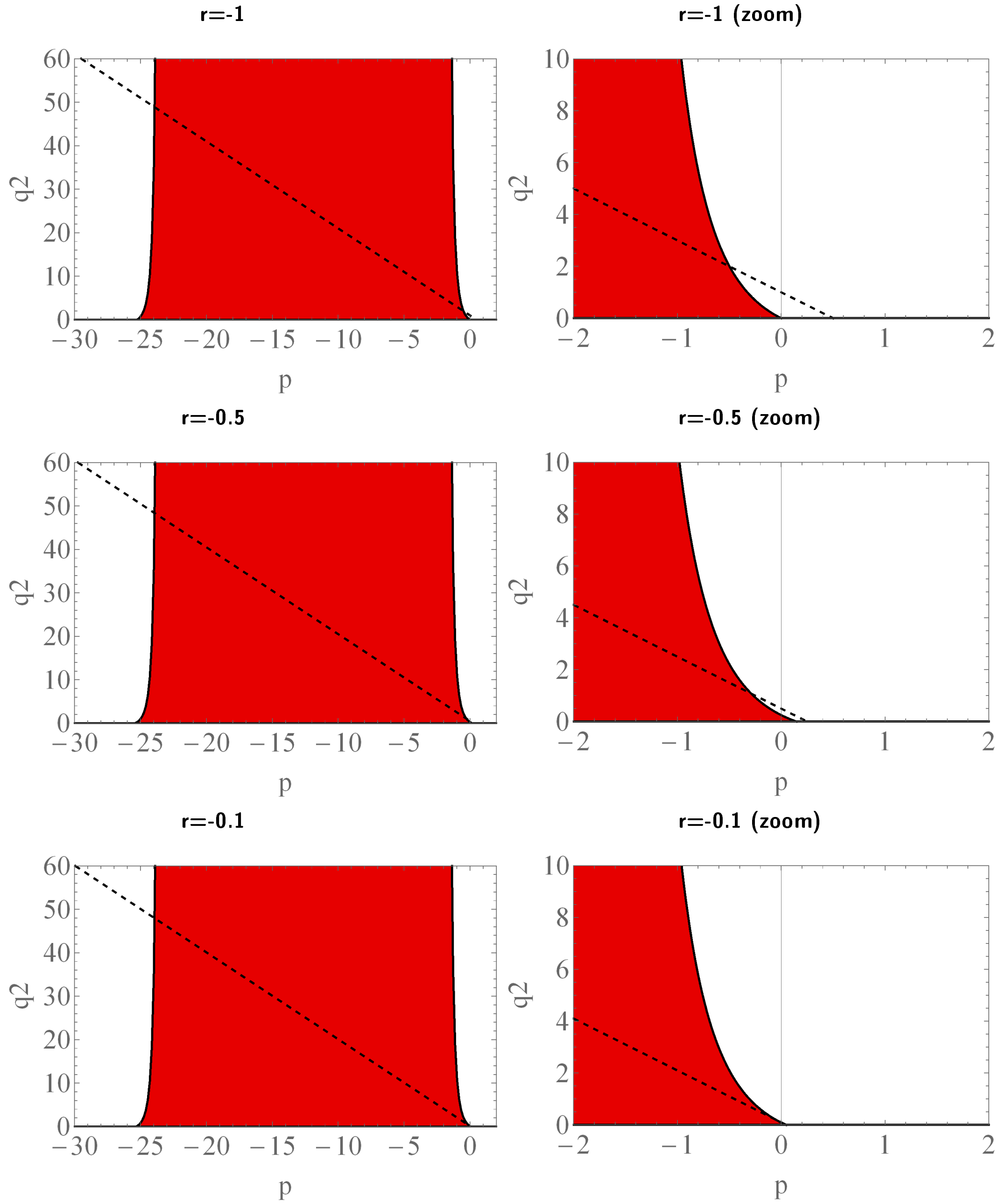

Figure 4.5: Stability regions of the S-ROCK1-JD method with $s = 6$ stages and for different values of $r \in \{-1, -0.5, -0.1\}$. The true stability region is indicated by the area below the dotted line.

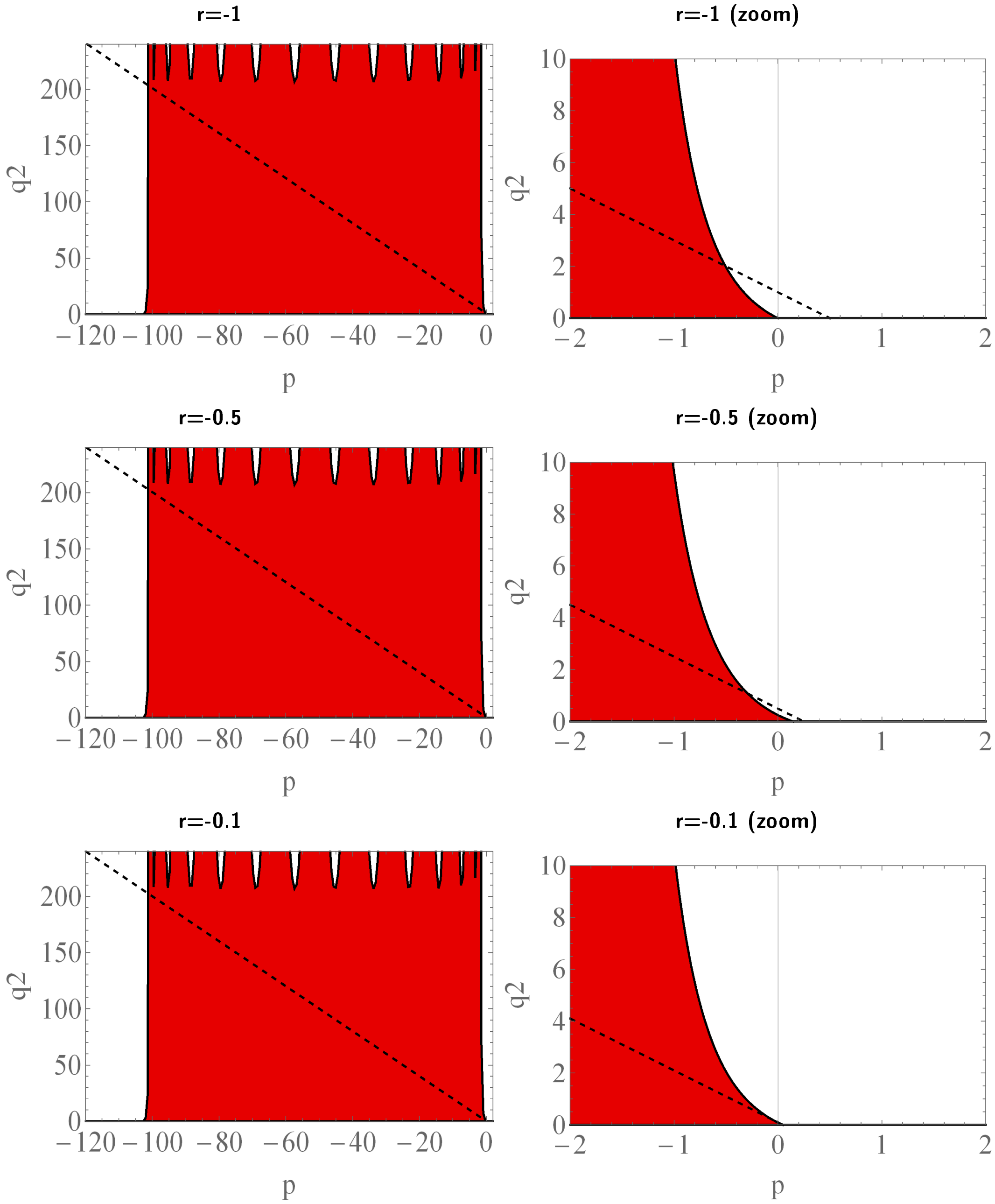

Figure 4.6: Stability regions of the S-ROCK1-JD method with $s = 13$ stages and for different values of $r \in \{-1, -0.5, -0.1\}$. The true stability region is indicated by the area below the dotted line.

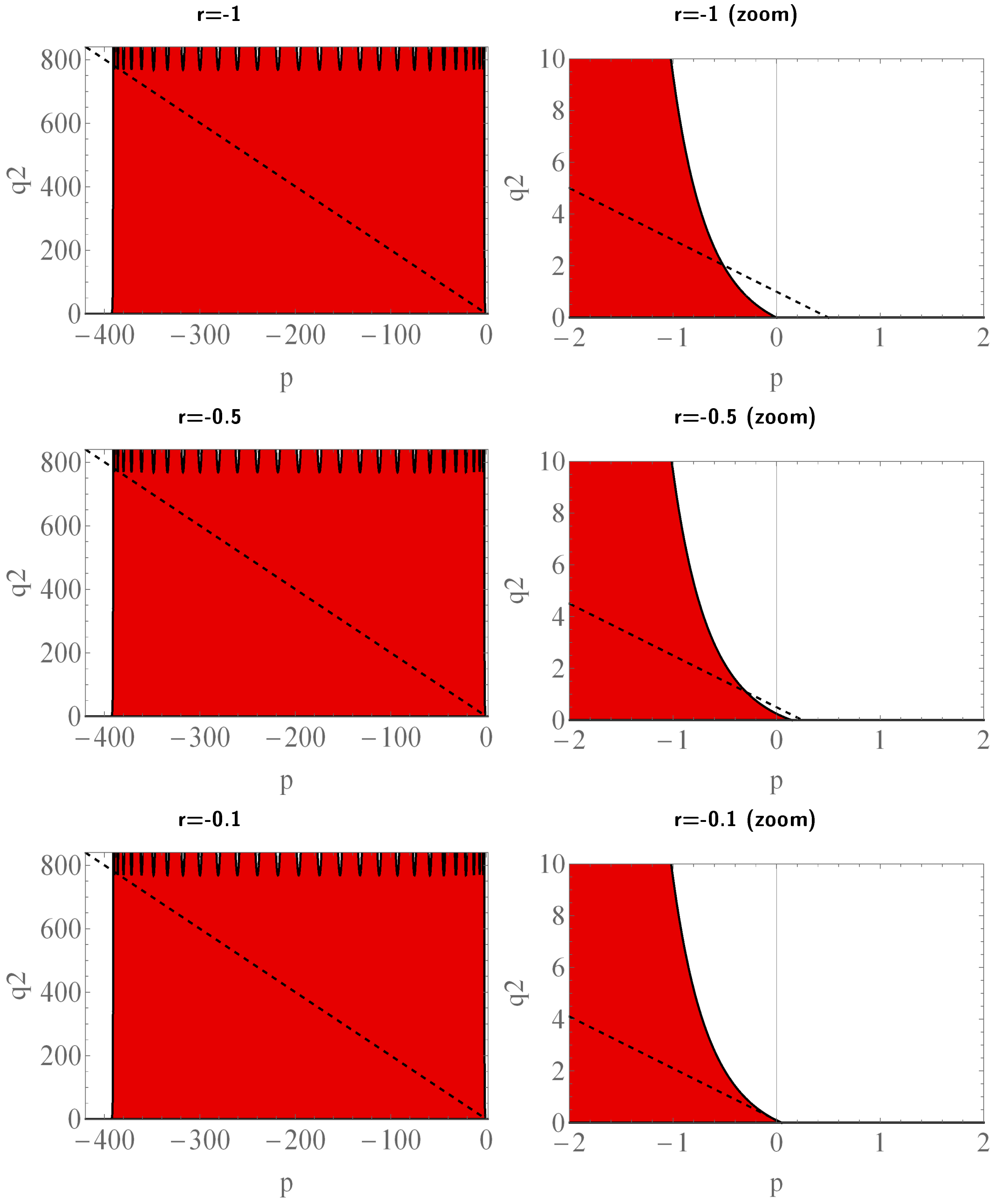

Figure 4.7: Stability regions of the S-ROCK1-JD method with $s = 28$ stages and for different values of $r \in \{-1, -0.5, -0.1\}$. The true stability region is indicated by the area below the dotted line.

Proposition 4.4.3. *Consider the test problem* (4.8). *Furthermore, consider the S-ROCK1-JD method with s stages. Let* μ,σ,γ,ρ *be parameters that lie in the true stability domain* (4.11). *If* $\gamma\in]-2,0[$ *and* $\sigma^2=(-2+\delta)\left(\mu+\frac{\rho\gamma(\gamma+2)}{2}\right)$ *for some* $\delta\in]0,2[$*, then one has to choose a time stepsize* Δt *such that*

$$\frac{T_s^2\left(\omega_0\right)}{T_s^2\left(\omega_0+\mu\Delta t\omega_1\right)}-1+(2-\delta)\,\mu\Delta t-\frac{\delta}{2}c_1\Delta t-c_2\Delta t^2>0$$

with $c_1=\rho\gamma\left(\gamma+2\right)$ *and* $c_2=\rho^2\gamma^2$ *to guarantee that the chosen values of* μ,σ,γ,ρ *lie in the stability domain of the numerical scheme.*

Proof. Since $\delta\in]0,2[$ and $\sigma^2=(-2+\delta)\left(\mu+\frac{\rho\gamma(\gamma+2)}{2}\right)$ we have that $\sigma^2<-2\left(\mu+\frac{\rho\gamma(\gamma+2)}{2}\right)$, and thus, the parameters lie in the true stability domain. To guarantee that the parameters also lie in the stability domain of the numerical scheme, by (4.14), we require that

$$\sigma^2\Delta t<\frac{T_s^2\left(\omega_0\right)}{T_s^2\left(\omega_0+\mu\Delta t\omega_1\right)}-1-\rho\gamma\left(\gamma+2\right)\Delta t-\rho^2\gamma^2\Delta t^2.$$

Rearranging the terms and using the choice of σ, we obtain the desired condition. □

In the case where r is negative, necessarily we have $\gamma\in]-2,0[$ since the other terms of r are positive. Hence, we can apply Proposition 4.4.3. In fact, suppose that we use as numerical integrator the S-ROCK1-JD method with s stages and suppose that the parameters μ,σ,γ,ρ are given and lie in the true stability domain. If $\gamma\in]-2,0[$, then we can pick the largest Δt such that the condition of Proposition 4.4.3 holds.

4.5 Numerical Experiments

In this section we analyze numerically the strong convergence of the S-ROCK1-JD method for three different models. Further, we compare the S-ROCK1-JD method to the Euler-Maruyama method for jump-diffusions for a one-dimensional linear SDE and a two-dimensional nonlinear SDE, both times driven by a jump-diffusion process. In the following we focus solely on the S-ROCK1-JD method, the numerical study of the PIROCK-JD method can be carried out in a similar way.

4.5.1 Numerical Study of the Strong Convergence

Here, we distinguish between two different time grids. First there is the regular time grid consisting of a uniform time stepsize and then we consider a jump-adapted time grid, where we add to the the time steps obtained by a fixed time stepsize the times of the jumps. Let $T=10$ be the time endpoint and let the number of uniform time steps be given by $L\in\left\{2^5,2^6,2^7,2^8,2^9\right\}$. Then the uniform time stepsize of the regular S-ROCK1-JD method is given by $\Delta t=\frac{T}{L}$. For the jump-adapted S-ROCK1-JD method, the jump times are added to the time grid obtained by Δt. To determine the strong convergence of the two S-ROCK1-JD schemes, we estimate the strong error

$$e_{\Delta t}^{strong} := \sqrt{\mathbb{E}\left[|X(T) - Y_L|^2\right]},$$

at the time endpoint T, where $X(T)$ represents the exact solution at T and Y_L the approximate solution at T obtained with a regular or a jump-adapted grid based on $\Delta t = \frac{T}{L}$. To approximate the expectation a sample average over 10000 simulations is taken. In the subsequent sections we present the numerical results for the Merton model, the Kou model and a model with constants marks (jumps with a deterministic jump size).

Strong Convergence for the Merton Model

The first model that we consider is the so-called Merton model, which was first introduced by Robert C. Merton in [80] in the mid-seventies. It was the first model in the financial sector that includes jumps. The Merton model is characterized by the stochastic differential equation

$$\begin{cases} \mathrm{d}X(t) & = & \mu X(t-)\mathrm{d}t + \sigma X(t-)\mathrm{d}W(t) + X(t-)\mathrm{d}J(t), \; 0 \le t \le T, \\ \mathrm{d}X(0) & = & 1, \end{cases} \tag{4.15}$$

where $J(t) = \sum_{i=1}^{N(t)} (V_i - 1)$ with $(N(t))_{t\in[0,T]}$ a Poisson process with intensity ρ and $V_1, V_2, \ldots, V_{N(t)}$ independent and identically distributed log-normal variables, i.e.

$$\log(V_i) \overset{\text{iid}}{\sim} \mathcal{N}\left(\eta, v^2\right)$$

for all i and with $\eta = 0$ and $v = 0.01$. The exact solution of the Merton model has been computed in Proposition 2.1.14 and is given by

$$X(T) = \exp\left\{\left(\mu - \frac{\sigma^2}{2}\right) T + \sigma W(T)\right\} \prod_{i=1}^{N(t)} V_i. \tag{4.16}$$

Figure 4.8 shows the results for four different stage numbers ($s \in \{3, 10, 50, 100\}$). The drift parameter is fixed to $\mu = -0.05$ and the diffusion parameter to $\sigma = 0.1$. As jump intensity we have chosen $\rho = 1$. One can observe that for any of the four values of s the curves of the regular as well as the jump-adapted S-ROCK1-JD method have a slope of roughly 1/2 underlining the theoretical finding that the S-ROCK1-JD methods are of order of strong convergence 1/2. Both numerical schemes seem to perform similarly and the corresponding curves nearly coincide.

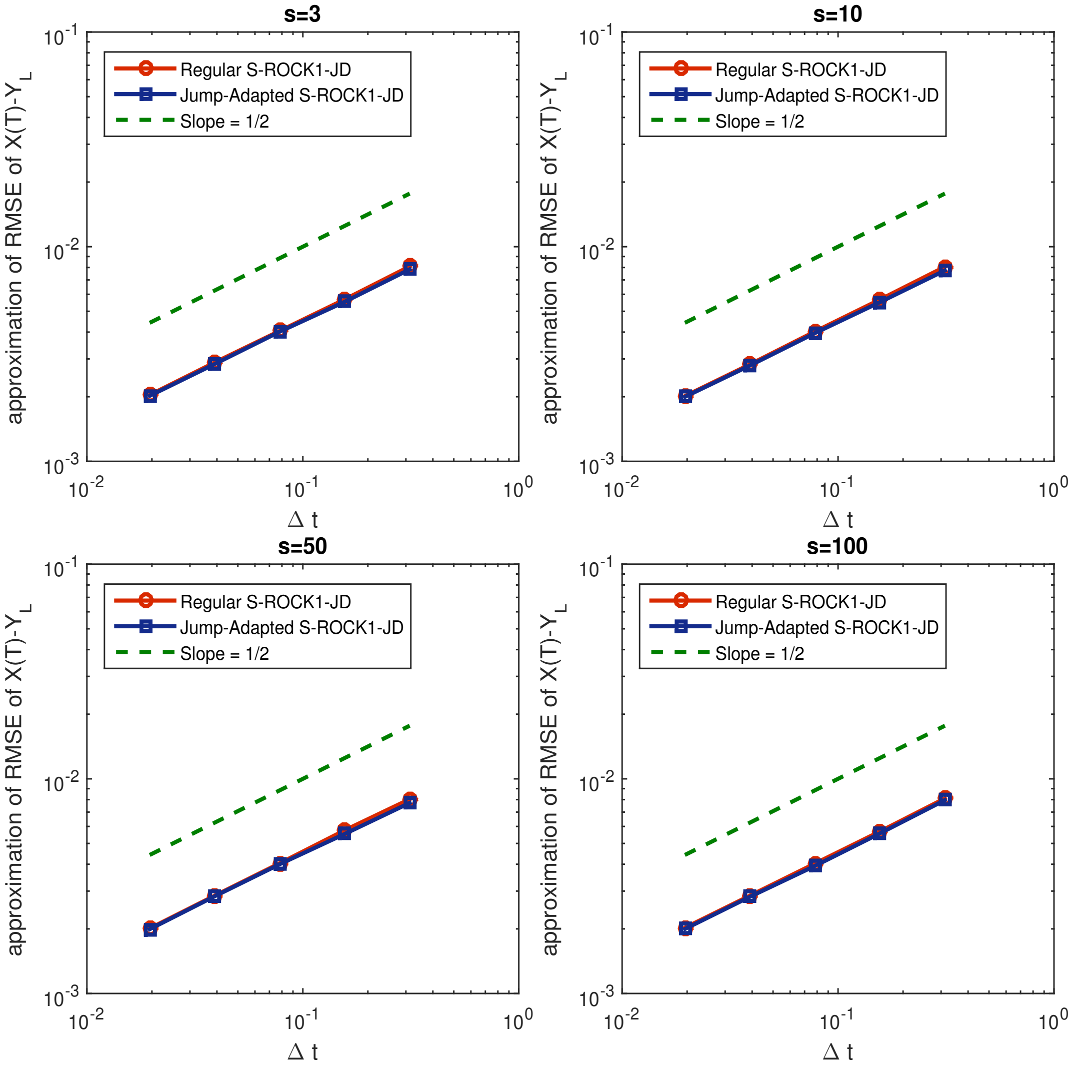

Figure 4.8: Strong convergence of the Merton model using different stage numbers s.

Strong Convergence for the Kou Model

The next model that we consider is the Kou model, which was proposed in 2002 by Steven G. Kou (see [67]). The Kou model is also characterized by the stochastic differential equation (4.15) of the Merton model, but here the jump sizes are not log-normally distributed, they are double exponentially distributed. In particular for the variables V_i it holds that

$$\log\left(V_i\right) \overset{\text{iid}}{\sim} \mathscr{K}\left(\eta_1, \eta_2, p\right)$$

with $\mathscr{K}$ an asymmetric exponential distribution defined by its density function

$$f_{\mathscr{K}}\left(x\right) := p\eta_1 e^{-\eta_1 x} 1_{\{x\geq 0\}} + \left(1-p\right)\eta_2 e^{\eta_2 x} 1_{\{x<0\}},$$

where x is any real number, $\eta_1 > 1$, $\eta_2 > 0$ (indicating the behaviour of the tails of the positive and negative jumps, respectively) and $p \in [0,1]$ (the probability of an upward jump). In the following we pick $\eta_1 = 50$, $\eta_2 = 25$ and $p = 0.3$. The other model parameters are chosen as in the previous section. The exact solution of the Kou model is again given by (4.16).

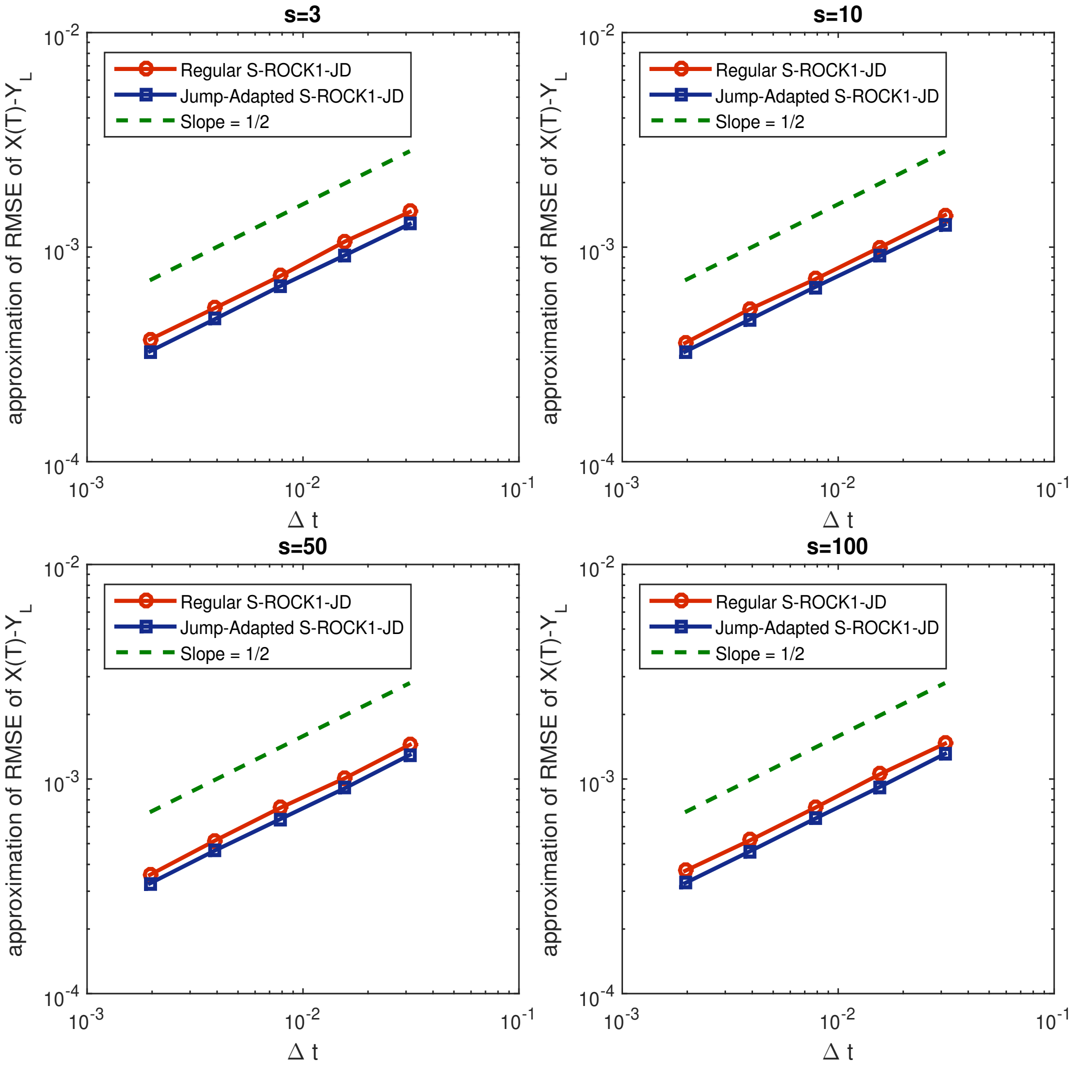

Figure 4.9: Strong convergence of the Kou model using different stage numbers s.

Figure 4.9 gives the numerical results of the study of the strong convergence of S-ROCK1-JD applied to the Kou model. As for the Merton model one observes that the curves of both numerical methods (regular and jump-adapted) are approximately of slope 1/2. Furthermore one can see that for this model, the jump-adapted S-ROCK1-JD method performs slightly better than the regular S-ROCK1-JD method.

Strong Convergence for Mark independent Jumps

The last model that we consider for the numerical study of the strong convergence is the model given by constant marks, i.e. the jump sizes are deterministic. This model can also be expressed by the stochastic differential equation (4.15), where we have

$$J(t) = \sum_{i=1}^{N(t)} V_i - 1 = \gamma^{N(t)},$$

i.e. $V_i = \gamma + 1$. Here we choose $\gamma = 0.1$ and the other parameters are chosen as in Section 4.5.1. The exact solution of this model is given by

$$X(T) = \exp\left\{\left(\mu - \frac{\sigma^2}{2}\right) T + \sigma W(T)\right\} (1+\gamma)^{N(T)}.$$

In Figure 4.10 the numerical results for different stage numbers of S-ROCK1-JD are illustrated. The numerical results suggest that the S-ROCK1-JD method is of strong order 1/2, which is consistent with the theoretical results in Section 4.3. One can observe that for this model the S-ROCK1-JD method based on a jump-adapted time grid is more accurate than the one based on a regular grid.

4.5.2 Comparison S-ROCK1-JD and Euler-Maruyama for Jump-Diffusions

In this section we compare the S-ROCK1-JD method to the Euler-Maruyama method for jump-diffusions (see Definition 4.2.3). In [59] it has been shown, that the Euler-Maruyama-JD method applied to the test problem (4.8) is mean square stable if for the time stepsize it holds that

$$\Delta t < \frac{|2\mu + \sigma^2 + \rho\gamma(\gamma+2)|}{(\mu + \rho\mu)^2}.$$

Observe that we can rewrite this as

$$\rho_{EM} \Delta t < 1, \tag{4.17}$$

with

$$\rho_{EM} := \frac{(\mu + \rho\mu)^2}{|2\mu + \sigma^2 + \rho\gamma(\gamma+2)|}, \tag{4.18}$$

which represents the stiffness parameter of the Euler-Maruyama-JD method. Note that by setting the jump-related terms to zero we recover the stiffness parameter of the Euler-Maruyama method for diffusions (see (3.7)). Choosing $(\mu, \sigma, \gamma, \rho) \in \mathbb{R}^4$ such that the test problem (4.8) is mean square stable, the stepsize Δt of the Euler-Maruyama-JD method has to be chosen such that

$$\Delta t < \frac{1}{\rho_{EM}}$$

to guarantee that the numerical integrator is also mean square stable.

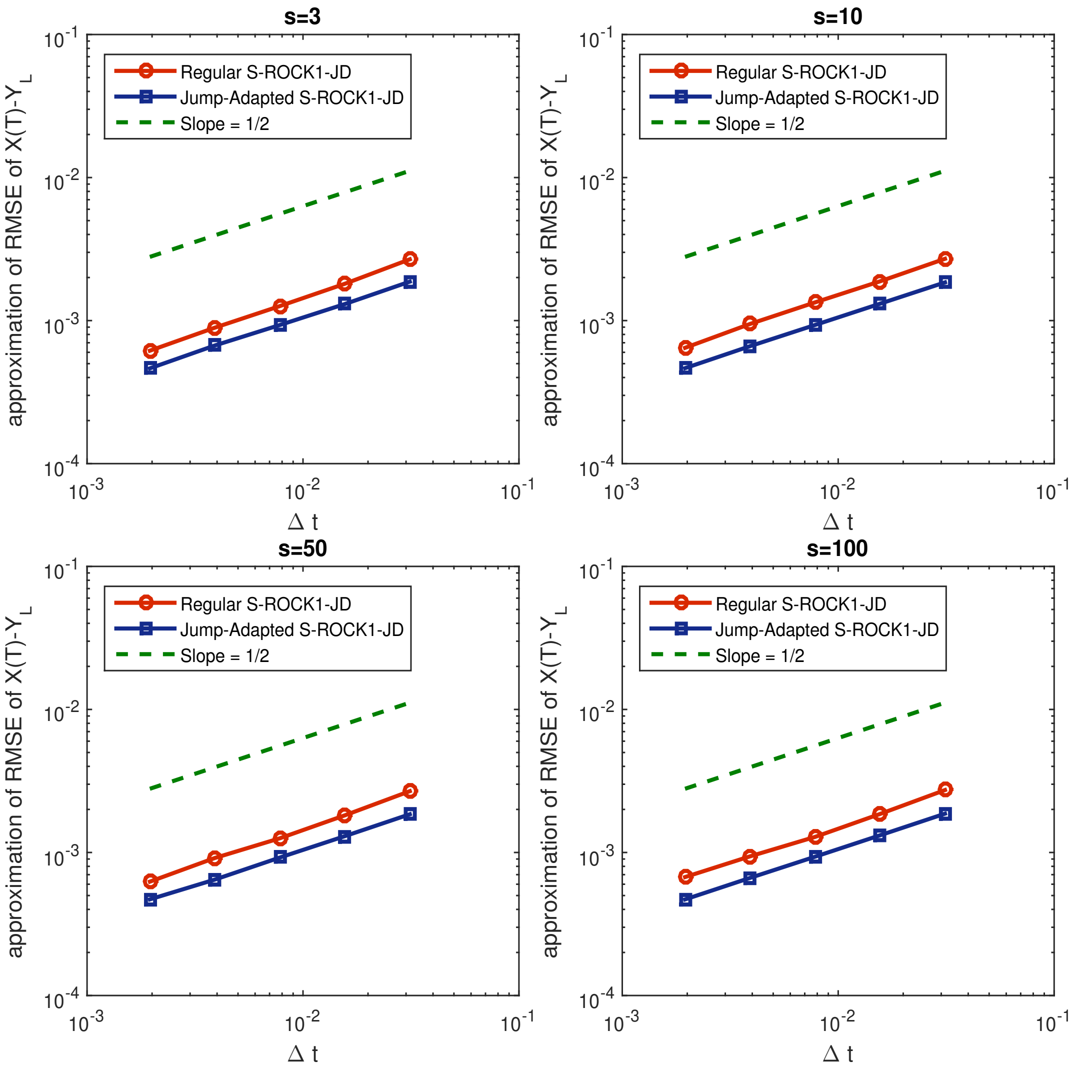

Figure 4.10: Strong convergence of the model with constant marks using different stage numbers s.

