ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Numerical Simulations of the Weak Approximation Error for Parabolic Stochastic Partial Differential Equations

Master Thesis

Helga Thum

August 2013

Advisor: Prof. Dr. Arnulf Jentzen

Department of Mathematics, ETH Zürich

Abstract

We consider the stochastic heat equation with additive and multiplicative space-time white noise and look at a Galerkin spectral method combined with a linear implicit Euler. The goal of this thesis is to observe how the weak approximation error of a stochastic partial differential equation with nonlinear diffusion coefficient behaves. We expect to see a weak convergence rate of $1 - \varepsilon$.
First the theoretical background of SPDEs from e.g. [DPZ92] [PR07] is recapitulated. Then the numerical scheme is presented and theoretically proven results from the literatur on error convergence rates are reviewed. Finally, we use numerical experiments and simulations to look at the convergence of the weak approximation error for the different diffusion coefficient functions $b(x,y) = 1$, $b(x,y) = 1 - \frac{y}{5}$, $b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$ for $x \in (0,1)$ and $y \in \mathbb{R}$. In the case of the nonlinear diffusion coefficient we observe that a rate of $1 - \varepsilon$ could be possible.

Acknowledgements

I would like to thank my supervisor Prof. Dr. Arnulf Jentzen for the support he gave me while writing this thesis. Without his help, comments and support it would not have been possible for me to write this thesis.
Additionally I would like to thank my partner and friends for their support, help and proofreading.

Contents

List of Figures

List of Tables

Introduction

In this thesis we want to take a closer look at the weak approximation error for parabolic stochastic partial differential equations (SPDEs). Working on the numerical approximation for SPDEs we face many difficulties. On the one hand we have to consider problems known from numerically solving deterministic partial differential equations. On the other hand we are faced with problems triggered by numerically solving stochastic ordinary differential equations (SODEs). And additionally new issues arise resulting from the infinite dimensional nature of the underlying noise processes. (cf. [JK11])
For an overview of the up to now published papers in this topic we refer to the papers cited in [GK03] or the survey article [JK09].
Stochastic partial differential equations are used as a model in many applications. This area of mathematics is especially motivated by the need to describe random phenomena studied in natural sciences like physics, chemistry, biology, and in control theory [DPZ92].
So, how can we define SPDEs? Basically, we combine deterministic partial differential equations with some kind of noise. We then get equations of the form

$$dX(t) = (AX(t) + F(X(t))dt + B(X(t))dW(t).$$

This will be described in Chapter 1. In this thesis, we concentrate on the stochastic heat equation. Thus, we simplify the above equation to

$$dX(t) = \Delta X(t)dt + B(X(t))dW(t),$$

where B is a multiplication operator of the form

$$(B(v)u)(x) = b(x, v(x)) \cdot u(x)$$

(see Section 1.3 for details).
Taking a closer look at the noise in this equation we see that we can split it

into two types, additive and multiplicative noise. We speak of *additive noise* if the operator B is a constant operator and of *multiplicative noise* if B is not constant.
Our goal in this thesis is to simulate numerically the weak approximation error

$$\varepsilon_N = \left| \mathbb{E}\left[\varphi(X_T)\right] - \mathbb{E}\left[\varphi(Y_T^N)\right] \right|$$

(for an exact definition see Section 2.4.2). The weak error of numerical methods for SPDEs is already the topic of many papers. As examples we can list [DP09] [Deb11] [LS13] [KLL11] [KLL12] [Bré12] [DBD06] [GKL09] [Hau03b] [Hau10] [DPJR10] [Kru12]. But to the best of my knowledge no simulations were made in any of these papers. This might be due to the computational effort necessary for Monte Carlo and Multilevel Monte Carlo methods. However, in the case of multiplicative noise it makes sense to use such methods as it is in general no longer possible to calculate exact solutions of these SPDEs explicitly.
Additionally, there are no theoretical results known for the weak approximation error of this kind of numerical schemes for SPDEs with nonlinear diffusion coefficients. We find results for different diffusion coefficients in [Deb11] or [AL12]. However, for the non-linear case the assumptions made in these papers are very restrictive.
We study in Section 3.3 two different cases. First, a linear diffusion coefficient

$$b(x,y) = 1 - \frac{y}{5}$$

and then a nonlinear diffusion coefficient

$$b(x,y) = \frac{1 - \frac{y}{5}}{1 + y^2}.$$

Especially for the second case the there are no theoretical results on the weak approximation error available.
Our results coincide for additive noise and multiplicative noise with linear diffusion coefficient with the results found e.g. in [WG12] [Deb11] [AL12]. We are able to observe a behaviour that may let us suspect a convergence rate $1 - \varepsilon$ for the weak approximation error of the stochastic heat equation. Our main result is the following: we observe that in simulations a similar result could be possible for the multiplicative noise with nonlinear diffusion coefficient.
To achieve these results we are not only using the classical Monte Carlo and Multilevel Monte Carlo schemes, but also the Difference Monte Carlo scheme. The scheme will be shown in detail in Section 3.2.4.

Structure of the Thesis

Starting with a bit of theoretical background we will recapitulate in Chapter 1 some important definitions and results concerning Hilbert space valued Wiener Processes, stochastic integrals with respect to these processes and the different kind of solutions of SPDEs. The chapter is not intended to be a self-contained introduction into this field of mathematics. Such an introduction can be found e.g. in [PR07] or [DPZ92], for example. The goal is to introduce the notation used here and give some results needed later on in the thesis. Additionally, the SPDE setting used in this thesis will be described.
In Chapter 2 we take a detailed look at the numerical methods used in the simulations. The basic numerical method is a spectral Galerkin method in space combined with the Linear Implicit Euler method in time. This discretization is then combined with Monte Carlo or Multilevel Monte Carlo schemes. The basic idea of these schemes is quite simple. We take a certain number of independent samples of the desired value, sum them up and divide by the number of samples. Thus, we are able to approximate the expected value of this random variable. We will take a brief look at the performance of these schemes. In addition, in this chapter we will take a look at two different types of approximation errors, the strong and the weak one. We will summarize a few theoretical results on these errors for SPDEs.
Finally, in Chapter 3 we describe our numerical experiments. We simulate the cases of additive and multiplicative noise. The case of additive noise is used to test our numerical scheme against former results. Here, the Difference Monte Carlo scheme is described. The idea is to calculate for different grid sizes an approximate solution and a reference solution on a grid with double the number of points and approximating this in one step with Monte Carlo. Thus, we can use that the random variables used in one sample are not independent and can reduce the number of samples needed.
With all these approaches we simulate the SPDE introduced in the setting in Section 1.3. It seems as if the convergence rate $1 - \varepsilon$ could be achieved.

Chapter 1

Stochastic Partial Differential Equations (SPDEs)

With this chapter we aim at giving a short introduction to stochastic partial differential equations (SPDEs). We will go mainly along the lines of [PR07] and [DPZ92] and use also [Kru12] who gave a self-contained and short introduction using the two sources mentioned before.
First we look at Hilbert space-valued Wiener processes. Then we define the stochastic integral with respect to such processes and look at different kinds of solutions for SPDEs.
Finally we introduce the SPDE setting used further on in this thesis.

1.1 Infinite dimensional Wiener Processes

Let $(H, \langle\cdot,\cdot\rangle_H, \|\cdot\|_H)$ and $(U, \langle\cdot,\cdot\rangle_U, \|\cdot\|_U)$ be separable Hilbert spaces and let $(\Omega, \mathcal{F}, P)$ be a probability space with a normal filtration $(\mathcal{F}_t)_{t\geq 0}$. We consider a operator Q that has to be linear, bounded, self-adjoint and positive definite. This gives us an eigenbasis $(e_k)_{k\in\mathbb{N}}$ and eigenvalues $(\mu_k)_{k\in\mathbb{N}}$ for this operator. Then we define the following.

Definition 1.1 *[DPZ92, page 86] A stochastic process $W : [0,T]\times\Omega \to U$ is called a* (standard) Q-Wiener process *on $(\Omega, \mathcal{F}, P)$ if*

(i) $W(0) = 0$,

(ii) *W has P-a.s. continuous trajectories,*

(iii) *W has independent increments and*

(iv) $\forall\ 0 \leq s < t \leq T$, $W(t) - W(s)$ *is distributed like a Gaussian random variable with mean* 0 *and covariance* $(t-s)Q$.

Very useful is the following representation of a Q-Wiener process.

Theorem 1.2 *[PR07, Prop. 2.1.10] Let $(e_k)_{k\in\mathbb{N}}$ be an orthonormal basis of U consisting of eigenvectors of Q with corresponding eigenvalues $(\mu_k)_{k\in\mathbb{N}}$. Then an U-valued stochastic process $W(t), t \in [0,T]$, is a Q-Wiener process if and only if*

$$W(t) = \sum_{k\in\mathbb{N}} \sqrt{\mu_k}\beta_k(t)e_k, \qquad t \in [0,T], \tag{1.1}$$

where $\beta_k, k \in \{n \in \mathbb{N} \mid \mu_n > 0\}$, are independent real-valued Brownian motions on a probability space $(\Omega, \mathcal{F}, P)$.
The series (1.1) even converges in $L^2(\Omega, \mathcal{F}, P; C([0,T], U))$, and thus always has a P-a.s. continuous modification. In particular, for any Q as above there exists a Q-Wiener process on U.

The proof can be found e.g. in [PR07, Prop. 2.1.10]. This result needs the assumption that the operator $Q \in L(U)$ has finite trace. Especially if $Q = Id_U$ this is not the case. However, this is also an important case. A Wiener process with such a covariance matrix is called *white noise* (cf. e.g. [Kru12, page 20]).
To define Wiener processes for more general covariance operators $Q \in L(U)$ we could e.g. look at [DPZ92, Ch. 4.3.1] or [PR07, Ch. 2.5.1]. First we need the following result.

Theorem 1.3 *[PR07, Prop. 2.3.4] If $Q \in L(U)$ is nonnegative and symmetric then there exists exactly one element $Q^{\frac{1}{2}} \in L(U)$ nonnegative and symmetric such that $Q^{\frac{1}{2}} \circ Q^{\frac{1}{2}} = Q$.*
If, in addition, tr $Q < \infty$ we have that $Q^{\frac{1}{2}} \in L^2(U)$ where $\left\|Q^{\frac{1}{2}}\right\|^2_{HS(U,H)} = tr\ Q$ and of course $L \circ Q^{\frac{1}{2}} \in HS(U,H)$ for all $L \in L(U,H)$.

In [PR07] it is stated that the proof can be found in [RS72]. With this result we are able to define two useful spaces. The space $U_0 = Q^{\frac{1}{2}}(U)$ with the inner product $\langle u_0, v_0\rangle_{U_0} = \left\langle Q^{-\frac{1}{2}}u_0, Q^{-\frac{1}{2}}v_0\right\rangle_U$, for all $u_0, v_0 \in U_0$ and the space $(U_1, \langle\cdot,\cdot\rangle_{U_1}, \|\cdot\|_{U_1})$. The second space is defined through the Hilbert-Schmidt embedding $J : U_0 \to U_1$. Such an embedding always exists. A proof can e.g. be found in [PR07, Remark 2.5.1]. Finally we can define a so called *cylindrical Q-Wiener process* using the following theorem.

Theorem 1.4 *[PR07, Prop. 2.5.2] Let $(e_k)_{k\in\mathbb{N}}$ be an orthonormal basis of $U_0 = Q^{\frac{1}{2}}(U)$ and $(\beta_k)_{k\in\mathbb{N}}$ a family of independent real-valued Brownian motions. Define $Q_1 = JJ^*$. Then $Q_1 \in L(U_1)$, nonnegative definite and symmetric with finite trace and the series*

$$W(t) = \sum_{k=1}^{\infty} \beta_k(t)Je_k, \qquad t \in [0,T],$$

converges in $\mathcal{M}_T^2(U_1)$ *and defines a* Q_1*-Wiener process on* U_1*. Moreover, we have that* $Q_1^{\frac{1}{2}}(U_1) = J(U_0)$ *and for all* $u_0 \in U_0$

$$\|u_0\|_{U_0} = \left\| Q_1^{-\frac{1}{2}} J u_0 \right\|_{U_1} = \|Ju_0\|_{Q_1^{\frac{1}{2}} U_1},$$

i.e. $J : U_0 \to Q_1^{\frac{1}{2}} U_1$ *is an isometry.*

1.2 Stochastic Integral

Now we take a look at the H-valued stochastic Itô-integral. For every stochastic process $\Phi : [0,T] \times \Omega \to L(U,H)$ we denote the stochastic integral with respect to the Wiener process W as

$$\int_0^t \Phi(\sigma) dW(\sigma). \tag{1.2}$$

A detailed construction of this integral can be found e.g. in [PR07, Ch. 2.3.2] or [DPZ92, Ch. 4.2]. The idea is to first look at *elementary integrands*.

Definition 1.5 *[PR07, Def. 2.3.1] An* $L(U,H)$*-valued process* $\Phi(t), t \in [0,T]$*, on* $(\Omega, \mathcal{F}, P]$ *with normal filtration* $(\mathcal{F}_t)_{t\in[0,T]}$ *is said to be* elementary *if there exists a partition* $0 = t_0 < \ldots < t_k = T, k \in \mathbb{N}$*, such that*

$$\Phi(t) = \sum_{i=0}^{n-1} \Phi_i \mathbb{1}_{(t_i, t_i+1]}(t), \qquad t \in [0,T],$$

where

- $\Phi_i : \Omega \to L(U,H)$ *is* $\mathcal{F}_{t_i}$*-measurable, w.r.t. strong Borel* σ*-algebra on* $L(U,H)$*,* $0 \le m \le k-1$*,*
- Φ_i *takes only a finite number of values in* $L(U,H)$*,* $0 \le m \le k-1$*.*

In that case the stochastic integral can be defined as

$$\int_0^t \Phi(\sigma) dW(\sigma) = \sum_{i=0}^{n-1} \Phi_i (W(t_{i+1} \wedge t) - W(t_i \wedge t)), \qquad t \in [0,T].$$

As a second step a norm on the set of elementary integrands needs to be defined such that we get an isometry between that space and the space of square-integrable martingales. This then implies that the definition of the stochastic integral can be extended to an abstract completion of the space

of elementary integrands. And finally, the definition can be extended to all stochastically integrable processes on $[0,T]$ defined by

$$\mathcal{N}_W(0,T;H) = \left\{ \Phi : \Omega_T \to L^2(U_0,H) \;\middle|\; \Phi \text{ is predictable with } P\left(\int_0^T \|\Phi(s)\|^2_{L^2(U_0,H)}\, ds < 0 \right) = 1 \right\}$$

(see e.g. [PR07, page 22ff] for further details).
The following properties will be useful in the remainder of this thesis.