Similarly, one can establish for S-ROCK methods with s stages, and in particular for the S-ROCK1-JD method, a stability criterion

$$\frac{\Delta t \rho_{SR}}{c_{SR} s^2} < 1 \tag{4.19}$$

with c_{SR} some constant that can numerically be estimated and ρ_{SR} the stiffness parameter associated to the S-ROCK method (see [12, 15]). For the test problem (4.8) the stiffness parameter is given by

$$\rho_{SR} := |\mu|, \tag{4.20}$$

where we consider the parameters $(\mu, \sigma, \gamma, \rho) \in \mathbb{R}^4$. Note that the stiffness parameter of the S-ROCK methods is independent of the diffusion term σ, which is due to the fact that with an adequate damping and the right choice of the stage number s a whole portion of the true stability region of the test problem is covered by the stability region of the numerical integrator (see [12, 15]).

In the following we use Monte Carlo to approximate the second moment of the corresponding SDE with jumps at the time endpoint. To measure the performance of the different approaches, we count the number of function evaluations required to reach a certain precision, which we measure by the root mean square accuracy. In Section 2.3 we have shown that to reach a mean square accuracy of $\mathcal{O}\left(\varepsilon^2\right)$ a computational cost (given as the number of function evaluations) of $\mathcal{O}\left(\varepsilon^{-3}\right)$ is required.

Comparison based on a Linear SDE with Jumps

Here, we consider the one-dimensional linear stochastic differential equation with jumps characterized by

$$\left\{\begin{array}{lcl} \mathrm{d}X(t) & = & \mu X(t-)\mathrm{d}t + \sigma X(t-)\mathrm{d}W_t + \gamma X(t-)\mathrm{d}J_t, \ 0 < t < T, \\ \\ X(0) & = & X_0, \end{array}\right.$$

where $J(t)$ is the jump component of the Merton model (see (4.15)). In the following we consider the parameters $T = 1$, $X_0 = 1$, $\gamma = 1$, $\rho = 1$, $\eta = 0$ and $\nu = 0.01$. In addition, as time stepsizes we take into account

$$\Delta t \in \left\{2^{-1}, 2^{-2}, \ldots, 2^{-13}\right\}.$$

For the drift term we consider different values $\mu \in \{-10, -100, -1000\}$ and we pick as diffusion coefficient $\sigma = \sqrt{|\mu|}$. Note that all sets of parameters $\left(\mu, \sigma, \gamma, \rho\right)$ lie in the true stability domain.

In Figure 4.11 we illustrate the number of function evaluations of the two approaches using as numerical integrator the S-ROCK1-JD method and the Euler-Maruyama-JD method, respectively, against the precision measured by the root mean square stability. One observes that as the stiffness of the SDE increases, i.e. as $|\mu|$ increases, the S-ROCK1-JD approach clearly prevails. This is due to the time stepsize restriction that suffers the Euler-Maruyama-JD method through the stability constraint (4.17). As the stiffness of the problems increases, the stepsize restriction becomes more severe and the Monte Carlo approach can only be applied if the time stepsize is sufficiently small. In contrast, the S-ROCK1-JD does not suffer from any restriction and by adjusting the stage number any time stepsize can be used (see (4.19)). Note that in the case where both numerical methods can be applied, the Euler-Maruyama approach is cheaper, since the S-ROCK method uses a certain number of stages s (with $s \geq 2$). However, usually one defines the S-ROCK methods by stating that in the case where the stage number $s < 2$ the S-ROCK method coincides with the Euler-Maruyama scheme.

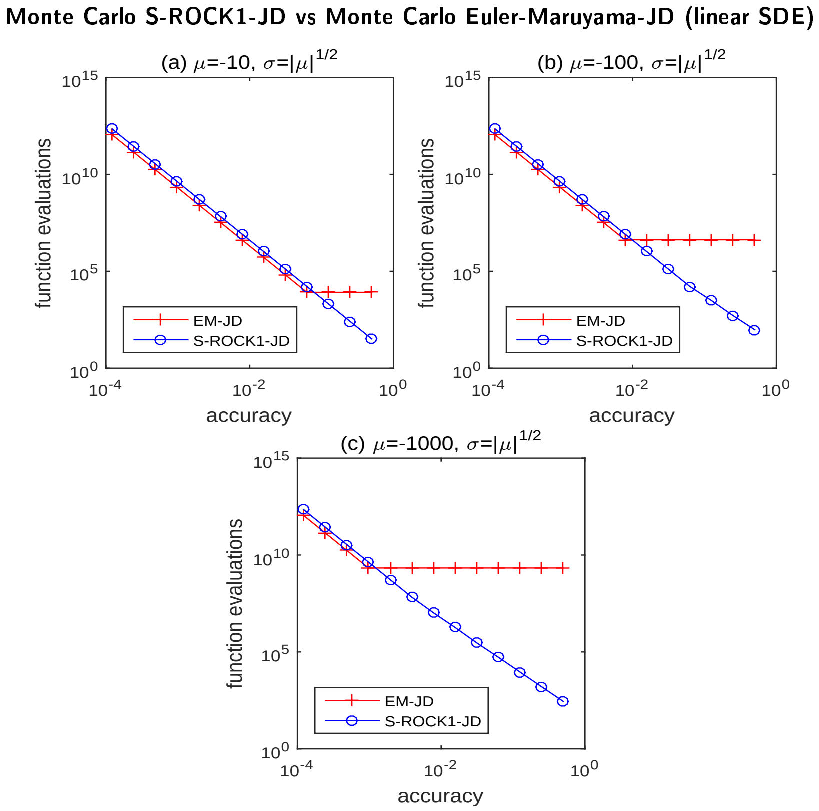

Figure 4.11: One-dimensional linear SDE with jumps: Number of function evaluations against the time stepsize Δt comparing the Monte Carlo method using S-ROCK1-JD and Euler-Maruyama-JD.

Comparison based on a Nonlinear SDE with Jumps

Here, we consider a two-dimensional noncommutative stiff stochastic differential equations with jumps, which is inspired by the one-dimensional population model (see [84]) and to which we have added a jump component. The stochastic model is defined by

$$\mathrm{d}\begin{pmatrix} X_1(t) \\ X_2(t) \end{pmatrix} = \begin{pmatrix} \alpha a_2(t-) - \mu_1 b_1(t-) \\ -\mu_2 b_2(t-) \end{pmatrix} \mathrm{d}t + \begin{pmatrix} -\sigma_1 b_1(t-) & \sigma_2 a_1(t-) \\ -\sigma_2 b_2(t-) & 0 \end{pmatrix} \begin{pmatrix} \mathrm{d}W_t^1 \\ \mathrm{d}W_t^2 \end{pmatrix} + \begin{pmatrix} X_1(t-) \\ X_2(t-) \end{pmatrix} \mathrm{d}J_t$$

for $0 \le t \le T$, $a_i(t) = X_i(t) - 1$ and $b_i(t) = X_i(t)(1 - X_i(t))$ with $i = 1, 2$. As initial condition we take $(X_1(0), X_2(0)) = (0.98, 0.98)$. The processes $\left(W^1(t)\right)_{t\in[0,T]}$ and $\left(W^2(t)\right)_{t\in[0,T]}$ are independent one-dimensional standard Brownian motions. The process $(J(t))_{t\in[0,T]}$ is independent from the Brownian motions and corresponds to the jump term of the Merton model (see (4.15)). We carry out similar simulations to the previous section. We consider as parameters $T = 1$, $\alpha = 2$, $\gamma = 1$, $\rho = 1$, $\eta = 0$ and $\nu = 0.01$. Furthermore, we take $\mu_2 = -2$ and $\sigma_2 = 0.25$ and we look at the influence of the parameters $\mu_1 \in \{-10, -100, -1000\}$ and $\sigma_1 = \sqrt{|\mu_1|}$. As time stepsize we consider

$$\Delta t \in \left\{2^{-1}, 2^{-2}, \ldots, 2^{-13}\right\}.$$

Observe that the parameter sets $(\mu_1, \sigma_1, \gamma, \rho)$ and $(\mu_2, \sigma_2, \gamma, \rho)$ both lie in the true stability domain with the former one regulating the stiffness of the stochastic problem.

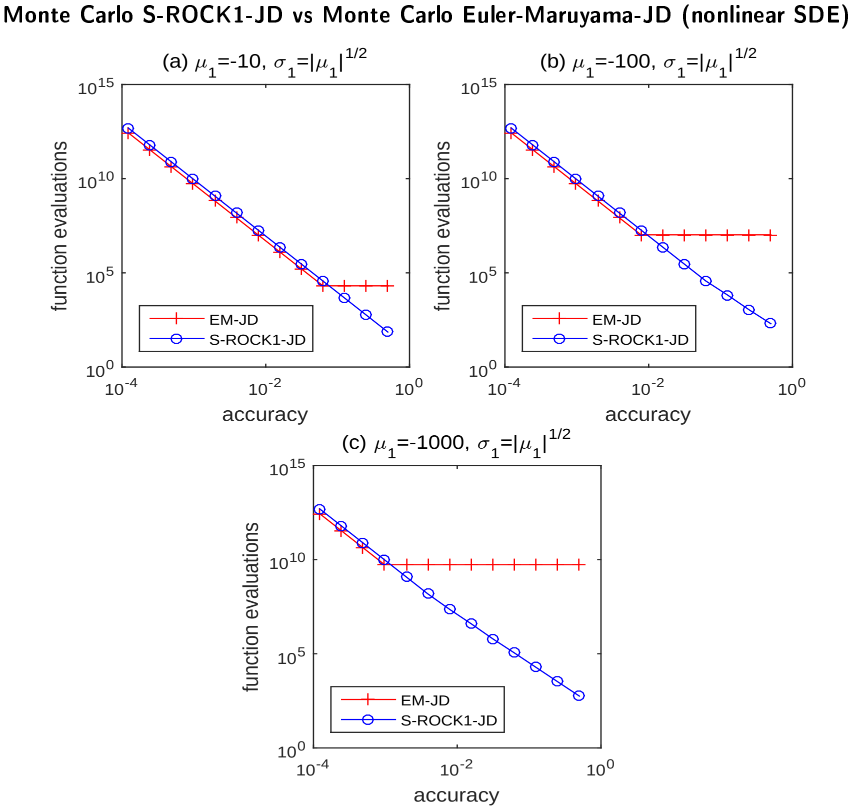

Figure 4.12: Two-dimensional nonlinear SDE with jumps: Number of function evaluations against the time stepsize Δt comparing the Monte Carlo method using S-ROCK1-JD and Euler-Maruyama-JD.

Figure 4.12 shows the result for the nonlinear SDE with jumps. As in the linear case the S-ROCK1-JD approach performs generally better than the one that uses the Euler-Maruyama method for jump-diffusions. In fact, for a large stiffness the Euler-Maruyama-JD scheme faces severe time stepsize restrictions, which can be omitted using the S-ROCK1-JD integrator. We note that the stability has been verified by looking at the second moment of the stochastic process at the time endpoint T.

4.6 Conclusion

In this chapter we have extended two S-ROCK methods so that they account for jumps in a stochastic problem driven by jump-diffusions. The S-ROCK1-JD method and the PIROCK-JD method have both been studied in detail. We have proven that the new numerical methods are of strong order of convergence 1/2. This has been numerically verified by carrying out simulations on the Merton and the Kou model and on a model based on mark independent jumps. The mean square stability domains of the S-ROCK1-JD and the PIROCK-JD schemes have been characterized. Numerical experiments have been realized to study the stability of the numerical schemes for a linear SDE with jumps and a nonlinear SDE with jumps.

5 Multilevel Monte Carlo Method for Stochastic Differential Equations driven by Jump-Diffusion Processes

There are many applications in economy, biology, chemistry, physics and so on that based on jump-diffusion models (see Section 2.1). Often one is interested in the expectation of some functional based on underlying the jump-diffusion process. In this chapter we extend the multilevel Monte Carlo method to stochastic differential equations driven by jump-diffusion processes, which allows to speed up the simulation procedure in the situations mentioned above. We show that the MLMC method based on jump-diffusions for a reasonable jump intensity reduces the computational complexity of the algorithm significantly compared to the standard Monte Carlo approach for a given mean square accuracy. Numerical experiments are carried out to underline our theoretical findings. In the numerical part we also study the antithetic variates and the control variates, two variance reduction techniques, and we compare them to the MLMC approach. In the last part of this chapter we combine the results of MLMC for jump-diffusion and the stabilized MLMC approach from Chapter 3 to obtain a stabilized multilevel Monte Carlo method for stiff stochastic differential equations driven by jump-diffusion processes.

The following is mainly taken from the scientific paper [8].

5.1 Introduction

Monte Carlo methods are commonly applied when we are interested in computing expectations of functionals depending on a stochastic process. Here, we assume that the stochastic process is given by a stochastic differential equation (SDE) incorporating a jump term with a finite rate intensity. In a Monte Carlo (MC) approach, sample paths of the solution of an SDE are computed by a numerical integrator and the expected value of the given functional is approximated by the average over those samples. This procedure represents computing a statistical estimator of the desired quantity, and bias and statistical errors are introduced due to the numerical method and the approximation of the expectation, respectively. The bound on the statistical error for the MC method involves the inverse of the square root of the number of samples, as well as the variance of the process. The nature of this bound can not be changed, however many strategies to reduce the variance of the estimators and hence the complexity of the procedure have been proposed in the past few years. Among them, we mention variance reduction techniques such as estimators based on control variates or antithetic variates (see e.g. [49]).

A recent approach, originating with Heinrich [56], proposed by Kebaier [65] as a statistical Romberg method with two levels and extend by Giles [46] to the so-called multilevel Monte Carlo (MLMC) method, allows to significantly speed up the classical MC method thanks to hierarchical sampling. By applying the Monte Carlo method for several nested time stepsizes and choosing the right balance between the stepsizes and the number of simulations at each level, it is possible to reduce the computational complexity of the Monte Carlo method for a given mean square accuracy. More precisely, to compute the expectation of functionals with an accuracy (here, the square root of the mean square error is chosen as a measure of the accuracy) of $\mathscr{O}(\varepsilon)$, the computational cost of $\mathscr{O}(\varepsilon^{-3})$ for the MC method is reduced to $\mathscr{O}(\varepsilon^{-2}(\log\varepsilon)^2)$ for the MLMC method.

In this chapter, we study the MLMC method for jump-diffusion processes. This class of processes becomes important for example in financial modeling, when stock prices based on diffusion processes should be modelled by taking into account sudden, unforeseeable events [18, 98]. Then, models based on jump-diffusion processes are required for more realistic modeling [80]. Furthermore, some physical processes cannot be modeled by continuous processes and need to take into account single events. We mention here the modeling of biological network dynamics [103] or chemical kinetics [48].

To the best of our knowledge, the MLMC method for jump-diffusions with finite rate activity has first been discussed in [23] and [105], whereas the case of Lévy processes with infinite rate activity has been studied in [35, 79]. The purpose of this chapter is to give a rigorous proof of the complexity theorem of the MLMC for jump-diffusion problems, in particular for the jump-adapted version of the Euler-Maruyama method. This appears not to have been treated in the literature. We show that the above reduction factor in the computational cost obtained by replacing MC methods with MLMC methods remains valid for the estimation of the expectation of functionals depending on jump-diffusion processes. We test the MLMC for jump-diffusion problems on various examples and show significant speed-up compared to standard MC computations. The new approach is also compared to estimators based on variance reduction techniques. The results show that for sufficiently small errors, the MLMC method always outperforms these other techniques for the models considered.

This chapter is organized as follows. In Section 5.2 we discuss jump-diffusion processes and numerical methods used to approximate such processes. In Section 5.3 we construct the MLMC method for jump-diffusions and we give an extended version of the complexity theorem presented in [46]. Numerical experiments to illustrate our theoretical findings are presented in Section 5.4. Finally, we create a stabilized multilevel Monte Carlo method for jump-diffusion processes by combining the results of Section 5.3 and Chapter 3.

5.2 Preliminaries

Throughout this chapter let $(\Omega, \mathscr{F}, \mathbb{P}, \mathscr{F}_t)$ be a filtered probability space where the filtration $(\mathscr{F}_t)_{t\geq 0}$ satisfies the usual conditions (see e.g. [77] or [95]). We use in this chapter the terms *computational cost* and *computational complexity* synonymously to represent overall for an algorithm the number of time steps of a numerical discretization, the number of samples generated and the number of function evaluations. These terms measure the complexity of algorithms and they will be used when we compare the performance of algorithms.

Here we consider stochastic processes $(S(t))_{t\in[0,T]}$ on the bounded interval $[0,T]$ described by the stochastic differential equation incorporating diffusion and jump terms

$$\begin{cases} \mathrm{d}S(t) &= a(t,S(t-))\mathrm{d}t + b(t,S(t-))\mathrm{d}W(t) + c(t,S(t-))\mathrm{d}J(t), \quad 0 \leq t \leq T, \\ S(0) &= S_0, \end{cases} \tag{5.1}$$

with $S(t-)$ denoting $S(t-) = \lim_{s\nearrow t} S(s)$. Here $(W(t))_{t\in[0,T]}$ is an $m-$dimensional Wiener process and $(J(t))_{t\in[0,T]}$ an $r-$dimensional compound Poisson process, $J(t) = (J^1(t), J^2(t), \ldots, J^r(t))$. Each component $J^k(t)$ is defined by

$$J^k(t) = \sum_{i=1}^{N^k(t)} (V_i^k - 1),$$

where $N^k(t)$ is a Poisson process with intensity λ_k and where the jump sizes are characterized by V_i^k. Further, the functions $a: [0,T] \times \mathbb{R}^d \to \mathbb{R}^d$, $b: [0,T] \times \mathbb{R}^d \to \mathbb{R}^{d\times m}$, and $c: [0,T] \times \mathbb{R}^d \to \mathbb{R}^{d\times r}$ represent the drift, the diffusion and the jump coefficient, respectively. We assume standard Lipschitz and linear growth conditions on a, b and c to ensure the existence of a strong solution of the SDE (5.1).

Remark 5.2.1. *Note that if γ_i^k corresponds to the i-th jump time of $J^k(t)$, then the jump size of this jump is given by*

$$S\left(\gamma_i^k\right) - S\left(\gamma_i^k-\right) = c^k\left(\gamma_i^k, S\left(\gamma_i^k-\right)\right)\left(V_i^k - 1\right).$$

If we now assume $c^k\left(\gamma_i^k, S\left(\gamma_i^k-\right)\right) = S\left(\gamma_i^k-\right)$, as this is the case for the models we consider in the following, then

$$S\left(\gamma_i^k\right) - S\left(\gamma_i^k-\right) = S\left(\gamma_i^k-\right)\left(V_i^k - 1\right),$$

and consequently

$$S\left(\gamma_i^k\right) = S\left(\gamma_i^k-\right)V_i^k,$$

which means V_i^k corresponds to the ratio of the stochastic process before and after the i-th jump of $J^k(t)$. Thus the choice of $V_i^k - 1$ above (see also e.g. [49]).

5.2.1 Numerical Schemes

In this section we recall two numerical schemes, the regular and the jump-adapted Euler method, to approximate solutions of the SDE (5.1). The former is iterated using a uniform stepsize, whereas the latter includes the jump times into the otherwise uniform time grid. we discuss the strong and the weak convergence of these numerical schemes. For details and additional references on numerical methods for jump-diffusion problems see [26, 25, 44, 58, 71, 73, 83, 28].

Regular Euler Method

The first numerical method that we consider is a natural extension of the Euler-Maruyama method for SDEs driven by diffusion processes. The regular Euler method is defined by a uniform stepsize h and the grid is given by

$$\tau_h^{T/h} = \{\tau_0, \tau_1, \ldots, \tau_{T/h}\}, \tag{5.2}$$

where $\tau_j = jh \; \forall j \in \{0, 1, \ldots, T/h\}$. Now, let S_j be an approximation of the stochastic process $S(t)$ at $t = \tau_j$, i.e. $S_j \approx S(\tau_j)$.

Definition 5.2.2 (Regular Euler method)**.** *Consider the time grid* (5.2)*. The regular Euler scheme is defined by*

$$S_j = S_{j-1} + a(\tau_{j-1}, S_{j-1})(\tau_j - \tau_{j-1}) + b(\tau_{j-1}, S_{j-1})\Delta W_j + c(\tau_{j-1}, S_{j-1})\Delta J_j \tag{5.3}$$

with $j \in \{1, 2, \ldots, T/h\}$. By the initial condition, we have $S_0 = S(0)$ and the increments of the m-dimensional Wiener process and the r-dimensional compound Poisson process are given by $\Delta W_j = W(\tau_j) - W(\tau_{j-1})$ and $\Delta J_j = J(\tau_j) - J(\tau_{j-1})$, respectively.

Note that, by setting the jump coefficient c to zero, we get the Euler-Maruyama method, which can be used for numerical approximations of SDEs driven by diffusions (see e.g. [57]).

Jump-Adapted Euler Method

The second numerical method that we consider, is the jump-adapted Euler scheme. Unlike the previous scheme, this method does not have a uniform stepsize if there is at least one jump. In the case of the jump-adapted Euler scheme, the jump times have to be added to the regular grid with uniform stepsize h defined in (5.2). Recall that we consider a r-dimensional jump process, and thus, the number of jumps in the time interval $[0, T]$ are specified by the Poisson variables $N^1(T), N^2(T), \ldots, N^r(T)$ with intensities $\lambda_1 T, \lambda_2 T, \ldots, \lambda_r T$. Hence, the grid for the jump-adapted Euler scheme is given by

$$\{\tau_0, \tau_1, \ldots, \tau_{T/h}\} \cup \{\gamma_1^1, \gamma_2^1, \ldots, \gamma_{N^1(T)}^1\} \cup \cdots \cup \{\gamma_1^r, \gamma_2^r, \ldots, \gamma_{N^r(T)}^r\}, \tag{5.4}$$

where $\gamma_1^k, \gamma_2^k, \dots, \gamma_{N^k(T)}^k$ are the jump times in the interval $[0, T]$ of the jump component $J^k(t)$. Thus, there are in total $T/h + \sum_{i=1}^r N^i(T)$ time steps. Adding the jump times to the regular grid and rearranging the grid such that the j-th entry τ_j corresponds to the j-th time step.

Definition 5.2.3 (Jump-Adapted Euler method)**.** *Consider the time grid* (5.4)*. The jump-adapted Euler scheme for* (5.1) *is defined by*

$$S_j = S_{j-1} + a\left(\tau_{j-1}, S_{j-1}\right)\left(\tau_j - \tau_{j-1}\right) + b\left(\tau_{j-1}, S_{j-1}\right)\Delta W_j + c\left(\tau_{j-1}, S_{j-1}\right)\Delta J_j \tag{5.5}$$

with $j \in \left\{1, 2, \dots, T/h + \sum_{i=1}^r N^i(T)\right\}$*. By the initial condition, we have* $S_0 = S(0)$ *and the increments of the* m*-dimensional Wiener process and the* r*-dimensional compound Poisson process are given by* $\Delta W_j = W\left(\tau_j\right) - W\left(\tau_{j-1}\right)$ *and* $\Delta J_j = J\left(\tau_j\right) - J\left(\tau_{j-1}\right)$*, respectively.*

Note that the definition of the jump-adapted Euler scheme is identical to the regular method in (5.3), but the time grid and thus the range of the iteration parameter j changes.

Remark 5.2.4. *To ease the notation we describe in the sequel both numerical schemes by (5.3) with* $j \in \{1, 2, \dots, G\}$*, where* $G = T/h + \sum_{i=1}^r N^i(T)$ *in the jump-adapted case and* $G = T/h$ *in the regular case.*

Convergence of Numerical Methods

In this chapter we consider two types of convergence: the strong convergence and the weak convergence, respectively. The are properly defined in Section 2.2.1 and we briefly recall them here.

Let $S(t_j)$ be the exact solution of the SDE (5.1) at $t = t_j$ and let $(S_j)_{j\in\mathbb{N}}$ be the approximate solution by a numerical method at the same time point.

1. A numerical method is converging with a strong order of convergence γ_{strong} if

$$\exists C \in \mathbb{R}_+ \text{ such that } \max_{0\le j\le T/h} \left(\mathbb{E}\left[|S_j - S(\tau_j)|^2\right]\right)^{1/2} \le C h^{\gamma_{strong}}, \tag{5.6}$$

   where $\tau_j = jh \in [0, T]$ and h is tending to 0.

2. A numerical method is converging with weak order of convergence γ_{weak} if there exists $C \in \mathbb{R}_+$ such that for all functions p in a certain class (usually p satisfies smoothness and polynomial growth conditions) we have

$$|\mathbb{E}[p(S_j)] - \mathbb{E}[p(S(\tau_j))]| \le C h^{\gamma_{weak}} \tag{5.7}$$

   for any $\tau_j = jh \in [0, T]$ fixed and h tending to 0.

Note that a possible class for the functions p is given by $\mathscr{C}_P^l(\mathbb{R})$. This class contains functions of the type $p : \mathbb{R} \to \mathbb{R}$ that are l times continuously differentiable and that, together with their partial derivatives up to order l, have polynomial growth (see e.g. [66]).

The regular Euler method and the jump-adapted Euler scheme have, under appropriate conditions for the coefficient functions a, b and c, a strong convergence of order 1/2 (see e.g. [26, 25]) .

Under appropriate conditions for the drift function a, the diffusion function b, the jump function c, the initial condition S_0 and the jump intensities $\lambda_1, \lambda_2, \ldots, \lambda_r$, the regular Euler scheme as well as the jump-adapted Euler scheme have a weak convergence of order 1 (see e.g. [26, 25]) .

5.3 Multilevel Monte Carlo Method for Jump-Diffusion Processes

In this section we generalize the multilevel Monte Carlo method to stochastic differential equations driven by jump-diffusions. We present here the construction of the Monte Carlo method and the multilevel Monte Carlo method for jump-diffusions. Furthermore, we state and prove the corresponding complexity theorem. Consider the jump-diffusion process $(S(t))_{t\in[0,T]}$ solution of the SDE (5.1) and a numerical approximation (e.g. the Euler method or the jump-adapted Euler method previously introduced). For a Lipschitz continuous function $f : \mathbb{R}^n \to \mathbb{R}$ we want to estimate the expectation $\mathbb{E}\left[f\left(S(T)\right)\right]$ from many realizations of the numerical solution of (5.1).

For simplicity of the presentation we describe in the sections 5.3.1 and 5.3.2 the Monte Carlo and the multilevel Monte Carlo method for one-dimensional jump processes with intensity λ. We emphasize that our complexity theorem will be presented for the general case of a r-dimensional jump process.

5.3.1 Monte Carlo Method for Jump-Diffusions

We recall the standard Monte Carlo estimator, which is given by

$$\mathbb{E}\left[f(S(T))\right] \approx \frac{1}{N}\sum_{i=1}^{N} f\left(S_G^{(i)}\right) =: \widehat{Y}, \tag{5.8}$$

where we take a sample average over N independent paths with S_G a numerical approximation of $S(t)$ at the time end point T (see Section 5.2.1).

Remark 5.3.1. *Note that h, as defined in Section 5.2.1, is the uniform time stepsize of the regular Euler scheme. For the jump-adapted Euler scheme, the jump times are added to the regular grid. Hence, if there is at least one jump, the grid is not regular any more. However, in that case h corresponds to the maximum stepsize, i.e.*

$$h = \max_{j\in\{1,2,\ldots,G\}} \left(\tau_j - \tau_{j-1}\right).$$

Applying the Monte Carlo method, two types of error arise (see e.g. [81]). Firstly, there is an error due to the numerical approximation of $S(t)$. This error introduces a bias. In fact, we approximate the stochastic process $(S(t))_{t\in[0,T]}$ at $t = T$ using a numerical scheme, so that $S(T) \approx S_G$. Evaluating the function f and taking the expectation on both sides leads to

$$\mathbb{E}\left[f(S(T))\right] \approx \mathbb{E}\left[f(S_G)\right].$$

By linearity of the expectation and the fact that the samples $S_G^{(i)}$ are identically distributed, we have

$$\mathbb{E}\left[\widehat{Y}\right] = \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N} f\left(S_G^{(i)}\right)\right] = \mathbb{E}\left[f(S_G)\right].$$

Hence, we obtain

$$\operatorname{bias}\left(\widehat{Y}\right) = \mathbb{E}\left[\widehat{Y}\right] - \mathbb{E}\left[f(S(T))\right] = \mathbb{E}\left[f(S_G)\right] - \mathbb{E}\left[f(S(T))\right] = \mathcal{O}(h), \tag{5.9}$$

where we have used the first order weak convergence of the numerical schemes (see Section 5.2.1) for the last equality.

Secondly, there is an error arising from the estimation of the expectation. The expectation, which is an integral, is approximated by taking the sample average over N simulations. Due to the strong law of large numbers (see e.g. [42]) and the central limit theorem (see e.g. [31]), this approximation is almost surely unbiased. However, there is a certain variance that depends on the number of simulations N (see e.g. [45]). Indeed, for the variance of the estimator we have

$$\operatorname{Var}\left(\widehat{Y}\right) = \frac{1}{N^2}\sum_{i=1}^{N} \operatorname{Var}\left(f\left(S_G^{(i)}\right)\right) = \frac{\operatorname{Var}\left(f\left(S_G^{(1)}\right)\right)}{N} = \mathcal{O}\left(N^{-1}\right), \tag{5.10}$$

where we have used first the independence of $S_G^{(1)}, S_G^{(2)}, \dots, S_G^{(N)}$ and then the fact that they are identically distributed.

One way to describe the trade-off between the bias and the variance is given by the mean square error, which can be decomposed as

$$\operatorname{MSE}\left(\widehat{Y}\right) = \mathbb{E}\left[\left(\widehat{Y} - \mathbb{E}\left[f(S(T))\right]\right)^2\right] = \operatorname{Var}\left(\widehat{Y}\right) + \left(\operatorname{bias}\left(\widehat{Y}\right)\right)^2. \tag{5.11}$$

By (5.9) and (5.10), we get

$$\operatorname{MSE}\left(\widehat{Y}\right) = \mathcal{O}(N^{-1}) + (\mathcal{O}(h))^2 = \mathcal{O}\left(N^{-1} + h^2\right). \tag{5.12}$$

In other words, for N large enough and h sufficiently small, there exists two constants C_1 and C_2 such that

$$\operatorname{MSE}\left(\widehat{Y}\right) \approx C_1 N^{-1} + C_2 h^2.$$

Now, let ε, a positive constant, be the desired mean square accuracy in the sense that

$$\mathrm{MSE}\left(\widehat{Y}\right) = \mathcal{O}\left(\varepsilon^2\right).$$

To achieve such an accuracy, one requires $N = \mathcal{O}\left(\varepsilon^{-2}\right)$ simulations and a regular stepsize $h = \mathcal{O}(\varepsilon)$. We have to distinguish now between the two numerical schemes. The regular Euler method has T/h steps, which is proportional to $h^{-1} = \mathcal{O}\left(\varepsilon^{-1}\right)$, and thus, a computational complexity of $\mathcal{O}\left(\varepsilon^{-3}\right)$ is required for the regular scheme. In the jump-adapted case, the jump times are added to the regular grid. The number of jumps over the time interval $[0,T]$ is given by the random variable $N(T)$ which follows a Poisson distribution with intensity λT. The expected number of jumps is given by $\mathbb{E}[N(T)] = \lambda T$. Therefore, for the jump-adapted scheme, there are $T/h + \lambda T = \mathcal{O}\left(\varepsilon^{-1} + \lambda\right)$ steps, and thus, the computational complexity amounts to $\mathcal{O}\left(\varepsilon^{-2}\left(\frac{1}{\varepsilon} + \lambda\right)\right)$.

Summing up the results for the standard Monte Carlo method for jump-diffusions, to achieve a mean square error of order $\mathcal{O}\left(\varepsilon^2\right)$, the regular Euler approach requires a computational cost of $\mathcal{O}\left(\varepsilon^{-3}\right)$. For the jump-adapted Euler approach, a computational cost of $\mathcal{O}\left(\varepsilon^{-2}\left(\frac{1}{\varepsilon} + \lambda\right)\right)$ is necessary. Note that, by setting the jump intensity λ to zero, we reproduce the result for diffusion processes in [46].

5.3.2 Multilevel Monte Carlo Method for Jump-Diffusions

The idea of the multilevel Monte Carlo method [46] is to apply the Monte Carlo method for several nested levels of time stepsizes and to compute different numbers of paths on each level, from a few paths when the time stepsize is small to many paths when the stepsize is large. By choosing the right balance between the stepsizes and the number of simulated trajectories at each level it is possible to reduce the computational complexity compared to that of the standard Monte Carlo method for a given mean square accuracy.

We introduce now the multilevel Monte Carlo method for stochastic differential equations driven by jump-diffusions. Fix a positive number T as the time end point, an integer $M \geq 2$ as the refinement factor and an integer L as the total number of levels. Define the uniform nested time stepsizes

$$h_l = \frac{T}{M^l}, \quad l = 0, 1, \ldots, L.$$

Furthermore, we fix an m-dimensional Wiener process $(W(t))_{t\in[0,T]}$ and a one-dimensional compound Poisson process $(J(t))_{t\in[0,T]}$. Let P denote the payoff function (e.g. $P = f(S(T))$) and approximate P by P_l, where $P_l = f\left(S_{M^l}\right)$ is an approximation of P based on the numerical discretization of $S(t)$ with a regular stepsize h_l. Applying the telescopic sum, we can write

$$\begin{aligned} P_L &= P_0 + P_1 - P_0 + P_2 - P_1 \pm \ldots + P_{L-1} - P_{L-2} + P_L - P_{L-1} \\ &= P_0 + \sum_{l=1}^{L} \left(P_l - P_{l-1}\right). \end{aligned}$$

Taking the expectation on both sides and using the linearity of the expectation we obtain

$$\mathbb{E}[P_L] = \mathbb{E}[P_0] + \sum_{l=1}^{L} \mathbb{E}[P_l - P_{l-1}]. \tag{5.13}$$

The idea of the multilevel Monte Carlo method is to approximate each term on the right-hand side independently. In fact for the first term we have

$$\mathbb{E}[P_0] \approx \frac{1}{N_0} \sum_{i=1}^{N_0} P_0^{(i)} =: \widehat{Y}_0,$$

where we take the average over N_0 independent samples. The other terms are estimated using N_l independent samples such that

$$\mathbb{E}[P_l - P_{l-1}] \approx \frac{1}{N_l} \sum_{i=1}^{N_l} \left(P_l^{(i)} - P_{l-1}^{(i)}\right) =: \widehat{Y}_l,$$

for $l \in \{1, 2, \ldots, L\}$. We emphasize that the estimates $P_l^{(i)}$ and $P_{l-1}^{(i)}$ are based on the same jump-diffusion path, i.e. the same Brownian motion path and also on the same sample path of the compound Poisson process. Therefore the estimator for the MLMC method is given by

$$\mathbb{E}[P_L] \approx \frac{1}{N_0} \sum_{i=1}^{N_0} P_0^{(i)} + \sum_{l=1}^{L} \frac{1}{N_l} \sum_{i=1}^{N_l} \left(P_l^{(i)} - P_{l-1}^{(i)}\right) = \sum_{l=0}^{L} \widehat{Y}_l =: \widehat{Y}. \tag{5.14}$$

Next we derive the variance and the computational cost for the MLMC estimator $\widehat{Y}$. Firstly we point out that in the partial estimator $\widehat{Y}_l = \frac{1}{N_l} \sum_{i=1}^{N_l} \left(P_l^{(i)} - P_{l-1}^{(i)}\right)$ each term in the sum is produced from a jump-diffusion process that is independent of the jump-diffusion processes used for the other summands. Using this independence combined with the fact that the jump-diffusion processes are identically distributed and denoting the variance of a single sample of $P_l^{(i)} - P_{l-1}^{(i)}$ by V_l, the variance of the partial estimator $\widehat{Y}_l$ is given by

$$\mathrm{Var}\left(\widehat{Y}_l\right) = \mathrm{Var}\left(\frac{1}{N_l} \sum_{i=1}^{N_l} \left(P_l^{(i)} - P_{l-1}^{(i)}\right)\right) = \frac{1}{N_l^2} \sum_{i=1}^{N_l} \underbrace{\mathrm{Var}\left(P_l^{(i)} - P_{l-1}^{(i)}\right)}_{=:V_l} = \frac{V_l}{N_l}.$$

Thus the variance of the combined estimator $\widehat{Y}$ is given by

$$\mathrm{Var}\left(\widehat{Y}\right) = \mathrm{Var}\left(\sum_{l=0}^{L} \widehat{Y}_l\right) = \sum_{l=0}^{L} \mathrm{Var}\left(\widehat{Y}_l\right) = \sum_{l=0}^{L} \frac{V_l}{N_l}. \tag{5.15}$$

Note that we have used the independence of the partial estimators $\widehat{Y}_l$ resulting from the independence of the jump-diffusion processes.

Secondly, concerning the computational complexity of $\widehat{Y}$, at each level l there are N_l Monte Carlo simulations required to approximate the expectation. Furthermore, d numerical discretizations (one for each component of the d-dimensional SDE (5.1)) are carried out with a regular stepsize h_l. For the jump-adapted Euler scheme, the expected number of jumps λT of each discretization has to be added to the number of steps resulting from the regular grid, $\frac{T}{h_l}$. Hence, there are $d\left(\frac{T}{h_l}+\lambda T\right)$ steps necessary in the jump-adapted case. Considering the evaluation of the function f as a single operation, the computational cost is given by

$$\operatorname{Cost}\left(\widehat{Y}\right)=\sum_{l=0}^{L} 2N_l d\left(\frac{T}{h_l}+\lambda T\right)=\sum_{l=0}^{L} 2N_l d\frac{T}{\left(\frac{h_l}{1+\lambda h_l}\right)}.$$

For the regular Euler scheme, the jumps do not affect the time grid, and thus there are $\frac{T}{h_l}$ steps. The computational cost, given by $\operatorname{Cost}\left(\widehat{Y}\right)=\sum_{l=0}^{L}2N_l d\frac{T}{h_l}$, is the same as for diffusion problems. Taking also into account that the order of weak and strong convergence are the same as for the Euler-Maruyama method, the construction for regular Euler is identical to the one in the diffusion case. Therefore we concentrate in the following on the jump-adapted Euler approach.

Finally, fixing a positive constant D as the fixed computational budget, we can set up an optimization problem, which minimizes the variance of $\widehat{Y}$ for a fixed computational complexity:

$$\begin{cases} \text{minimize } \operatorname{Var}\left(\widehat{Y}\right)=\displaystyle\sum_{l=0}^{L}\frac{V_l}{N_l} \\ \\ \text{subject to } \operatorname{Cost}\left(\widehat{Y}\right)=\displaystyle\sum_{l=0}^{L}2N_l d\frac{T}{\left(\frac{h_l}{1+\lambda h_l}\right)}=D\,, \end{cases} \tag{5.16}$$

where we want to find a solution with respect to the positive variables $N_l>0$, with $l\in\{0,1,\dots,L\}$. The optimization problem (5.16) can be solved by standard methods (e.g. using Lagrange multipliers). The following proposition holds.

Proposition 5.3.2. *The solution to the optimization problem for the continuous variables $N_0,N_1,\dots,N_L$ given in (5.16) satisfies for $l\in\{0,1,\dots,L\}$*

$$N_l=\frac{D}{2dT}\frac{\sqrt{V_l\frac{h_l}{1+\lambda h_l}}}{\displaystyle\sum_{k=0}^{L}\sqrt{\frac{V_k}{\frac{h_k}{1+\lambda h_k}}}}. \tag{5.17}$$

Now we show that $V_l=\mathcal{O}(h_l)$. Note that since the jump-diffusion processes are i.i.d., we work in the following with the notation P_l instead of $P_l^{(i)}$. The strong order of convergence 1/2 of the jump-adapter Euler method (see Section 5.2.1) yields

$$\mathbb{E}\left[|S_{M^l}-S(T)|^2\right]=\mathcal{O}(h_l)\quad\text{as }\ l\to\infty. \tag{5.18}$$

For the variance of one single sample we have

$$V_l = \mathrm{Var}(P_l - P_{l-1}) \leq \left(\mathrm{Var}(P_l - P)^{1/2} + \mathrm{Var}(P_{l-1} - P)^{1/2}\right)^2,$$

where we have used the Cauchy-Schwarz inequality.

Furthermore, using the property of f being Lipschitz continuous, we have

$$\begin{aligned} \mathrm{Var}(P_l - P) &\leq \mathbb{E}\left[(P_l - P)^2\right] \\ &= \mathbb{E}\left[\left(f(S_{M^l}) - f(S)\right)^2\right] \\ &\leq C\mathbb{E}\left[|S_{M^l} - S|^2\right], \end{aligned}$$

where $C \in \mathbb{R}_+$. Thus, by (5.18),

$$\mathrm{Var}\left(P_l - P\right) = \mathcal{O}(h_l) \quad \text{as} \quad l \to \infty.$$

Note that we also have $\mathrm{Var}\left(P_{l-1} - P\right) = \mathcal{O}(h_l)$, as $h_{l-1} = Mh_l$ with M being constant. We thus have $V_l = \mathcal{O}(h_l)$ and in other words, for h_l sufficiently small, there is a positive constant K such that $V_l = Kh_l$.

We can then write the number of simulations per level as

$$N_l = \frac{\frac{D}{2dT}\sqrt{Kh_l \frac{h_l}{1+\lambda h_l}}}{\sum_{k=0}^{L}\sqrt{\frac{Kh_k}{\frac{h_k}{1+\lambda h_k}}}} = \frac{D}{2dT}\frac{h_l\sqrt{\frac{1}{1+\lambda h_l}}}{\sum_{k=0}^{L}\sqrt{1+\lambda h_k}}. \tag{5.19}$$

Combining this with (5.15), we obtain for the variance of the MLMC estimator

$$\mathrm{Var}\left(\widehat{Y}\right) = \sum_{l=0}^{L}\frac{V_l}{N_l} = \sum_{l=0}^{L}\frac{Kh_l}{\frac{D}{2dT}\frac{h_l\sqrt{\frac{1}{1+\lambda h_l}}}{\sum_{k=0}^{L}\sqrt{1+\lambda h_k}}} = \frac{2dTK}{D}\left(\sum_{l=0}^{L}\sqrt{1+\lambda h_l}\right)^2. \tag{5.20}$$

Note that by increasing the computational budget the variance can be made as small as desired. However, usually the computational budget is limited. Thus we fix now a mean square accuracy of $\mathcal{O}\left(\varepsilon^2\right)$ and we determine the corresponding computational budget D and the number of levels L.

We consider now the mean square error (5.11). For the bias we have

$$\operatorname{bias}\left(\widehat{Y}\right)=\mathbb{E}\left[\widehat{Y}\right]-\mathbb{E}[P]=\mathbb{E}\left[P_L\right]-\mathbb{E}[P]=\mathbb{E}\left[P_L-P\right],$$

where we have used the linearity of the expectation. Using the first order weak convergence of the jump-adapted Euler scheme (see Section 5.2.1), we obtain

$$\operatorname{bias}\left(\widehat{Y}\right)=\mathbb{E}\left[P_L-P\right]=\mathcal{O}\left(h_L\right). \tag{5.21}$$

Hence, to achieve a mean square error of $\operatorname{MSE}\left(\widehat{Y}\right)=\mathcal{O}\left(\varepsilon^2\right)$, we require in particular that the bias satisfies $\operatorname{bias}\left(\widehat{Y}\right)=\mathcal{O}\left(\varepsilon\right)$, and thus,

$$h_L=\frac{T}{M^L}=\widetilde{K}\varepsilon,$$

for L large enough and where $\widetilde{K}$ is a positive constant. Rearranging terms and taking the natural logarithm we obtain $L=\frac{1}{\log M}\left(\log T+\log\widetilde{K}^{-1}+\log\varepsilon^{-1}\right)$. Hence, the number of levels L satisfies

$$L=\frac{\log\varepsilon^{-1}}{\log M}+\mathcal{O}\left(1\right). \tag{5.22}$$

This shows us how to choose L. Finally, to achieve a mean square accuracy of $\mathcal{O}\left(\varepsilon^2\right)$ we also require $\operatorname{Var}\left(\widehat{Y}\right)=\mathcal{O}\left(\varepsilon^2\right)$. Considering (5.20), this is equivalent to

$$\frac{2dTK}{D}\left(\sum_{l=0}^{L}\sqrt{1+\lambda h_l}\right)^2=\widehat{K}\varepsilon^2,$$

where $\widehat{K}$ is a positive constant. Rearranging terms and taking an upper bound, we get

$$\begin{aligned} D &= 2dTK\widehat{K}^{-1}\varepsilon^{-2}\left(\sum_{l=0}^{L}\sqrt{1+\lambda h_l}\right)^2 \\ &\leq 2dTK\widehat{K}^{-1}\varepsilon^{-2}\left(\sqrt{1+\lambda T}\sum_{l=0}^{L}1\right)^2 \\ &= 2dTK\widehat{K}^{-1}\varepsilon^{-2}\left(1+\lambda T\right)\left(L+1\right)^2. \end{aligned} \tag{5.23}$$

Hence, the computational budget satisfies

$$D=\mathcal{O}\left(d\varepsilon^{-2}(\log\varepsilon)^2(1+\lambda T)\right),$$

where we have used the result for L in (5.22).

Therefore, considering the multilevel Monte Carlo method for jump-diffusions and using the jump-adapted Euler method, to achieve a mean square error of

$$\mathrm{MSE}\left(\widehat{Y}\right) = \mathcal{O}\left(\varepsilon^2\right)$$

a computational complexity of

$$\mathrm{Cost}\left(\widehat{Y}\right) = \mathcal{O}\left(d\varepsilon^{-2}(\log\varepsilon)^2(1+\lambda T)\right)$$

is necessary.

Remark 5.3.3. *Note that when considering the jump-adapted method instead of the regular one, the additional term* $(1+\lambda T)$ *appears in the computational cost. Usually the time end point* T *is fixed and thus the parameter of interest is the jump intensity* λ. *Observe that by setting* λ *to zero, we produce the results for the MLMC method for multi-dimensional diffusions. Hence, the approach in this paper delivers a natural extension of the MLMC method to jump-diffusion processes.*

Remark 5.3.4. *As mentioned earlier, the construction of the MLMC method for jump-diffusions is based on a one-dimensional jump process. In a more general approach, where jumps are driven by an* r*-dimensional compound Poisson process, the computational cost for a fixed mean square accuracy of* $MSE\left(\widehat{Y}\right) = \mathcal{O}\left(\varepsilon^2\right)$ *is given by*

$$Cost\left(\widehat{Y}\right) = \mathcal{O}\left(d\varepsilon^{-2}(\log\varepsilon)^2\left(1+\sum_{i=1}^{r}\lambda_i T\right)\right).$$

5.3.3 Complexity Theorem for Jump-Diffusions

In this section we give an extended version of the complexity theorem, which was stated and proven for diffusion processes in the one-dimensional case in [46]. We extend the theorem to jump-diffusions and consider a d-dimensional SDE driven by an m-dimensional Wiener process and an r-dimensional compound Poisson process.

Theorem 5.3.5 (Complexity Theorem for Jump-Adapted Schemes)**.** *Fix two positive integers* T *and* M *such that* $M \geq 2$ *and let* $\lambda_1, \lambda_2, \ldots, \lambda_r$ *be* r *positive numbers. Let* $(S(t))_{t\in[0,T]} \subset \mathbb{R}^d$ *be a solution to the stochastic differential equation (5.1). Let* P *be a functional of* $S(t)$. *Denote by* P_l *an approximation of* P *using a jump-adapted numerical approximation with a regular time stepsize* $h_l = T/M^l$.

Suppose that there exist independent estimators $\widehat{Y}_l$ *(based on* N_l *Monte Carlo simulations) and that there exist positive constants* $\alpha \geq 1/2$, $\beta > 0$, $c_i > 0$ *(*$i \in \{1,2,3\}$*) such that:*

(i) $\mathbb{E}[P_l - P] \leq c_1 h_l^{\alpha}$,

(ii) $\mathbb{E}\left[\widehat{Y}_l\right] = \begin{cases} \mathbb{E}[P_0], & l = 0 \\ \mathbb{E}[P_l - P_{l-1}], & l > 0 \end{cases}$,

(iii) $Var\big(\widehat{Y}_l\big) \leq c_2 \frac{h_l^\beta}{N_l}$,

(iv) $Cost\big(\widehat{Y}_l\big) \leq c_3 d \dfrac{N_l}{h_l}(1+\tilde{\lambda} h_l)$,

where $\tilde{\lambda} = \sum_{i=1}^r \lambda_i$*. If Conditions (i)-(iv) hold, then there exists a positive constant* c_4 *such that for all* $\varepsilon < \frac{1}{e}$ *there exist positive integers* $L \in \mathbb{N}_*$ *and* $N_l \in \mathbb{N}_*$ *such that the combined estimator of* $\mathbb{E}[P]$,

$$\widehat{Y} = \sum_{l=0}^{L} \widehat{Y}_l,$$

has a mean square error that is bounded by

$$MSE\big(\widehat{Y}\big) = \mathbb{E}\left[\big(\widehat{Y} - \mathbb{E}[P]\big)^2\right] \leq \varepsilon^2$$

with a computational complexity bounded by

$$Cost\big(\widehat{Y}\big) \leq \begin{cases} c_4 d\varepsilon^{-2}(1+\tilde{\lambda}T), & \beta > 1, \\ c_4 d\varepsilon^{-2}(\log\varepsilon)^2(1+\tilde{\lambda}T), & \beta = 1, \\ c_4 d\varepsilon^{-2-(1-\beta)/\alpha}(1+\tilde{\lambda}T), & 0 < \beta < 1, \end{cases}$$

where the logarithm is taken with the natural basis.

An immediate consequence of the theorem above is the result for the regular Euler-Maruyama scheme presented in the following corollary, which also holds for multi-dimensional diffusion problems.

Corollary 5.3.6 (Regular Schemes)**.** *Suppose we use a regular scheme for the numerical approximation. Subject to the assumptions (i)-(iii) as in Theorem 5.3.5, and replacing (iv) with* $Cost\big(\widehat{Y}_l\big) \leq c_3 d \dfrac{N_l}{h_l}$, *the mean square error is bounded by* $MSE\big(\widehat{Y}\big) \leq \varepsilon^2$ *and the bound for the computational cost is characterized by*

$$Cost\big(\widehat{Y}\big) \leq \begin{cases} c_4 d\varepsilon^{-2}, & \beta > 1, \\ c_4 d\varepsilon^{-2}(\log\varepsilon)^2, & \beta = 1, \\ c_4 d\varepsilon^{-2-(1-\beta)/\alpha}, & 0 < \beta < 1. \end{cases}$$

Proof of Theorem 5.3.5. Throughout this proof the notation $\lceil x \rceil$ is used for rounding up the real number x to the next higher integer. First, let $\varepsilon < \frac{1}{e}$ and we choose the total number of levels L to be equal to

$$L = \left\lceil \frac{\log\big(\sqrt{2}c_1 T^\alpha \varepsilon^{-1}\big)}{\alpha \log M} \right\rceil. \tag{5.24}$$

In the first part we prove that the squared bias of $\widehat{Y}$ is bounded above by $\frac{\varepsilon^2}{2}$. In the second part we prove that the variance of $\widehat{Y}$ has an upper bound given by $\frac{\varepsilon^2}{2}$. Combining the results of the two parts leads to an upper bound of

$$\mathrm{MSE}\left(\widehat{Y}\right) = \mathrm{Var}\left(\widehat{Y}\right) + \left(\mathrm{bias}\left(\widehat{Y}\right)\right)^2 \leq \frac{\varepsilon^2}{2} + \frac{\varepsilon^2}{2} = \varepsilon^2. \tag{5.25}$$

(a) Estimation of $\left(\mathit{bias}\left(\widehat{Y}\right)\right)^2$

Using L as defined in (5.24), the following inequalities can be established:

$$\begin{array}{rcccl} \dfrac{\log\left(\sqrt{2}c_1 T^{\alpha}\varepsilon^{-1}\right)}{\alpha \log M} & \leq & L & < & \dfrac{\log\left(\sqrt{2}c_1 T^{\alpha}\varepsilon^{-1}\right)}{\alpha \log M} + 1 \\ \Longleftrightarrow \qquad \sqrt{2}c_1\varepsilon^{-1} & \leq & \underbrace{\left(\dfrac{M^L}{T}\right)^{\alpha}}_{=h_L^{-\alpha}} & < & \sqrt{2}c_1\varepsilon^{-1}M^{\alpha}. \end{array}$$

Therefore we get the inequalities

$$\frac{M^{-\alpha}\varepsilon}{\sqrt{2}} < c_1 h_L^{\alpha} \leq \frac{\varepsilon}{\sqrt{2}}. \tag{5.26}$$

Recall that $\widehat{Y}$ is an estimator of $\mathbb{E}[P]$, and thus the bias of the combined estimator is given by

$$\mathrm{bias}\left(\widehat{Y}\right) = \mathbb{E}\left[\widehat{Y}\right] - \mathbb{E}[P].$$

Taking into account the linearity of the expectation and condition (ii), we have

$$\mathbb{E}\left[\widehat{Y}\right] = \mathbb{E}\left[\sum_{l=0}^{L} \widehat{Y}_l\right] = \sum_{l=0}^{L} \mathbb{E}\left[\widehat{Y}_l\right] = \mathbb{E}[P_0] + \sum_{l=1}^{L} \mathbb{E}[P_l - P_{l-1}] = \mathbb{E}[P_L].$$

Using in addition Condition (i), the squared bias satisfies $\left(\mathrm{bias}\left(\widehat{Y}\right)\right)^2 \leq c_1^2 h_L^{2\alpha}$, and taking into account the upper bound of (5.26), we obtain

$$\left(\mathrm{bias}\left(\widehat{Y}\right)\right)^2 \leq \frac{\varepsilon^2}{2}.$$

(b) Estimation of $\mathit{Var}\left(\widehat{Y}\right)$

Now it remains to show that the variance of $\widehat{Y}$ is bounded by $\frac{\varepsilon^2}{2}$. First we establish the inequality

$$\sum_{l=0}^{L} h_l^{-1} < \frac{M^2}{M-1}\left(\sqrt{2}c_1\right)^{1/\alpha}\varepsilon^{-2}. \tag{5.27}$$

In fact we have by the definition of the time stepsize h_l

$$\sum_{l=0}^{L} h_l^{-1} = \sum_{l=0}^{L} \left(\frac{T}{M^l}\right)^{-1} = h_L^{-1} \sum_{l=0}^{L} M^{l-L} < h_L^{-1} \frac{M}{M-1}.$$

In addition, by the lower limit of (5.26), we have $h_L^{-1} < M\left(\frac{\varepsilon}{\sqrt{2}c_1}\right)^{-1/\alpha}$, and thus

$$\sum_{l=0}^{L} h_l^{-1} < \frac{M^2}{M-1} \left(\sqrt{2}c_1\right)^{1/\alpha} \varepsilon^{-1/\alpha}.$$

Finally, since by assumption $\varepsilon < 1$ the following inequalities are equivalent:

$$\varepsilon^{-1/\alpha} \leq \varepsilon^{-2} \Leftrightarrow \alpha \geq \frac{1}{2}.$$

As by assumption $\alpha \geq \frac{1}{2}$ we obtain (5.27).

To pursue the proof we need to distinguish three cases.

Case 1: $\beta = 1$. Inspired by (5.19), we set the number of samples at level l to

$$N_l = \left\lceil 2\varepsilon^{-2}(L+1)c_2 \frac{h_l}{\sqrt{1+\tilde{\lambda} h_l}} \sqrt{1+\tilde{\lambda} T} \right\rceil. \tag{5.28}$$

Now, by considering first the independence of the partial estimators $\widehat{Y}_l$, followed by Condition (iii) and then the definition of N_l in (5.28), we have for the variance an upper bound given by $\frac{\varepsilon^2}{2}$.

Indeed, using $\mathrm{Var}\left(\widehat{Y}\right) = \mathrm{Var}\left(\sum_{l=0}^{L} \widehat{Y}_l\right) = \sum_{l=0}^{L} \mathrm{Var}\left(\widehat{Y}_l\right)$, we have

$$\begin{aligned} \sum_{l=0}^{L} \mathrm{Var}\left(\widehat{Y}_l\right) &\leq \sum_{l=0}^{L} c_2 N_l^{-1} h_l \\ &\leq \sum_{l=0}^{L} c_2 h_l \left(\frac{\varepsilon^2}{2} (L+1)^{-1} c_2^{-1} \frac{\sqrt{1+\tilde{\lambda} h_l}}{h_l} \frac{1}{\sqrt{1+\tilde{\lambda} T}} \right) \\ &= \tfrac{\varepsilon^2}{2} (L+1)^{-1} \tfrac{1}{\sqrt{1+\tilde{\lambda} T}} \sum_{l=0}^{L} \sqrt{1+\tilde{\lambda} h_l}, \ \leq \ \frac{\varepsilon^2}{2}, \end{aligned}$$

where we used $\sum_{l=0}^{L} \sqrt{1+\tilde{\lambda} h_l} \leq \sqrt{1+\tilde{\lambda} T}(L+1)$. Therefore, in the case of $\beta = 1$, the mean square error of $\widehat{Y}$ is bounded by ε^2, i.e. (5.25) is satisfied. We derive now an upper bound for the computational complexity of the combined estimator in the case of $\beta = 1$. The idea is to bound first N_l and then to use condition (iv). By definition of N_l (see (5.28)), we have in particular

$$N_l < 2\varepsilon^{-2}(L+1)c_2\frac{h_l}{\sqrt{1+\tilde{\lambda}h_l}}\sqrt{1+\tilde{\lambda}T}+1. \tag{5.29}$$

We aim to find an upper bound for $L+1$. By definition of L (see (5.24)), the number of levels is bounded above by

$$L \quad < \quad \frac{\log\left(\sqrt{2}c_1T^\alpha\varepsilon^{-1}\right)}{\alpha\log M}+1 \quad = \quad \frac{\log\varepsilon^{-1}}{\alpha\log M}+\frac{\log\left(\sqrt{2}c_1T^\alpha\right)}{\alpha\log M}+1.$$

Furthermore we notice that $\varepsilon < \frac{1}{e} \Leftrightarrow e < \varepsilon^{-1} \Leftrightarrow 1 < \log\varepsilon^{-1}$. Thus we get

$$L+1 \quad < \quad \frac{\log\varepsilon^{-1}}{\alpha\log M}+\frac{\log\left(\sqrt{2}c_1T^\alpha\right)}{\alpha\log M}+2 \quad \le \quad c_5\log\varepsilon^{-1}, \tag{5.30}$$

where $c_5 = \frac{1}{\alpha\log M}+\max\left(0,\frac{\log(\sqrt{2}c_1T^\alpha)}{\alpha\log M}\right)+2$. Using first condition (iv), followed by the upper bound (5.29) of N_l and the inequality (5.27) as well as the bound (5.30) for $L+1$, we end up with the computational complexity for $\mathrm{Cost}\left(\widehat{Y}\right)=\sum_{l=0}^{L}\mathrm{Cost}\left(\widehat{Y}_l\right)$ bounded by

$$\begin{aligned}
\mathrm{Cost}\left(\widehat{Y}\right) \quad &\le \quad \sum_{l=0}^{L}c_3dN_l\frac{1+\tilde{\lambda}h_l}{h_l} \\
&< \quad \sum_{l=0}^{L}c_3d\frac{1+\tilde{\lambda}h_l}{h_l}\left(2\varepsilon^{-2}(L+1)c_2\frac{h_l}{\sqrt{1+\tilde{\lambda}h_l}}\sqrt{1+\tilde{\lambda}T}+1\right) \\
&= \quad c_3d2\varepsilon^{-2}(L+1)c_2\sqrt{1+\tilde{\lambda}T}\sum_{l=0}^{L}\sqrt{1+\tilde{\lambda}h_l}+c_3d\sum_{l=0}^{L}\frac{1+\tilde{\lambda}h_l}{h_l} \\
&\le \quad c_3d2\varepsilon^{-2}(L+1)^2c_2(1+\tilde{\lambda}T)+c_3d(1+\tilde{\lambda}T)\sum_{l=0}^{L}h_l^{-1} \\
&\le \quad (1+\tilde{\lambda}T)d\varepsilon^{-2}\left(\log\varepsilon\right)^2\left[c_32c_5^2c_2+c_3\tfrac{M^2}{M-1}\left(\sqrt{2}c_1\right)^{1/\alpha}\right].
\end{aligned}$$

Hence an upper bound for the computational cost is given by

$$\mathrm{Cost}\left(\widehat{Y}\right) \le c_4d\varepsilon^{-2}(\log\varepsilon)^2(1+\tilde{\lambda}T),$$

where $c_4 = 2c_3c_5^2c_2+c_3\frac{M^2}{M-1}\left(\sqrt{2}c_1\right)^{1/\alpha}$.