Theorem 1.6 *For all $\Phi \in \mathcal{N}_W(0,T;H)$ with $\int_0^T \mathbb{E}[\|\Phi(\sigma)\|^2_{L^2(U_0,H)}]d\sigma < \infty$*

- *[PR07, Prop. 2.3.5]* Itô's isometry

$$\begin{aligned} \mathbb{E}\left[\left\| \int_0^T \Phi(\sigma) dW(\sigma) \right\|^2_H \right] &= \mathbb{E}\left[\int_0^T \|\Phi(\sigma)\|^2_{L^2(U_0,H)}\, d\sigma \right] \\ &= \int_0^t \|\Phi(\sigma)\|^2_{L^2(U_0,H)}\, d\sigma \end{aligned} \tag{1.3}$$

- *[DPZ92, Prop. 4.13]* Expectation

$$\mathbb{E}\left[\int_0^t \Phi(\sigma) dW(\sigma) \right] = 0 \tag{1.4}$$

This definition of the stochastic integral can again be expanded to the stochastic Itô-integrals with respect to cylindrical Wiener processes. Details can be found e.g. in [PR07, Ch. 2.5.2] or [DPZ92, Ch. 4.3.2].

1.3 The SPDE-Setting considered in this thesis

Finally we get to the stochastic partial differential equations. From now on we will use the now described setting on $(0,1)$. It can e.g. be found in [JK11].
Let $T \in (0,\infty)$, let $H = U = L^2((0,1),\mathbb{R})$ and let $(\Omega,\mathcal{F},P)$ be a probability space with a normal filtration $(\mathcal{F}_t)_{t\in[0,T]}$. From now on we fix $Q = Id_H$ and let $(W_t)_{t\in[0,T]}$ be a cylindrical Q-Wiener process with respect to the filtration $(\mathcal{F}_t)_{t\in[0,T]}$. We consider the stochastic partial differential equation (SPDE)

$$\begin{aligned} dX_t &= [AX_t + F(X_t)]\, dt + B(X_t) dW_t, \\ X(0) &= \xi, \end{aligned} \tag{1.5}$$

for $t \in [0,T]$. Consider the following assumptions:

Assumption 1.7 *[JK11, cf. Section 5.3]*

- ***linear operator*** *A: Let $A : D(A) \subset H \to H$ be equal to the Laplacian operator with Dirichlet boundary conditions times a constant $\kappa \in (0,\infty)$. For this operator we get the eigenvalues $\lambda_j \in \mathbb{R}, j \in \mathbb{N}$, given by*

$$\lambda_j = \kappa \pi^2 j^2 \tag{1.6}$$

and the eigenfunctions $e_j \in H, j \in \mathbb{N}$, given by

$$e_j(x) = \sqrt{2} \sin(j\pi x) \tag{1.7}$$

for $x \in (0,1)$.
Then we can write the linear operator $A : D(A) \subset H \to H$ as

$$Av = \sum_{i \in I} \lambda_i \langle e_i, v \rangle_H e_i$$

for all $v \in D(A)$ with $D(A) = \left\{ w \in H \;\middle|\; \sum_{i \in I} |\lambda_i|^2 |\langle e_i, w \rangle_H|^2 < \infty \right\}$.
Additionally, define for $r \in \mathbb{R}$ the real Hilbert spaces of domains of fractional powers of the linear operator $-A : D(A) \subset H \to H$ by $H_r := D((-A)^r)$ with norm $\|\cdot\|_{H^r} := \|(-A)^r(\cdot)\|_H$.

- ***drift term*** *F: Let $c \in [0,\infty)$ be a real number. Let $f : (0,1) \times \mathbb{R} \to \mathbb{R}$ be a Borel measurable function with*

$$\int_0^1 |f(x,0)|^2 \, dx < \infty$$
$$|f(x,y_1) - f(x,y_2)| \leq c\,|y_1 - y_2|$$

for all $x \in (0,1)$ and all $y_1, y_2 \in \mathbb{R}$. Then $F : H \to H$

$$(F(v))(x) = f(x, v(x))$$

for all $x \in (0,1)$, $v \in H$ defines a globally Lipschitz continuous mapping. This operator F is known as Nemytskii operator.

- ***diffusion term*** *B: Let $\beta \in (-\frac{1}{2}, -\frac{1}{4})$ and let $c \in [0,\infty)$. Let $b : (0,1) \times \mathbb{R} \to \mathbb{R}$ be a Borel measurable function with*

$$\int_0^1 |b(x,0)|^2 \, dx < \infty$$
$$|b(x,y_1) - b(x,y_2)| \leq c\,|y_1 - y_2|$$

for all $x \in (0,1)$ and all $y_1, y_2 \in \mathbb{R}$.
Define $B : H \to HS(H, H_\beta)$ by

$$(B(v)u)(w) = \int_0^1 b(x, v(x)) \cdot u(x) \cdot w(x) dx$$

for all $u, v \in H$, $w \in H_{-\beta}$ as a globally Lipschitz continuous mapping.

- ***initial value*** *ξ: The treatment of the initial condition is not the main difficulty in the treatment of SPDEs. For our setting we choose to set the initial value to zero, $\xi \equiv 0$ (cf. e.g. [Kru12, page 132]).*

1.3.1 Notions of solutions

Such SPDEs can have different notions of solutions. For instance, in [PR07] we find

Definition 1.8 *[PR07, Def. F.0.2] A $D(A)$-valued predictable process $X(t), t \in [0,T]$ is called an* analytically strong solution *of problem* (1.5) *if*

$$X(t) = \int_0^t AX_s + F(X_s) ds + \int_0^t B(X_s) dW_s, \qquad P - a.s.,$$

for each $t \in [0,T]$. In particular, the integrals on the right-hand side have to be well-defined.

Definition 1.9 *[PR07, Def. F.0.3] A H-valued predictable process $X(t), t \in [0,T]$ is called an* analytically weak solution *of problem* (1.5) *if*

$$\langle X(t), \zeta \rangle = \int_0^t \langle X(s), A^*\zeta \rangle + \langle F(X_s), \zeta \rangle \, ds + \int_0^t \langle \zeta, B(X_s) dW_s \rangle, \quad P - a.s.,$$

for each $t \in [0,T]$ and $\zeta \in D(A^*)$. In particular, the integrals on the right-hand side have to be well-defined.

Definition 1.10 *[PR07, Def. F.0.1] A H-valued predictable process $X(t), t \in [0,T]$ is called a* mild solution *of problem* (1.5) *if*

$$X(t) = \int_0^t e^{A(t-s)} F(X_s) ds + \int_0^t e^{A(t-s)} B(X_s) dW_s, \qquad P - a.s., \tag{1.8}$$

for each $t \in [0,T]$. In particular, the integrals on the right-hand side have to be well-defined.

With the above assumptions we get e.g. by [JK11, Theorem 5.1] that

$$dX_t(x) = [\kappa \Delta X_t(x) + f(x, X_t(x))]\, dt + b(x, X_t(x)) dW_t(x) \tag{1.9}$$

with $X_0(x) = 0$ and $X_t(0) = X_t(1) = 0$ for $x \in (0,1), t \in [0,T]$ has a unique (up to modifications) mild solution $X : [0,T] \times \Omega \to H_{\beta+\frac{1}{2}}$.
Mostly, we will use the case of $f(x,y) = 0$ for all $x \in (0,1)$, $y \in \mathbb{R}$. For b we choose different functions. In the case $b(x,y) = 1$ for all $x \in (0,1)$, $y \in \mathbb{R}$ we get the stochastic heat equation with additive noise, and with $b(x,y) = 1 - \frac{y}{5}$ for all $x \in (0,1)$, $y \in \mathbb{R}$ and $b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$ for all $x \in (0,1)$, $y \in \mathbb{R}$ we get multiplicative noise.
In summary we can conclude that we get the following three settings:

$$dX_t(x) = \kappa \Delta X_t(x) dt + dW_t(x) \tag{1.10}$$

$$dX_t(x) = \kappa \Delta X_t(x) dt + \frac{X_t(x)}{5} dW_t(x) \tag{1.11}$$

$$dX_t(x) = \kappa \Delta X_t(x) dt + \frac{1 - \frac{X_t(x)}{5}}{1 + X_t(x)^2} dW_t(x) \tag{1.12}$$

for $x \in (0,1)$, $t \in [0,T]$.

Chapter 2

Numerical Discretisation Methods for Stochastic Partial Differential Equations

In this chapter we again look at the setting introduced in Section 1.3. Now we want to revisit a method for the numerical discretization of this specific SPDE.
First, we will take a look at the spatial and noise discretization, in particular the spectral Galerkin method. Then we will add a temporal discretization and consider a fully discrete scheme for the stochastic heat equation with space-time white noise.
Additionally, we will present different ways to look at the error of a numerical scheme.
Finally, the classical Monte Carlo method and the Multilevel Monte Carlo method are presented.

2.1 Spatial and Noise Discretization

Spatial discretizations are normally achieved by finite elements methods, finite difference methods or spectral Galerkin methods. Examples of finite elements and finite differences can for example be found in [ANZ98], [BTZ04], [Hau08], [Rot02], [Wal05], [GKL09]. Here we only use the spectral Galerkin method and give a short introduction to this method (see e.g. [GK96]).

Spectral Galerkin Method

Again we consider the setting of Section 1.3 and thus the SPDE

$$dX_t(x) = [AX_t(x) + f(x, X_t(x))]\,dt + b(x, X_t(x))dW_t(x), \tag{2.1}$$

where $(W_t)_{t\in[0,T]}$ is a cylindrical Id-Wiener process with respect to $(\mathcal{F}_t)_{t\in[0,T]}$ (see Section 1.3.1). As introduced in Section 1.3 we will use the linear operator $A = \kappa\left(\frac{\partial^2}{\partial x^2}\right)$. Using its eigenfunctions 1.7 as an orthonormal basis we can write every function as

$$v = \sum_{n=1}^{\infty} \langle e_n, v\rangle\, e_n.$$

Thus, we can use the projection $P_N : H \to H, N \in \mathbb{N}$, defined by

$$P_N(v) = \sum_{n=1}^{N} \langle e_n, v\rangle\, e_n$$

for all $v \in H, N \in \mathbb{N}$, to project each of these functions on the N-dimensional subspace $S_N \subset H, N \in \mathbb{N}$, defined by

$$S_N = P_N(H) = \left\{ \sum_{n=1}^{N} c_n e_n \;\middle|\; c_n \in \mathbb{R} \right\} \cong \mathbb{R}^N$$

spanned by the first N eigenfunctions of A.
If we now look at the solution X_t of our SPDE we get the stochastic process $X^N : [0,T] \times \Omega \to S_N, N \in \mathbb{N}$

$$X_t^N = e^{At}P_N(X_0) + \int_0^t e^{A(t-s)}F_N(X_s^N)ds + \int_0^t e^{A(t-s)}B_N(X_s^N)dW_s, \quad P-a.s.,$$

where $F_N : S_N \to S_N$ is defined by

$$F_N(v) = P_N(F_N(v))$$

and $B_N : S_N \to HS(H, S_N)$ by

$$B_N(v)u = P_N(B(v)(P_N(u)).$$

It is important to notice that this is not the same as just projecting the solution X_t onto the space S_N as we would then get

$$P_N(X_t) = e^{At}P_N(X_0) + \int_0^t e^{A(t-s)}P_N(F(X_s))ds + \int_0^t e^{A(t-s)}P_N(B(X_s)dW_s),$$

P-a.s.
The first version can also be written in the form of a finite dimensional SPDE. We write the N-dimensional Galerkin SDE that corresponds to the SPDE (2.1) as

$$dX_t^N = \left(AX_t^N + F_N(X_t^N)\right) dt + B_N(X_t^N)dW_t. \tag{2.2}$$

2.2 Temporal Discretization

After the spatial discretization we now need a temporal discretization to get a completely discretized numerical scheme. This can be found for instance in [Tho97] and [Kru12]. First, we define a rational function approximating the exponential function.

Definition 2.1 *A rational function $r : \mathbb{C} \to \mathbb{C}$ is called a rational approximation of the exponential e^λ of order $p \geq 1$ if*

$$r(\lambda) = e^\lambda + \mathcal{O}\left(\lambda^{p+1}\right). \tag{2.3}$$

Using such a rational function we can now take a look at the homogeneous heat equation

$$dX_t = AX_t, \qquad\qquad D \times [0, \infty)$$

where A is the Laplacian operator with Dirichlet boundary conditions and D is a bounded domain in $\mathbb{R}^d$. With the technique described in Section 2.1 we get a semidiscrete version of this equation reading as

$$dX_t^N = A_N X_t^N. \tag{2.4}$$

Our goal is now to further discretize this in time. We first define time steps of size Δt. Thus, we want to define the solution X_k^N at time $t_k = k \cdot \Delta t, k = 1, \dots K$. As described e.g. in [Tho97] we can assume that the solution operator of equation (2.4) has to be the exponential $e^{(tA_N)}$. Therefore, it makes sense to define a discrete solution through

$$X_{k+1}^N = r(\Delta t A_N) X_k^N.$$

Here, r is the rational approximation (2.3).
For choosing this function r there are different possibilities. One would be

$$r(x) = \frac{1}{1-x}$$

which leads to the implicit Euler approximation used in the remainder of this thesis. Other possibilities would be

$$r(x) = 1 + x$$

for the explicit Euler or

$$r(x) = \frac{1 + \frac{1}{2}x}{1 - \frac{1}{2}x}$$

for the Crank-Nicolson scheme.
This technique can be expanded to our general problem (2.1).

2.3 Fully Discrete Scheme

Combining the Galerkin method and the linear implicit Euler scheme we get the scheme (cf. e.g. in [JR10]) $Y^{N,K} : \Omega \times [0,T] \to H, N, K \in \mathbb{N}$ with

$$Y^{N,K}_{t_{k+1}} = P_N \left(I - \frac{T}{K}A\right)^{-1} \left(Y^{N,K}_{t_k} + \frac{T}{K} \cdot F_N(Y^{N,K}_{t_k})\right) \tag{2.5}$$

$$+ \int_{t_k}^{t_{k+1}} P_N \left(I - \frac{T}{K}A\right)^{-1} B_N(Y^{N,K}_{t_k}) dW_s, \tag{2.6}$$

where $t_k = \frac{kT}{K}$. Thus, in MATLAB we get the following basic scheme. It is a slightly changed version of the code in [JR10].

```
Y = zeros(1,N); %initalisation for process
A = -pi^2*(1:N).^2/100;
for k = 1:K %for time stepping
    y = dst(Y) * sqrt(2);
    dW = dst( randn(1,N) .* sqrt(2/K));
    y = y + f(y)/K + b(y).*dW;
    Y = idst(y) / sqrt(2) ./ (1-A/K);
end
z=sum(Y.^2); %norm
```

We want to have a closer look at this numerical scheme. First we show an example.

Example 2.2 (Stochastic Heat Equation) *As introduced in 1.3 we look at the stochastic heat equation*

$$dX_t(x) = \left[\kappa \left(\frac{\partial^2}{\partial x^2}\right) X_t(x) + f(x, X_t(x))\right] dt + b(x, X_t(x)) dW_t(x)$$

where κ is a small parameter, we will have $\kappa = \frac{1}{100}$. Using

$$f(x,y) = 0$$
$$A = \kappa \frac{\partial^2}{\partial x^2}$$

for all $x \in (0,1)$ in equation (2.1) and apply scheme (2.5) we can plot random paths. In Figure 2.1 we see this for three different choices of the function b. In (1) $b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$ was used, in (2) $b(x,y) = 1 - \frac{y}{5}$ and in (3) $b(x,y) = 1$. The behaviour of these realizations is in general very similar. These will also be the choices of b used later on in this thesis.

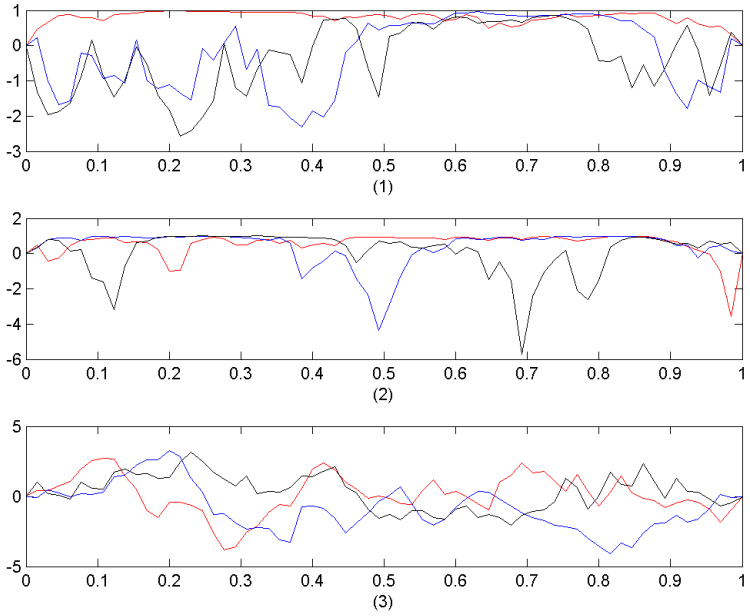

Figure 2.1: Realizations for the Stochastic Heat Equation

2.4 Error Analysis

Here we give a short review on results in literature on error analysis for SPDEs.

2.4.1 Strong Approximation Error

The strong approximation is e. g. defined in [BJLS12].

Definition 2.3 *[BJLS12, page 102f] Let $(Y^N)_{N\in\mathbb{N}}$ and X be stochastic processes. $(Y^N)_{N\in\mathbb{N}}$ converges to X at time T in the strong L^p-sense if*

$$\lim_{N\to\infty} \mathbb{E}\left[\left\|X_T - Y_T^N\right\|_H^p\right] = 0.$$

In addition, $(Y^N)_{N\in\mathbb{N}}$ converges to X at time T in the strong L^p-sense with order α if there exists a real number $C \in [0,\infty)$ such that for every $N \in \mathbb{N}$ it holds that

$$\left\|X_T - Y_T^N\right\|_{L^p(\Omega,H)} \leq \frac{C}{N^\alpha}.$$

This strong error for SPDEs is the subject of many papers. It is, for example, analysed in [KLL10] [Kru12] and [Hau03a]. The settings in these papers are somewhat different from the one considered here. However, we can state the following well known facts. For the spatially discretized X_t^N defined in (2.2) we get this (cf. e.g. [GK96]).

Theorem 2.4 *Assume the setting in Section 1.3. For every $\varepsilon \in (0, \frac{1}{2})$ and every $p \in [0,\infty)$ there exists a real number $C_{\varepsilon,p} \in [0,\infty)$ such that for every $N \in \mathbb{N}$ it holds that*

$$\left\|X_T - X_T^N\right\|_{L^p(\Omega,H)} \leq \frac{C_{\varepsilon,p}}{N^{\left(\frac{1}{2}-\varepsilon\right)}}. \tag{2.7}$$

If we look at the fully discretized scheme we find (cf. e.g. [Wal05])

Theorem 2.5 *Assume the setting in Section 1.3. For every $\varepsilon \in (0, \frac{1}{2})$ and every $p \in [0,\infty)$ there exists a real number $C_{\varepsilon,p} \in [0,\infty)$ such that for every $N \in \mathbb{N}$ and every $K \in \mathbb{N}$ it holds that*

$$\left\|X_T - Y_T^{N,K}\right\|_{L^p(\Omega,H)} \leq C_{\varepsilon,p}\left(\frac{1}{N^{\left(\frac{1}{2}-\varepsilon\right)}} + \frac{1}{K^{\left(\frac{1}{4}-\varepsilon\right)}}\right). \tag{2.8}$$

2.4.2 Weak Approximation Error

More important for this thesis is, however, the weak error. Again e.g. in [BJLS12] we find this definition:

Definition 2.6 *[BJLS12, page 103] Let $(Y^N)_{N\in\mathbb{N}}$ and X be stochastic processes. $(Y^N)_{N\in\mathbb{N}}$ converges to X at time T in the numerically weak sense if it holds for every infinitely often differentiable function $\varphi : H \to \mathbb{R}$*

$$\lim_{N\to\infty} \mathbb{E}\left[\varphi(Y_T^N)\right] = \mathbb{E}\left[\varphi(X_T)\right].$$

In addition, $(Y^N)_{N\in\mathbb{N}}$ converges to X at time T in the numerically weak sense with order α if for the same φ there exists a real number $C \in [0,\infty)$ such that for every $N \in \mathbb{N}$ it holds that

$$\left|\mathbb{E}\left[\varphi(X_T)\right] - \mathbb{E}\left[\varphi(Y_T^N)\right]\right| \leq \frac{C}{N^\alpha}$$

This weak error for stochastic partial differential equations is the topic of many papers, e.g. [DP09] [Deb11] [LS13] [KLL11] [KLL12] [Bré12] [DBD06] [GKL09] [Hau03b] [Hau10] [Kru12] [DPJR10]. So why look at it again in detail? There are mainly two reasons:

- *Simulation:* To the best of my knowledge none of these papers contain simulations for SPDEs using some kind of Monte Carlo scheme. In [Kru12] some numerical experiments are made. However, they use the special properties of the stochastic heat equation and of geometric Brownian motion to explicitly calculate a specific solutions for these equations and eliminate any randomness. Here, this is also used as a first step in Section 3.2.1 and 3.2.2. But for the multiplicative noise, especially with nonlinear diffusion coefficient, this approach is in general no longer possible. Therefore, in Section 3.3 we have to use a Monte Carlo method.
- *Nonlinear Case:* Most of the papers dealing with the weak approximation error for SPDEs restrict themselves to the case of additive noise. Even papers looking at the case of multiplicative noise, like [Deb11], make very restrictive assumptions. This can for example be found in Remark 2.3 in [Deb11] right after the main result Theorem 2.2. The remark restricts the results found in that paper to affine linear diffusion coefficients. These results are also cited in Theorem 2.7 in this thesis. Later on in Section 3.3 we will choose the nonlinear diffusion coefficient

  $$b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$$

  and use simulations to look at the weak approximation error.

Before going on we want to collect some results about the weak approximation error. In [Deb11] we find the following.

Theorem 2.7 *[Deb11, Theorem 2.2] Assume that F and B are C_b^3 functions from H to H and $L(H)$. Additionally, B satisfies for any $x, y, h \in H$*

$$\begin{aligned}\|B(x)\|_{L(H)} &\leq C(\|x\|_H + 1)\\ \|B(x) - B(y)\|_{L(H)} &\leq C\|x - y\|_H\\ \|B''(x)\cdot(h,h)\|_{L(H)} &\leq C\left\|(-A)^{-1/4}h\right\|_H^2.\end{aligned}$$

Then for any initial value $\xi \in H, T > 0, \varepsilon > 0$[1]*, the temporal semi-discretization $X_T^K : \Omega \times [0,T] \to H, K \in \mathbb{N}$, satisfies the following weak error estimate:*

$$\left|\mathbb{E}\left[\varphi(X_T)\right] - \mathbb{E}\left[\varphi(X_T^K)\right]\right| \leq \frac{C(T, |\varphi|_{C_b^3}, \|\xi\|_H, \varepsilon)}{K^{\left(\frac{1}{2}-\varepsilon\right)}}.$$

As we can also read in [Deb11] Remark 2.3 the assumptions in Theorem 2.7 are quite restrictive. They are fulfilled for additive noise or noise of the form $BXdW$, where B is a linear operator from H to $L(H)$. Otherwise, it implies that the noise is a perturbation of such noise. Therefore, our diffusion coefficient $b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$ is not subject of this theorem.
For additive noise we find in [WG12] the following

Theorem 2.8 *[WG12, Theorem 2.1, $\beta = \frac{1}{2}$] Suppose that for all $y \in H$*

$$\begin{aligned}&\|F(y)\|_H \leq L(\|y\|_H + 1)\\ &\|F'(y)\|_{L(H)} \leq L\\ &\|F''(y)\|_{L(H\times H;H)} \leq L\end{aligned}$$

and let X_t be the exact solution of (2.1) *and $X_t^K : [0,T]\times\Omega \to H, K \in \mathbb{N}$, its temporal semi-discretization. Assume additionally that $\varphi \in C_b^2(H;\mathbb{R})$ and the initial data $\xi \in H$. Then for every $K \in \mathbb{N}$*

$$\left|\mathbb{E}\left[\varphi(X_T)\right] - \mathbb{E}\left[\varphi(X_T^K)\right]\right| \leq \frac{C(\|\xi\|_H, T, \varepsilon, \varphi, L)}{K^{\left(\frac{1}{2}-\varepsilon\right)}}$$

for arbitrary small $\varepsilon < \frac{1}{2}$ and $\frac{T}{K} \leq \frac{T}{2}$.

For the spatial discretization of the nonlinear stochastic heat equation we find in [AL12]

[1] There is a misprint in the original version of this theorem. Not only should the assumption 2.4 be fulfilled but also assumption 2.5. The second one is the restrictive one.

Theorem 2.9 *[AL12, Theorem 1.1] Assume either $B(x) = Id$ or $B(x) = C + Ax + \tilde{b}(x)$ where $C \in L(H), A \in L(H, L(H))$ and $\tilde{b} \in C_b^2(H_{-\frac{1}{2}}, L(H))$ and let X and X^N be the solutions of the equations* (2.1) *and* (2.2)*, respectively. Then, for every test function $\varphi \in C_b^2(H; \mathbb{R})$ and $\gamma \in [0, \frac{1}{2})$, we have*

$$\left| \mathbb{E}\left[\varphi(X_T)\right] - \mathbb{E}\left[\varphi(X_T^N)\right] \right| \leq \frac{C}{N^{2\gamma}}.$$

In Chapter 3 we will use numerical simulations and experiments to take a look at these rates of weak approximation.

2.5 Monte Carlo Method

Now we have a fully discretized numerical scheme. But we still got the problem of simulating a solution that is a random realization. We are more interested in the expectation or second moment of these realizations as the realizations can vary largely with every simulation. Here, one possibility is the Monte Carlo method. In this method we use basically the strong law of large numbers.
The following well-known definition can be, for instance, found in [BJLS12].

Definition 2.10 *[BJLS12, Def. 2.3.1] Let $(\Omega, \mathcal{F}, P)$ be a probability space and $X_n \in L^1(P; \mathbb{R}), n \in \mathbb{N}$ be independent, identically distributed (i.i.d.) random variables. Then the random variables*

$$E_N[X] = \frac{X_1 + \ldots + X_N}{N}$$

for $N \in \mathbb{N}$ are defined as Monte Carlo approximation *of* $\mathbb{E}[X]$.

To look at the behaviour of the so defined random variables we now review the widely known root mean square error of the Monte Carlo method.

Theorem 2.11 *[BJLS12, Thm. 2.5.1] Let $X_n \in L^2(P; \mathbb{R}), n \in \mathbb{N}$, be i.i.d. random variables. Then*

$$\begin{aligned} \left\| \mathbb{E}[X_1] - \frac{X_1 + \ldots + X_N}{N} \right\|_{L^2(P;\mathbb{R})} &= \frac{1}{\sqrt{N}} \sqrt{Var(X_1)} \\ &\leq \frac{1}{\sqrt{N}} \|X_1\|_{L^2(\Omega,\mathbb{R})} \end{aligned}$$

for all $N \in \mathbb{N}$.