Case 2: $\beta > 1$**.** The number of simulations at level l is chosen such that

$$N_l = \left\lceil 2\varepsilon^{-2} c_2 T^{(\beta-1)/2} \left(1 - M^{-(\beta-1)/2}\right)^{-1} \frac{h_l^{(\beta+1)/2}}{\sqrt{1+\tilde{\lambda} h_l}} \sqrt{1+\tilde{\lambda} T} \right\rceil . \tag{5.31}$$

Observe that the choice of N_l is the same as in the first case if the parameter β is fixed at $\beta = 1$. In the following the derivation of the upper bound of the variance of the combined estimator is very similar as in the case with $\beta = 1$. Due to the independence of the partial estimators $\widehat{Y}_l$, condition (iii) and the definition of N_l in (5.31), we have

$$\sum_{l=0}^{L} \operatorname{Var}\left(\widehat{Y}_l\right) \leq \sum_{l=0}^{L} c_2 \frac{h_l^{\beta}}{N_l} \leq \tfrac{\varepsilon^2}{2} T^{-(\beta-1)/2} \left(1 - M^{-(\beta-1)/2}\right) \sum_{l=0}^{L} h_l^{(\beta-1)/2}.$$

Using the above upper bound with the inequality

$$\sum_{l=0}^{L} h_l^{(\beta-1)/2} < T^{(\beta-1)/2} \frac{M^{(\beta-1)/2}}{M^{(\beta-1)/2} - 1}, \tag{5.32}$$

we obtain

$$\operatorname{Var}\left(\widehat{Y}\right) \leq \tfrac{\varepsilon^2}{2} T^{-(\beta-1)/2} \left(1 - M^{-(\beta-1)/2}\right) \sum_{l=0}^{L} h_l^{(\beta-1)/2} < \tfrac{\varepsilon^2}{2}.$$

Hence the inequality (5.25) for the mean square error of $\widehat{Y}$ holds also in this case. It remains to find the appropriate upper limit of the computational complexity of the combined estimator $\widehat{Y}$ in this case where $\beta > 1$. By the choice of N_l in (5.31), we have in particular

$$N_l < 2\varepsilon^{-2} c_2 T^{(\beta-1)/2} \left(1 - M^{-(\beta-1)/2}\right)^{-1} \frac{h_l^{(\beta+1)/2}}{\sqrt{1+\tilde{\lambda} h_l}} \sqrt{1+\tilde{\lambda} T} + 1.$$

Combining this with condition (iv) and the inequalities (5.27) and (5.32), we arrive at

$$\begin{aligned}
\operatorname{Cost}\left(\widehat{Y}\right) &\le \sum_{l=0}^{L} c_3 d N_l \frac{1+\tilde{\lambda} h_l}{h_l} \\
&< \sum_{l=0}^{L} c_3 d \frac{(1+\tilde{\lambda} h_l)}{h_l} \Bigg[2\varepsilon^{-2} c_2 T^{(\beta-1)/2} \left(1 - M^{-(\beta-1)/2}\right)^{-1} \frac{h_l^{(\beta+1)/2}\sqrt{1+\tilde{\lambda} T}}{\sqrt{1+\tilde{\lambda} h_l}} \\
&\qquad +1\Bigg] \\
&\le (1+\tilde{\lambda} T) d \Bigg[2\varepsilon^{-2} c_2 c_3 T^{(\beta-1)/2} \left(1 - M^{-(\beta-1)/2}\right)^{-1} \sum_{l=0}^{L} h_l^{(\beta+1)/2} \\
&\qquad + c_3 \sum_{l=0}^{L} h_l^{-1} \Bigg] \\
&< (1+\tilde{\lambda} T) d \left[2\varepsilon^{-2} c_2 c_3 T^{(\beta-1)} \left(1 - M^{-(\beta-1)/2}\right)^{-2} + c_3 \tfrac{M^2}{M-1} \left(\sqrt{2} c_1\right)^{1/\alpha} \varepsilon^{-2} \right].
\end{aligned}$$

Rearranging this expression we get the required upper bound for the computational cost of $\widehat{Y}$:

$$\operatorname{Cost}\left(\widehat{Y}\right) \le c_4 d \varepsilon^{-2} (1+\tilde{\lambda} T)$$

with $c_4 = 2c_2 c_3 T^{\beta-1} \left(1 - M^{-(\beta-1)/2}\right)^{-2} + c_3 \frac{M^2}{M-1} \left(\sqrt{2} c_1\right)^{1/\alpha}$.

Case 3: $0 < \beta < 1$**.** In the last case, we set the number of simulations for level l to

$$N_l = \left\lceil 2\varepsilon^{-2} c_2 h_L^{-(1-\beta)/2} \left(1 - M^{-(1-\beta)/2}\right)^{-1} \frac{h_l^{(\beta+1)/2}}{\sqrt{1+\tilde{\lambda} h_l}} \sqrt{1+\tilde{\lambda} T} \right\rceil. \tag{5.33}$$

Similarly to the previous cases, taking into account the independence of the partial estimators $\widehat{Y}_l$, condition (iii) and the definition of N_l in (5.33), we obtain

$$\sum_{l=0}^{L} \operatorname{Var}\left(\widehat{Y}_l\right) \le \sum_{l=0}^{L} c_2 \frac{h_l^{\beta}}{N_l} \le \tfrac{\varepsilon^2}{2} h_L^{(1-\beta)/2} \left(1 - M^{-(1-\beta)/2}\right) \sum_{l=0}^{L} h_l^{-(1-\beta)/2}.$$

In addition we observe that

$$\sum_{l=0}^{L} h_l^{-(1-\beta)/2} = h_L^{-(1-\beta)/2} \sum_{l=0}^{L} \left(M^{-(1-\beta)/2}\right)^l,$$

where we applied the definition of the time stepsize $h_l = \frac{T}{M^l}$. Using next (recall that $\beta \in]0,1[$)

$$\sum_{l=0}^{L} \left(M^{-(1-\beta)/2}\right)^l < \sum_{l=0}^{\infty} \left(M^{-(1-\beta)/2}\right)^l = \left(1 - M^{-(1-\beta)/2}\right)^{-1},$$

we have

$$\sum_{l=0}^{L} h_l^{-(1-\beta)/2} < h_L^{-(1-\beta)/2}\left(1-M^{-(1-\beta)/2}\right)^{-1}. \tag{5.34}$$

Therefore the variance upper bound is given by

$$\begin{aligned}\operatorname{Var}\left(\widehat{Y}\right) &\le \tfrac{\varepsilon^2}{2} h_L^{(1-\beta)/2}\left(1-M^{-(1-\beta)/2}\right)\sum_{l=0}^{L} h_l^{-(1-\beta)/2}\\ &< \tfrac{\varepsilon^2}{2} h_L^{(1-\beta)/2}\left(1-M^{-(1-\beta)/2}\right) h_L^{-(1-\beta)/2}\left(1-M^{-(1-\beta)/2}\right)^{-1}\\ &= \tfrac{\varepsilon^2}{2}.\end{aligned}$$

Finally we have to find the appropriate upper bound for the computational complexity of the combined estimator in the case where $0<\beta<1$. First, using condition (iv), the upper bound of N_l as in (5.33), we obtain

$$\begin{aligned}\operatorname{Cost}\left(\widehat{Y}\right) &\le \sum_{l=0}^{L} c_3 d N_l \frac{1+\tilde{\lambda} h_l}{h_l}\\ &< \sum_{l=0}^{L} c_3 d \frac{1+\tilde{\lambda} h_l}{h_l}\left[2\varepsilon^{-2} c_2 h_L^{-(1-\beta)/2}\left(1-M^{-(1-\beta)/2}\right)^{-1}\frac{h_l^{(\beta+1)/2}\sqrt{1+\tilde{\lambda}T}}{\sqrt{1+\tilde{\lambda} h_l}}\right.\\ &\qquad \left.+1\right]\\ &\le (1+\tilde{\lambda}T) d\left[2\varepsilon^{-2} c_2 c_3 h_L^{-(1-\beta)/2}\left(1-M^{-(1-\beta)/2}\right)^{-1}\sum_{l=0}^{L} h_l^{(\beta-1)/2}\right.\\ &\qquad \left.+c_3\sum_{l=0}^{L} h_l^{-1}\right].\end{aligned}$$

Taking into account inequality (5.34), we observe

$$h_L^{-(1-\beta)/2}\left(1-M^{-(1-\beta)/2}\right)^{-1}\sum_{l=0}^{L} h_l^{-(1-\beta)/2} < h_L^{-(1-\beta)}\left(1-M^{-(1-\beta)/2}\right)^{-2}$$

and then using the lower bound given in (5.26), we obtain

$$\begin{aligned}& h_L^{-(1-\beta)/2}\left(1-M^{-(1-\beta)/2}\right)^{-1}\sum_{l=0}^{L} h_l^{-(1-\beta)/2}\\ &< \left(\sqrt{2}c_1\right)^{(1-\beta)/\alpha} M^{1-\beta}\varepsilon^{-(1-\beta)/\alpha}\left(1-M^{-(1-\beta)/2}\right)^{-2}.\end{aligned}$$

In addition we notice that since $\beta\in]0,1[$, $\alpha>0$ and $\varepsilon<e^{-1}<1$,

$$\varepsilon^{-2}<\varepsilon^{-2-(1-\beta)/\alpha}.$$

Combining this results with the inequality (5.27), we end up with

$$\mathrm{Cost}\left(\widehat{Y}\right) < c_4 d\varepsilon^{-2-(1-\beta)/\alpha}(1+\tilde{\lambda}T),$$

where

$$c_4 = 2c_3c_2\left(\sqrt{2}c_1\right)^{(1-\beta)/\alpha} M^{1-\beta}\left(1-M^{-(1-\beta)/2}\right)^{-2} + c_3\frac{M^2}{M-1}\left(\sqrt{2}c_1\right)^{1/\alpha}.$$

This completes the last case and therefore the proof of the complexity theorem.

□

Remark 5.3.7. *Note that by setting the jump intensities $\lambda_1, \lambda_2, \dots, \lambda_r$ to zero, and by considering the one-dimensional case, we reproduce the complexity theorem for the multilevel Monte Carlo method based on diffusions stated in [46].*

Now we briefly discuss the conditions of the complexity theorem. The first condition, (i), defines an upper bound for the bias of the estimator P_l. The constant α can be obtained by looking at the order of weak convergence of the numerical approximation method. For the regular and the jump-adapted Euler method, under appropriate conditions for $a(t,S(t))$, $b(t,S(t))$, $c(t,S(t))$ and the jump intensities $\lambda_1,\lambda_2,\dots,\lambda_r$, the weak order is one and therefore $\alpha = 1$ (see Section 5.2.1). The second condition, (ii), and the last condition, (iv), limit the choice of the partial estimators $\widehat{Y}_l$. Condition (ii) fixes the mean of $\widehat{Y}_l$ and the fourth condition (iv) defines an upper bound for the computational complexity of the partial estimators $\widehat{Y}_l$. Note that these two conditions can usually be met by choosing partial estimators that are located around a given point (can be obtained by deducting the bias of the estimator from the estimator and using this difference as a new estimator) and by considering an upper limit for the computational complexity of the estimators. The most delicate condition is the third one, (iii). This condition demands the variance of $\widehat{Y}$ to be bounded by the term given on the right-hand side of (iii). In the case of the regular and the jump-adapted Euler method, as in the approach in Section 5.3.2, an upper bound for the variance, in particular β, can be found using the strong convergence property of the numerical method. In particular for the Euler method, we have $\beta = 1$ (see Section 5.2.1).

5.4 Numerical Examples

In this section we consider two jump-diffusion models, the Merton and the Kou model, and compare numerically the performance of the proposed multilevel Monte Carlo method to the standard Monte Carlo method without any variance reduction technique, as well as to the standard Monte Carlo method with variance reduction techniques, in this case we use antithetic variates and control variates.

5.4.1 The Merton and the Kou model

The Merton model, introduced in 1976 by Robert C. Merton in [80], is historically the first jump-diffusion model in finance. The Kou model was first presented in 2002 by Steven G. Kou in [67]. The two models are studied in detail in Section 2.1.2. Here we briefly recall their definition and exact solutions. Both models are specified by the particular form ($d = 1$, $m = 1$, $r = 1$) of the SDE (5.1):

$$\begin{cases} \mathrm{d}S(t) & = & \mu S(t-)\mathrm{d}t + \sigma S(t-)\mathrm{d}W(t) + S(t-)\mathrm{d}J(t), \quad 0 \le t \le T, \\ S(0) & = & S_0, \end{cases} \tag{5.35}$$

where $J(t) = \sum_{i=1}^{N(t)} (V_i - 1)$, and where $N(t)$ is a Poisson process with intensity λ. In the Merton model the jump sizes are characterized by

$$\log(V_i) \overset{\text{iid}}{\sim} \mathcal{N}\left(\eta, v^2\right)$$

with $\eta \in \mathbb{R}$ and $v > 0$. In the Kou model the jump sizes are double exponentially distributed, i.e.

$$\log(V_i) \overset{\text{iid}}{\sim} \mathcal{K}\left(\eta_1, \eta_2, p\right),$$

where $\mathcal{K}$ is an asymmetric exponential distribution, whose density function is given by

$$f_k(x) = p\eta_1 e^{-\eta_1 x} 1_{\{x \ge 0\}} + (1-p)\eta_2 e^{\eta_2 x} 1_{\{x<0\}} \tag{5.36}$$

with $x \in \mathbb{R}$, $\eta_1 > 1$, $\eta_2 > 0$ and $p \in [0,1]$. The parameters η_1 and η_2 define the decay of the tails in the distribution of positive and negative jumps and the parameter p specifies the probability of an upward jump (see e.g. [32]).

Both jump-diffusion models admit an analytical solution given by

$$S(t) = S_0 \exp\left\{\left(\mu - \frac{\sigma^2}{2}\right) t + \sigma W(t)\right\} \prod_{i=1}^{N(t)} V_i$$

(see Proposition 2.1.14).

In the following we focus on pricing European call options, that is we intend to estimate the expectation $\mathbb{E}\left[f(S(T))\right]$ with f given as

$$f(S(T)) = e^{-rT} \max(S(T) - K, 0),$$

where r is the risk-free interest rate, T the maturity and K the strike price of the option, and $S(T)$ is specified by equation (5.35) (see e.g. [32]).

Remark 5.4.1. *To price options one works with risk-neutral measures. Under the risk-neutral probability measure, i.e. under the measure such that the discounted underlying* $\left(e^{-rt}S(t)\right)_{t\ge 0}$

is a martingale (see [70]), the Merton model is given by (5.35) with drift

$$\mu = r - \lambda\left(\mathbb{E}\left[V_i\right] - 1\right) = r - \lambda\left(\exp\left(\eta + \frac{\nu^2}{2}\right) - 1\right)$$

(see e.g. [80] and [49]). Similarly, the drift for the Kou model in the risk-neutral case can be specified by

$$\mu = r - \lambda\left(\frac{p\eta_1}{\eta_1 - 1} + \frac{q\eta_2}{\eta_2 + 1} - 1\right)$$

(see e.g. [68]).

5.4.2 Two Variance Reduction Techniques

A brief overview of the most common variance reduction techniques is given in Section 2.3.2. Here we describe now the two variance reduction techniques which we compare in the following numerically with the MLMC method.

Antithetic Variates

The idea of the antithetic variates is to produce for every sample path an antithetic one, which is based on the realizations of the original path, and thus is computationally cheap to get. For instance, if the random variable U is uniformly distributed over the interval $[0,1]$, then so is $1-U$. Suppose sample paths are generated by realizations $u_1, u_2, \ldots$ of U. Then the antithetic paths are produced using $1-u_1, 1-u_2, \ldots$ as realizations (see e.g. [49]).

The antithetic variates estimator is given by

$$\widehat{Y}_{AV} = \frac{1}{2N}\left(\sum_{i=1}^{N} f\left(S_G^{(i)}\right) + \sum_{i=1}^{N} f\left(\tilde{S}_G^{(i)}\right)\right),$$

where the realizations $\tilde{S}_G^{(i)}$ are based on an antithetic path. To produce sample paths of the two jump-diffusion models presented in Section 5.4.1 one requires realizations of the normal distribution for the diffusion part, and the log-normal and the double exponential distribution for either of the jump parts. If x is a realization of the normal distribution, then we take $\tilde{x} = -x$ for the antithetic path. For the Merton model, if x is log-normally distributed, then the antithetic realization is obtained by

$$\tilde{x} = \exp\left(\mu - \sigma\left(\frac{\log(x) - \mu}{\sigma}\right)\right).$$

For the Kou model, observe that a realization of the double exponential distribution with density given in (5.36) can be obtained by

$$x = \begin{cases} \frac{1}{\eta_2} \ln\left(\frac{u}{1-p}\right), & \text{if } u < 1-p, \\ -\frac{1}{\eta_1}\left[\ln\left(\frac{1}{p}\right) + \ln(1-u)\right], & \text{if } u \geq 1-p, \end{cases}$$

where u is a realization of a standard uniform distribution. Hence, for the antithetic path we take into account $1-u$ instead of u, i.e.

$$\tilde{x} = \begin{cases} \frac{1}{\eta_2} \ln\left(\frac{1-u}{1-p}\right), & \text{if } 1-u < 1-p, \\ -\frac{1}{\eta_1}\left[\ln\left(\frac{1}{p}\right) + \ln(u)\right], & \text{if } 1-u \geq 1-p. \end{cases}$$

Control Variates

Suppose we would like to estimate the mean of a random variable Y. Let $\bar{Y}$ be an unbiased estimator of $\mathbb{E}[Y]$ and let X be another random variable (called control variate) with known mean. Then $Y^* = \bar{Y} - \xi(X - \mathbb{E}[X])$ is also an unbiased estimator of $\mathbb{E}[Y]$ for any coefficient ξ, but the variance of Y^* can be minimized. Note that the parameter ξ can be estimated using a least-squares approach (see e.g. [49]). The same idea also applies for functionals of random variables.

For our example, under the risk-neutral measure the process $\left(e^{-rt}S(t)\right)_{t\geq 0}$ is a martingale. Hence,

$$\mathbb{E}\left[e^{-rT}S(T)\right] = S_0,$$

and thus, $S(T)$ can be used as control variate (see [49]). The control variates estimator is given by

$$\widehat{Y}_{CV} = \frac{1}{N}\sum_{i=1}^{N}\left(f\left(S_G^{(i)}\right) - \widehat{\xi}_N\left(S_G^{(i)} - e^{rT}S_0\right)\right),$$

where $\widehat{\xi}_N$ is the least-squares estimator of ξ (see e.g. [49]).

5.4.3 Numerical Results

We consider Equation (5.35) in the setting of Section 5.4.1 with $T = 1$, $S_0 = 1$, $K = 1$, $r = 0.05$, $\sigma = 0.2$, $\lambda = 1$. Further we employ the refinement factor $M = 4$ in the MLMC method. For the Merton model, as in Example 10.2 in [32], we fix $\eta = -0.1$ and $\nu = 0.1$. For the Kou model, we set the probability of an upward jump to $p = 0.3$ and we fix $\eta_1 = 50$ and $\eta_2 = 25$, as, for instance, in [68].

Remark 5.4.2. *With regard to a financial model, the chosen data have the following meaning: The parameter T represents the maturity of the option and the initial share price S_0 as well as the strike price are* 1 *(we do not specify the currency here). The risk-free interest rate is set to* 5%

and the implied volatility is chosen to be 20% *(see e.g. [61]). In a financial context the jump intensity typically lies between* 0.05 *and* 2*, see e.g. [25]. Here we have chosen* $\lambda = 1$ *as this is often the case in [32].*

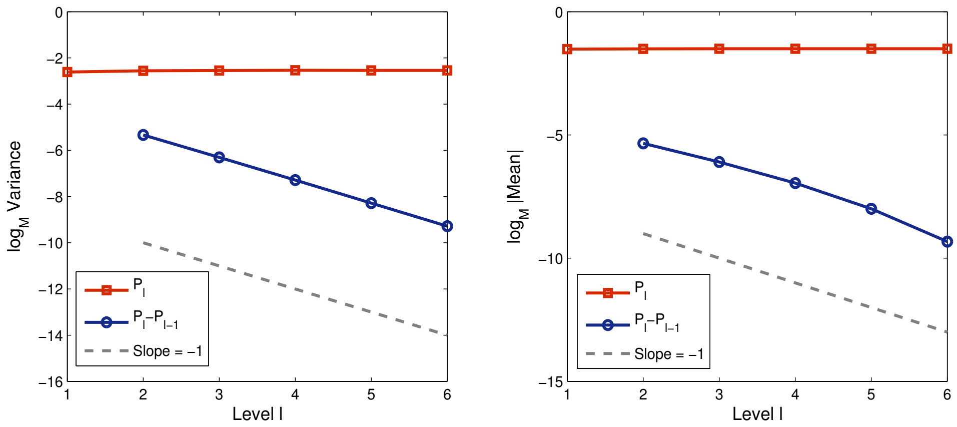

Figure 5.1: Comparison of the variance (left figure) and the mean (right figure) of the MLMC and the MC method over different levels using the Merton model and jump-adapted Euler.

In Figure 5.1 we compare the variance and the mean of the MLMC method and the standard Monte Carlo method without any variance reduction technique using the jump-adapted Euler method applied to the Merton model. To produce the two plots, 2×10^4 sample paths have been generated. The left plot shows the logarithm with base M of the variance of P_l, the discrete approximation of the variable P using the time stepsize $h_l = \frac{T}{M^l}$, and P_{l-1}, respectively, against the number of levels l. One observes that the curve for $P_l - P_{l-1}$ is almost parallel to the straight line of slope minus one. This indicates that the variance of a single sample verifies $\mathrm{Var}\,(P_l - P_{l-1}) = \mathcal{O}\,(h_l)$ as suggested by the theory, see Section 5.3.2. Note that the variance of P_l, used for the standard MC method, is more or less constant whereas the variance of $P_l - P_{l-1}$, used for the MLMC method, decreases as l increases. At level $l = 6$, $\mathrm{Var}\,(P_l - P_{l-1})$ is approximately 4^6 times smaller than $\mathrm{Var}\,(P_l)$.

The right plot of Figure 5.1 represents the logarithm with base M of the absolute value of the mean of P_l and $P_l - P_{l-1}$, respectively, against the number of levels l. The curve for $\mathbb{E}\,[P_l - P_{l-1}]$ is almost linear with slope minus one. Therefore, we have $\mathbb{E}\,[P_l - P_{l-1}] = \mathcal{O}\,(h_l)$, which corresponds to the weak convergence of order one of the jump-adapted Euler method, see Section 5.2.1. At level $l = 6$, the absolute value of $\mathbb{E}\,[P_l - P_{l-1}]$, used for the MLMC method, is about 4^8 times smaller than the absolute value of $\mathbb{E}\,[P_l]$, used for the standard Monte Carlo method.

We next also compare the MLMC method to the standard MC method with two variance reduction techniques described in Section 5.4.2. As a measure of the effectiveness of the methods we use the mean square error (MSE) described in Section 5.3.1 (see also e.g. [49]).

The MSE can be approximated by a sample average over N^* simulations:

$$\mathrm{MSE}\left(\widehat{Y}\right) = \mathbb{E}\left[\left(\widehat{Y} - \mathbb{E}\left[f(S(T))\right]\right)^2\right] \approx \frac{1}{N^*}\sum_{i=1}^{N^*}\left(\widehat{Y}^{(i)} - Y\right)^2, \tag{5.37}$$

where $\widehat{Y}^{(i)}$ are independent realizations at time T. For our computations we have used $N^* = 1000$. We also report the computational cost of the different methods for ε being the desired parameter in the mean square accuracy of $\mathrm{MSE}\left(\widehat{Y}\right) = \mathcal{O}\left(\varepsilon^2\right)$ for the MLMC method. Based on (5.23) in Section 5.3.2 we can bound the computational cost of the MLMC method in the jump-adapted case by

$$\mathrm{Cost}\left(\widehat{Y}\right) \le 2T\varepsilon^{-2}(1+\lambda T)(L+1)^2, \tag{5.38}$$

where we have set the constants K and $\widehat{K}$ in (5.23) to 1. Similarly we get in the regular case the same upper bound without the factor $(1+\lambda T)$. The computational cost of the other three methods is given by

$$\mathrm{Cost}\left(\widehat{Y}\right) = \begin{cases} 2N\left(\frac{T}{h} + \lambda T\right), & \text{for jump-adapted Euler,} \\ 2N\frac{T}{h}, & \text{for regular Euler.} \end{cases} \tag{5.39}$$

Remark 5.4.3. *The only additional cost, compared to the standard Monte Carlo method, of the control variates estimator is due to the computation of* $\widehat{\xi}_N$*. However, this cost is negligible compared to the other cost since* $\widehat{\xi}_N$ *can often be computed using vector multiplication. For the antithetic variates, antithetic paths are generated, but these are based on realizations of the original sample paths, and thus, no significant additional cost occurs.*

ε	MLMC	MC	AV	CV
0.1	1.00e-2	1.10e-3	2.89e-4	2.44e-5
0.01	1.00e-4	7.98e-5	3.96e-5	1.99e-5
0.005	2.50e-5	3.68e-5	2.56e-5	1.31e-5
0.002	4.00e-6	2.81e-5	2.33e-5	1.14e-5
0.001	1.00e-6	2.27e-5	2.21e-5	1.06e-5
0.0001	1.00e-8	2.18e-5	2.15e-5	1.03e-5

Table 5.1: Estimated mean square error of the methods MLMC, standard MC, antithetic variates (AV) and control variates (CV) for different values of ε using the Merton model and jump-adapted Euler.

Table 5.1 shows the estimated mean square error of the four methods for different values of ε using the Merton model and jump-adapted Euler. Similar results have been obtained in the regular case and for the Kou model. The results are obtained through the following procedure. For a given ε, using (5.38), the upper bound for the computational cost of the

MLMC is computed and then fixed. Furthermore, the total number of levels L is determined according to (5.22). For the MC, the antithetic and the control variates methods, we fix the stepsize $h = h_L = T/M^L$ as in [65] or [46]. Then, the number of simulations N is computed for these methods by (5.39). Finally, we run the N simulations and approximate the MSE of the different methods according to (5.37). Figure 5.2 illustrates the results in a loglog-plot. The accuracy ε is taken from the set

$$\varepsilon \in \{0.1, 0.01, 0.005, 0.002, 0.001, 0.0001\}.$$

We observe that as ε gets smaller, i.e. as the accuracy increases, the MLMC method has the lowest mean square error of all methods, followed by the control variates method, the antithetic method and the standard Monte Carlo method. In addition we notice that the MSE of the MLMC method decays linearly as ε tends to zero, whereas the MSE of the other three methods first roughly decays linearly to finally hardly decay as ε decreases.

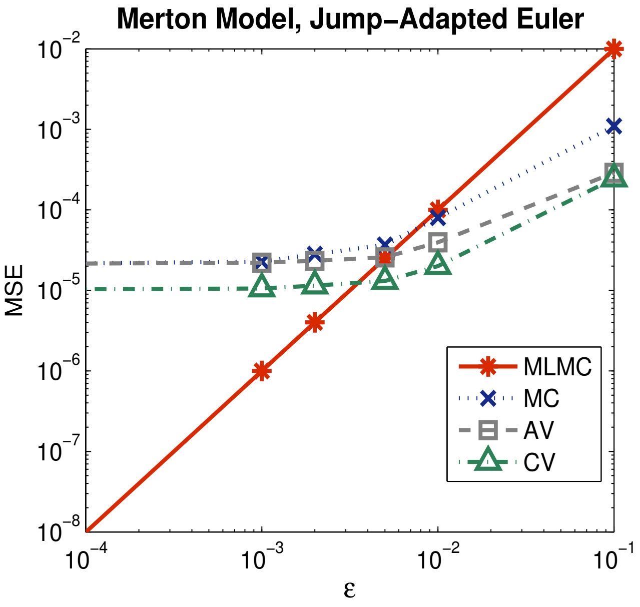

Figure 5.2: Estimated mean square error of the methods MLMC, standard MC, antithetic variates (AV) and control variates (CV) for different values of ε using the Merton model and jump-adapted Euler.

5.5 Stabilized Multilevel Monte Carlo Method for Jump-Diffusion Processes

In this section we combine the approach of the previous sections with the concepts of Chapter 3 and Chapter 4, i.e. we introduce a stabilized multilevel Monte Carlo method for jump-diffusion processes. This is especially useful for applications characterized by stiff stochastic differential equations based on jump-diffusions. Before we start investigating the stabilized

MLMC approach for jump-diffusions, we briefly present the stabilized Monte Carlo method for jump-diffusions.

In what follows we consider SDE (5.1) and the numerical integrator S-ROCK1-JD introduced in Definition 4.2.1. Without loss of generality we assume $d = 1$. The extension to the multi-dimensional case is straightforward. Additionally to the strong order of convergence 1/2 of S-ROCK1-JD that we have proven in Chapter 4, we assume that this integrator is of weak order of convergence 1. Similar to (3.18) we consider the following stability constraint:

$$\frac{M^{-l}\rho}{c_1 s_l^2} \leq 1, \tag{5.40}$$

for $s_l \geq 2$ indicating the stage number, $h_l = \frac{T}{M^l}$ the time stepsize, ρ the stiffness parameter and c_1 some constant similar to c_{SR1} but for the S-ROCK1-JD method instead of the S-ROCK1 method.