Proof First we need to look at the variance of a sum of square integrable random variables. By definition and the linearity of expectation

$$\begin{aligned}\operatorname{Var}\left(\sum_{i=1}^{N} X_i\right) &= \mathbb{E}\left[\left(\sum_{i=1}^{N} X_i - \mathbb{E}\left[X_i\right]\right)^2\right] \\ &= \sum_{i,j=1}^{N} \mathbb{E}\left[\left(X_i - \mathbb{E}\left[X_i\right]\right)\left(X_j - \mathbb{E}\left[X_j\right]\right)\right] \\ &= \sum_{i,j=1}^{N} \operatorname{Cov}\left(X_i, X_j\right).\end{aligned}$$

As the random variables in this theorem are independent they are uncorrelated, too. So for $i \neq j$ we get

$$\operatorname{Cov}\left(X_i, X_j\right) = 0.$$

and thus,

$$\operatorname{Var}\left(\sum_{i=1}^{N} X_i\right) = \sum_{i=1}^{N} \operatorname{Var}\left(X_i\right).$$

By the definition of the variance and the above equality we thus get

$$\begin{aligned}\mathbb{E}\left[\left|\mathbb{E}\left[X_1\right] - \frac{X_1 + \ldots + X_N}{N}\right|^2\right] &= \operatorname{Var}\left(\frac{X_1 + \ldots + X_N}{N}\right) \\ &= \frac{\operatorname{Var}\left(X_1\right) + \ldots + \operatorname{Var}\left(X_N\right)}{N^2} \\ &= \frac{N \cdot \operatorname{Var}\left(X_1\right)}{N^2} = \frac{\operatorname{Var}\left(X_1\right)}{N}\end{aligned}$$

for all $N \in \mathbb{N}$.
For the second inequality we use that

$$\begin{aligned}\operatorname{Var}\left(X\right) &= \mathbb{E}\left[\left(X - \mathbb{E}\left[X\right]\right)^2\right] \\ &= \mathbb{E}\left[X^2\right] - \left(\mathbb{E}\left[X\right]\right)^2 \\ &\leq \mathbb{E}\left[X^2\right] = \|X\|^2_{L^2(P;\mathbb{R})}\end{aligned} \tag{2.9}$$

which finishes the proof. □

Therefore, we conclude that the Monte Carlo approximation converges to the expectation in the root mean square sense with order $\frac{1}{2}$.

2.6 Multilevel Monte Carlo Method

One problem of the Monte Carlo method is the large number of samples needed to achieve a good result. To reduce this problem we try a different approach, the Multilevel Monte Carlo Method. This method can be found, for instance, in [Gil08] [Hei01] or in [BL12].
There we find the following approach. Let Y be again a random variable on the probability space $(\Omega, \mathcal{A}, P)$ with values in the Hilbert space H. The goal is to approximate this variable. Now let $(V_\ell, \ell \in \mathbb{N}_0)$ be a sequence of finite-dimensional subspaces of H and $(Y_\ell, \ell \in \mathbb{N}_0)$ a sequence of approximations of Y, where $Y_\ell \in V_\ell$. Now, a trick is used: the telescopic sum. Thus, we get

$$Y_L = Y_0 + \sum_{\ell=1}^{L} (Y_\ell - Y_{\ell-1}) .$$

As in the case of the Monte Carlo method we are interested in the expectation of this random variable. By using the linearity of expectation we see

$$\mathbb{E}[Y_L] = \mathbb{E}[Y_0] + \sum_{\ell=1}^{L} \mathbb{E}[Y_\ell - Y_{\ell-1}] . \tag{2.10}$$

Now each of this summands can be approximated by the Monte Carlo scheme described in Section 2.5. We approximate $\mathbb{E}[Y_\ell - Y_{\ell-1}]$ by the corresponding Monte Carlo estimator $E_{M_\ell}[Y_\ell - Y_{\ell-1}]$ with M_ℓ independent samples and thus

$$E^L[Y_L] = E_{M_0}[Y_0] + \sum_{\ell=1}^{L} E_{M_\ell}[Y_\ell - Y_{\ell-1}] .$$

M_ℓ depends here on the level ℓ.
We still have to decide how to choose the number of samples at each level. We will take a closer look at this question during our numerical simulations in Section 3.2.5.
How does this method behave compared with the classical Monte Carlo scheme? E.g. in [BL12] we find

Theorem 2.12 *[BL12, Thm. 2.3] Let $(Y_\ell, \ell \in \mathbb{N}_0)$ converge weakly to Y of order $\alpha > 0$ and assume*

$$Var[Y_\ell - Y_{\ell-1}] \leq (C_2)^2 2^{-2\beta\ell}$$

and $Var[Y_0] = (C_3)^2$. Now we choose $M_0 = 2^{2\alpha L}$ for the first level and for every level $\ell = 1, \ldots, L$ $M_\ell = 2^{2(\alpha L - \beta\ell)} \ell^{2(1+\varepsilon)}$. Then, the error is bounded by

$$\left\| \mathbb{E}[Y_L] - E^L[Y_l] \right\|_{L^2(\Omega;B)} \leq (C_1 + C_3 + C_2\zeta(1+\varepsilon)) 2^{-\alpha L},$$

and thus we get the same order of convergence as for the Monte Carlo scheme.

Chapter 3

Numerical Simulation of the Weak Approximation Error

With our numerical experiments we aim at using Monte Carlo and Multilevel Monte Carlo simulations to make conclusions about possible convergence rates for stochastic partial differential equations (SPDEs) with multiplicative noise.
For SPDEs with additive noise there are already papers which give us information about the convergence rates and the weak errors. Some of them were listed in Section 2.4.2. That gives us the possibility to use the results of these papers as a reference.
Hence, we first look at SPDEs with additive noise to get a reference about how well the beforehand presented methods work. We look at different scenarios: The convergence of the infinite sum of the exact solution, the deterministic numerical error and the weak error of the Monte Carlo and the Multilevel Monte Carlo method. Furthermore we use another method, the Difference Monte Carlo method. This method will be described in detail in 3.2.4.
Then, we go on to SPDEs with multiplicative noise and look at different diffusion coefficients. In these cases we no longer have the possibility to determine an exact solution or look at the deterministic error. Therefore, we restrict our analysis to different Monte Carlo schemes. We plot again the weak error of the Monte Carlo, the Difference Monte Carlo and the Multilevel Monte Carlo method.
But first, we give some preliminary remarks about the numerical calculation of the weak approximation error and the use of random numbers.

3.1 Preliminary Remarks

3.1.1 Weak Error

For the whole chapter we will again assume the setting from Section 1.3. In all following numerical experiments we are interested in the weak error

$$e_{N,K}(T) = \left| \mathbb{E}\left[\varphi\left(Y_T^{N,K} \right) - \varphi\left(X(T) \right) \right] \right|, \tag{3.1}$$

where we choose always the test function

$$\varphi(h) = \|h\|^2$$

as mentioned before in Section 2.4.2.
We will analyse the convergence rate of this error. If the error is plotted in a loglog-plot we get the convergence rate as the slope of the resulting line. This slope can be calculated in MATLAB using the "polyfit" command.
Another possibility would be the *experimental order of convergence* (EOC) as e.g. described in [Kru12]. Taking two successive values of N, for example N_1 and N_2 compute

$$EOC(N_1, N_2) = \frac{\log(e_{N_2,K}(T)) - \log(e_{N_1,K}(T))}{\log(N_1) - \log(N_2)}. \tag{3.2}$$

This EOC is given with every error.

3.1.2 Random Numbers

Another important point regarding our numerical simulations is the generation of random numbers. Throughout this thesis I used MATLAB for the numerical experiments and simulations. As these simulations are very time consuming I used the ETH cluster Brutus (see [wik13]). On this cluster the default MATLAB version is 7.14(2012a). To generate random variables I used that version and first set a random seed with `rng('shuffle')`. This random seed was saved for every simulation to guarantee reproducibility.
To speed up computations it makes sense to run a Monte Carlo simulation in parallel. Using the "Parallel Computing Toolbox" provided by MATLAB this works quite uncomplicated. We first have to start a number of parallel workers or sessions. Then we can use the command `parfor` instead of the normally used `for`. One important question remains. How does MATLAB handle random numbers on parallel workers? In the documentation [mat13] we find

> "By default, each worker in a cluster working on the same job has a unique random number stream."

Thus, we do not need to initialize a random seed for every worker, i.e. MATLAB session.
Another problem is that we can not predict the order in which the `parfor`-loop calculates. As all our calculations are independently and we do not need to guarantee a special order in which the Monte Carlo samples are calculated we do not need to look further into this.
Therefore, we are able to get reproducible results.

3.2 SPDEs with additive noise

3.2.1 Exact Solution

As e.g. described in [Kru12] the exact solution of (1.9) with $b(x, y) = 1$ can be written as

$$\begin{aligned} X_t &= \int_0^t e^{A(t-s)} dW_s \\ &= \sum_{n=1}^{\infty} e_n \int_0^t e^{-\lambda_n (t-s)} d\beta_s, \end{aligned}$$

where we use the eigenvalues λ_n and eigenvectors e_n of the operator A (see (1.6) and (1.7)) and the fact that we can write the Wiener process as a sum of finite dimensional Brownian motions (see 1.1).
Now we look at the second moment of this stochastic process and get

$$\begin{aligned} \mathbb{E}\left[\|X_t\|_H^2\right] &= \sum_{i=1}^{\infty} \mathbb{E}\left[|\langle e_i, X_t\rangle|^2\right] \\ &= \sum_{i=1}^{\infty} \mathbb{E}\left[\left|\int_0^t e^{-\lambda_i (t-s)} d\beta_s^i\right|^2\right] \\ &= \sum_{i=1}^{\infty} \int_0^t e^{-2\lambda_i (t-s)} ds \\ &= \sum_{i=1}^{\infty} \int_0^t e^{-2\lambda_i s} ds \\ &= \sum_{i=1}^{\infty} \frac{1 - e^{-2\lambda_i t}}{2\lambda_i}. \end{aligned}$$

Table 3.1: Error of Infinite Series of Stochastic Heat Equation

N	Weak Error	EOC
2	1.8905	
4	1.1197	0.75569
8	0.59532	0.91131
16	0.30694	0.95572
32	0.15587	0.97765
64	0.078542	0.98878
128	0.039424	0.99438
256	0.019751	0.99719
512	0.0098848	0.9986
1024	0.0049448	0.99932
2048	0.0024729	0.99969

Here we used Itô's formula in the third step (see (1.3)).
This infinite sum can be approximated by choosing a number N and calculating

$$\tilde{E}_N(X_t) = \sum_{i=1}^{N} \frac{1 - e^{-2\lambda_i t}}{2\lambda_i}.$$

For large N we get a very good approximation of the exact solution. We can use the following MATLAB program.

```
N=2^11;
y=0;
t=1;
lam = @(j) 1/100*pi^2*j.^2;
exact_solution=sum((1-exp(-2*lam(1:N)*t))./(2*lam(1:N)));
```

By using different N and calculate the error to a reference solution calculated with a very high N (here $N_{ref} = 2^{25}$) we get a convergence rate of this first approximation towards the exact solution. This can be seen in Figure 3.1. We see that the numerically calculated points form a straight line in the loglog-plot. As this line is parallel to the line with slope 1 (the red line in Figure 3.1) we can conclude that we have a convergence rate of 1.
Another way to present this fact can be found in tabel 3.1. We see the error for each step size N and the corresponding EOC (see (3.2)). The first EOC is a little smaller than expected. But from then on we get values that are always $1 - \varepsilon$. Additionally we realize that the EOC gets larger for larger N. We suspect that this is due to getting closer and closer to the reference solution.

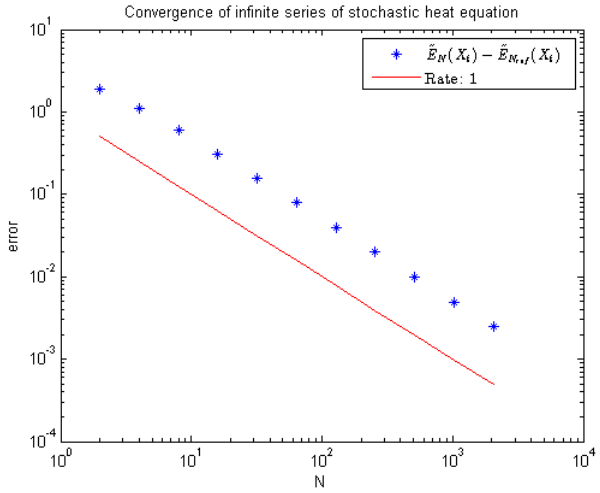

Figure 3.1: Convergence of Infinite Sum to Exact Solution

3.2.2 Deterministic Numerical Error

Now we no longer want to calculate the exact solution but want to use the numerical method introduced in Chapter 2.
We want to analyse the deterministic numerical error of our method. That means we use the properties of equation (1.9) to eliminate the randomness in the solution and just look at the approximation error of the numerical method.
First we have to discretize (1.9) in space as described in Section 2.1 and then we use the implicit Euler method in Section 2.2 to discretize in time and get

$$X_n^N = (1 - \Delta t A_N)^{-1} \left(X_{n-1}^N + \Delta W_{n-1} \right).$$

This formula can be used recursively. As we use $X_0 = 0$ as initial condition we get

$$X_n^N = \sum_{k=0}^{n-1} (I - \Delta t A_N)^{k-n} \Delta W_k.$$

We are interested in the second moment of this formula. Thus, by using the linearity of expectation we calculate

$$\begin{aligned} \mathbb{E}\left[\left\| X_n^N \right\|_H^2 \right] &= \mathbb{E}\left[\left\| \sum_{k=0}^{n-1} (I - \Delta t A_N)^{n-k} \Delta W_k \right\|_H^2 \right] \\ &= \sum_{k=0}^{n-1} \mathbb{E}\left[\left\| (I - \Delta t A_N)^{k-n} \Delta W_k \right\|_H^2 \right]. \end{aligned}$$

By the definition of the stochastic integral in Section 1.2 we get

$$\mathbb{E}\left[\left\|X_n^N\right\|_H^2\right] = \sum_{k=0}^{n-1} \mathbb{E}\left[\left\|\int_0^{\Delta t} (I - \Delta t A_N)^{k-n}\, dW_s\right\|_H^2\right].$$

For this we can now use Itô's isometry (1.3) and get a deterministic integral. Thus, we are able to calculate

$$\begin{aligned}
\mathbb{E}\left[\left\|X_n^N\right\|_H^2\right] &= \sum_{k=0}^{n-1} \int_0^{\Delta t} \left\|(I - \Delta t A_N)^{k-n}\right\|_{HS(H,H)}^2 ds \\
&= \sum_{k=0}^{n-1} \left\|(I - \Delta t A_N)^{k-n}\right\|_{HS(H,H)}^2 \Delta t \\
&= \sum_{k=0}^{n-1} \sum_{j=1}^{N} \left\|(I - \Delta t A_N)^{k-n} e_j\right\|_H^2 \Delta t \\
&= \sum_{k=0}^{n-1} \sum_{j=1}^{N} \left\|\frac{1}{(1+\Delta t \lambda_j)^{n-k}} e_j\right\|_H^2 \Delta t \\
&= \sum_{k=0}^{n-1} \sum_{j=1}^{N} \frac{1}{(1+\Delta t \lambda_j)^{2(n-k)}} \Delta t.
\end{aligned}$$

By changing the order of the sums and using the geometric series starting from one

$$\sum_{k=1}^{N} q^k = \frac{q\left(q^N - 1\right)}{q-1}$$

we get

$$\begin{aligned}
\mathbb{E}\left[\left\|X_n^N\right\|_H^2\right] &= \sum_{j=1}^{N} \sum_{k=1}^{n-1} \frac{1}{(1+\Delta t \lambda_j)^{2k}} \Delta t \\
&= \sum_{j=1}^{N} \frac{\left((1+\Delta t \lambda_j)^{-2n} - 1\right)(1+\Delta t \lambda_j)^{-2}}{(1+\Delta t \lambda_j)^{-2} - 1} \Delta t.
\end{aligned}$$

This can be calculated, for specific N and $\Delta t = \frac{1}{K}$, by

```
N=2^5; K=50000;
lambda = @(j) pi^2*j.^2./100;
dt=1/K;

q=(1+dt*lambda(1:N)).^(-2);
data=sum((q.*(q.^n-1))./(q-1))*dt;
```

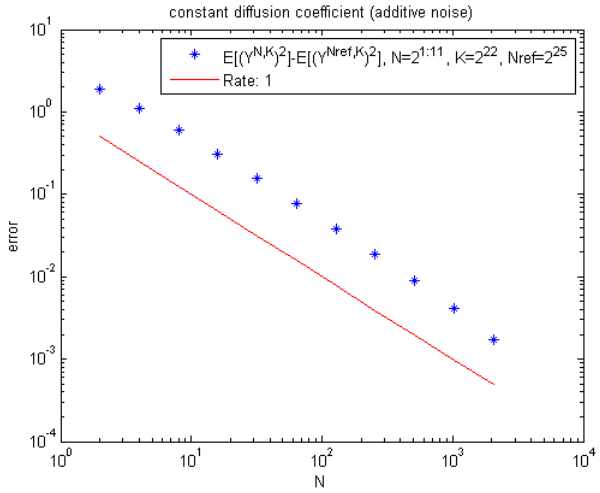

Figure 3.2: Error of Deterministic Numerical Scheme

Table 3.2: Error of Deterministic Numerical Scheme

N	Weak Error	EOC
2	1.8896	
4	1.1188	0.75614
8	0.59446	0.91229
16	0.30608	0.95768
32	0.15501	0.98158
64	0.077683	0.99665
128	0.038569	1.0102
256	0.018903	1.0288
512	0.0090523	1.0622
1024	0.0041426	1.1278
2048	0.00173	1.2597

Again, we can now use different N and calculate the error to a reference solution with $N_{ref} = 2^{25}$. Plotting this in a loglog-plot gives us the convergence rate of the numerical method. In Figure 3.2 we can compare the numerical error with the line plotted with slope 1. We see again that first we have a little bit slower convergence but then we are parallel to the red line. In Table 3.2 we see again the EOC. Here this rate is a little bit higher than in the case of the exact solution. And as in the case before we get higher convergence rates with larger N.

3.2.3 Monte Carlo Scheme

The next step is now to finally use the Monte Carlo Scheme. As described in Section 2.5 the idea is to sum up a large number of samples and divide them by the quantity of these. As proven in Theorem 2.11 the error of the Monte Carlo Method is of order $\mathcal{O}(M^{-1/2})$, where M is the number of repetitions.

```
% parallel on cluster:
cluster = parcluster('BrutusLSF8h')
matlabpool(cluster,100)

% number of Monte Carlo repetitions
M=20000;

% set random seet based on current time and save seed.
rng('shuffle');
s=rng('shuffle');

NN = 2.^(1:11);

K=50000;

data=zeros(1,length(NN));
f = @(x) 0;
b = @(x) 1;

for n=1:length(NN) %for different number of steps in space
    N=NN(n); %number of steps in space
        tic
    z=zeros(1,M); %initialisation for numerical expectation

    parfor m=1:M
        Y = zeros(1,N); %initalisation for process
        A = -pi^2*(1:N).^2/100;
        for k = 1:K %for time stepping
            y = dst(Y) * sqrt(2);
            dW = dst( randn(1,N) .* sqrt(2/K));
            y = y + f(y)/K + b(y).*dW;
            Y = idst(y) / sqrt(2) ./ (1-A/K);
        end
        z(m)=sum(Y.^2);
    end
    %Monte Carlo
    data(n)=sum(z)/M;
        time(n)=toc;
end

% save the data
```

```
42 name = [datestr(now, 'yymmddTHHMMSS') '.mat'];
43 save(name,'NN','data','time','s')
44
45 matlabpool close
```

The above code shows how this can be implemented in MATLAB. Here the basic code described in Section 2.2 is used again .
To calculate the Monte Carlo scheme more efficiently I used the BRUTUS cluster of ETH. A wiki can be found at [wik13]. There we also read that Brutus was ranked the 88th fastest computer in the world in November 2009[1]. Using this cluster and the Parallel Computing Toolbox offered by MATLAB (see 3.1.2) we run the code for the Monte Carlo method in parallel. This is possible as the individual repetitions for the Monte Carlo scheme are independent and do not need any information from each other (cf. Section 3.1.2). The data displayed in Figure 3.3 and Table 3.3 is calculated parallel on 100 workers on the Brutus cluster. Nevertheless, we need about 5 hours to calculate just the approximation for $N = 2^{11}$. In total this calculation takes about 14 hours. The calculation times for every step can be found in Table 3.3.
As in the two above cases we again have the results plotted and displayed in a table. In Figure 3.3 we see two simulations of the error of the Monte Carlo scheme. As a reference we use the exact solution calculated in 3.2.1 and we choose 20′000 samples. It gets clear that for the first few N the simulations are very much alike. Then the results start to have a quite big variance. We suppose that the variance gets smaller with more Monte Carlo samples. However, while the simulations are consistent we again see a convergence rate of about one. This can be seen as the simulated error is parallel to the line with slope 1 (printed in red). Additionally, we can take a look at Table 3.3. As long as we do not have a big variance the EOC is again about one. Afterwards it gets quite chaotic and does not yield any valuable information. To take a better look at this variance an additional simulation was done with 80′000 sampes for $N = 2^7, 2^8, 2^9$. In Figure 3.3 we see this simulation as green points. And it really seems as if the variance can be significantly reduced by the usage of more samples.

This step of our experiments is the first one that also makes sense for multiplicative noise. It can be found in 3.3.1.

3.2.4 Difference Monte Carlo Scheme

As we are only interested in the behaviour of the weak error we now look at another scheme that directly gives us this error. Schemes like this have been used before. One example can be found in [JKN09, Page 54]. This scheme

[1] http://www.clusterwiki.ethz.ch/brutus/Brutus_cluster

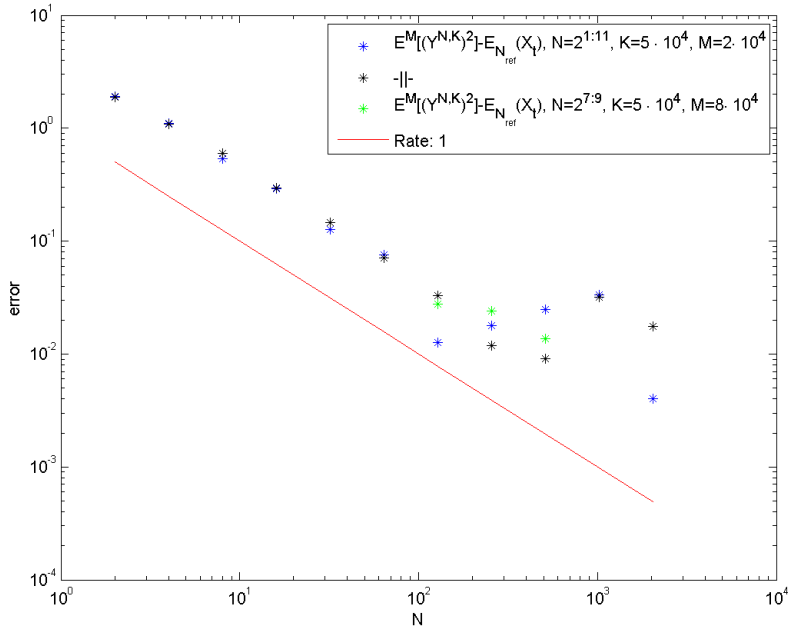

Figure 3.3: Error of MC, $b(x,y) = 1$

Table 3.3: Error of MC, $b(x,y) = 1$

N	Weak Error	EOC	Weak Error	EOC	time (in sec)
2	1.9193	0	1.8763		2613.6
4	1.1038	0.79803	1.0932	0.7793	2624.9
8	0.538	1.0369	0.59619	0.87474	2671.4
16	0.29245	0.87939	0.29563	1.012	2779.3
32	0.12725	1.2006	0.14601	1.0177	2852.2
64	0.075813	0.74712	0.07064	1.0475	3020.1
128	0.01264	2.5844	0.032744	1.1093	3438.9
256	0.017794	-0.49339	0.011778	1.4751	4273.4
512	0.024692	-0.47264	0.0090901	0.37372	5018.3
1024	0.033175	-0.42608	0.031936	-1.8128	8592.6
2048	0.0040396	3.0378	0.017532	0.86518	17585

N	Weak Error	EOC	time (in sec)
128	0.027865		13796
256	0.02413	0.20762	17147
512	0.013692	0.81751	20073

helps to reduce the time needed to compute the error. For estimating the weak error

$$\left|\mathbb{E}\left[\varphi(X_T)\right] - \mathbb{E}\left[\varphi(X_T^N)\right]\right| = \left|\mathbb{E}\left[\varphi(X_T) - \varphi(X_T^N)\right]\right|$$

as above not the exact solution $\mathbb{E}\left[\varphi(X_T)\right]$ but a reference solution is used. The idea is now to always look at a reference that uses only double the points to discretize in space. These references are calculated for every step. Additionally, as we are not able to calculated the expectation directly, we use again the classical Monte Carlo scheme. Thus, we get the approximation

$$\begin{aligned}\left|\mathbb{E}\left[\varphi(X_T^{2N})\right] - \mathbb{E}\left[\varphi(X_T^N)\right]\right| &= \mathbb{E}\left[\varphi(X_T^{2N}) - \varphi(X_T^N)\right] \\ &\approx \frac{1}{M(N)}\left[\sum_{m=1}^{M(N)} \varphi(X_T^{2N,m}) - \varphi(X_T^{N,m})\right].\end{aligned}$$

If we take a closer look at the error of this approximation we see

$$\begin{aligned}\varepsilon_N &= \left\|\mathbb{E}\left[\varphi(X_T^{2N}) - \varphi(X_T^N)\right] - \frac{1}{M(N)}\left[\sum_{m=1}^{M(N)} \varphi(X_T^{2N,m}) - \varphi(X_T^{N,m})\right]\right\|_{L^2(\Omega,\mathbb{R})} \\ &= \frac{1}{\sqrt{M(N)}}\sqrt{\mathrm{Var}\left(\varphi(X_T^{2N}) - \varphi(X_T^N)\right)} \\ &\leq \frac{1}{\sqrt{M(N)}}\left\|\varphi(X_T^{2N}) - \varphi(X_T^N)\right\|_{L^2(\Omega,\mathbb{R})},\end{aligned}$$

by the mean square error of the Monte Carlo method from Theorem 2.11. Using now the Lipschitz constant of φ we get

$$\begin{aligned}\varepsilon_N &\leq \frac{\|\varphi\|_{\mathrm{Lip}}\left\|X_T^{2N} - X_T^N\right\|_{L^2(\Omega,H)}}{\sqrt{M(N)}} \\ &\leq \frac{\|\varphi\|_{\mathrm{Lip}}\left(\left\|X_T^{2N} - X_T\right\|_{L^2(\Omega,H)} + \left\|X_T^N - X_T\right\|_{L^2(\Omega,H)}\right)}{\sqrt{M(N)}} \\ &\leq \frac{\|\varphi\|_{\mathrm{Lip}}\, const \cdot N^{\left(\varepsilon-\frac{1}{2}\right)}}{\sqrt{M(N)}},\end{aligned} \tag{3.3}$$

where in the last step the strong error described in Theorem 2.4 was used. The question is how to choose the number of Monte Carlo samples. As we are mainly interested in the convergence rate of our approximation we need to look at the slope of the line printed in a loglog-plot. The idea is now that the plot of the Difference Monte Carlo approximation and the plot of the real weak error should be parallel. If we now look at Figure 3.4 we see two parallel lines in a loglog-plot with distance δ. We see that

$$\left|\log(g(N)) - \log(f(N))\right| = \delta$$

and therefore

$$\log\left(\frac{g(N)}{f(N)}\right) = \delta$$
$$\frac{g(N)}{f(N)} = e^{\delta}$$

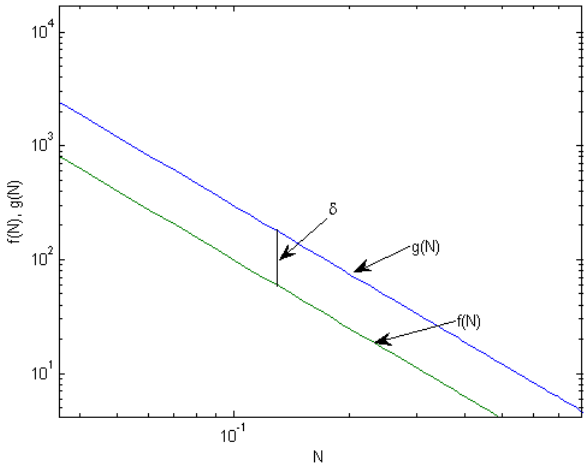

Figure 3.4: Distance of two Lines in loglog-Plot

In our case one of the lines is the weak error of our scheme, say $f(N)$ and the other $g(N)$ is the approximated error calculated with our scheme. The approximated error is the weak error plus our approximation error, thus

$$g(N) = f(N) + \varepsilon_N.$$

Combining the last two equations we get

$$\frac{\varepsilon_N}{f(N)} = e^{\delta} - 1.$$

As we have a weak error of order $1 - \varepsilon$ we get

$$\frac{e^{\delta} - 1}{N} = f(N) \cdot (e^{\delta} - 1) = \varepsilon_N \approx \frac{\text{const}}{\sqrt{MN}},$$

where the last part follows from equation (3.3). Thus, we can conclude that we need to choose M as

$$M = \frac{\text{const}}{(e^{\delta} - 1)^2} N.$$

In Table 3.4 we see how the function $(e^{\delta} - 1)^{-2}$ behaves.