5.5.1 Stabilized Monte Carlo Method for Jump-Diffusions

Here, we proceed as in Section 5.3.1. The stabilized Monte Carlo estimator for jump-diffusions is defined by

$$\mathbb{E}\left[f\left(S(T)\right)\right] \approx \frac{1}{N}\sum_{i=1}^{N} f\left(S_G^{(i)}\right) =: \tilde{Y}, \tag{5.41}$$

where the $S_G^{(i)}$ are independent numerical approximations of $S(T)$ using the S-ROCK1-JD method with s stages and a uniform time stepsize h. In the jump-adapted case the jump times are added to the grid (see Remark 5.3.1). Similar to (5.9) one can show that for the bias of the estimator $\tilde{Y}$ it holds that

$$\operatorname{bias}\left(\tilde{Y}\right) = \mathcal{O}\left(h\right),$$

where we have used in particular the weak order of convergence one of the S-ROCK1-JD method. By taking into account the strong order of convergence 1/2 of the S-ROCK1-JD method and by proceeding as in (5.10) one obtains for the variance

$$\operatorname{Var}\left(\tilde{Y}\right) = \mathcal{O}\left(N^{-1}\right).$$

Therefore, for the mean square error we get

$$\operatorname{MSE}\left(\tilde{Y}\right) = \operatorname{Var}\left(\tilde{Y}\right) + \left(\operatorname{bias}\left(\tilde{Y}\right)\right)^2 = \mathcal{O}\left(N^{-1} + h^2\right).$$

Assume now that a mean square accuracy of $\mathcal{O}\left(\varepsilon\right)$ is desired for some $\varepsilon > 0$. It follows that we require $N = \mathcal{O}\left(\varepsilon^{-2}\right)$ samples and a regular time stepsize $h = \mathcal{O}\left(\varepsilon\right)$.

Distinguishing between the regular and the jump-adapted S-ROCK1-JD method, we can compute the corresponding computational cost. First, for the regular approach we have T/h

steps per sample, and thus, the computational cost is given by

$$\operatorname{Cost}\left(\tilde{Y}\right) = N\frac{T}{h}\,(s+m) = \mathcal{O}\left(\varepsilon^{-5/2}\sqrt{\rho}+\varepsilon^{-3}\right),$$

where we have used that $s = \max\left(\sqrt{\frac{\rho}{c_1}h}, 2\right) = \mathcal{O}\left(\sqrt{\rho\varepsilon}\right)$ by (5.40). This coincides with the result found in (3.22). Second, for the jump-adapted method we add to the T/h steps the expected number of jumps, i.e. $\mathbb{E}[N(T)] = \lambda T$. It follows that the computational cost in this case is specified by

$$\operatorname{Cost}\left(\tilde{Y}\right) = N\frac{T}{h}\,(s+m)\,(1+\lambda T) = \mathcal{O}\left(\left(\varepsilon^{-5/2}\sqrt{\rho}+\varepsilon^{-3}\right)(1+\lambda T)\right),$$

where we have used again that $s = \max\left(\sqrt{\frac{\rho}{c_1}h}, 2\right) = \mathcal{O}\left(\sqrt{\rho\varepsilon}\right)$ by (5.40). Hence, by using the jump-adapted instead of the regular S-ROCK1-JD method the term $(1+\lambda T)$ appears.

5.5.2 Stabilized Multilevel Monte Carlo Method for Jump-Diffusions

We introduce now the stabilized multilevel Monte Carlo method for jump-diffusions. Consider the following sequence of nested time stepsizes

$$h_l = \frac{T}{M^l}, \quad l = 0, 1, \ldots, L,$$

where T is some fixed time endpoint, M the refinement factor and L the total number of levels. Furthermore, let

$$P_l = f\left(S_{M^l}\right) \approx f\left(S(T)\right)$$

with S_{M^l} the approximation of $S(T)$ using S-ROCK1-JD with s_l stages and time stepsize h_l. The stabilized MLMC estimator for jump-diffusions is defined by

$$Y^* := \sum_{l=0}^{L} Y_l^* \ \text{ with } \ Y_l^* = \frac{1}{N_l}\sum_{i=1}^{N_l}\left(P_l^{(i)} - P_{l-1}^{(i)}\right)$$

with $P_{-1} \equiv 0$ and where the $P_l^{(i)}$ are independent and $P_l^{(i)}$ and $P_{l-1}^{(i)}$ are based on the same sample path. Using the weak order of convergence 1 of the S-ROCK1-JD method, we obtain for the bias of the estimator Y^*

$$\operatorname{bias}\left(Y^*\right) = \mathcal{O}\left(h_L\right). \tag{5.42}$$

To get this result we can proceed similar to Section 3.3.2. By taking into account the strong order of convergence 1/2 of the S-ROCK1-JD scheme and the Lipschitz continuity of f we can show that (see Section 3.3.2)

$$\operatorname{Var}\left(Y^*\right) = C\sum_{l=0}^{L}\frac{M^{-l}}{N_l}. \tag{5.43}$$

We suppose now that a mean square accuracy of $\operatorname{MSE}(Y^*) = \mathcal{O}\left(\varepsilon^2\right)$ with $\varepsilon = M^{-L}$ is desired.

Then we obtain for the bias $\text{bias}(Y^*) = \mathcal{O}(\varepsilon)$, where we used (5.42). Inspired by (5.43), we put

$$N_l = M^{2L} M^{-l} L,$$

and thus, for the variance it holds $\text{Var}(Y^*) = \mathcal{O}\left(\varepsilon^2\right)$. Therefore, the mean square error is as desired $\text{MSE}(Y^*) = \mathcal{O}\left(\varepsilon^2\right)$.

We compute now the corresponding computational cost. We show the result for the jump-adapted S-ROCK1-JD method. The computational cost is given by

$$\begin{aligned}
\text{Cost}(Y^*) &= \sum_{l=0}^{L} N_l M^l (s_l + m)(1 + \lambda T) \\
&= (1 + \lambda T) \sum_{l=0}^{L} M^{2L} M^{-l} L M^l (s_l + m) \\
&= (1 + \lambda T) M^{2L} L \left(\sum_{l=0}^{L} s_l + m(L+1) \right) \\
&= (1 + \lambda T) M^{2L} L \left(\sqrt{\tfrac{\rho}{c_1}} \sum_{l=0}^{L} M^{-l/2} + m(L+1) \right) \\
&= (1 + \lambda T) M^{2L} L \left(\sqrt{\tfrac{\rho}{c_1}} \tfrac{\sqrt{M} - M^{-L/2}}{\sqrt{M} - 1} + m(L+1) \right) \\
&= \mathcal{O}\left(\varepsilon^{-2} (\log \varepsilon)^2 \left(\tfrac{\sqrt{\rho}}{|\log \varepsilon|} + 1 \right) (1 + \lambda T) \right),
\end{aligned} \tag{5.44}$$

where we have used (5.40) to determine the number of stages s_l. Note that this result corresponds to the one for diffusion processes (3.21) except there is the additional term $(1 + \lambda T)$, which results from the jumps. The computational cost for the regular S-ROCK1-JD approach can be obtained by the same procedure. In the regular case there is no $(1 + \lambda T)$ term though, and thus, to achieve a mean square accuracy of $\text{MSE}(Y^*) = \mathcal{O}\left(\varepsilon^2\right)$ one requires a computational cost of

$$\text{Cost}\left(Y^*\right) = \mathcal{O}\left(\varepsilon^{-2} (\log \varepsilon)^2 \left(\frac{\sqrt{\rho}}{|\log \varepsilon|} + 1 \right) \right).$$

Remark 5.5.1. *Similar to Section 3.3.1, due to the time stepsize restriction, which results from stability issues of stiff SDEs, the multilevel Monte Carlo method for jump-diffusions based on the Euler-Maruyama integrator cannot exploit all levels, and thus, the MLMC approach becomes less efficient and the corresponding computational cost increases. In contrast, the suggested stabilized MLMC method for jump-diffusions can use all the levels by adapting the stage number accordingly and therefore preserves the speeding-up feature of the multilevel Monte Carlo procedure.*

Furthermore, note that by setting the jump-related terms equal to zero, we recover the results from Chapter 3, and thus, the suggested stabilized MLMC method for jump-diffusion processes is a natural extension of the stabilized MLMC approach for diffusions. In addition, by setting the terms corresponding to the stiffness equal to zero we rediscover the results from the previous sections of this chapter. Hence, the new stabilized multilevel Monte Carlo method for jump-

diffusions is also a natural extension of the MLMC approach for jump-diffusions.

5.6 Conclusion

In this chapter we have extended the multilevel Monte Carlo method to multi-dimensional stochastic differential equations driven by jump-diffusions. We have stated and proven a complexity theorem for estimating the expectation of functionals depending on d-dimensional SDEs driven by an m-dimensional Wiener process and an r-dimensional compound Poisson process. Numerical experiments have been carried out to compare the MLMC method to the Monte Carlo method without any variance reduction technique as well as with two variance reduction techniques, the antithetic variates and the control variates. The numerical results confirm our theoretical findings and show for a sufficiently small mean square accuracy a significant reduction of the computational complexity of the MLMC method compared to the other methods. We have further extended the MLMC method by suggesting a stabilized multilevel Monte Carlo method for stiff stochastic differential equations driven by jump-diffusion processes.

6 S-ROCK Method with Variable Time Stepping for Stiff Stochastic Differential Equations

In this chapter we introduce a variable time stepping algorithm that uses the S-ROCK method as numerical integrator to approximate the weak solution of stiff stochastic differential equations. A computable leading term of the error resulting from the time discretization is derived here. Furthermore, an algorithm is presented that adapts the time grid and the number of stages of the S-ROCK method per time step at the same time. Realizing two numerical experiments we show that the algorithm is suitable to deal with stiffness in the underlying stochastic model, whereas approaches based on classical integrators such as the Euler-Maruyama method struggle in the presence of stiffness due to mean square stability issues.

6.1 Introduction

There are various approaches that offer variable stepsize solutions for stochastic differential equations. We mention here a few of them. Gaines and Lyons describe in [43] a variable time stepping algorithm for pathwise solutions to SDEs, i.e. for strong solutions. They also discuss Brownian trees and the approximation of Lévy areas. Burrage and Burrage in [29] present a variable stepsize implementation for strong solutions to SDEs using an embedding strategy. Ilie and Teslya propose another adaptive time grid approach to approximate strong solutions of the chemical Langevin equation (see [63]). They use the Milstein scheme as numerical integrator. Römisch and Winkler present a variable time stepping method for strong approximations of stochastic differential equations with small noise that is based on the mean square of the pth mean of the local error. As numerical integrators they have considered the drift-implicit Euler method and the drift-implicit Milstein scheme. Valinejad and Hosseini in [99, 100] suggest a variable time stepping algorithm for weak solutions to stochastic differential equations and they also provide an algorithm for SDEs with small noise. Küpper, Lehn and Rößler introduce in [33, 90] an adaptive time stepping algorithm for the weak solution of stochastic differential equations that is based on embedding.

A lot of research on variable time stepping methods has also been carried out by Szepessy et al. (see for instance [96, 82, 38, 83]). In [96] Szepessy et al. suggest a variable time stepping algorithm for the approximation of the expectation of functionals depending on stochastic processes. Their approach is based on the Euler-Maruyama method. In the following we describe an adaptive algorithm for approximating weak solutions of SDEs using the S-ROCK method as numerical integrator. The approach that we present in this chapter can readily handle models defined by stochastic differential equations with multiple scales. The Euler-Maruyama method can face in such a setting severe time stepsize restrictions due to mean square stability issues. We call such stochastic problems stiff. The S-ROCK methods are explicit integrators that use orthogonal Chebyshev polynomials and that have an extended stability domain which is very useful if one is to solve stiff stochastic models. We introduce here an algorithm that adapts the time grid and at the same time adjusts the number of stages of the S-ROCK method to account for the stiffness of the problem.

This chapter is organized as follows. First, we specify which kind of stochastic processes we consider here and which numerical integrators are used. We briefly recall the concept of mean square stability and define the stability domains of the numerical schemes. Next, we study an algorithm that generates a time grid with variable stepsize for the S-ROCK method to approximate weak solutions of the stochastic problem. In particular we derive a computable leading term to estimate the time discretization error. After describing the adaptive algorithm in detail we finally carry out some numerical experiments to corroborate the theoretical findings.

The following is part of a scientific paper that is in preparation [7].

6.2 Preliminaries

In this chapter we consider stochastic processes $(X(t))_{t\in[0,T]}$ that are defined on a bounded time interval $[0,T]$ and that are characterized by a stochastic differential equation

$$\left\{\begin{array}{lcl} \mathrm{d}X(t) & = & f(t,X(t))\mathrm{d}t+\sum_{r=1}^{m} g^r(t,X(t))\mathrm{d}W^r(t), \quad 0\le t\le T, \\ X(0) & = & X_0, \end{array}\right. \tag{6.1}$$

where $X(t)\in\mathbb{R}^d$, $f:[0,T]\times\mathbb{R}^d\to\mathbb{R}^d$ defines the drift, $g^r:[0,T]\times\mathbb{R}^d\to\mathbb{R}^d$ specify the diffusion terms and $(W^r(t))_{t\in[0,T]}$ are independent one-dimensional standard Brownian motions (with $r=1,2,\dots,m$). To guarantee that a strong solution to the SDE (6.1) exists, we assume standard Lipschitz continuity and linear growth conditions on the drift and the diffusion functions (see for instance [17, 66, 81]).

6.2.1 Numerical Schemes

As a numerical integrator we consider in this chapter two different schemes, the Euler-Maruyama method and the S-ROCK1 method. Both these schemes have been properly introduced in Section 2.2 and we recall them here briefly. Without loss of generality we define here the numerical schemes for autonomous SDEs of (6.1). A simply transformation can be used to pass from a non-autonomous system to an autonomous one (see Remark 2.1.3). Let

$$\tau_N := \{t_0, t_1, \dots, t_N\} \tag{6.2}$$

be a time grid of $[0, T]$ with $t_0 = 0$ and $t_N = T$.

Euler-Maruyama Method

The first numerical integrator that we use in this chapter is the well-known Euler-Maruyama method, which we have defined in Definition 2.8. The Euler-Maruyama method based on the time grid (6.2) is specified by

$$\overline{X}_{n+1} = \overline{X}_n + f\left(\overline{X}_n\right)\Delta t_n + \sum_{r=1}^{m} g^r\left(\overline{X}_n\right)\Delta W_n^r,$$

where $\Delta t_n = t_{n+1} - t_n$ and $\Delta W_n^r = W^r(t_{n+1}) - W^r(t_n)$.

S-ROCK1 Method

The second numerical scheme that we take into account here is the so-called S-ROCK1 method that we have introduced in Definition 2.2.8. This numerical scheme is the S-ROCK method with weak order of convergence 1 and strong order of convergence 1/2 (see [12]). Considering the time grid (6.2) the S-ROCK1 method with s stages ($s \geq 2$) is defined by

$$\begin{aligned}
K_0 &= \overline{X}_n \\
K_1 &= \overline{X}_n + \Delta t_n \tfrac{\omega_1}{\omega_0} f(K_0) \\
K_i &= 2\Delta t_n \omega_1 \tfrac{T_{i-1}(\omega_0)}{T_i(\omega_0)} f(K_{i-1}) + 2\omega_0 \tfrac{T_{i-1}(\omega_0)}{T_i(\omega_0)} K_{i-1} - \tfrac{T_{i-2}(\omega_0)}{T_i(\omega_0)} K_{i-2}, \quad i = 2, 3, \dots, s, \\
\overline{X}_{n+1} &= K_s + \sum_{r=1}^{m} g^r(K_{s-1})\Delta W_n^r,
\end{aligned} \tag{6.3}$$

where $\omega_0 = 1 + \frac{\eta}{s^2}$, $\omega_1 = \frac{T_s(\omega_0)}{T_s'(\omega_0)}$, $\Delta t_n = t_{n+1} - t_n$ and $\Delta W_n^r = W_r(\tau_{n+1}) - W_r(\tau_n)$. We recall that $(T_i(x))_{i\geq 0}$ represent the orthogonal Chebyshev polynomials, which can recursively be computed by

$$T_0(x) = 1, \;\; T_1(x) = x, \;\; T_i(x) = 2xT_{i-1}(x) - T_{i-2}(x) \text{ for } i \geq 2, \;\; x \in \mathbb{R}.$$

We also recall that the parameter η is the so-called damping of the S-ROCK1 method and it can be used to adjust the stability domain of the numerical method in the vertical direction. If there is no noise, the S-ROCK1 method coincides with the Chebyshev method discussed in

[101]. For simplicity we call the S-ROCK1 method in the following S-ROCK method.

Due to stability issues stiffness can lead to severe time stepsize restrictions of some classical explicit integrators such as the Euler-Maruyama method. S-ROCK methods are explicit integrators with an extended stability domain, and thus, they can handle stiff stochastic differential equations very well (see e.g. [12, 10, 15, 14]). In the next section we briefly recall the concept of mean square stability and we illustrate, why the S-ROCK method is powerful when it comes to stiff problems.

6.2.2 Mean Square Stability

Here, we recall the notion of mean square stability. For more details we refer to Section 2.2.2. A stochastic process $(X(t))_{t\geq 0}$ is called mean square stable if

$$\lim_{t\to\infty} \mathbb{E}\left[X(t)^2\right] = 0.$$

To study the stability of a numerical method we consider the linear test problem

$$\begin{cases} \mathrm{d}X(t) = \mu X(t)\mathrm{d}t + \sigma X(t)\mathrm{d}W(t), \quad 0 < t \leq T, \\ \\ X(0) = 1, \end{cases} \tag{6.4}$$

where $X(t) \in \mathbb{R}$, μ is the drift coefficient and σ the diffusion coefficient. The initial condition is given by $X(0) = 1$. The stochastic process $(W(t))_{t\in[0,T]}$ is a one-dimensional standard Brownian motion. The exact solution to (6.4) is given by

$$X(t) = \exp\left\{\left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W(t)\right\}$$

(see Proposition 2.1.5). It can be shown that the stability domain of the test problem (6.4) is given by

$$\mathcal{S}_{exact} := \left\{(\mu,\sigma) \in \mathbb{C}^2 \mid \mathcal{R}\{\mu\} + \frac{1}{2}|\sigma|^2 < 0\right\}$$

(see (2.7)).

A numerical integrator $\left(\overline{X}_n\right)_{n\geq 0}$ is said to be mean square stable if

$$\lim_{n\to\infty} \mathbb{E}\left[\overline{X}_n^2\right] = 0.$$

The mean square stability domain of the Euler-Maruyama method is given by

$$\mathcal{S}_{EM} := \left\{(p,q) \in \mathbb{C}^2 \mid |1+p|^2 + q^2 < 1\right\},$$

where $(p,q) = (h\mu, \sqrt{h}|\sigma|)$. Assuming the parameters μ and σ to be real-valued, for the test problem it can be established that the time stepsize Δt_n of the Euler-Maruyama method has to be chosen such that

$$\rho_{EM}\Delta t_n < 1 \quad \text{with} \quad \rho_{EM} := \frac{|\mu|^2}{2|\mu| - |\sigma|^2}, \tag{6.5}$$

where ρ_{EM} represents the stiffness parameter of the Euler-Maruyama scheme. For more general problems we adjust the stiffness parameter ρ_{EM} accordingly.

To define the stability domain of S-ROCK methods, we need to introduce the notion of a portion of the true stability domain first:

$$\mathscr{S}_{SDE,a} = \left\{(p,q) \in [-a,0] \times \mathbb{R} \mid |q| \le \sqrt{-2p}\right\}.$$

In addition, we define a parameter

$$a^* = \sup\left\{a > 0 \mid \mathscr{S}_{SDE,a} \subset \mathscr{S}_{num}\right\}$$

with $\mathscr{S}_{num}$ indicating the stability domain of the corresponding numerical integrator. It can be shown that S-ROCK methods have large parameters a^* and that they grow quadratically with the stage number s, i.e. usually there is some constant c_{SR} depending solely on s that can be estimated numerically (see [12]). The number of function evaluations grows only linearly with the stage number s. That makes S-ROCK methods so powerful.

Similar to (6.5), one can set a stability criterion up for the S-ROCK method. In fact, we consider here

$$\frac{\rho_{SR}\Delta t_n}{c_{SR} s_n^2} < 1, \tag{6.6}$$

where Δt_n is the time stepsize, s_n the corresponding stage number, c_{SR} some constant depending on s_n (that can be estimated numerically and lies between 0.33 and 1.01, see [12]) and where ρ_{SR} is the stiffness parameter, which for the test problem is equal to $|\mu|$.

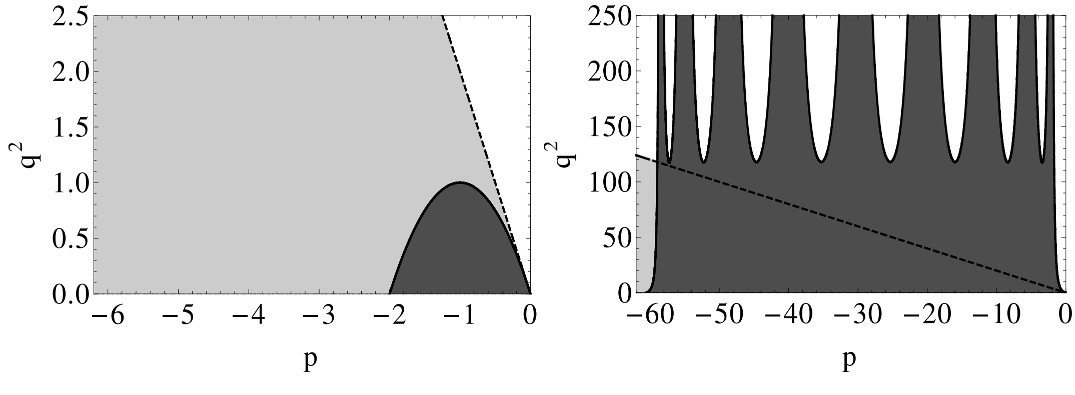

Figure 6.1: Comparison of the stability domains (dark gray) of the Euler-Maruyama scheme (left-hand side) and the S-ROCK scheme with $s = 10$ stages and damping $\eta = 5.9$ (right-hand side). The stability domain (light gray) of the linear test problem is delimited by the dashed line.

In Figure 6.1 we illustrate the stability domain of the Euler-Maruyama method and the S-ROCK method with $s = 10$ stages and a damping of $\eta = 5.9$. The true stability domain of the test problem is indicated by the area beneath the dashed line. To avoid stability issues, a numerical integrator should cover as much as possible of the true stability domain. One can observe that the S-ROCK scheme covers significantly more of the true stability domain than the Euler-Maruyama method, which only covers a small part of it. That is why the Euler-Maruyama method can face severe time stepsize restrictions due to stability. In contrast, the stability domain of the S-ROCK method can be enlarged by increasing the stage number s, and thus, the S-ROCK methods can avoid time stepsize restrictions by choosing the stage number accordingly.

6.3 Variable Time Stepping S-ROCK Method Using A Posteriori Error Control

In the following we introduce an adaptive S-ROCK method which is based on a posteriori error estimates to get weak approximations of stochastic differential equations. This approach is inspired by the deterministic time stepping algorithm of Szepessy, Tempone and Zouraris in [96], which uses Euler-Maruyama as numerical scheme.

Let $(X(t))_{t\in[0,T]}$ be a stochastic process defined by (6.1). Let $\phi : \mathbb{R}^d \to \mathbb{R}$ be some functional. The goal is to estimate the expectation of the functional ϕ depending on the stochastic process $(X(t))_{t\in[0,T]}$, i.e. we aim to approximate

$$E := \mathbb{E}\left[\phi(X(T))\right] \tag{6.7}$$

using Monte Carlo simulations and the S-ROCK method (6.3) based on variable time steps. The estimator of E, that we consider here, is given by

$$\widehat{E} := \frac{1}{M}\sum_{j=1}^{M} \phi\left(\overline{X}_N^j\right) \tag{6.8}$$

a sample average over M independent identically distributed samples of $\overline{X}_N \approx X(T)$, the numerical S-ROCK approximation based on a time grid, that is not necessarily uniform. We aim to bound the error by some given tolerance, denoted by TOL, i.e.

$$E - \widehat{E} \leq \text{TOL}.$$

Observe that this error can be decomposed as

$$\begin{aligned} E-\widehat{E} &= \mathbb{E}\left[\phi(X(T))\right]-\tfrac{1}{M}\sum_{j=1}^{M}\phi\left(\overline{X}_N^j\right) \\ &= \left(\mathbb{E}\left[\phi(X(T))\right]-\mathbb{E}\left[\phi(\overline{X}_N)\right]\right)+\left(\mathbb{E}\left[\phi(\overline{X}_N)\right]-\tfrac{1}{M}\sum_{j=1}^{M}\phi\left(\overline{X}_N^j\right)\right) \\ &= \mathrm{err}_T+\mathrm{err}_S \end{aligned} \tag{6.9}$$

with

$$\mathrm{err}_T := \left(\mathbb{E}\left[\phi(X(T))\right]-\mathbb{E}\left[\phi(\overline{X}_N)\right]\right)$$

and

$$\mathrm{err}_S := \left(\mathbb{E}\left[\phi(\overline{X}_N)\right]-\frac{1}{M}\sum_{j=1}^{M}\phi\left(\overline{X}_N^j\right)\right),$$

i.e. the error resulting from the time discretization is denoted by err_T and the statistical error due to approximating the expectation by a sample average by err_S. The discretization error component will be used to refine the time grid and the statistical error one to adjust the number of simulations M.

6.3.1 Derivation of a Computable Leading Term of the Time Error

In this section we derive analytically a computable leading term of the time discretization error err_T. Before we state the theorem which suggests an a posterior error estimate for a variable time stepping S-ROCK method, we introduce some notation. To ease this notation, we apply in what follows the summation convention, i.e. as soon as we have an index appearing twice in a term, the sum over this index is used. For instance, if we write

$$f_k(t_1,X(t_1))f_k(t_2,X(t_2))$$

this can be interpreted as

$$\sum_{k=1}^{d} f_k(t_1,X(t_1))f_k(t_2,X(t_2)).$$

Furthermore, observe that the S-ROCK method with s stages (6.3) can be reformulated as

$$\begin{aligned} \overline{X}_{n+1} &= \overline{X}_n+\tfrac{K_s-\overline{X}_n}{\Delta t_n}\Delta t_n+\sum_{r=1}^{m} g^r(K_{s-1})\Delta W_n^r \\ &=: \overline{X}_n+\alpha(t_n,\overline{X}_n)\Delta t_n+\sum_{r=1}^{m}\beta^r(t_n,\overline{X}_n)\Delta W_n^r. \end{aligned} \tag{6.10}$$

Then for each $k \in \{1,2,\dots,d\}$ we define the piecewise constant functions

$$\begin{aligned}\overline{\alpha}_k(t,\overline{X}) &:= \alpha_k(t_n,\overline{X}_n) \text{ and}\\ \overline{\beta}^r_k(t,\overline{X}) &:= \beta^r_k(t_n,\overline{X}_n)\end{aligned}$$

for all $t \in [t_n, t_{n+1}[$, $n = 0,1,\dots,N-1$ and $r = 1,2,\dots,m$.

Moreover, we define a function φ and its first variation φ' by a dual backward problem. First, let

$$c_i(t_n,x) := x_i + \Delta t_n \alpha_i(t_n,x) + \sum_{r=1}^{m} \Delta W^r_n \beta^r_i(t_n,x)$$

with the functions α and β as defined above. The function φ is characterized by

$$\begin{cases}\varphi_i(t_n) &= \frac{\partial}{\partial x_i} c_j\left(t_n,\overline{X}_n\right)\varphi_j(t_{n+1}), \ \ t_n < T,\\ \varphi_i(T) &= \frac{\partial}{\partial x_i}\phi\left(\overline{X}_N\right)\end{cases} \tag{6.11}$$

and its first variation

$$\varphi'_{ik}(t_n) = \partial_{x_k(t_n)}\varphi_i(t_n) := \frac{\partial \varphi_i}{\partial x_k}\left(t_n,\overline{X}_n = x\right)$$

satisfies

$$\begin{cases}\varphi'_{ik}(t_n) &= \partial_i c_j\left(t_n,\overline{X}_n\right)\partial_k c_p\left(t_n,\overline{X}_n\right)\varphi'_{jp}(t_{n+1})\\ &\quad +\partial_{ik} c_j\left(t_n,\overline{X}_n\right)\varphi_j(t_{n+1}), \quad t_n < T,\\ \varphi'_{ik}(T) &= \partial_{ik}\phi\left(\overline{X}_N\right).\end{cases} \tag{6.12}$$

We have now everything at hand to state the theorem that gives us a computable leading term of the time error.

Theorem 6.3.1. *Let $(X(t))_{t\in[0,T]}$ be the stochastic process defined in* (6.1) *and let $\overline{X}_n$ be its S-ROCK approximation* (6.3) *at $t = t_n$. Suppose that for $m_0 > \left\lceil \frac{d}{2} \right\rceil + 10$ there exist two positive constants $C_1 \in \mathbb{N}$ and $C_2 \in \mathbb{R}$ such that*

(i) $\phi \in C^{m_0}_{loc}\left(\mathbb{R}^d\right)$ *with* $\left|\partial_\xi \phi(x)\right| \le C_2\left(1+|x|^{C_1}\right)$ $\forall |\xi| \le m_0$ *(all derivatives up to order ξ have polynomial growth);*

(ii) $\mathbb{E}\left[|X(0)|^{2C_1+d+1} + |X_0|^{2C_1+d+1}\right] \le C_2$*;*

(iii) *f and g^r are bounded in $C^{m_0}\left([0,T]\times\mathbb{R}^d\right)$;*

(iv) *$X(0)$ and X_0 have the same distribution.*

Then the time discretization error satisfies

$$\begin{aligned}
&\mathbb{E}\left[\phi(X(T)) - \phi\left(\overline{X}_N\right)\right] \\
= \; & \sum_{n=0}^{N-1} \frac{1}{M} \sum_{j=1}^{M} \left[\left(f_k\left(t_{n+1}, \overline{X}_{n+1}(\omega_j)\right) - \overline{\alpha}_k\left(t_n, \overline{X}(\omega_j)\right)\right)\varphi_k(t_{n+1}, \omega_j)\right. \\
& \left. + \left(f_k\left(t_n, \overline{X}_n(\omega_j)\right) - \overline{\alpha}_k\left(t_n, \overline{X}(\omega_j)\right)\right)\varphi_k(t_n, \omega_j)\right] \tfrac{\Delta t_n}{2} \\
+ \; & \sum_{n=0}^{N-1} \sum_{r=1}^{m} \frac{1}{2} \frac{1}{M} \sum_{j=1}^{M} \left[\left(\left(g_k^r g_\ell^r\right)\left(t_{n+1}, \overline{X}_{n+1}(\omega_j)\right) - \left(\overline{\beta}_k^r \overline{\beta}_\ell^r\right)\left(t_n, \overline{X}(\omega_j)\right)\right)\varphi'_{kl}(t_{n+1}, \omega_j)\right. \\
& \left. + \left(\left(g_k^r g_\ell^r\right)\left(t_n, \overline{X}_n(\omega_j)\right) - \left(\overline{\beta}_k^r \overline{\beta}_\ell^r\right)\left(t_n, \overline{X}(\omega_j)\right)\right)\varphi'_{kl}(t_n, \omega_j)\right] \tfrac{\Delta t_n}{2} \\
+ \; & \sum_{n=0}^{N-1} \Delta t_n^2 \mathcal{O}\left(\Delta t_n + \sum_{m=n+1}^{N-1} \Delta t_m^2 + \frac{\Delta t_{n-1}}{\Delta t_n} \sum_{m=n}^{N-1} \Delta t_m^2\right),
\end{aligned}$$

where we neglect the statistical error, which depends on the number of simulations M.

The theorem leads to a proposition that we can use in the following to define the algorithm.

Proposition 6.3.2. *Suppose the assumptions (i)-(iv) from Theorem 6.3.1 hold. Define the piecewise constant function ρ as follows:*

$$\rho(t) := \rho_n \;\; \forall t \in [t_n, t_{n+1}[\;\; \textit{and} \;\; n = 0, 1, \ldots, N-1$$

with

$$\begin{aligned}
\rho_n \; = \; & \left[\left(f_k\left(t_{n+1}, \overline{X}_{n+1}\right) - \overline{\alpha}_k\left(t_n, \overline{X}\right)\right)\varphi_k(t_{n+1})\right. \\
+ \; & \left.\left(f_k\left(t_n, \overline{X}_n\right) - \overline{\alpha}_k\left(t_n, \overline{X}\right)\right)\varphi_k(t_n)\right] \tfrac{1}{2\Delta t_n} \\
+ \; & \sum_{r=1}^{m} \frac{1}{2} \left[\left(g_k^r\left(t_{n+1}, \overline{X}_{n+1}\right) g_\ell^r\left(t_{n+1}, \overline{X}_{n+1}\right) - \overline{\beta}_k^r\left(t_n, \overline{X}\right)\overline{\beta}_\ell^r\left(t_n, \overline{X}\right)\right)\varphi'_{kl}(t_{n+1})\right. \\
+ \; & \left.\left(g_k^r\left(t_n, \overline{X}_n\right) g_\ell^r\left(t_n, \overline{X}_n\right) - \overline{\beta}_k^r\left(t_n, \overline{X}\right)\overline{\beta}_\ell^r\left(t_n, \overline{X}\right)\right)\varphi'_{kl}(t_n)\right] \tfrac{1}{2\Delta t_n}.
\end{aligned}$$

Then for the time discretization error (as in the previous theorem, neglecting the statistical error depending on M)

$$\mathbb{E}\left[\phi(X(T)) - \phi\left(\overline{X}_N\right)\right] = \frac{1}{M} \sum_{j=1}^{M} \sum_{n=0}^{N-1} \rho_n(\omega_j)\Delta t_n^2 + \mathcal{O}\left(\Delta t_n^3\right)$$

holds and

$$\frac{1}{M} \sum_{j=1}^{M} \sum_{n=0}^{N-1} \rho_n(\omega_j)\Delta t_n^2$$

can be used as a posteriori error estimate.

Proof of Proposition 6.3.2. Follows immediately from Theorem 6.3.1. □

To prove Theorem 6.3.1 four lemmas are required. The first lemma gives a description of the time discretization error by some function $u(t,x)$ (to be specified later). The second lemma quantifies the quadrature error obtained by approximating integrals. In the third lemma the function $u(t,x)$ is replaced by some computable function $\overline{u}(t,x)$ (for definition see below) and the resulting error is quantified. Finally the last lemma shows how the derivatives of the function $\overline{u}$ can be expressed by dual functions.

Before we state the first lemma, observe that using (6.10) the S-ROCK method can be written (for theoretical purposes only) as

$$\overline{X}(t)-\overline{X}(t_n)=\int_{t_n}^{t}\overline{\alpha}\left(\tau,\overline{X}\right)\mathrm{d}\tau+\int_{t_n}^{t}\sum_{r=1}^{m}\overline{\beta}^{r}(\tau,\overline{X})\mathrm{d}W_{\tau}^{r} \tag{6.13}$$

with $\overline{\alpha}\left(\tau,\overline{X}\right)$ and $\overline{\beta}\left(\tau,\overline{X}\right)$ as defined above.

Lemma 6.3.3. *Let us assume that conditions (i)-(iv) of Theorem 6.3.1 hold. Then one can show that*

$$\begin{aligned}
&\mathbb{E}\left[\phi(X(T))-\phi(\overline{X}(T))\right]\\
=\;&\int_0^T\mathbb{E}\left[\sum_{k=1}^{d}\left(f_k\left(t,\overline{X}(t)\right)-\overline{\alpha}_k\left(t,\overline{X}\right)\right)\partial_k u\left(t,\overline{X}(t)\right)\right]\mathrm{d}t\\
+\;&\sum_{r=1}^{m}\frac{1}{2}\int_0^T\mathbb{E}\left[\sum_{k,\ell=1}^{d}\left(g_k^r\left(t,\overline{X}(t)\right)g_\ell^r\left(t,\overline{X}(t)\right)-\overline{\beta}_k^r\left(t,\overline{X}\right)\overline{\beta}_\ell^r\left(t,\overline{X}\right)\right)\partial_{k\ell}u\left(t,\overline{X}(t)\right)\right]\mathrm{d}t
\end{aligned}$$

with $u(t,x)=\mathbb{E}\left[\phi(X(T))\,|X(t)=x\right]$.

Proof of Lemma 6.3.3. Similar to the proof of Lemma 2.1. in [96], using the Kolmogorov Backward Equation, the Feynman-Kac formula with zero potential, applying Itô's formula to (6.13) and using some basic properties from stochastic calculus leads to the result. □

Lemma 6.3.4. *Suppose that the assumptions (i)-(iv) of Theorem 6.3.1 hold. For the quadrature error the following equalities can be established:*

$$\begin{aligned}
&\int_{t_n}^{t_{n+1}}\mathbb{E}\left[\left(f_k\left(t,\overline{X}(t)\right)-\overline{\alpha}_k\left(t,\overline{X}\right)\right)\partial_k u\left(t,\overline{X}(t)\right)\right]\mathrm{d}t\\
=\;&\tfrac{\Delta t_n}{2}\mathbb{E}\left[\left(f_k\left(t_{n+1},\overline{X}(t_{n+1})\right)-\overline{\alpha}_k\left(t_n,\overline{X}\right)\right)\partial_k u\left(t_{n+1},\overline{X}(t_{n+1})\right)\right.\\
&\left.+\left(f_k\left(t_n,\overline{X}(t_n)\right)-\overline{\alpha}_k\left(t_n,\overline{X}\right)\right)\partial_k u\left(t_n,\overline{X}(t_n)\right)\right]\\
&+\mathcal{O}\left(\Delta t_n^3\right)
\end{aligned} \tag{6.14}$$

and

$$\begin{aligned}
&\int_{t_n}^{t_{n+1}} \mathbb{E}\left[\frac{1}{2}\left(g_k^r\left(t,\overline{X}(t)\right)g_\ell^r\left(t,\overline{X}(t)\right)-\overline{\beta}_k^r\left(t,\overline{X}\right)\overline{\beta}_\ell^r\left(t,\overline{X}\right)\right)\partial_{k\ell}u\left(t,\overline{X}(t)\right)\right]\mathrm{d}t \\
= \quad & \tfrac{\Delta t_n}{2}\mathbb{E}\Big[\tfrac{1}{2}\left(\left(g_k^r g_\ell^r\right)\left(t_{n+1},\overline{X}\left(t_{n+1}\right)\right)-\left(\overline{\beta}_k^r\overline{\beta}_\ell^r\right)\left(t_n,\overline{X}\right)\right)\partial_{k\ell}u\left(t_{n+1},\overline{X}\left(t_{n+1}\right)\right) \\
& +\tfrac{1}{2}\left(g_k^r\left(t_n,\overline{X}\left(t_n\right)\right)g_\ell^r\left(t_n,\overline{X}\left(t_n\right)\right)-\overline{\beta}_k^r\left(t_n,\overline{X}\right)\overline{\beta}_\ell^r\left(t_n,\overline{X}\right)\right)\partial_{k\ell}u\left(t_n,\overline{X}\left(t_n\right)\right)\Big] \\
& +\mathcal{O}\left(\Delta t_n^3\right).
\end{aligned} \tag{6.15}$$

Proof of Lemma 6.3.4. This proof follows the idea of the proof of Lemma 2.3. in [96]. Let $\gamma\left(t,\overline{X}(t)\right) := \left(f_k\left(t,\overline{X}(t)\right)-\overline{\alpha}_k\left(t,\overline{X}\right)\right)\partial_k u\left(t,\overline{X}(t)\right)$ and

$$\begin{aligned}
& \overline{\gamma}(t) \\
:= \quad & \gamma\left(t_n,\overline{X}\left(t_n\right)\right) \\
+ \quad & \tfrac{t-t_n}{\Delta t_n}\left(\left(f_k\left(t_{n+1},\overline{X}\left(t_{n+1}\right)\right)-\overline{\alpha}_k\left(t_n,\overline{X}\right)\right)\partial_k u\left(t_{n+1},\overline{X}\left(t_{n+1}\right)\right)-\gamma\left(t_n,\overline{X}\left(t_n\right)\right)\right).
\end{aligned}$$

Using the definitions of γ and $\overline{\gamma}$ it is straightforward to show that

$$\begin{aligned}
& \int_{t_n}^{t_{n+1}}\left(\mathbb{E}\left[\gamma\left(t,\overline{X}(t)\right)\right]-\mathbb{E}\left[\overline{\gamma}(t)\right]\right)\mathrm{d}t \\
= \quad & \int_{t_n}^{t_{n+1}}\mathbb{E}\left[\gamma\left(t,\overline{X}(t)\right)\right]\mathrm{d}t-\frac{\Delta t_n}{2}\mathbb{E}\Big[\left(f_k\left(t_{n+1},\overline{X}(t_{n+1})\right)-\overline{\alpha}_k\left(t_n,\overline{X}\right)\right) \\
& \cdot\partial_k u\left(t_{n+1},\overline{X}(t_{n+1}\right)+\left(f_k\left(t_n,\overline{X}(t_n)\right)-\overline{\alpha}_k\left(t_n,\overline{X}\right)\right)\partial_k u\left(t_n,\overline{X}(t_n\right)\Big].
\end{aligned}$$

Since this corresponds to a linear interpolation, the interpolation error can be bounded by

$$\left|\int_{t_n}^{t_{n+1}}\left(\mathbb{E}\left[\gamma\left(t,\overline{X}(t)\right)\right]-\mathbb{E}\left[\overline{\gamma}(t)\right]\right)\mathrm{d}t\right| \le \int_{t_n}^{t_{n+1}}\frac{1}{8}\Delta t_n^2\left|\frac{d^2}{dt^2}\mathbb{E}\left[\gamma\left(t,\overline{X}(t)\right)\right]\right|\mathrm{d}t.$$

Furthermore, applying Itô's lemma twice and using assumption (iii) one obtains the desired upper bound of (6.14). The proof of (6.15) is similar. □

Before we pass on to the next lemma, we have to introduce some more notation (stochastic flow representation). Recall that $u(t,x)=\mathbb{E}\left[\phi\left(X(T)\right)|X(t)=x\right]$. By the chain rule it follows that

$$\begin{aligned}
\partial_k u(t,x) \quad &= \quad \tfrac{\partial}{\partial x_k}\mathbb{E}\left[\phi\left(X(T)\right)|X(t)=x\right] \\
&= \quad \mathbb{E}\left[\tfrac{\partial}{\partial x_i}\phi\left(X(T)\right)\tfrac{\partial X_i(T)}{\partial x_k}|\tfrac{\partial X_i(t)}{\partial x_k}=\delta_{ik},X(t)=x\right],
\end{aligned}$$

where we denote $\frac{\partial X_i(T)}{\partial x_k} =: X'_{ik}(T)$ and where δ_{ik} represents the Kronecker delta. The first variation is defined by

$$\begin{cases} \mathrm{d}X'_{ij}(\tau) &= \frac{\partial}{\partial x_k} f_i\left(\tau, X(\tau)\right) X'_{kj}(\tau)\mathrm{d}\tau + \frac{\partial}{\partial x_k} g_i^r\left(\tau, X(\tau)\right) X'_{kj}(\tau)\mathrm{d}W^r(\tau), \;\; t < \tau \leq T, \\ X_{ij}(t) &= \delta_{ij}. \end{cases}$$

Similarly one can define the second, third and fourth variation. Therefore, we can set up the following system. Let $Y := \left(X, X', X'', X''', X''''\right)^T$ and

$$\begin{cases} \mathrm{d}Y(t) &= F(t,Y)\,\mathrm{d}t + \sum_{r=1}^{m} G^r(t,Y)\,\mathrm{d}W^r(t), \;\; t > t_0, \\ Y(t_0) &= (x, I, 0, 0, 0)^T. \end{cases}$$

The S-ROCK approximation of this system is given by

$$\mathrm{d}\overline{Y} = \overline{\mathscr{A}}\left(t, \overline{Y}\right)\mathrm{d}t + \sum_{r=1}^{m} \overline{\mathscr{B}}^r\left(t, \overline{Y}\right)\mathrm{d}W^r(t)$$

with $\overline{\mathscr{A}}$ and $\overline{\mathscr{B}}$ piecewise constant functions (similar to the definition of $\overline{\alpha}$ and $\overline{\beta}$). Moreover, we introduce the function

$$\overline{u}(t,x) := \mathbb{E}\left[\phi\left(\overline{X}(T)\right) | \overline{X}(t) = x\right],$$

which is purely based on the approximated solution. In the next lemma we show how we can replace the function u by $\overline{u}$.

Lemma 6.3.5. *Let $\mathscr{F}_{t_n}$ be the σ-algebra generated by $\{W(\tau) \mid \tau \leq t_n\}$. Further, let $\Delta t(\tau) := \Delta t_n$ for all $\tau \in [t_n, t_{n+1}[$ for all n. Suppose that the assumptions of Theorem 6.3.1 hold. Replacing u by $\overline{u}$ results in the following errors*

(i) $\partial_i u\left(t, \overline{X}(t)\right) = \partial_i \overline{u}\left(t, \overline{X}(t)\right) + \int_t^T \mathscr{O}\left(\Delta t(\tau)\right)\mathrm{d}\tau$ *(the same holds for the derivatives up to order four);*

(ii)
$$\begin{aligned} &\mathbb{E}\left[\left(f_k\left(t, \overline{X}(t)\right) - \overline{\alpha}_k\left(t, \overline{X}\right)\right)\left(\partial_k u\left(t, \overline{X}(t)\right) - \partial_k \overline{u}\left(t, \overline{X}(t)\right)\right)\right] \\ =\;& \Delta t_n \textstyle\int_t^T \mathscr{O}\left(\Delta t(\tau)\right)\mathrm{d}\tau; \end{aligned}$$

(iii)
$$\begin{aligned} &\mathbb{E}\left[\tfrac{1}{2}\left(g_k^r\left(t, \overline{X}(t)\right) g_\ell^r\left(t, \overline{X}(t)\right) - \overline{\beta}_k^r\left(t, \overline{X}\right)\overline{\beta}_\ell^r\left(t, \overline{X}\right)\right)\right. \\ &\left.\cdot\left(\partial_{k\ell} u\left(t, \overline{X}(t)\right) - \partial_{k\ell}\overline{u}\left(t, \overline{X}(t)\right)\right)\right] \\ =\;& \Delta t_n \textstyle\int_t^T \mathscr{O}\left(\Delta t(\tau)\right)\mathrm{d}\tau. \end{aligned}$$

Proof of Lemma 6.3.5. By the above (stochastic flow representation) we define

$$\begin{aligned} \varphi_i(Y) &:= \partial_k \phi(X) X'_{ki} \\ \varphi_{ij}(Y) &:= \partial_k \phi(X) X''_{kij} + \partial_{kn}\phi(X) X'_{ki} X'_{nj} \end{aligned}$$

and similiarly $\varphi_{ijm}(Y)$ and $\varphi_{ijmn}(Y)$. In the following we prove the result for the derivative of order 1, the proof for the orders 2, 3 and 4 is a natural extension. Observe that

$$\begin{aligned} & \partial_i u\left(t, \overline{X}(t)\right) - \partial_i \overline{u}\left(t, \overline{X}(t)\right) \\ = \; & \mathbb{E}\left[\varphi_i\left(Y(T)\right) - \varphi_i\left(\overline{Y}(T)\right) | Y(t) = \overline{Y}(t) = (x, I, 0, 0, 0)^T\right] \\ = \; & \int_t^T \mathbb{E}\left[\Gamma\left(\tau, \overline{Y}(\tau)\right) | \mathscr{F}_t\right] \mathrm{d}\tau \end{aligned}$$

with $\Gamma\left(\tau, \overline{Y}(\tau)\right) := \left(F - \overline{\mathscr{A}}\right)_k \partial_k v^i\left(\tau, \overline{Y}(\tau)\right) + \frac{1}{2}\left(G^r_k G^r_\ell - \overline{\mathscr{B}}^r_k \overline{\mathscr{B}}^r_\ell\right) \partial_{k\ell} v^i\left(\tau, \overline{Y}(\tau)\right)$ and

$$\begin{cases} -\frac{\partial v^i}{\partial t} - F_k \partial_k v^i - \frac{1}{2} G^r_k G^r_\ell \partial_{k\ell} v^i &= 0, \;\; t < T, \\ v^i(T, \cdot) &= \varphi_i, \end{cases}$$

where we have used Lemma 6.3.3.

Let $\mathscr{L}_Y \omega\left(\tau, \overline{Y}(\tau)\right) := \left(\frac{\partial}{\partial t}\omega + \overline{\mathscr{A}}_k \partial_k \omega + \frac{1}{2}\overline{\mathscr{B}}^r_k \overline{\mathscr{B}}^r_\ell \partial_{k\ell}\omega\right)\left(\tau, \overline{Y}(\tau)\right)$ and $t_m \leq \tau \leq t_{m+1}$. By Itô's lemma one obtains

$$\begin{aligned} \mathbb{E}\left[\Gamma\left(\tau, \overline{Y}(\tau)\right) | \mathscr{F}_{t_m}\right] &= \mathbb{E}\left[\Gamma\left(t_m, \overline{Y}(t_m)\right) | \mathscr{F}_{t_m}\right] + \int_{t_m}^{\tau} \mathbb{E}\left[\mathscr{L}_Y \Gamma\left(\varepsilon, \overline{Y}(\varepsilon)\right) | \mathscr{F}_{t_m}\right] \mathrm{d}\varepsilon \\ &\leq \mathbb{E}\left[\Gamma\left(t_m, \overline{Y}(t_m)\right) | \mathscr{F}_{t_m}\right] + C\Delta t_m, \end{aligned}$$

where we have bounded $\mathbb{E}\left[\mathscr{L}_Y \Gamma\left(\varepsilon, \overline{Y}(\varepsilon)\right) | \mathscr{F}_{t_m}\right]$ by C due to the smoothness of the coefficients (drift and diffusion). The former term of the upperbound can be decomposed as

$$\begin{aligned} & \mathbb{E}\left[\Gamma\left(t_m, \overline{Y}(t_m)\right) | \mathscr{F}_{t_m}\right] \\ = \; & \mathbb{E}\left[\left(F - \overline{\mathscr{A}}\right)_k \left(t_m, \overline{Y}(t_m)\right) \partial_k v^i\left(t_m, \overline{Y}(t_m)\right) | \mathscr{F}_{t_m}\right] \\ + \; & \mathbb{E}\left[\frac{1}{2}\left(G^r_k G^r_\ell - \overline{\mathscr{B}}^r_k \overline{\mathscr{B}}^r_\ell\right)\left(t_m, \overline{Y}(t_m)\right) \partial_{k\ell} v^i\left(t_m, \overline{Y}(t_m)\right) | \mathscr{F}_{t_m}\right] \end{aligned} \tag{6.16}$$

with $v^i\left(t_m, \overline{Y}(t_m)\right) = \mathbb{E}\left[\varphi_i\left(Y(T)\right) | Y(t_m) = \overline{Y}(t_m)\right]$. The first term is of order $\mathscr{O}(\Delta t_m)$. In fact, one can show that for the S-ROCK method (since the method converges)

$$\overline{\mathscr{A}}\left(t_m, \overline{Y}(t_m)\right) = F\left(t_m, \overline{Y}(t_m)\right) \omega_1 \frac{T'_s(\omega_0)}{T_s(\omega_0)} + \mathscr{O}(\Delta t_m) = F\left(t_m, \overline{Y}(t_m)\right) + \mathscr{O}(\Delta t_m).$$

And thus,

$$\begin{aligned}
&\mathbb{E}\left[\left(F-\overline{\mathscr{A}}\right)_k\left(t_m,\overline{Y}(t_m)\right)\partial_k v^i\left(t_m,\overline{Y}(t_m)\right)|\mathscr{F}_{t_m}\right]\\
=\;&\mathbb{E}\left[\mathscr{O}(\Delta t_m)\partial_k v^i\left(t_m,\overline{Y}(t_m)\right)|\mathscr{F}_{t_m}\right]\\
=\;&\mathscr{O}(\Delta t_m)\mathbb{E}\left[\partial_k v^i\left(t_m,\overline{Y}(t_m)\right)|\mathscr{F}_{t_m}\right]\\
=\;&\mathscr{O}(\Delta t_m),
\end{aligned}$$

where we used the smoothness of the drift and diffusion coefficients. The second term of (6.16) is also of order $\mathscr{O}(\Delta t_m)$. In fact, for the S-ROCK method we have

$$\overline{\mathscr{B}}_k^r\left(t_n,\overline{Y}(t_n)\right)=G_k^r(K_{s-1})=G_k^r\left(\overline{Y}(t_n)+\mathscr{O}(\Delta t_n)\right)$$

since

$$K_{s-1}=\overline{Y}(t_n)+\Delta t_n\omega_1\frac{T'_{s-1}(\omega_0)}{T_{s-1}(\omega_0)}F\left(t_n,\overline{Y}(t_n)\right)+\mathscr{O}\left(\Delta t_n^2\right)=\overline{Y}(t_n)+\mathscr{O}(\Delta t_n).$$

By Taylor expansion we get

$$\overline{\mathscr{B}}_k^r\left(t_n,\overline{Y}(t_n)\right)=G_k^r\left(\overline{Y}(t_n)+\mathscr{O}(\Delta t_n)\right)=G_k^r\left(\overline{Y}(t_n)\right)+\mathscr{O}(\Delta t_n)$$

and hence,

$$\begin{aligned}
&G_k^r\left(t_m,\overline{Y}(t_m)\right)G_\ell^r\left(t_m,\overline{Y}(t_m)\right)-\overline{\mathscr{B}}_k^r\left(t_m,\overline{Y}(t_m)\right)\overline{\mathscr{B}}_\ell^r\left(t_m,\overline{Y}(t_m)\right)\\
=\;&G_k^r\left(t_m,\overline{Y}(t_m)\right)G_\ell^r\left(t_m,\overline{Y}(t_m)\right)\\
&-\left(G_k^r\left(\overline{Y}(t_m)\right)+\mathscr{O}(\Delta t_m)\right)\left(G_\ell^r\left(\overline{Y}(t_m)\right)+\mathscr{O}(\Delta t_m)\right)\\
=\;&G_k^r\left(t_m,\overline{Y}(t_m)\right)G_\ell^r\left(t_m,\overline{Y}(t_m)\right)-G_k^r\left(t_m,\overline{Y}(t_m)\right)G_\ell^r\left(t_m,\overline{Y}(t_m)\right)+\mathscr{O}(\Delta t_m)\\
=\;&\mathscr{O}(\Delta t_m).
\end{aligned}$$

It follows that

$$\begin{aligned}
&\mathbb{E}\left[\tfrac{1}{2}\left(G_k^rG_\ell^r-\overline{\mathscr{B}}_k^r\overline{\mathscr{B}}_\ell^r\right)\left(t_m,\overline{Y}(t_m)\right)\partial_{k\ell}v^i\left(t_m,\overline{Y}(t_m)\right)|\mathscr{F}_{t_m}\right]\\
=\;&\mathbb{E}\left[\mathscr{O}(\Delta t_m)\partial_{k\ell}v^i\left(t_m,\overline{Y}(t_m)\right)|\mathscr{F}_{t_m}\right]\\
=\;&\mathscr{O}(\Delta t_m),
\end{aligned}$$

where we used once more the smoothness of the coefficients. Therefore, we have

$$\mathbb{E}\left[\Gamma\left(t_m,\overline{Y}(t_m)\right)|\mathscr{F}_{t_m}\right]=\mathscr{O}(\Delta t_m).$$

This completes the proof of part (i) for derivatives of order 1.

To prove part (ii), we define

$$\gamma\left(t,\overline{X}(t)\right) := \left(f_k-\overline{\alpha}_k\right)\left(\partial_k u-\partial_k\overline{u}\right)\left(t,\overline{X}(t)\right).$$

For $t_n \le t \le t_{n+1}$, we have

$$\mathbb{E}\left[\gamma\left(t,\overline{X}(t)\right)\right] = \mathbb{E}\left[\gamma\left(t_n,\overline{X}\left(t_n\right)\right)\right] + \int_{t_n}^{t}\mathbb{E}\left[\mathscr{L}\gamma\left(\tau,\overline{X}\left(\tau\right)\right)\right]$$

with $\mathscr{L}\gamma\left(\tau,\overline{X}\left(\tau\right)\right) = \left(\frac{\partial}{\partial t}\gamma + f_k\partial_k\gamma + \frac{1}{2}g_k^r g_\ell^r\partial_{k\ell}\gamma\right)\left(\tau,\overline{X}\left(\tau\right)\right)$. Observe that

$$\mathscr{L}\gamma\left(\tau,\overline{X}\left(\tau\right)\right) = \gamma_1\left(\tau,\overline{X}\left(\tau\right)\right) + \gamma_2\left(\tau,\overline{X}\left(\tau\right)\right),$$

where $\gamma_1\left(\tau,\overline{X}\left(\tau\right)\right)$ regroups the terms of the form $\left(f_k-\overline{\alpha}_k\right)v$ with v a smooth function of $\left(t,\overline{X}(t)\right)$ and $\gamma_2\left(\tau,\overline{X}\left(\tau\right)\right)$ the terms of the form $v\left(\partial_k u-\partial_k\overline{u}\right)$. By (i) we have that $\mathbb{E}\left[\gamma_2\right](s) = \int_s^T \mathscr{O}\left(\Delta t(\tau)\right)\mathrm{d}\tau$. By Itô's formula,

$$\mathbb{E}\left[\gamma_1\right](s) = \mathbb{E}\left[\gamma_1\left(t_m\right)\right] + \int_{t_m}^{s}\mathbb{E}\left[\mathscr{L}\gamma_1\right](\tau)\,\mathrm{d}\tau = \mathscr{O}\left(\Delta t_m\right)$$

since $\left(f_k-\overline{\alpha}_k\right)\left(t_m\right) = \mathscr{O}\left(\Delta t_m\right)$ as above and since $\mathbb{E}\left[\mathscr{L}\gamma_1\right](\tau)$ is bounded. Hence,

$$\begin{aligned}
& \int_{t_n}^{t}\mathbb{E}\left[\mathscr{L}\gamma\left(\tau,\overline{X}\left(\tau\right)\right)\right]\mathrm{d}\tau \\
= & \int_{t_n}^{t}\mathbb{E}\left[\gamma_1\left(\tau,\overline{X}\left(\tau\right)\right)\right]\mathrm{d}\tau + \int_{t_n}^{t}\mathbb{E}\left[\gamma_2\left(\tau,\overline{X}\left(\tau\right)\right)\right]\mathrm{d}\tau \\
= & \int_{t_n}^{t}\mathscr{O}\left(\Delta t_n\right)\mathrm{d}\tau + \int_{t_n}^{t}\int_{\tau}^{T}\mathscr{O}\left(\Delta t\left(\varepsilon\right)\right)\mathrm{d}\varepsilon\mathrm{d}\tau \\
= & \mathscr{O}\left(\Delta t_n^2\right) + \int_{t_n}^{T}\mathscr{O}\left(\Delta t\left(\varepsilon\right)\right)\mathrm{d}\varepsilon\Delta t_n \\
= & \Delta t_n\int_{t_n}^{T}\mathscr{O}\left(\Delta t\left(\varepsilon\right)\right)\mathrm{d}\varepsilon.
\end{aligned}$$

Furthermore, using from above $\left(f_k - \overline{\alpha}_k\right)\left(t_n, \overline{X}(t_n)\right) = \mathcal{O}(\Delta t_n)$ and the result from (i) we get

$$\begin{aligned}
\mathbb{E}\left[\gamma\left(t_n, \overline{X}(t_n)\right)\right] &= \mathbb{E}\left[\left(f_k - \overline{\alpha}_k\right)\left(t_n, \overline{X}(t_n)\right)\left(\partial_k u - \partial_k \overline{u}\right)\left(t_n, \overline{X}(t_n)\right)\right] \\
&= \mathbb{E}\left[\mathcal{O}(\Delta t_n)\left(\partial_k u - \partial_k \overline{u}\right)\left(t_n, \overline{X}(t_n)\right)\right] \\
&= \mathcal{O}(\Delta t_n)\,\mathbb{E}\left[\left(\partial_k u - \partial_k \overline{u}\right)\left(t_n, \overline{X}(t_n)\right)\right] \\
&= \mathcal{O}(\Delta t_n)\,\mathbb{E}\left[\int_{t_n}^{T} \mathcal{O}(\Delta t(\tau))\,\mathrm{d}\tau\right] \\
&= \mathcal{O}(\Delta t_n) \int_{t_n}^{T} \mathcal{O}(\Delta t(\tau))\,\mathrm{d}\tau \\
&= \Delta t_n \int_{t_n}^{T} \mathcal{O}(\Delta t(\tau))\,\mathrm{d}\tau.
\end{aligned}$$

Therefore,

$$\mathbb{E}\left[\gamma\left(t, \overline{X}(t)\right)\right] = \Delta t_n \int_{t_n}^{T} \mathcal{O}(\Delta t(\tau))\,\mathrm{d}\tau,$$

which concludes the proof of (ii). The proof of (iii) is similar to the one of (ii), and thus, we omit it here. □

The next lemma shows how $\partial_i \overline{u}$ can be represented by dual functions.