Table 3.4: Magnitude of M for given δ

delta	M
0.5	2.3762
0.1	90.408
0.01	9900.4
0.001	9.99e+05
0.0001	9.999e+07

Implementing this in MATLAB gives us the following code:

```
% parallel on cluster:
cluster = parcluster('BrutusLSF8h')
matlabpool(cluster,100)

% set random seet based on current time and save seet
rng('shuffle');
s=rng('shuffle');

NN = 2.^(1:11);

K=50000; %number of steps in time

data=zeros(1,length(NN));
time=zeros(1,length(NN));

%f: drift, b: diffusion coefficient
f = @(x) 0;
b = @(x) 1;

for n=1:length(NN) %for different number of steps in space
       tic

   N1=NN(n); % number of steps in space
   N2=2*N1;
   M=N1*10; % number of Monte Carlo sampes

   z=zeros(1,M);
   parfor m=1:M %for Monte Carlo

       Y1 = zeros(1,N1); %initalisation for process
       Y2 = zeros(1,N2);
       A1 = -pi^2*(1:N1).^2/100;
       A2 = -pi^2*(1:N2).^2/100;

       for k = 1:K %for time stepping
           % generate random numbers

           r=randn(1,max(N1,N2));
```

```
39
40              % calculate one step for first stepsize
41              y1 = dst(Y1) * sqrt(2);
42              dW1 = dst( r(1:N1) .* sqrt(2/K));
43              y1 = y1 + f(y1)/K + b(y1).*dW1;
44              Y1 = idst(y1) / sqrt(2) ./ (1-A1/K);
45
46              % calculate one step for second stepsize
47              y2 = dst(Y2) * sqrt(2);
48              dW2 = dst( r(1:N2) .* sqrt(2/K));
49              y2 = y2 + f(y2)/K + b(y2).*dW2;
50              Y2 = idst(y2) / sqrt(2) ./ (1-A2/K);
51          end
52          z1=sum(Y1.^2);
53          z2=sum(Y2.^2);
54          z(m)=abs(z1-z2);
55      end
56      %Monte Carlo
57      data(n)=sum(z)/M;
58
59      time(n)=toc;
60  end
61
62  % save the data
63  name = [datestr(now, 'yymmddTHHMMSS') '.mat'];
64  save(name,'NN','data','time','s')
65
66  matlabpool close
```

In Figure 3.5 and Table 3.5 we see the result of these calculations. First we have a convergence rate less than one and in the end the rate gets larger. That could be again due to the proximity to the exact solution. However, for the bigger part of the diagram the simulated error is parallel to the red line with slope one. The same can be seen in Table 3.5. An EOC of approximately one can be calculated.
Additionally, there is another very important difference between this calculation and the classical Monte Carlo scheme: the derivation time. Approximating the weak error with the Difference Monte Carlo scheme is much more efficient in terms of computation time. Especially for small N only a fraction of the time is required to calculate the classical Monte Carlo scheme. Unfortunately, as we need a number of Monte Carlo samples of order $\mathcal{O}(N)$ much more computation time for larger N is used. In that case we need more time than, for example, for the calculations in Figure 3.3. However, our numerical results here are also much better than in Section 3.2.3.

For multiplicative noise the corresponding results can be found in 3.3.2.

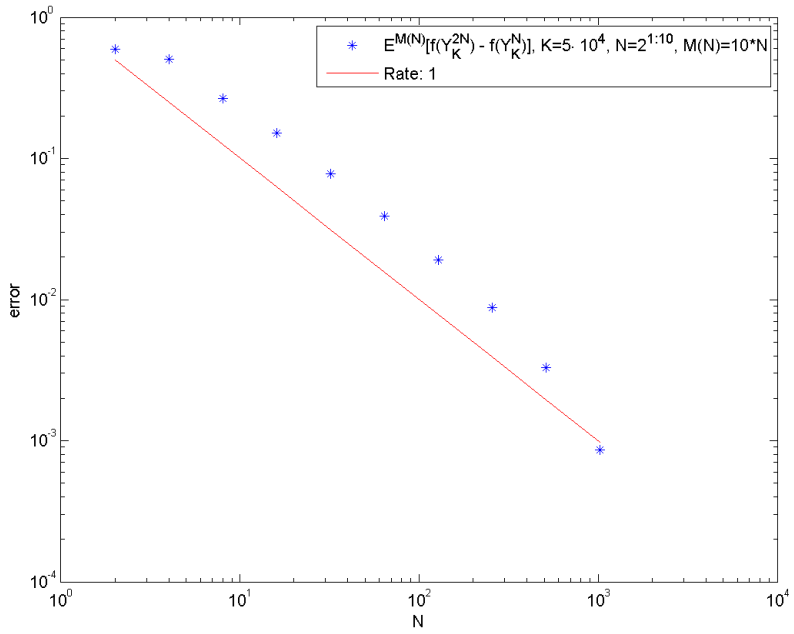

Figure 3.5: Error of DMC: $b(x,y) = 1$

Table 3.5: Error of DMC: $b(x,y) = 1$

N	Weak Error	EOC	time (in sec)
2	0.59539		24.996
4	0.50422	0.23977	19.169
8	0.26695	0.91748	20.234
16	0.15105	0.82156	40.424
32	0.077567	0.9615	83.986
64	0.038987	0.99243	165.18
128	0.019069	1.0318	405.79
256	0.0087564	1.1228	982.19
512	0.0033073	1.4047	3021.6
1024	0.00086166	1.9404	12995

3.2.5 Multilevel Monte Carlo Scheme

The next simulations use the Multilevel Monte Carlo scheme described in Section 2.6. We have to keep in mind that this scheme approximates again the solution of an SPDE and not directly the weak error as the Difference Monte Carlo described in 3.2.4. The question is now how we should choose the number of Monte Carlo samples on each level of the calculation. In Theorem 2.12 we already see a possible way to choose the number of samples. Here we will look at our setting described in Section 1.3 and derive the number of samples needed using this specific SPDE.
To get a reasonable number of samples we have to guarantee that the variance of our solution is bounded and, thus, stays small. We know that the solution of the spatial discretisation of the heat equation (1.9) is

$$\int_0^T e^{A_N(T-s)} dW_s.$$

Using the representation of the Wiener process in (1.1) we can write this as

$$\int_0^T e^{A_N(T-s)} dW_s = \sum_{j=1}^{N} e_j \int_0^T e^{-\lambda_j(T-s)} d\beta_s^j \tag{3.4}$$

Additionally we know from Theorem 1.6, equations (1.4) and (1.3) that

$$\mathbb{E}\left[\int_0^t \Phi(\sigma) dW(\sigma)\right] = 0$$

and

$$\mathrm{Var}\left[\int_0^t \Phi(\sigma) dW(\sigma)\right] = \int_0^t \mathbb{E}\left[\Phi^2\right] dt.$$

Therefore, we know that the stochastic integral in (3.4) has a normal distribution with expectation equal to zero and variance equal to

$$\int_0^T e^{-2\lambda_j(T-s)} ds.$$

Thus, it is true in distribution that

$$\begin{aligned}
\int_0^T e^{A_N(T-s)} dW_s &= \sum_{j=1}^N e_j \int_0^T e^{-\lambda_j (T-s)} d\beta_s^j \\
&= \sum_{j=1}^N e_j \mathcal{N}\left(0, \int_0^T e^{-2\lambda_j (T-s)} ds\right) \\
&= \sum_{j=1}^N e_j \mathcal{N}\left(0, \int_0^T e^{-2\lambda_j (s)} ds\right) \\
&= \sum_{j=1}^N e_j \sqrt{\frac{\int_0^T e^{-2\lambda_j s} ds}{T}} \mathcal{N}(0,T) \\
&= \sum_{j=1}^N e_j \sqrt{\frac{\int_0^T e^{-2\lambda_j s} ds}{T}} \beta_T^j.
\end{aligned}$$

Calculating this integral and re-substituting $\sum_{j=1}^N e_j \beta_T^j$ by W_T^N gives us

$$\begin{aligned}
Y^N &:= \sum_{j=1}^N e_j \sqrt{\frac{1 - e^{-2\lambda_j T}}{2\lambda_j T}} \beta_T^j \\
&= \sqrt{\frac{e^{2A_N T} - I}{2A_N T}} W_T^N.
\end{aligned} \tag{3.5}$$

To simplify the calculations we assume that $N = 2^L$ and thus $L = \log_2(N)$. By using the definition of the Multilevel Monte Carlo scheme in Section 2.10 with the above defined stochastic process Y^N we get

$$\begin{aligned}
\mathbb{E}\left[\left\|Y^N\right\|_H^2\right] &= \mathbb{E}\left[\left\|Y^{2^0}\right\|_H^2\right] + \sum_{l=1}^L \mathbb{E}\left[\left\|Y^{2^l}\right\|_H^2 - \left\|Y^{2^{l-1}}\right\|_H^2\right] \\
&= \frac{1 - e^{-2\pi^2 T}}{2\pi^2 T} + \mathbb{E}\left[\left\|W_T^1\right\|_H^2\right] + \sum_{l=1}^L \mathbb{E}\left[\left\|Y^{2^l}\right\|_H^2 - \left\|Y^{2^{l-1}}\right\|_H^2\right] \\
&= \frac{1 - e^{-2\pi^2 T}}{2\pi^2} + \sum_{l=1}^L \sum_{j=1}^{M_l} \frac{\left\|Y^{2^l}\right\|_H^2 - \left\|Y^{2^{l-1}}\right\|_H^2}{M_j}.
\end{aligned} \tag{3.6}$$

Here we calculated the first summand directly.
Now we want to look at the variance corresponding to the expectation in

(3.6). We look at each level independently and get

$$\begin{aligned}\operatorname{Var}\left(\sum_{j=1}^{M_l}\frac{\left\|Y^{2^l,j}\right\|_H^2-\left\|Y^{2^{l-1},j}\right\|_H^2}{M_l}\right)&=\sum_{j=1}^{M_l}\operatorname{Var}\left(\frac{\left\|Y^{2^l,j}\right\|_H^2-\left\|Y^{2^{l-1},j}\right\|_H^2}{M_l}\right)\\&=\sum_{j=1}^{M_l}\frac{\operatorname{Var}\left(\left\|Y^{2^l,j}\right\|_H^2-\left\|Y^{2^{l-1},j}\right\|_H^2\right)}{M_l^2}\\&\leq\sum_{j=1}^{M_l}\frac{\left\|\left\|Y^{2^l}\right\|_H^2-\left\|Y^{2^{l-1}}\right\|_H^2\right\|_{L^2(\Omega;\mathbb{R})}^2}{M_l^2}\end{aligned}$$

This is true as the random variables in each Monte Carlo sample are independent of each other and thus are uncorrelated. The last step is due to equation (2.9).
Now we look at each of the summands individually. By definition of the L^2-norm we get

$$\begin{aligned}\left\|\left\|Y^{2^l}\right\|_H^2-\left\|Y^{2^{l-1}}\right\|_H^2\right\|_{L^2(\Omega;\mathbb{R})}^2&=\mathbb{E}\left[\left\|\left\|Y^{2^l}\right\|_H^2-\left\|Y^{2^{l-1}}\right\|_H^2\right\|_H^2\right]\\&=\mathbb{E}\left[\left\|Y^{2^l}\right\|_H^2-\left\|Y^{2^{l-1}}\right\|_H^2\right].\end{aligned}$$

Using the calculations above in (3.5) we conclude

$$\begin{aligned}&\left\|\left\|Y^{2^l}\right\|_H^2-\left\|Y^{2^{l-1}}\right\|_H^2\right\|_{L^2(\Omega;\mathbb{R})}^2\\&\quad=\mathbb{E}\left[\sum_{j=1}^{2^l}\frac{\left|e^{2\lambda_j T}-1\right|}{2\lambda_j T}\left\|e_j\beta_T^j\right\|_H^2-\sum_{j=1}^{2^{l-1}}\frac{\left|e^{2\lambda_j T}-1\right|}{2\lambda_j T}\left\|e_j\beta_T^j\right\|_H^2\right]\\&\quad=\sum_{j=2^{l-1}+1}^{2^l}\frac{1-e^{2\lambda_j T}}{2\lambda_j T}\mathbb{E}\left[\left\|e_j\beta_T^j\right\|_H^2\right].\end{aligned}$$

As $\mathbb{E}\left[\left\|e_j\beta_T^j\right\|_H^2\right]=T$ as in Theorem 1.6 equation (1.3) we get

$$\left\|\left\|Y^{2^l}\right\|_H^2-\left\|Y^{2^{l-1}}\right\|_H^2\right\|_{L^2(\Omega;\mathbb{R})}^2=\sum_{j=2^{l-1}+1}^{2^l}\frac{1-e^{2\lambda_j T}}{2\lambda_j}.$$

Now we want to estimate this sum. This requires the following idea. If we look at the integral of x^{-p} we know for $p < 1$

$$\begin{aligned}\int_N^\infty \frac{1}{x^p} dx &= \left[\frac{x^{1-p}}{1-p}\right]_{x=N}^{x=\infty} \\ &= \frac{1}{(1-p)\, N^{p-1}}.\end{aligned}$$

Therefore, we can conclude for the corresponding sum that

$$\sum_{k=n}^{\infty} \frac{1}{k^p} \leq \frac{\text{const}}{n^{p-1}}.$$

Thus, we get

$$\begin{aligned}\left\| \left\| Y^{2^l} \right\|_H^2 - \left\| Y^{2^{l-1}} \right\|_H^2 \right\|_{L^2(\Omega;\mathbb{R})}^2 &\leq \sum_{j=2^{l-1}+1}^{\infty} \frac{1 - e^{2\lambda_j T}}{2\lambda_j} \\ &\leq \sum_{j=2^{l-1}+1}^{\infty} \frac{1}{\lambda_j} \\ &\leq \sum_{j=2^{l-1}+1}^{\infty} \frac{1}{j^2} \\ &\leq \frac{\text{const}}{2^{l-1}}.\end{aligned}$$

Finally we get

$$\text{Var}\left(\sum_{j=1}^{M_l} \frac{\left\| Y^{2^l,j} \right\|_H^2 - \left\| Y^{2^{l-1},j} \right\|_H^2}{M_l} \right) \leq \sum_{j=1}^{M_l} \frac{\text{const}}{M_l^2 \cdot 2^{l-1}}.$$

Now we want to balance this variance with the error $N^{(\varepsilon-1)}$. Thus, we look at

$$\frac{1}{N} = \frac{\text{const}}{M_l^2 \cdot 2^{l-1}}$$

and conclude that we should choose

$$M_l = \frac{\text{const} \cdot N}{2^{l-1}}$$

at each level l.
If we remember the result in Theorem 2.12 we realize that we had there

$$M_l = 2^{2(\alpha L - \beta l)} l^{2(1+\varepsilon)}.$$

As we defined $L = \log_2(N)$ this is of the same form as the result calculated here.