Lemma 6.3.6. *Suppose the assumptions (i)-(iv) of Theorem 6.3.1 hold. Let φ and φ' be the solutions of the dual backward problem defined in* (6.11) *and* (6.12). *Then the following holds:*

(i)

$$\begin{aligned}
\partial_i \overline{u}\left(t_n, \overline{X}(t_n)\right) &= \mathbb{E}\left[\varphi_i(t_n) \,|\, \mathcal{F}_{t_n}\right], \\
\partial_{ij} \overline{u}\left(t_n, \overline{X}(t_n)\right) &= \mathbb{E}\left[\varphi'_{ij}(t_n) \,|\, \mathcal{F}_{t_n}\right];
\end{aligned}$$

(ii) *for $t = t_{n+1}$ or $t = t_n$*

$$\mathbb{E}\left[\left(f_i\left(t, \overline{X}(t)\right) - \overline{\alpha}_i\left(t, \overline{X}\right)\right)\mathbb{E}\left[\varphi_i(t) | \mathcal{F}_t\right]\right] = \mathbb{E}\left[\left(f_i\left(t, \overline{X}(t)\right) - \overline{\alpha}_i\left(t, \overline{X}\right)\right)\varphi_i(t)\right];$$

(iii)

$$\begin{aligned}
&\mathbb{E}\left[\tfrac{1}{2}\left(g_i^r\left(t, \overline{X}(t)\right) g_j^r\left(t, \overline{X}(t)\right) - \overline{\beta}_i^r\left(t, \overline{X}\right)\overline{\beta}_j^r\left(t, \overline{X}\right)\right)\mathbb{E}\left[\varphi'_{ij}(t) | \mathcal{F}_t\right]\right] \\
= \;& \mathbb{E}\left[\tfrac{1}{2}\left(g_i^r\left(t, \overline{X}(t)\right) g_j^r\left(t, \overline{X}(t)\right) - \overline{\beta}_i^r\left(t, \overline{X}\right)\overline{\beta}_j^r\left(t, \overline{X}\right)\right)\varphi'_{ij}(t)\right].
\end{aligned}$$

Proof of Lemma 6.3.6. Since the S-ROCK method is a Runge-Kutta scheme, we can exchange derivation and the application of the numerical method. Hence, we have

$$\partial_i \overline{u}\left(t, \overline{X}(t)\right) = \partial_i \mathbb{E}\left[\phi\left(\overline{X}(T)\right) | \overline{X}(t) = \overline{X}(t)\right] = \mathbb{E}\left[\partial_j \phi\left(\overline{X}(T)\right)\overline{X}'_{ji}(T, t) \,|\, \mathcal{F}_t\right],$$

where $\overline{X}'_{ji}$ represents the S-ROCK approximation of X' with initial condition $\overline{X}_{ji}(t,t) = \delta_{ji}$. Then one can proceed as in the proof of Lemma 2.5 in [96] to show that

$$\varphi_k(t_m) = \partial_i \phi\left(\overline{X}(T)\right)\overline{X}'_{ik}(T, t_m).$$

Therefore,

$$\partial_i \overline{u}\left(t_n, \overline{X}(t_n)\right) = \mathbb{E}\left[\partial_j \phi\left(\overline{X}(T)\right)\overline{X}'_{ji}(T, t_n) | \mathscr{F}_{t_n}\right] = \mathbb{E}\left[\varphi_i(t_n) | \mathscr{F}_{t_n}\right],$$

which proves part one of (i). Observe that

$$\partial_{ij}\overline{u}\left(t_n, \overline{X}(t_n)\right) = \partial_j\left(\partial_i \overline{u}\left(t_n, \overline{X}(t_n)\right)\right) = \partial_j \mathbb{E}\left[\varphi_i(t_n) | \mathscr{F}_{t_n}\right] = \mathbb{E}\left[\underbrace{\partial_{x_j(t_n)}\varphi'_i(t_n)}_{=\varphi'_{ij}(t_n)} | \mathscr{F}_{t_n}\right].$$

As in Lemma 2.5 of [96] one can show that φ'_{ij} satisfies the dual backward problem defined in (6.12), and thus, we obtain

$$\partial_{ij}\overline{u}\left(t_n, \overline{X}(t_n)\right) = \mathbb{E}\left[\varphi'_{ij}(t_n) | \mathscr{F}_{t_n}\right],$$

which completes the proof of (i). To prove (ii) and (iii) respectively, one uses that the corresponding functions are measurable and the properties of conditional expectations. □

We have now everything that is required to prove Theorem 6.3.1.

Proof of Theorem 6.3.1. Suppose the assumptions (i)-(iv) of Theorem 6.3.1 hold. For the time discretization error we have $\mathbb{E}\left[\phi(X(T)) - \phi\left(\overline{X}(T)\right)\right]$ is equal to

$$\begin{aligned}&\int_0^T \mathbb{E}\left[\left(f_k\left(t, \overline{X}(t)\right) - \overline{\alpha}_k\left(t, \overline{X}\right)\right)\partial_k u\left(t, \overline{X}(t)\right)\right] \mathrm{d}t \\ &+ \sum_{r=1}^{m} \frac{1}{2}\int_0^T \mathbb{E}\left[\left(g_k^r\left(t, \overline{X}(t)\right) g_\ell^r\left(t, \overline{X}(t)\right) - \overline{\beta}_k^r\left(t, \overline{X}\right)\overline{\beta}_\ell^r\left(t, \overline{X}\right)\right)\partial_{k\ell} u\left(t, \overline{X}(t)\right)\right] \mathrm{d}t,\end{aligned}$$

where we have used Lemma 6.3.3. Taking into account the addition property of integrals we obtain

$$\begin{aligned}&\sum_{n=0}^{N-1}\int_{t_n}^{t_{n+1}} \mathbb{E}\left[\left(f_k\left(t, \overline{X}(t)\right) - \overline{\alpha}_k\left(t, \overline{X}\right)\right)\partial_k u\left(t, \overline{X}(t)\right)\right] \mathrm{d}t \\ &+ \sum_{n=0}^{N-1}\sum_{r=1}^{m} \frac{1}{2}\int_{t_n}^{t_{n+1}} \mathbb{E}\left[\left(g_k^r\left(t, \overline{X}(t)\right) g_\ell^r\left(t, \overline{X}(t)\right) - \overline{\beta}_k^r\left(t, \overline{X}\right)\overline{\beta}_\ell^r\left(t, \overline{X}\right)\right)\partial_{k\ell} u\left(t, \overline{X}(t)\right)\right] \mathrm{d}t.\end{aligned}$$

By Lemma 6.3.4 it follows that this corresponds to

$$\begin{aligned}
&\sum_{n=0}^{N-1} \frac{\Delta t_n}{2} \mathbb{E}\left[\left(f_k\left(t_{n+1}, \overline{X}(t_{n+1})\right) - \overline{\alpha}_k\left(t_n, \overline{X}\right)\right) \partial_k u\left(t_{n+1}, \overline{X}(t_{n+1})\right)\right] \\
&+ \tfrac{\Delta t_n}{2} \mathbb{E}\left[\left(f_k\left(t_n, \overline{X}(t_n)\right) - \overline{\alpha}_k\left(t_n, \overline{X}\right)\right) \partial_k u\left(t_n, \overline{X}(t_n)\right)\right] \\
&+ \sum_{n=0}^{N-1} \sum_{r=1}^{m} \frac{\Delta t_n}{2} \mathbb{E}\left[\frac{1}{2}\left(\left(g_k^r g_\ell^r\right)\left(t_{n+1}, \overline{X}(t_{n+1})\right) - \left(\overline{\beta}_k^r \overline{\beta}_\ell^r\right)\left(t_n, \overline{X}\right)\right) \partial_{k\ell} u\left(t_{n+1}, \overline{X}(t_{n+1})\right)\right] \\
&+ \tfrac{\Delta t_n}{2} \mathbb{E}\left[\tfrac{1}{2}\left(\left(g_k^r g_\ell^r\right)\left(t_n, \overline{X}(t_n)\right) - \left(\overline{\beta}_k^r \overline{\beta}_\ell^r\right)\left(t_n, \overline{X}\right)\right) \partial_{k\ell} u\left(t_n, \overline{X}(t_n)\right)\right] + \mathcal{O}\left(\Delta t_n^3\right).
\end{aligned}$$

Next, applying Lemma 6.3.5 yields

$$\begin{aligned}
&\sum_{n=0}^{N-1} \frac{\Delta t_n}{2} \left(\mathbb{E}\left[\left(f_k\left(t_{n+1}, \overline{X}(t_{n+1})\right) - \overline{\alpha}_k\left(t_n, \overline{X}\right)\right) \partial_k \overline{u}\left(t_{n+1}, \overline{X}(t_{n+1})\right)\right]\right. \\
&\left. + \Delta t_n \textstyle\int_{t_{n+1}}^{T} \mathcal{O}\left(\Delta t(\tau)\right) \mathrm{d}\tau\right) \\
&+ \tfrac{\Delta t_n}{2} \left(\mathbb{E}\left[\left(f_k\left(t_n, \overline{X}(t_n)\right) - \overline{\alpha}_k\left(t_n, \overline{X}\right)\right) \partial_k \overline{u}\left(t_n, \overline{X}(t_n)\right)\right] + \Delta t_{n-1} \textstyle\int_{t_n}^{T} \mathcal{O}\left(\Delta t(\tau)\right) \mathrm{d}\tau\right) \\
&+ \sum_{n=0}^{N-1} \sum_{r=1}^{m} \frac{\Delta t_n}{2} \left(\mathbb{E}\left[\frac{1}{2}\left(\left(g_k^r g_\ell^r\right)\left(t_{n+1}, \overline{X}(t_{n+1})\right) - \left(\overline{\beta}_k^r \overline{\beta}_\ell^r\right)\left(t_n, \overline{X}\right)\right) \partial_{k\ell} \overline{u}\left(t_{n+1}, \overline{X}(t_{n+1})\right)\right]\right. \\
&\left. + \Delta t_n \textstyle\int_{t_{n+1}}^{T} \mathcal{O}\left(\Delta t(\tau)\right) \mathrm{d}\tau\right) \\
&+ \tfrac{\Delta t_n}{2} \left(\mathbb{E}\left[\tfrac{1}{2}\left(\left(g_k^r g_\ell^r\right)\left(t_n, \overline{X}(t_n)\right) - \left(\overline{\beta}_k^r \overline{\beta}_\ell^r\right)\left(t_n, \overline{X}\right)\right) \partial_{k\ell} \overline{u}\left(t_n, \overline{X}(t_n)\right)\right]\right. \\
&\left. + \Delta t_{n-1} \textstyle\int_{t_n}^{T} \mathcal{O}\left(\Delta t(\tau)\right) \mathrm{d}\tau\right) \\
&+ \mathcal{O}\left(\Delta t_n^3\right).
\end{aligned}$$

Finally, considering Lemma 6.3.6 we get

$$\begin{aligned}
&\mathbb{E}\left[\phi(X(T)) - \phi\left(\overline{X}(T)\right)\right] \\
= \quad & \sum_{n=0}^{N-1} \frac{\Delta t_n}{2} \mathbb{E}\left[\left(f_k\left(t_{n+1}, \overline{X}(t_{n+1})\right) - \overline{\alpha}_k\left(t_n, \overline{X}\right)\right) \varphi_k(t_{n+1})\right] + \frac{\Delta t_n^2}{2} \int_{t_{n+1}}^{T} \mathcal{O}\left(\Delta t(\tau)\right) \mathrm{d}\tau \\
&+ \tfrac{\Delta t_n}{2} \mathbb{E}\left[\left(f_k\left(t_n, \overline{X}(t_n)\right) - \overline{\alpha}_k\left(t_n, \overline{X}\right)\right) \varphi_k(t_n)\right] + \tfrac{\Delta t_{n-1} \Delta t_n}{2} \textstyle\int_{t_n}^{T} \mathcal{O}\left(\Delta t(\tau)\right) \mathrm{d}\tau + \mathcal{O}\left(\Delta t_n^3\right) \\
&+ \sum_{n=0}^{N-1} \sum_{r=1}^{m} \frac{1}{2} \left[\frac{\Delta t_n}{2} \mathbb{E}\left[\left(\left(g_k^r g_\ell^r\right)\left(t_{n+1}, \overline{X}(t_{n+1})\right) - \left(\overline{\beta}_k^r \overline{\beta}_\ell^r\right)\left(t_n, \overline{X}\right)\right) \varphi_{k\ell}'(t_{n+1})\right]\right. \\
&+ \tfrac{\Delta t_n^2}{2} \textstyle\int_{t_{n+1}^T} \mathcal{O}\left(\Delta t(\tau)\right) \mathrm{d}\tau \\
&+ \tfrac{\Delta t_n}{2} \mathbb{E}\left[\left(\left(g_k^r g_\ell^r\right)\left(t_n, \overline{X}(t_n)\right) - \left(\overline{\beta}_k^r \overline{\beta}_\ell^r\right)\left(t_n, \overline{X}\right)\right) \varphi_{k\ell}'(t_n)\right] \\
&\left. + \tfrac{\Delta t_{n-1} \Delta t_n}{2} \textstyle\int_{t_n}^{T} \mathcal{O}\left(\Delta t(\tau)\right) \mathrm{d}\tau\right] + \mathcal{O}\left(\Delta t_n^3\right).
\end{aligned}$$

By approximating the expectations by a sample average one obtains the result of the theorem. □

We use now the results form this section to define in the next section an adaptive algorithm.

6.3.2 Adaptive Algorithm

In this section we describe an algorithm to approximate E, given by (6.7), by $\widehat{E}$, defined by (6.8), using the S-ROCK method with variable time stepping. The aim is to generate a suitable time grid and to pick the right number of simulations M such that the error $E-\widehat{E}$ is bounded by the given tolerance TOL. The algorithm is partly based on [96], but we extend it to account for the choice of the stage numbers of the S-ROCK method.

Main Routine

Suppose an initial number of simulations M_T^0, a time grid τ^0 of the interval $[0,T]$ and some tolerance $\mathrm{TOL}>0$ are given at the start of the algorithm. Furthermore suppose that the tolerance is split into TOL_T and TOL_S such that $\mathrm{TOL}_T+\mathrm{TOL}_S=\mathrm{TOL}$. The aim is to achieve $\mathrm{err}_T \leq \mathrm{TOL}_T$ as well as $\mathrm{err}_S \leq \mathrm{TOL}_S$. Then (6.9) implies that $E-\widehat{E}\leq \mathrm{TOL}$.

First an adaptive time grid is generated for the interval $[0,T]$. Observe that usually the expectations in the time discretization error err_T cannot be computed explicitly. Using the expression of the computable leading term derived in Section 6.3.1 we can define some kind of error density ρ, which measures the time discretization error over $[0,T]$. This density is a piecewise constant function with $\rho(t)=\rho_n$ for $t\in[t_n,t_{n+1}[$ for $n=0,1,\ldots,N-1$. The time disretization error can further be approximated by

$$\begin{aligned}
|\mathrm{err}_T| &= \left|\mathbb{E}\left[\phi(X(T))\right]-\mathbb{E}\left[\phi(\overline{X}_N)\right]\right| \\
&\approx \left|\mathbb{E}\left[\sum_{n=0}^{N-1}(\Delta t_n)^2\rho_n\right]\right| \\
&\leq \left|\left|\mathbb{E}\left[\sum_{n=0}^{N-1}(\Delta t_n)^2\rho_n\right]\right|-\mathscr{A}\left(M_T,\sum_{n=0}^{N-1}(\Delta t_n)^2\rho_n\right)\right| \\
&\quad +\left|\mathscr{A}\left(M_T,\sum_{n=0}^{N-1}(\Delta t_n)^2\rho_n\right)\right| \\
&=: \widehat{\mathrm{err}}_{TS}+\widehat{\mathrm{err}}_{TT}
\end{aligned} \tag{6.17}$$

with $\mathscr{A}(M,Y)$ representing the sample average over M independent identically distributed samples of Y. The former term in the last line of (6.17), $\widehat{\mathrm{err}}_{TS}$, can further be approximated using the so-called Berry-Esseen theorem and approximating the standard deviation by the sample standard deviation $\mathscr{S}(M,Y)$:

$$\widehat{\mathrm{err}}_{TS} = \left\| \mathbb{E}\left[\sum_{n=0}^{N-1} (\Delta t_n)^2 \rho_n\right] \right| - \mathscr{A}\left(M_T, \sum_{n=0}^{N-1} (\Delta t_n)^2 \rho_n\right) \Bigg| \le c_0 \frac{\mathscr{S}\left(M_T, \sum_{n=0}^{N-1} (\Delta t_n)^2 \rho_n\right)}{\sqrt{M_T}}$$

with $c_0 \ge 1.65$ defining the confidence interval.

Splitting $\mathrm{TOL}_T = \mathrm{TOL}_{TS} + \mathrm{TOL}_{TT}$, at the start of the loop we set $\widehat{\mathrm{err}}^0_{TS}$ and $\widehat{\mathrm{err}}^0_{TT}$ to, for instance, $\widehat{\mathrm{err}}^0_{TS} = 2\mathrm{TOL}_{TS}$ and $\widehat{\mathrm{err}}^0_{TT} = 2\mathrm{TOL}_{TT}$, respectively. Moreover, taking into account the stiffness of the problem and the initial time grid we define the number of stages of the S-ROCK needed at each time step. Then we iterate as long as $\widehat{\mathrm{err}}^i_{TS} + \widehat{\mathrm{err}}^i_{TT} > \mathrm{TOL}_{TT} + \mathrm{TOL}_{TS}$. At each iteration i we recompute the estimates $\widehat{\mathrm{err}}^i_{TS}$ and $\widehat{\mathrm{err}}^i_{TT}$ based on the time grid τ^i and with M^i_T simulations. If $\widehat{\mathrm{err}}^i_{TS} > \mathrm{TOL}_{TS}$, then we increase the number of simulations M^i_T according to a subroutine (see below) which yields M^{i+1}_T. The time grid τ^i remains unchanged though, i.e. $\tau^{i+1} = \tau^i$. Also the number of stages of S-ROCK for each time stepsize remains the same. Otherwise if $\widehat{\mathrm{err}}^i_{TT} > \mathrm{TOL}_{TT}$ then we refine the time grid according to a subroutine (see below) which yields τ^{i+1}. Since the time stepsizes change, one has also to adapt the number of stages s of the S-ROCK method. The subroutine to update the stage number is given below. The number of simulations M^{i+1}_T is given by the previous number M^i_T.

At the end of the first part of the algorithm, we have generated a specific time grid τ^I, with I representing the final iteration. From now on the time grid and the associated number of stages are fixed and we are left to determine the number of simulations M to control the statistical error. The starting value of M is chosen such that $M^0 = M^I_T$. Similar as above the statistical error err_S can be approximated by

$$\begin{aligned} \mathrm{err}_S &= \mathbb{E}\left[\phi\left(\overline{X}_N\right)\right] - \tfrac{1}{M}\sum_{j=1}^{M} \phi\left(\overline{X}^j_N\right) \\ &\approx c_0 \frac{\mathscr{S}\left(M, \phi\left(\overline{X}_N\right)\right)}{\sqrt{M}} =: \widehat{\mathrm{err}}_S. \end{aligned}$$

Iterate as long as $\widehat{\mathrm{err}}^i_S > \mathrm{TOL}_S$. At each iteration i compute M^i samples of $\phi(X_N)$ and then compute $\widehat{E} = \mathscr{A}\left(M^i, \phi(X_N)\right)$ and $\widehat{\mathrm{err}}^{i+1}_S$ evaluating the sample standard deviation. Then increase the number of simulations to M^{i+1} according to a specified subroutine (see below). At the end of the second part of the algorithm we have computed an approximation $\widehat{E}$ of E such that $E - \widehat{E} \le \mathrm{TOL}$ as desired.

Subroutines

The subroutine to update M is given by

$$M^{new} = \min\left\{ \left\lceil \left(\frac{c_0 \mathscr{S}}{0.95\mathrm{TOL}}\right)^2 \right\rceil, N_1 \times M \right\}$$

with $\mathscr{S}$ the corresponding sample standard deviation and N_1 some integer used to avoid an explosion of the number of simulations due to a bad sample. The number of simulations M_T can be updated similarly.

The subroutine to refine the time grid takes as arguments the desired tolerance (here TOL_T), the current time grid (here τ^i at the ith iteration) and an approximation $\widehat{\rho}^i$ of the error density function ρ^i, which is obtained by taking a sample average (over M_T^i samples). First we compute an optimal (in terms of minimal number of steps to be within the given tolerance, Lagrange multipliers approach) time grid τ^* such that

$$\tau_n^* = \frac{\mathrm{TOL}_T}{\sqrt{\widehat{\rho}_n^i \sum_{k=0}^{N-1} \widehat{\rho}_k^i}}.$$

The new time grid τ^{i+1} is obtained by dividing $\Delta\tau_{n+1}^i$ into i_{n+1} subintervals with

$$i_{n+1} = \min\left\{\max\left\{\left\lceil \frac{\Delta\tau_{n+1}^i}{\Delta\tau_{n+1}^*} \right\rceil, 1\right\}, N_2\right\},$$

where $\Delta\tau_{n+1}^* = \tau_{n+1}^* - \tau_n^*$ and N_2 is some integer bound the number of increments. Observe that if the error density is large at say τ_n, then τ_n^* is small, and thus, the corresponding interval is refined by adding possibly many more steps.

If the time grid changes, the number of stages of S-ROCK has to be adapted (this is also the case for the initial time grid). The following routine shows how to update the stage numbers. Let s_n be the stage number corresponding to the time stepsize $\Delta\tau_n$. For the S-ROCK integrator we consider the stability constraint (6.6), i.e.

$$\frac{\Delta\tau_n \rho^\star}{c_{SR} s_n^2} < 1$$

with $\rho^\star$ a given stiffness parameter (e.g. for the test problem $\rho^\star = \rho_{SR}$) and c_{SR} a computable positive constant that with increasing s_n quickly settles at 0.33. Hence, the stage number corresponding to the time interval $[\tau_n, \tau_{n+1}[$ can be computed by

$$s_n = \left\lfloor \sqrt{\frac{\Delta\tau_n}{c_{SR}}\rho^\star} \right\rfloor + 1$$

for each n.

Recall that for the Euler-Maruyama method the time stepsize has to satisfy

$$\Delta\tau_n \rho < 1$$

due to stability issues, see (6.5). Therefore, if the time stepsize $\Delta\tau_n$ suggested by the adaptive algorithm is too large, one is forced to use a smaller time stepsize that satisfies the stability constraint. This drawback does not exist for S-ROCK. In fact by adjusting the stage number s_n any suggested (by the adaptive algorithm) time stepsize $\Delta\tau_n$ can be used. Adapting the time stepsize of the Euler-Maruyama approach to account for the stepsize restriction is beyond the scope of this chapter and we use the S-ROCK approach, when the variable time stepping algorithm based on Euler-Maruyama cannot be applied.

6.4 Numerical Experiments

In this section we carry out two numerical experiments. We show first that the adaptive algorithm of Section 6.3.2, that uses S-ROCK as numerical integrator, works well for a two-dimensional stochastic differential equation that is not stiff. Then we consider again a two-dimensional SDE but this time the stiffness of the stochastic problem depends on one of the parameters of the model. Numerical experiments show that in a scenario with stiffness the Euler-Maruyama approach cannot be used, whereas the S-ROCK approach yields an adaptive time grid by adjusting the number of stages to the stiffness of the problem.

6.4.1 Nonstiff Stochastic Differential Equation

The first example that we consider here is a two-dimensional nonstiff stochastic differential equation defined by the SDE

$$\begin{aligned} \mathrm{d}\begin{pmatrix} X_1(t) \\ X_2(t) \end{pmatrix} &= \begin{pmatrix} -X_2(t) \\ X_1(t) \end{pmatrix}\mathrm{d}t + \begin{pmatrix} 0 \\ \frac{\sin(X_1(t)+X_2(t))}{\sqrt{1+t}} \end{pmatrix}\mathrm{d}W^1(t) \\ &\quad + \begin{pmatrix} \frac{\cos(X_1(t)+X_2(t))}{\sqrt{1+t}} \\ 0 \end{pmatrix}\mathrm{d}W^2(t), \quad 0 < t \leq T, \end{aligned} \tag{6.18}$$

with initial condition $(X_1(0), X_2(0)) = (1,1)$ and $\left(W^1(t)\right)_{t\in[0,T]}$ and $\left(W^2(t)\right)_{t\in[0,T]}$ two independent one-dimensional standard Brownian motions. Moreover, as functional we consider $\phi(X(t)) = X_1(t)^2 + X_2(t)^2$. It can be shown that the exact solution of $\mathbb{E}\left[\phi(X(T))\right] = 2 + \log(1+T)$. This example was also used in [96] and [97]. In [96] it has been shown that the algorithm for Euler-Maruyama works well for this example. Here, we use this stochastic problem to illustrate that the variable time stepping algorithm that uses S-ROCK as numerical integrator can be applied too. To give us an idea how the solutions to (6.18) look like, Figure 6.2 shows a sample path of the two-dimensional SDE using the S-ROCK method with $s = 3$ stages and a uniform time grid with time stepsize 0.001.

In the following we use the parameters $T = 1$, $M_T^0 = 100$, $\mathrm{TOL} = 0.02$, $\mathrm{TOL}_{TT} = 2/9\mathrm{TOL}$, $\mathrm{TOL}_{TS} = 1/9\mathrm{TOL}$, $\mathrm{TOL}_S = 2/3\mathrm{TOL}$, $c_0 = 1.96$, $N_1 = 50$ and $N_2 = 3$. Furthermore, the initial time grid that we consider is a uniform one with time stepsize 0.1.

Table 6.1 shows the results of the algorithm described in Section 6.3.2 applied to the stochastic problem (6.18) using the S-ROCK integrator with $s = 3$ stages. One observers that first the

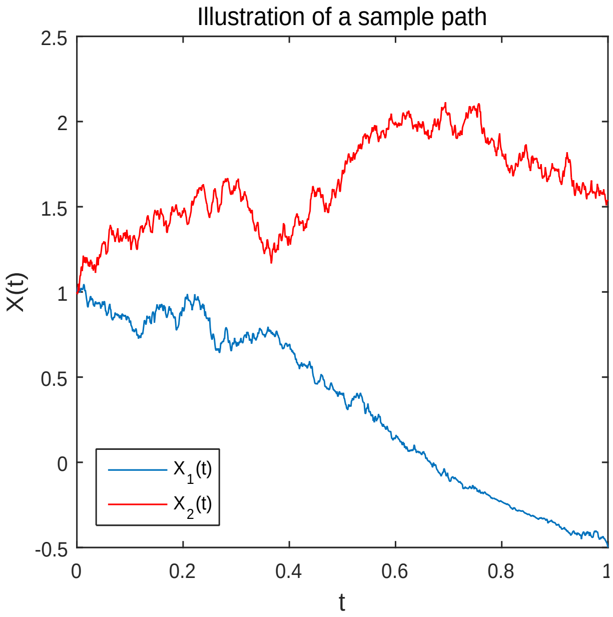

Figure 6.2: Illustration of a sample path of (6.18) using S-ROCK with $s = 3$ stages and a uniform time grid with 1000 steps.

Table 6.1: Results of the algorithm for variable time stepping using S-ROCK with $s = 3$ stages to solve (6.18).

iteration	N^i	M_T^i	$\widehat{\mathrm{err}}^i_{TT}$	$\widehat{\mathrm{err}}^i_{TS}$
0	10	100	0.0089	0.0044
1	10	5000	0.0568	0.0323
2	10	28922	0.0529	0.0051
3	30	28922	0.0522	0.0021
4	79	28922	0.0170	0.0022
5	137	28922	0.0056	0.0008

iteration	0	1	2
M^i	28922	28922	100932
$\widehat{\mathrm{err}}^i_S$	0.0267	0.0237	0.0126

number of simulations M_T^i increases, which leads to a reduced $\widehat{\mathrm{err}}^i_{TS}$. Then once $\widehat{\mathrm{err}}^i_{TS}$ falls below TOL_{TS}, the time grid is adjusted and the number of time steps N^i increases, which yields that $\widehat{\mathrm{err}}^i_{TT}$ decreases and eventually is smaller than TOL_{TT}. Once the time grid is fixed, the number of simulations used for the Monte Carlo approach is determined. The error of the approximation of $\mathbb{E}\left[\phi(X(T))\right]$ we obtain by the adaptive algorithm is $E - \widehat{E} = 0.0164$, and thus, smaller than the desired tolerance of $\mathrm{TOL} = 0.02$. Hence, the algorithm using S-ROCK with

$s = 3$ works well for this example. Various other simulations have been carried out for larger values of s and each time the algorithm delivered successfully an adaptive time grid.

6.4.2 Stiff Stochastic Differential Equation

The next example that we consider here is one derived from the one-dimensional population dynamic model (see [84]). It is characterized by the SDE

$$\left\{\begin{array}{rcl} \mathrm{d}\begin{pmatrix} X_1(t) \\ X_2(t) \end{pmatrix} & = & \begin{pmatrix} \alpha(X_2(t)-1)-\mu_1 X_1(t)(1-X_1(t)) \\ -\mu_2 X_2(t)(1-X_2(t)) \end{pmatrix}\mathrm{d}t \\ & + & \begin{pmatrix} -\sigma_1 X_1(t)(1-X_1(t)) \\ -\sigma_2 X_2(t)(1-X_2(t)) \end{pmatrix}\mathrm{d}W^1(t) \\ & + & \begin{pmatrix} -\sigma_2(1-X_1(t)) \\ 0 \end{pmatrix}\mathrm{d}W^2(t), \quad 0 \le t \le T, \\ \begin{pmatrix} X_1(0) \\ X_2(0) \end{pmatrix} & = & \begin{pmatrix} 0.95 \\ 0.95 \end{pmatrix}, \end{array}\right. \tag{6.19}$$

where $(W_1(t))_{t\in[0,T]}$ and $(W_2(t))_{t\in[0,T]}$ are two independent one-dimensional standard Brownian motions. As functional we consider again $\phi(X(t)) = X_1(t)^2 + X_2(t)^2$. The parameters of (6.19) that we take into account here are $T = 1$, $\alpha = 2$, $\mu_2 = -1$, $\sigma_2 = 0.5$. Depending on the values of (μ_1, σ_1), the SDE (6.19) is stiff (see also Section 3.5.2). Hence, we vary in the following the value of μ_1 while considering $\sigma_1 = \sqrt{|\mu_1|}$. Note that the choice of the sets of parameters that we pick here (μ_1, σ_1) and (μ_2, σ_2) lie both in the true stability domain of the test problem.

For the adaptive algorithm we use $M_T^0 = 100$, $\mathrm{TOL} = 0.02$, $\mathrm{TOL}_{TT} = 2/9\mathrm{TOL}$, $\mathrm{TOL}_{TS} = 1/9\mathrm{TOL}$, $\mathrm{TOL}_S = 2/3\mathrm{TOL}$, $c_0 = 1.96$, $N_1 = 50$, $N_2 = 3$ and an initial time grid with uniform time stepsize 0.2.

Table 6.2: Results of the algorithm for the variable time stepping Euler-Maruyama method for the stochastic problem (6.19) with $\mu_1 = -1$.

iteration	N^i	M_T^i	$\widehat{\mathrm{err}}_{TT}^i$	$\widehat{\mathrm{err}}_{TS}^i$
0	5	100	0.0089	0.0044
1	5	346	0.0004	0.0039

iteration	0	1	2
M^i	346	346	424
$\widehat{\mathrm{err}}_S^i$	0.0267	0.0140	0.0133

Choosing $\mu_1 = -1$ and using the adaptive algorithm of Section 6.3.2 with the Euler-Maruyama method as numerical integrator, we get the results from Table 6.2. The time grid is not refined, and thus, the final time grid coincides with the initial one. However, if we increase the stiffness by taking for instance $\mu_1 = -10$ the algorithm based on the Euler-Maruyama method does not converge in this setting due to stability issues. One would have to adjust the initial time stepsize according to the stability constraint of Euler-Maruyama (6.5). The same has been tested for $\mu_1 = -100$ or $\mu_1 = -1000$. In both cases the algorithm did not yield a result.

We apply now the variable time stepping algorithm of Section 6.3.2 with the S-ROCK method as numerical integrator. Table 6.3 shows the result for $\mu_1 = -1$. The algorithm works and the final time grid that one obtains is also illustrated in the table (the time stepsizes of the final grid are represented by Δt_n^I and the corresponding stage numbers by s_n^I). Since the stochastic problem is not stiff, a stage number of $s = 3$ is sufficient. We increase now the stiffness in the next step.

Table 6.3: Results of the algorithm for the variable time stepping S-ROCK method for the stochastic problem (6.19) with $\mu_1 = -1$.

iteration	N^i	M_T^i	$\widehat{\text{err}}_{TT}^i$	$\widehat{\text{err}}_{TS}^i$
0	5	100	0.0089	0.0044
1	5	259	0.0095	0.0034
2	11	259	0.0082	0.0020
3	11	259	0.0040	0.0009

iteration	0	1	2
M^i	259	259	306
$\widehat{\text{err}}_S^i$	0.0267	0.0138	0.0123

Δt_n^I	0.0667	0.0667	0.0667	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
s_n^I	3	3	3	3	3	3	3	3	3	3	3

In Table 6.4 the results for $\mu_1 = -10$ are illustrated. Recall that the Euler-Maruyama approach did not converge for this set of parameters, but in the table we can see, that the S-ROCK approach works perfectly well. Again the final time grid is shown with some refinement at the start of the interval and at the end. Since the stiffness is not that large, the stage number is still $s = 3$ for every time step.

Table 6.5 presents the results we obtain using $\mu_1 = -100$. The algorithm yields a refined time grid with variable time stepsize. In addition, due to the stiffness of the problem the stage numbers are adjusted. For the three first steps, the stage number is equal to $s = 5$. For the remaining steps the stage number is equal to $s = 6$. Note that the stage numbers corresponding

Table 6.4: Results of the algorithm for the variable time stepping S-ROCK method for the stochastic problem (6.19) with $\mu_1 = -10$.

iteration	N^i	M_T^i	$\widehat{\text{err}}_{TT}^i$	$\widehat{\text{err}}_{TS}^i$
0	5	100	0.0089	0.0044
1	5	329	0.0044	0.0038
2	7	329	0.0051	0.0019
3	7	329	0.0037	0.0014

iteration	0	1
M^i	329	329
$\widehat{\text{err}}_S^i$	0.0267	0.0033

Δt_n^I	0.1	0.1	0.2	0.2	0.2	0.1	0.1
s_n^I	3	3	3	3	3	3	3

Table 6.5: Results of the algorithm for the variable time stepping S-ROCK method for the stochastic problem (6.19) with $\mu_1 = -100$.

iteration	N^i	M_T^i	$\widehat{\text{err}}_{TT}^i$	$\widehat{\text{err}}_{TS}^i$
0	5	100	0.0089	0.0044
1	5	1700	0.0005	0.0044
2	11	1700	0.0078	0.0020
3	11	1700	0.0019	0.0018

iteration	0	1
M^i	1700	1700
$\widehat{\text{err}}_S^i$	0.0267	0.0010

Δt_n^I	0.0667	0.0667	0.0667	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
s_n^I	5	5	5	6	6	6	6	6	6	6	6

to the initial time grid were all equal to $s = 8$.

In Table 6.6 we have further increased the stiffness of the problem by taking $\mu_1 = -1000$. The resulting time grid is characterized by variable time stepsizes. The corresponding stage numbers are also adjusted. Note that the initial stage number was $s = 25$.

To sum up, the variable time stepping S-ROCK algorithm can account for the stiffness by

Table 6.6: Results of the algorithm for the variable time stepping S-ROCK method for the stochastic problem (6.19) with $\mu_1 = -1000$.

iteration	N^i	M_T^i	$\widehat{\text{err}}_{TT}^i$	$\widehat{\text{err}}_{TS}^i$
0	5	100	0.0089	0.0044
1	5	5000	0.0235	0.0271
2	5	17172	0.0046	0.0039
3	10	17172	0.0053	0.0021
4	10	17172	0.0012	0.0018

iteration	0	1
M^i	17172	17172
$\widehat{\text{err}}_S^i$	0.0267	0.0003

Δt_n^I	0.0667	0.0667	0.0667	0.2	0.1	0.1	0.1	0.1	0.2
s_n^I	15	15	15	25	18	18	18	18	25

adjusting the time grid and the corresponding stage numbers simultaneously, while the same approach using Euler-Maruyama as numerical integrator fails for stiff scenarios.

6.5 Conclusion

We have presented a variable time stepping algorithm for the S-ROCK method that can be used to tackle stiff stochastic differential equations. Analytically a leading term of the time discretization error has been derived. In addition, we have suggested here an algorithm that can be used to compute a time grid with variable time stepsizes and we have shown how to choose the number of stages of the S-ROCK method accordingly to avoid problems that can appear in stiff scenarios. Two different numerical experiments have been carried out, a nonstiff one and a stiff one. In both cases the adaptive S-ROCK approach works well, whereas the adaptive Euler-Maruyama approach struggles in the latter one.

7 Conclusion and Outlook

In this chapter we recapitulate the main findings of this thesis and we give a brief outlook what topics could be part of future research.

7.1 Conclusion

In this thesis, we have first recalled in Chapter 2 some definitions and results from stochastic calculus and numerical analysis. We have presented diffusion processes and we have also defined and motivated the use of jump-diffusion processes. Furthermore, numerical methods for stochastic differential equations have been discussed and concepts such as convergence and stability have been introduced. In Chapter 2 we have also presented some Monte Carlo techniques, in particular the standard Monte Carlo method, some variance reduction techniques and the multilevel Monte Carlo method.

Next, in Chapter 3 we have introduced a stabilized multilevel Monte Carlo method for stiff stochastic differential equations. We have shown that for the standard MLMC method that uses Euler-Maruyama, stiffness can prevent from exploiting all levels. The stabilized MLMC method that we have presented uses S-ROCK as numerical integrator. By adjusting the stage number of the S-ROCK method, this approach can use every level, and thus, for stiff problems the stabilized MLMC method performs better than the standard MLMC method. Another finding of this chapter is that even for nonstiff problems, that have a significant noise term, the stabilized MLMC method leads to a better performance. Using a higher weak order scheme on the finest time grid, we have also given an improved stabilized multilevel Monte Carlo method that can be used to further speeding up the simulation procedure. Various numerical experiments have been carried out to illustrate the performance of the stabilized MLMC method. We have also numerically shown that the improved stabilized MLMC method yields better results.

In Chapter 4 we have presented an extension of the S-ROCK methods to stochastic processes driven by jump-diffusions. It has been proven that the two numerical methods that we have considered in this chapter are both of strong order of convergence 1/2. Furthermore, the mean square stability domains have been characterized and plots of the stability regions have illustrated the theoretical findings. In the numerical part we have shown for two models from finance, the Merton and the Kou model, that the strong order of convergence 1/2 holds. In addition, we have carried out numerical experiments on stiff problems with jumps to show that the presented stabilized numerical integrators can handle stiffness.

Chapter 5 provides an extension of the multilevel Monte Carlo method to jump-diffusion processes. Using either a regular or a jump-adapted time grid, we have shown how the hierarchical sampling strategy of multilevel Monte Carlo can be combined with the inclusion of jumps to yield a technique that performs better than the standard Monte Carlo approach. Moreover, we have stated and proven a complexity theorem that specifies how much computational work is necessary for the newly introduced method to achieve a certain mean square accuracy. Various numerical experiments have been carried out that corroborate the theoretical results. In the last part of Chapter 5 we have combined the results of previous parts of the chapter with the ones from Chapter 4 to define a stabilized multilevel Monte Carlo method that can account for jumps.

Finally, in Chapter 6 we have proposed a variable time stepping algorithm to approximate weak solutions of stiff stochastic differential equations using the S-ROCK method. We have derived a computable leading term of the time discretization error for the stabilized algorithm. In addition, we have described an adaptive algorithm that specifies how to refine the time grid and to choose the stage numbers of the S-ROCK method simultaneously. Numerical experiments have shown that the algorithm based on Euler-Maruyama cannot be applied for stiff problems, unless one takes into account the stability constraint that leads to a reduction of the time stepsize. However, the stabilized approach based on S-ROCK methods that we have provided yields the expected numerical results even in presence of stiffness.

7.2 Outlook

In this section we briefly discuss how the existing research of this thesis could further be expanded.

One topic that future work could address is the rigorous study of the weak convergence of the newly introduced S-ROCK methods for jump-diffusions. The weak order of convergence of S-ROCK methods has been proven for stochastic problems based on diffusions in [12, 15]. One would have to take into account the additional jump term to establish a weak order of convergence result for the S-ROCK methods for stochastic differential equations driven by jump-diffusion processes.

Another part of research could be the numerical study of the stabilized multilevel Monte Carlo method for jump-diffusion that has been presented in this thesis. By considering various numerical experiments, in particular stiff problems, one could try to corroborate numerically the theoretical findings.

Future research could also try to combine some results of this thesis to provide a variable time stepping algorithm to approximate the weak solutions of stochastic differential equations with multiple scales driven by jump-diffusions processes. One could try to achieve this by considering the S-ROCK methods for jump-diffusions and by adapting the adaptive algorithm proposed in this thesis.

Finally, it could be very interesting to investigate how the efficient stabilized numerical techniques presented in this thesis can be used for applications in the financial sector, in natural sciences or in engineering to solve current problems.

Bibliography

[1] A. Abdulle. On roots and error constants of optimal stability polynomials. *BIT Numerical Mathematics*, 40(1):177–182, 2000.

[2] A. Abdulle. Fourth order Chebyshev methods with recurrence relation. *SIAM J. Sci. Comput.*, 23(6):2041–2054, 2002.

[3] A. Abdulle, A. Barth, and C. Schwab. Multilevel Monte Carlo methods for stochastic elliptic multiscale pdes. *SIAM Multiscale Model. Simul.*, 11(4):1033–1070, 2013.

[4] A. Abdulle and A. Blumenthal. Stabilized multilevel Monte Carlo method for stiff stochastic differential equations. *J. Comput. Phys.*, 251:445–460, 2013.

[5] A. Abdulle and A. Blumenthal. Improved Stabilized Multilevel Monte Carlo Method for Stiff Stochastic Differential Equations. *Lect. Notes Comput. Sci. Eng.*, 103:537–545, 2015.

[6] A. Abdulle and A. Blumenthal. S-ROCK Methods for Stiff Stochastic Differential Equations driven by Jump-Diffusion Processes. *Preprint*, 2015.

[7] A. Abdulle and A. Blumenthal. Variable Time Stepping S-ROCK Methods for Weak Solutions of Stiff Stochastic Differential Equations. *Preprint*, 2015.

[8] A. Abdulle, A. Blumenthal, and E. Buckwar. The multilevel monte carlo method for stochastic differential equations driven by jump-diffusion processes. *MATHICSE Technical Report*, 2011.

[9] A. Abdulle and S. Cirilli. Stabilized methods for stiff stochastic systems. *C. R. Math. Acad. Sci. Paris*, 345(10):593–598, 2007.

[10] A. Abdulle and S. Cirilli. S-ROCK: Chebyshev methods for stiff stochastic differential equations. *SIAM J. Sci. Comput.*, 30(2):997–1014, 2008.

[11] A. Abdulle, W. E, and T. Li. Effectiveness of implicit methods for stiff stochastic differential equations. *Commun. Comput. Phys.*, 3(2):295–307, 2008.

[12] A. Abdulle and T. Li. S-ROCK methods for stiff Ito SDEs. *Commun. Math. Sci.*, 6(4):845–868, 2008.

[13] A. Abdulle and A. Medovikov. Second order chebyshev methods based on orthogonal polynomials. *Numer. Math.*, 90(1):1–18, 2001.

[14] A. Abdulle and G. Vilmart. PIROCK: a swiss-knife partitioned implicit-explicit orthogonal Runge-Kutta Chebyshev integrator for stiff diffusion-advection-reaction problems with or without noise. *J. Comput. Phys.*, 242:869–888, 2013.

[15] A. Abdulle, G. Vilmart, and K. Zygalakis. Weak second order explicit stabilized methods for stiff stochastic differential equations. *SIAM J. Sci. Comput.*, 35(4):A1792–A1814, 2013.

[16] D. Applebaum. *Lévy Processes and Stochastic Calculus.* Cambridge University Press, Cambridge, 2004.

[17] L. Arnold. *Stochastic differential equations: theory and applications.* John Wiley and Sons, New york, 1974.

[18] Y. Aït-Sahalia. Telling from discrete data whether the underlying continuous-time model is a diffusion. *Journal of Finance*, 57:2075–2112, 2002.

[19] L. Bachelier. Théorie de la spéculation. *Annales scientifiques de l'École normale supérieure*, 17:21–86, 1900.

[20] A. Barth and A. Lang. Multilevel Monte Carlo method with applications to stochastic partial differential equations. *Int. Journal of Computer Mathematics*, 89(18):2479–2498, 2012.

[21] A. Belqadhi. Runge-Kutta-Chebyshev methods for jump-diffusion stiff stochastic differential equations. Master's thesis, EPFL, 2010.

[22] F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81:637–654, 1973.

[23] A. Blumenthal. The multilevel Monte Carlo method for SDEs driven by jump-diffusions with application in finance. Master's thesis, EPFL, 2011.

[24] R. Brown. A brief Account of Microscopical Observations made in the Months of June, July, and August, 1827, on the Particles contained in the Pollen of Plants; and on the general Existence of active Molecules in Organic and Inorganic Bodies. *Philosophical Magazine*, 4:161–173, 1828.

[25] N. Bruti-Liberati and E. Platen. Approximation of Jump Diffusions in Finance and Economics. *Computational Economics*, 29:283–312, 2007.

[26] N. Bruti-Liberati and E. Platen. *Numerical Solution of Stochastic Differential Equations with Jumps in Finance.* Springer, Sydney, 2010.

[27] E. Buckwar and C. Kelly. Towards a systematic linear stability analysis of numerical methods for systems of stochastic differential equations. *SIAM Journal on Numerical Analysis*, 48(1):298–321, 2010.

[28] E. Buckwar and M. G. Riedler. Runge-Kutta methods for jump-diffusion differential equations. *Journal of Computational and Applied Mathematics*, 236(6):1155–1182, 2011.

[29] P. Burrage and K. Burrage. A variable stepsize implementation for stochastic differential equations. *SIAM J. Sci. Comput.*, 24(3):848–864, 2002.

[30] Y. Cao, D. Gillespie, and L. Petzold. Adaptive explicit-implicit tau-leaping method with automatic tau selection. *J. Chem. Phys.*, 126(224101):1–9, 2007.

[31] G. M. Clarke and D. Cooke. *A basic course in statistics.* Edward Arnold, London, third edition, 1992.

[32] R. Cont and P. Tankov. *Financial Modelling With Jump Processes.* Chapman & Hall/CRC, London, 2004.

[33] D. Küpper, J. Lehn and A. Rößler. A step size control algorithm for the weak approximation of stochastic differential equations. *Numer Algor*, 44:335–346, 2007.

[34] K. Debrabant and A. Rößler. On the Acceleration of the Multi–Level Monte Carlo Method. *J. Appl. Probab.*, 52(2), 2015.

[35] S. Dereich. Multilevel Monte Carlo algorithms for Lévy-driven SDEs with Gaussian correction. *Ann. Appl. Probab.*, 21(1):283–311, 2011.

[36] D. Duffie. *Dynamic Asset Pricing Theory.* Princeton University Press, Princeton, 2001.

[37] T. Dumont, M. Duarte, S. Descombes, M.-A. Dronne, M. Massot, and V. Louvet. Simulation of human ischemic stroke in realistic 3D geometry. *Commun. Nonlinear. Sci. Numer. Simulat.*, 18(6):1539–1557, 2013.

[38] A. Dzougoutov, K.-S. Moon, von Schwerin E., A. Szepessy, and R. Tempone. Adaptive Monte Carlo algorithms for stopped diffusion. *Lect.*, 44:59–88, 2005.

[39] K. D. Elworthy, A. Truman, H. Z. Zhao, and J. G. Gaines. Approximate travelling waves for generalized KPP equations and classical mechanics. *Proc. Roy. Soc. London Ser. A*, 446(1928):529–554, 1994.

[40] A. Ern and M. Vohralik. Adaptive inexact newton methods with a posteriori stopping criteria for nonlinear diffusion pde. *to appear in SIAM J. Sci. Comput.*, 2013.

[41] M. Evans and T. Swartz. *Approximating Integrals via Monte Carlo and Deterministic Methods.* Oxford University Press, Oxford, 2000.

[42] V. Fabian and J. Hannan. *Introduction to Probability and Mathematical Statistics.* John Wiley & Sons, Michigan, 1985.

[43] J. G. Gaines and T. J. Lyons. Variable step size control in the numerical solution of stochastic differential equations. *SIAM J. Appl. Math.*, 57:1455–1484, 1997.

[44] A. Gardoń. The order of approximations for solutions of Itô-type stochastic differential equations with jumps. *Stochastic Anal. Appl.*, 22(3):679–699, 2004.

[45] J. E. Gentle. *Random Number Generation and Monte Carlo Methods.* Springer, New York, 2003.

[46] M. Giles. Multilevel Monte Carlo path simulation. *Operations Research*, 56(3):607–617, 2008.

[47] M. Giles and C. Reisinger. Stochastic finite differences and multilevel Monte Carlo for a class of SPDEs in finance. *SIAM Journal of Financial Mathematics*, 3(1):572–592, 2012.

[48] D. Gillespie. Stochastic simulation of chemical kinetics. *Annu. Rev. Phys. Chem.*, 58, 2007.

[49] P. Glasserman. *Monte Carlo methods in financial engineering*, volume 53 of *Applications of Mathematics (New York).* Springer-Verlag, New York, 2004.

[50] P. Glasserman and N. Merener. Numerical solution of jump-diffusion LIBOR market models. *Finance Stoch.*, 7(1):1–27, 2003.

[51] P. W. Glynn and W. Whitt. The Asymptotic Efficiency of Simulation Estimators. *Operations Research*, 40(3):505–520, 1992.

[52] E. Hairer, S. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I. Nonstiff Problems*, volume 8. Springer Verlag Series in Comput. Math., Berlin, 1993.

[53] E. Hairer and G. Wanner. *Solving ordinary differential equations II. Stiff and differential-algebraic problems.* Springer-Verlag, Berlin and Heidelberg, 1996.

[54] F. B. Hanson. *Applied Stochastic Processes and Control for Jump-Diffusions: Modeling, Analysis and Computation.* Society for Industrial and Applied Mathematics, Chicago, 2007.

[55] R. Hasminskii. *Stochastic stability of differential equations.* Sijthoff and Noordhoff, The Netherlands, 1980.

[56] S. Heinrich. Monte Carlo complexity of global solution of integral equations. *Journal of Complexity*, 14:151–175, 1998.

[57] D. Higham. An algorithmic introduction to numerical simulation of stochastic differential equations. *SIAM Review*, 43(3):525–546, 2001.

[58] D. J. Higham and P. E. Kloeden. Numerical methods for nonlinear stochastic differential equations with jumps. *Numerische Mathematik*, 101:101–119, 2005.

[59] D. J. Higham and P. E. Kloeden. Convergence and stability of implicit methods for jump-diffusion systems. *Int. J. Numer. Anal. Model.*, 3(2):125–140, 2006.

[60] D. J. Higham and X. Mao. Nonnormality and stochastic differential equations. *BIT*, 46(3):525–532, 2006.

[61] J. Hull. *Options, futures, and other derivatives.* Pearson Prentice Hall, New Jersey, 2009.

[62] M. Hutzenthaler, A. Jentzen, and P. Kloeden. Divergence of the multilevel Monte Carlo method. *ArXiv preprint*, 1105.0226, 2011.

[63] S. Ilie and A. Teslya. An adaptive stepsize method for the chemical langevin equation. *J. Chem. Phys.*, 136(184101):1–14, 2012.

[64] M. Jeanblanc, M. Yor, and M. Chesney. *Mathematical Methods for Financial Markets.* Springer, London, 2009.

[65] A. Kebaier. Statistical Romberg Extrapolation: A new variance reduction method and applications to option pricing. *Annals of Applied Probability*, 15(4):2681–2705, 2005.

[66] P. Kloeden and E. Platen. *Numerical solution of stochastic differential equations.* Springer-Verlag, Berlin and New York, 1992.

[67] S. G. Kou. A Jump-Diffusion model for Option Pricing. *Management Science*, 48:1086–1101, 2002.

[68] S. G. Kou and H. Wang. Option Pricing Under a Double Exponential Jump Diffusion Model. *Management Science*, 50:1178–1192, 2004.

[69] A. R. Krommer and C. W. Ueberhuber. *Computational Integration.* Society for Industrial and Applied Mathematics, Philadelphia, 1998.

[70] D. Lamberton and B. Lapeyre. *Introduction to Stochastic Calculus Applied to Finance.* ellipses, Paris, 1997.

[71] X. Q. Liu and C. W. Li. Weak approximation and extrapolations of stochastic differential equations with jumps. *SIAM J. Numer. Anal.*, 37(6):1747–1767, 2000.

[72] G. J. Lord, C. E. Powell, and T. Shardlow. *An Introduction to Computational Stochastic PDEs.* Cambridge University Press, Cambridge, 2014.

[73] Y. Maghsoodi. Mean square efficient numerical solution of jump-diffusion stochastic differential equations. *Sankhyā Ser. A*, 58(1):25–47, 1996.

[74] X. Mao. *Stability of stochastic differential equations with respect to semimartingales.* Longman Scientific and Technical, London, 1991.

[75] X. Mao. Stochastic stabilization and destabilization. *Systems Control Lett.*, 23(4):279–290, 1994.

[76] X. Mao. *Stochastic differential equations and applications.* Horwood, Chichester, 1997.

[77] X. Mao and C. Yuan. *Stochastic Differential Equations with Markovian Switching.* Imperial College Press, London, 2006.

[78] G. Maruyama. Continuous markov processes and stochastic equations. *Rend. Circ. Mat. Palermo*, 4:48–90, 1955.

[79] H. Marxen. The Multilevel Monte Carlo methof used on a Lévy driven SDE. *Monte Carlo methods and Applications*, 16(2):167–190, 2010.

[80] R. C. Merton. Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics*, 3:125–144, 1976.

[81] G. Milstein and M. Tretyakov. *Stochastic Numerics for Mathematical Physics.* Scientific Computing. Springer-Verlag, Berlin and New York, 2004.

[82] K.-S. Moon, A. Szepessy, R. Tempone, and G. E. Zouraris. Convergence rates for adaptive weak approximation of stochastic differential equations. *Stochastic Anal. Appl.*, 23, 2005.

[83] E. Mordecki, A. Szepessy, and R. Tempone. Adaptive weak approximation of diffusions with jumps. *SIAM J. Numer. Anal.*, 4:1732–1768, 2008.

[84] J. Murray. *Mathematical Biology I: An Introduction.* Springer, Seattle, 2002.

[85] B. Pachpatte. *Inequalities for Differential and Integral Equations.* Academic Press, San Diego, 1998.

[86] E. Platen and N. Bruti-Liberati. *Numerical solution of stochastic differential equations with jumps in Finance*, volume 64 of *Stoch. Model. and Appl. Prob.* Springer, Berlin Heidelberg, 2010.

[87] C. Profeta, B. Roynette, and M. Yor. *Option Prices and Probabilities: A New Look at Generalized Black-Scholes Formulae.* Springer, Berlin, 2010.

[88] A. Rathinasamy and K. Balachandran. Mean-square stability of second-order Runge-Kutta methods for multi-dimensional linear stochastic differential systems. *J. Comput. Appl. Math.*, 219(1):170–197, 2008.

[89] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods.* Springer, Paris, second edition, 2004.

[90] A. Rößler. An adaptive discretization algorithm for the weak approximation of stochastic differential equations. *Proc. Appl. Math. Mech.*, 4:19–22, 2004.

[91] P. Rué, J. Villà-Freixa, and K. Burrage. Simulation methods with extended stability for stiff biochemical Kinetics. *BMC Systems Biology*, 4(110):1–13, 2010.

[92] Y. Saito and T. Mitsui. Stability analysis of numerical schemes for stochastic differential equations. *SIAM J. Numer. Anal.*, 33:2254–2267, 1996.

[93] Y. Saito and T. Mitsui. Mean-square stability of numerical schemes for stochastic differential systems. *Vietnam J. Math.*, 30(suppl.):551–560, 2002.

[94] R. Seydel. *Tools for Computational Finance.* Springer, Köln, 2002.

[95] S. E. Shreve. *Stochastic Calculus for Finance II: Continuous-Time Models.* Springer Finance, Pittsburgh, 2004.

[96] A. Szepessy, R. Tempone, and G. E. Zouraris. Adaptive weak approximation of stochastic differential equations. *Comm. Pure Appl. Math.*, 54:1169–1214, 2001.

[97] D. Talay and L. Tubaro. Expansion of the global error for numerical schemes solving stochastic differential equations. *Stochastic Anal. Appl.*, 8(4):483–509 (1991), 1990.

[98] P. Tankov and E. Voltchkova. Jump-diffusion models: a practitioner's guide. *Banque et Marchés*, 2009.

[99] A. Valinejad and S. M. Hosseini. A variable step-size control algorithm for the weak approximation of stochastic differential equations. *Numer Algor*, 55(4):429–446, 2010.

[100] A. Valinejad and S. M. Hosseini. A stepsize control algorithm for SDEs with small noise based on stochastic Runge-Kutta Maruyama methods. *Numer Algor*, 61(3):479–498, 2012.

[101] P. Van der Houwen and B. Sommeijer. On the internal stage Runge-Kutta methods for large m-values. *Z. Angew. Math. Mech.*, 60:479–485, 1980.

[102] E. Vanden-Eijnden. Numerical techniques for multiscale dynamical system with stochastic effects. *Commun. Math. Sci.*, 1:385–391, 2003.

[103] D. Wilkinson. *Stochastic Modelling for Systems Biology.* Chapman and Hall/CRC, 2006.

[104] P. Wilmott, S. Howison, and J. Dewynne. *The Mathematics of Financial Derivatives.* Cambridge University Press, Cambridge, 1997.

[105] Y. Xia. Multilevel Monte Carlo method for jump-diffusion SDEs. Technical report, University of Oxford, http://arxiv.org/abs/1106.4730, 2011.

Curriculum Vitae

Personal Data

Name	Adrian Blumenthal
Date of birth	February 4th, 1985
Nationality	Swiss

Education

2011 - 2015	**PhD in Applied Mathematics** École Polytechnique Fédérale de Lausanne, Switzerland. Thesis advisor: Prof. A. Abdulle.
2009 - 2011	**Master of Science in Applied Mathematics** École Polytechnique Fédérale de Lausanne, Switzerland. Master thesis at Heriot-Watt University Edinburgh, United Kingdom.
2006 - 2009	**Bachelor of Science in Mathematics** École Polytechnique Fédérale de Lausanne, Switzerland. Exchange year at Heriot-Watt University Edinburgh, United Kingdom.

Publications

[1] A. Abdulle and A. Blumenthal. Improved Stabilized Multilevel Monte Carlo Method for Stiff Stochastic Differential Equations. *Lect. Notes Comp. Sci. Eng.*, 103:537–545, 2015.

[2] A. Abdulle and A. Blumenthal. Stabilized multilevel Monte Carlo method for stiff stochastic differential equations. *J. Comput. Phys.*, 251:445–460, 2013.

[3] A. Abdulle, A. Blumenthal, and E. Buckwar. The multilevel Monte Carlo Method for Stochastic Differential Equations driven by Jump-Diffusion Processes. *MATHICSE Technical Report*, 2011.

[4] A. Abdulle and A. Blumenthal. S-ROCK Methods for Stiff Stochastic Differential Equations driven by Jump-Diffusion Processes. *Preprint*, 2015.

[5] A. Abdulle and A. Blumenthal. Variable Time Stepping S-ROCK Methods for Weak Solutions of Stiff Stochastic Differential Equations. *Preprint*, 2015.

Academic Distinctions

- *Bourse d'excellence au niveau Master*, École Polytechnique Fédérale de Lausanne, Switzerland.
- *Roderick MacLeod Shearer Memorial Prize for excellence in mathematics in the final honours course*, Heriot-Watt University Edinburgh, United Kingdom.

Presentations

- Swiss Numerics Colloquium (contributed talk), Lausanne, Switzerland, 2013.
  Stabilized Multilevel Monte Carlo Method for Stiff Stochastic Differential Equations.
- Doctoral Seminar EPFL (invited talk), Lausanne, Switzerland, 2013.
  Efficient Numerical Methods for Stochastic Differential Equations.
- ENUMATH conference (contributed talk), Lausanne, Switzerland, 2013.
  Stabilized Multilevel Monte Carlo Method for Stiff Stochastic Problems.
- SciCADE (minisymposium), Valladolid, Spain, 2013.
  Solving Stiff Stochastic Differential Equations with a Stabilized Multilevel Monte Carlo Method.
- University of Rennes (invited talk), Rennes, France, 2013.
  Une nouvelle méthode multilevel Monte Carlo stabilisée pour des problèmes stochastiques raides.
- Swiss Numerics Colloquium (poster), Zurich, Switzerland, 2014.
  Stabilized Multilevel Monte Carlo Method for Stochastic Differential Equations.