Semidiscrete Version

Using once again the stochastic heat equation with additive noise (diffusion coefficient $b(x,y) = 1$), we derived in (3.5) the spatial semi-discretization

$$Y^N = \sqrt{\frac{e^{2A_N T} - I}{2A_N T}} W_T^N.$$

As W_T^N is distributed as a N-dimensional Gaussian random variable with expectation 0 and variance T we can simulate the Multilevel Monte Carlo scheme

$$E^L\left[\varphi(Y^N)\right] = E_{N_0}\left[\varphi(Y_0)\right] + \sum_{\ell=1}^{L} E_{N_\ell}\left[\varphi(Y_\ell) - \varphi(Y_{\ell-1})\right],$$

with $\varphi(x) = \|x\|_H^2$. This is how this could be solved in MATLAB

```
NN=2.^(1:13);

T=1;
kappa=1/100;

A=-(1:max(NN)).^2*pi^2*kappa;
B=sqrt((exp(2*A*T)-1)./(2*A*T))';

MLMC=zeros(size(NN));

for n=1:length(NN)
    tic
    N=NN(n);

    L=log2(N);

    %Level 0
    MLMCn=B(1)^2*T;

    %Level 1 to L
    for l=1:L
        Ml=N/2^(l-6); %number of repetitions
        v=repmat(B(2^(l-1)+1:2^l),1,Ml).*randn(2^(l-1),Ml)*sqrt(T);
        level=sum(sum(v.^2))/Ml;

% %         for better readability:
%         for j=1:Ml
%             v=B(2^(l-1)+1:2^l).*randn(1,2^(l-1))*sqrt(T);
%             level=level+sum(v.^2)/Ml;
%         end

```

```
32          MLMCn=MLMCn+level;
33      end
34      MLMC(n)=MLMCn;
35      time(n)=toc;
36  end
```

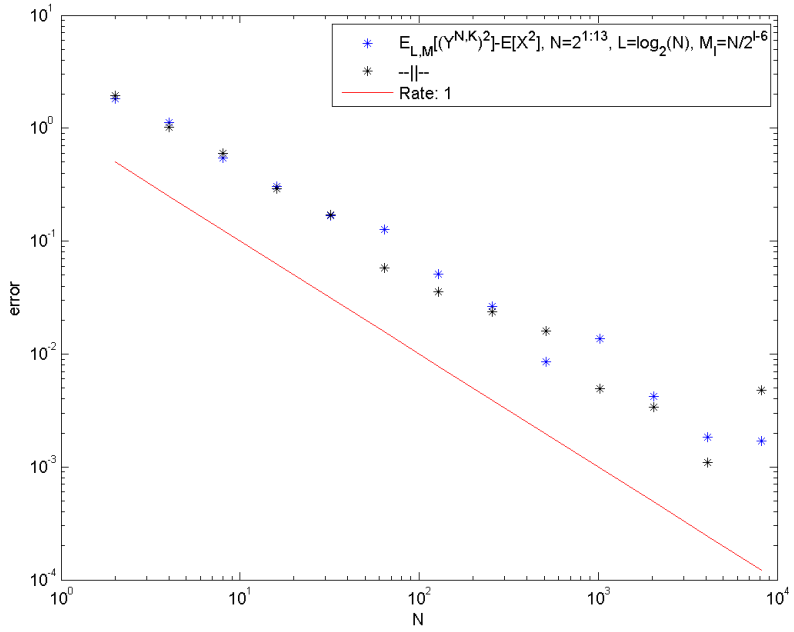

Figure 3.6: Error of MLMC, semi-discretized, $b(x,y) = 1$

As we see in Table 3.6 these calculations are much faster then the above. That is however not very surprising as we use a semi-linear approach here. Nevertheless there is quite a bit of variance left as we see in Figure 3.6. Setting the constant in the calculation of the number of Monte Carlo samples per level M_l to a higher value helps but gives us also higher calculation times. Both the plot and the table show again that the convergence rate is about one. The variance leads to up and downs in the results. Therefore, the EOC is not very readable and there are even negative values. However, looking at the general behaviour of the points gives us a quite good picture of the semi-linear scheme.

Fully Discrete Version

Again, the semi-linear approach is only a possible way to go in the case of additive noise ($b(x,y) = 1$). Therefore, we need to look at the fully dis-

Table 3.6: Error of MLMC, semi-discretized, $b(x,y) = 1$

N	Weak Error	EOC	Weak Error	EOC	time (in sec)
2	1.8285		1.9521		0.00014397
4	1.1229	0.70338	1.0127	0.94677	0.00015283
8	0.54502	1.0429	0.59577	0.76541	0.00016861
16	0.30663	0.82984	0.29058	1.0358	0.00023944
32	0.16835	0.86499	0.1709	0.7658	0.00036109
64	0.12663	0.41083	0.057853	1.5627	0.00061901
128	0.051004	1.312	0.035849	0.69047	0.0012049
256	0.026409	0.94956	0.023772	0.59267	0.0024687
512	0.0084901	1.6372	0.01606	0.56575	0.0042314
1024	0.013685	-0.68877	0.0049233	1.7058	0.0086449
2048	0.0042183	1.6979	0.0033584	0.55188	0.019086
4096	0.0018389	1.1978	0.0010929	1.6196	0.046592
8192	0.0016868	0.12456	0.0048125	-2.1386	0.10679

cretized scheme and use the Multilevel Monte Carlo again. We use the same number of samples per level M_l as above and the code uses parts of the code for the Difference Monte Carlo method described in 3.2.4. Thus, we get the following code.

```
% parallel on cluster:
cluster = parcluster('BrutusLSF8h')
matlabpool(cluster,100)

% set random seet based on current time and save seed.
rng('shuffle');
s=rng('shuffle');

NN = 2.^(1:11);

K=50000; %number of steps in time

data=zeros(1,length(NN));
time=zeros(1,length(NN));

%f: drift, b: diffusion coefficient
f = @(x) 0;
b = @(x) 1;


for n=1:length(NN) %for different number of steps in space
        tic

    N=NN(n); %number of steps in space
    L=log2(N); %number of levels

```

```
    %level 0
    Ml=1/2^(-6);
    z=zeros(1,Ml); %initialisation for numerical expectation
    parfor m=1:Ml
        z(m)=EulerGalerkin(1,K,f,b);
    end
    MLMCn=sum(z)/Ml;

    for l=1:L %level 1 to L
        Ml=N/2^(l-6); %number of repetitions
        clear z;
        z=zeros(1,Ml);
        N1=2^l;
        N2=2^(l-1);
        parfor m=1:Ml %for Monte Carlo
            Y1 = zeros(1,N1); %initalisation for process
            Y2 = zeros(1,N2);
            A1 = -pi^2*(1:N1).^2/100;
            A2 = -pi^2*(1:N2).^2/100;

                for k = 1:K %for time stepping
                    % generate random numbers
                    r=randn(1,max(N1,N2));

                    % calculate one step for first stepsize
                    y1 = dst(Y1) * sqrt(2);
                    dW = dst( r(1:N1) .* sqrt(2/K));
                    y1 = y1 + f(y1)/K + b(y1).*dW;
                    Y1 = idst(y1) / sqrt(2) ./ (1-A1/K);

                    % calculate one step for second stepsize
                    y2 = dst(Y2) * sqrt(2);
                    dW = dst( r(1:N2) .* sqrt(2/K));
                    y2 = y2 + f(y2)/K + b(y2).*dW;
                    Y2 = idst(y2) / sqrt(2) ./ (1-A2/K);
                end
            z1=sum(Y1.^2);
            z2=sum(Y2.^2);
            z(m)=abs(z1-z2);
        end
        %Monte Carlo
        MLMCn=MLMCn+sum(z)/Ml;
    end
    data(n)=MLMCn;
    time(n)=toc;
end

% save the data
name = [datestr(now, 'yymmddTHHMMSS') '.mat'];
```

```
76 save(name,'data','time','NN','s')
77
78 matlabpool close
```

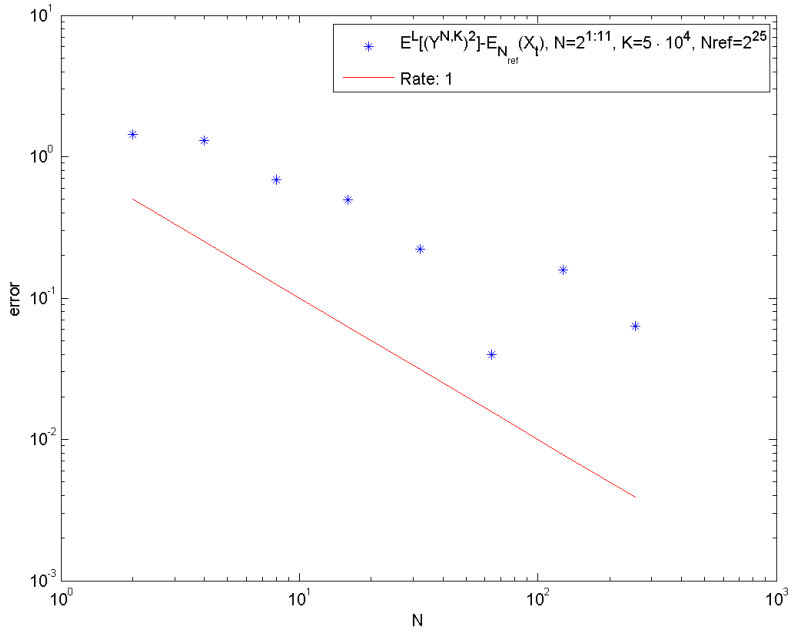

Figure 3.7: Error of MLMC: fully discretized, $b(x,y) = 1$

Table 3.7: Error of MLMC: fully discretized, $b(x,y) = 1$

N	Weak Error	EOC	time (in sec)
2	1.4333		34.176
4	1.2958	0.14542	80.735
8	0.68269	0.92459	156.81
16	0.49321	0.46903	305.64
32	0.22197	1.1519	645.62
64	0.039609	2.4864	1331.1
128	0.1581	-1.9969	2279.6
256	0.063306	1.3204	4600.8

As in the above cases we again notice that in Figure 3.7 the error of the MLMC decreases with about the same slope as the red line (slope 1). The EOC in Table 3.7 does not give us very convincing results. This is due to the variance that is still quite high. With a calculation time of about 2.5 hours for the first eight powers of two we are slower than with the Differ-

ence Monte Carlo scheme but still considerably faster than with the classical Monte Carlo scheme.
The corresponding results with multiplicative noise can be found in Section 3.3.3.

3.3 SPDEs with multiplicative noise

Finally, we are in the position to look at a SPDE with multiplicative noise. Our simulations will again be limited to the stochastic heat equation (1.9). However, we now no longer choose a constant diffusion coefficient but use another function $b(t, X(t))$. In the following sections we will always look at two different kinds of diffusion coefficients. First we look at the case of affine linear b. There we will choose

$$b(x, y) = 1 - \frac{y}{5}$$

as an example.
So in the MATLAB code we have to replace the line `b = @(x) 1` by `b = @(x) 1-x/5`.
There are already some theoretical papers looking at cases like this, for example [Deb11].
Additionally we also look at the case of nonlinear diffusion coefficients. Our example will be

$$b(x, y) = \frac{1 - \frac{y}{5}}{1 + y^2}.$$

This leads to replacing `b = @(x) 1` by `b = @(x) (1 - x/5)./(1 + x.^2)`.
To the best of my knowledge there are up till now no theoretical results about the convergence of the weak approximation error of this kind of numerical schemes using this diffusion coefficient (cf. Section 2.4.2).
In these cases it is no longer possible to calculate an exact solution, to look at the numerical error directly or use a semi-linear approach as we did in the additive case. Thus, we have to directly use the different Monte Carlo schemes described above.
The idea is now to use the code used in Section 3.2 and only change the diffusion coefficient. We use the same parameters for e.g. the number of Monte Carlo samples, the number of steps in time or space. We would expect to see the same convergence rates as before.
We start with the classical Monte Carlo scheme, then go on to the Difference Monte Carlo scheme and finish with the Multilevel Monte Carlo scheme.
In this new setting we are confronted with another problem: We are no longer able to calculate an exact solution or at least a very near approximation of the exact solution. Thus, we have to use a reference solution to

calculate the weak approximation error. From now on we always use the solution calculated with the highest number of grid points N in space as a reference solution.

3.3.1 Monte Carlo Scheme

With the classical Monte Carlo scheme we see results that are very similar to the first case. For the reference solution we used a number of grid points of 2^{11}. At first the convergence rate is very near to one as can be seen in Figures 3.8 and 3.9. Then we again see quite a big variance. Again we calculate for three grid sizes a simulation with the higher number of samples (80′000 instead of 20′000). And we see once more an improvement in the result.
Not surprisingly we have the same calculation times as before in the case described in Section 3.2.3.

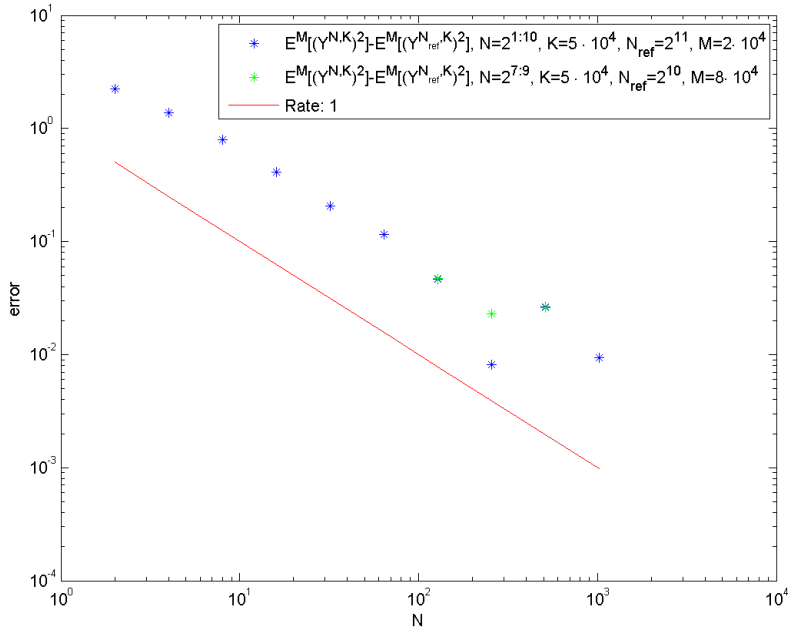

Figure 3.8: Error of MC: $b(x,y) = 1 - \frac{y}{5}$

3.3.2 Difference Monte Carlo Scheme

The difference Monte Carlo gives us now two advantages. We have lower computation times and we do no longer need a reference solution. As we

Table 3.8: Error MC: $b(x,y) = 1 - \frac{y}{5}$

N	Weak Error	EOC	time (in sec)	Weak Error	EOC	time (in sec)
2	2.2516		2679.5			
4	1.3829	0.70327	2681.1			
8	0.79076	0.8064	2737.2			
16	0.41245	0.93901	2861.7			
32	0.20604	1.0013	2904.8			
64	0.1156	0.8338	3107.2			
128	0.046333	1.319	3525.9	0.047539		13985
256	0.008129	2.5109	4385.6	0.023	1.0475	17477
512	0.026563	-1.7083	5098.1	0.025955	-0.1744	20252
1024	0.0094281	1.4944	8796.3			

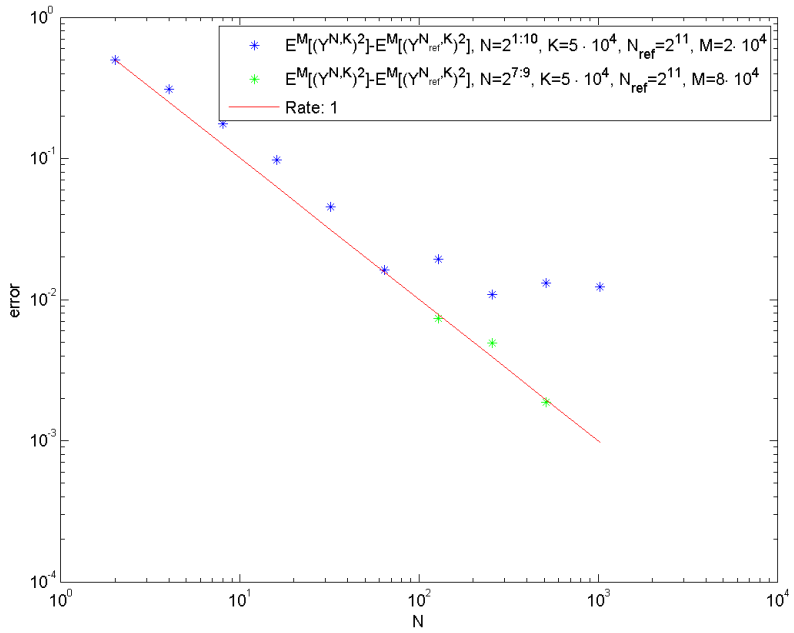

Figure 3.9: Error of MC: $b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$

Table 3.9: Error MC: $b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$

N	Weak Error	EOC	time (in sec)	Weak Error	EOC	time (in sec)
2	0.49855		1949.3			
4	0.30834	0.69323	2015.2			
8	0.17673	0.80296	2068.5			
16	0.097547	0.85738	2191.1			
32	0.045575	1.0978	2248.4			
64	0.016146	1.497	2406.1			
128	0.019401	-0.26492	2881.6	0.0073638	0	14017
256	0.010802	0.84479	3877.5	0.0049461	0.57415	17439
512	0.013072	-0.27518	4850	0.0018689	1.4041	20706
1024	0.012282	0.08999	8914.1			

Table 3.10: Error DMC: $b(x,y) = 1 - \frac{y}{5}$

N	Weak Error	EOC	time (in sec)
2	0.54524		27.519
4	0.75882	-0.47686	27.342
8	0.60647	0.32331	28.568
16	0.3346	0.85798	59.641
32	0.19701	0.7642	113.6
64	0.13499	0.54544	218.79
128	0.08815	0.61477	522.87
256	0.054696	0.68853	1236.7
512	0.033554	0.70494	3569.6
1024	0.01793	0.90409	13849
2048	0.0073789	1.2809	49113

see in Figure 3.11 we have a much lower convergence rate for small N in the nonlinear case $b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$. However, increasing N we again achieve a convergence rate of $1-\varepsilon$. In this case we simulated an additional point for $N = 2^{12}$. This simulations needs with a parallelization on 100 workers about 70 hours. Then we realize that as in the case of constant diffusion coefficient we get a higher convergence rate. Here about 1.5.
We have to mention that we have close to no variance at all. With each calculation we get the same result.

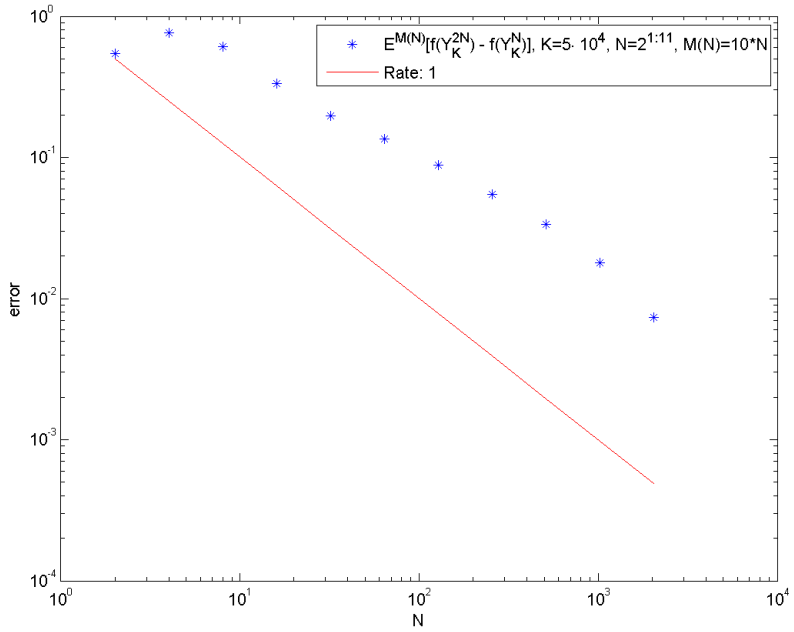

Figure 3.10: Error of DMC: $b(x,y) = 1 - \frac{y}{5}$

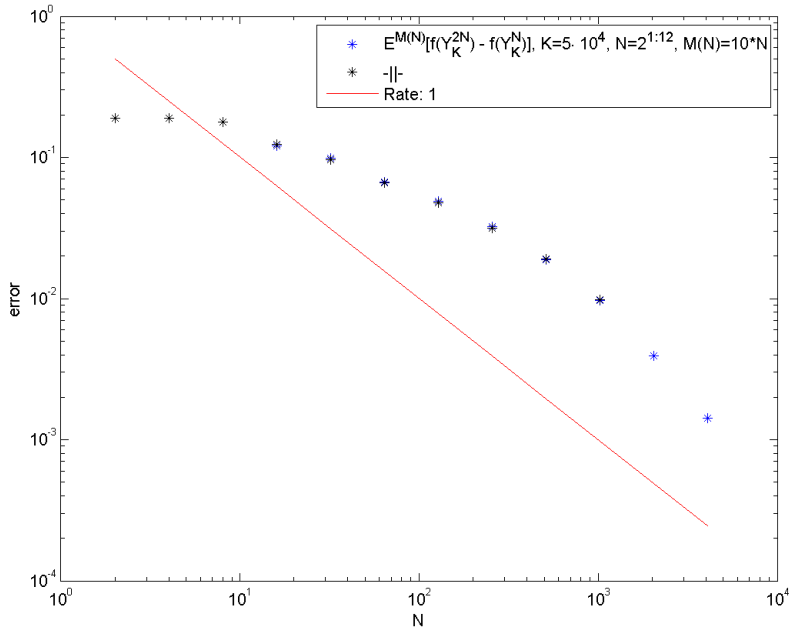

Figure 3.11: Error of DMC: $b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$

Table 3.11: Error DMC: $b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$

N	Weak Error	EOC	time (in sec)
2	0.18998		26.869
4	0.19063	-0.0049592	28.065
8	0.17754	0.10267	32.538
16	0.12097	0.55346	57.133
32	0.099037	0.28863	114.47
64	0.067153	0.5605	221.66
128	0.048655	0.46488	527.5
256	0.032357	0.58848	1250.1
512	0.018922	0.77405	3641.8
1024	0.0097295	0.95961	13810
2048	0.0039311	1.3074	52689
4096	0.0014239	1.4651	247220

In Figure 3.10 for the linear $b(x,y) = 1 - \frac{y}{5}$ it looks a little bit different. The convergence rate seems very stable right from the beginning (ignoring the first, very small N). It seem though that the convergence rate does not reach $1-\varepsilon$ but stays more around 0.7. A simulation with higher N could perhaps give us a better understanding of the behaviour observed here.

3.3.3 Multilevel Monte Carlo Scheme

With the Multilevel Monte Carlo Scheme we are back at our old problems. We need to use the solution calculated with the finest grid as reference solution. The calculation time is, however, not as bad as in the classical Monte Carlo case.
We see again a quite big variance and thus the EOC calculated in the Tables 3.12 and 3.13 is of not much use. If we look at the plots 3.12 and 3.13, though, we see that a convergence rate of $1-\varepsilon$ is a good approximation of the one seen there.
Again the result could be made more clear by using higher N or increasing the number of samples on each level.

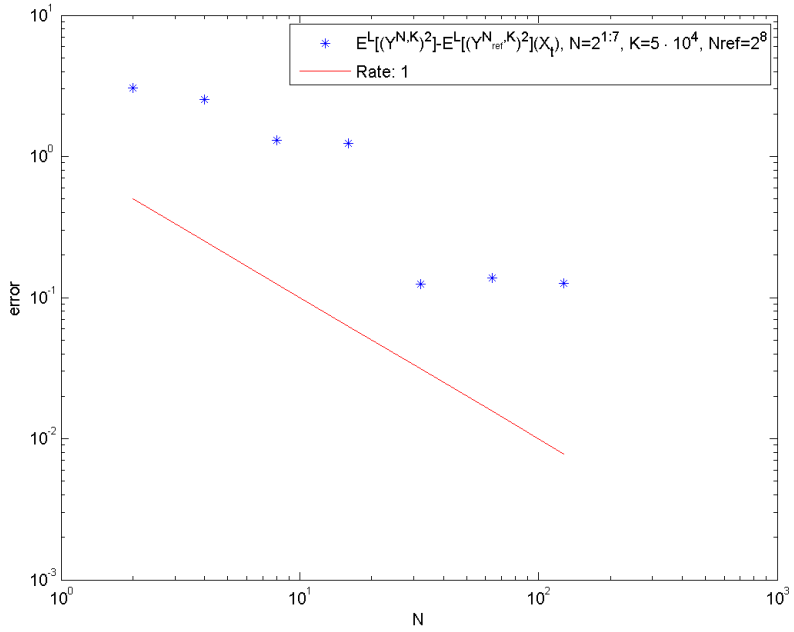

Figure 3.12: Error of MLMC: $b(x,y) = 1 - \frac{y}{5}$

Table 3.12: Error MLMC: $b(x,y) = 1 - \frac{y}{5}$

N	Weak Error	EOC	time (in sec)
2	3.0553		35.112
4	2.5259	0.27452	80.733
8	1.3049	0.95291	155.74
16	1.233	0.081696	300.28
32	0.1242	3.3114	575.08
64	0.13852	-0.15745	1096
128	0.12536	0.14405	2170.1

Table 3.13: Error MLMC: $b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$

N	Weak Error	EOC	time (in sec
2	0.80534		36.78
4	0.68563	0.23218	83.497
8	0.48445	0.50108	164
16	0.33526	0.53105	315.95
32	0.21817	0.61985	609.11
64	0.08021	1.4436	1128.3
128	0.06925	0.21196	2194.6

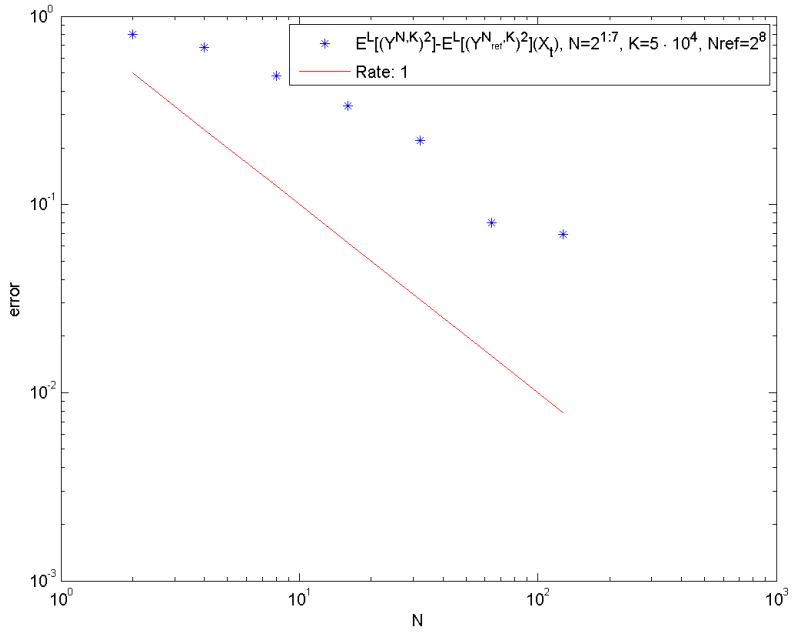

Figure 3.13: Error of MLMC: $b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$

Conclusion

The main goal of this thesis was to give an experimental convergence rate for the weak approximation error of the stochastic heat equation with space time white noise and a nonlinear diffusion coefficient.
We were starting from the assumption that the same setting with additive noise has a convergence rate of $1 - \varepsilon$. Therefore, we expected to see the same rate also in the more complex setting of nonlinear diffusion coefficients.
After revisiting the theory behind stochastic partial differential equations and the above mentioned former results, we started numerical experiments. The first bunch of experiments was made using a constant diffusion coefficient. This helped to understand the used numerical scheme, a spectral Galerkin method combined with the linear implicit Euler scheme. While using different methods to cope with the randomness of the solution we observed that it was always possible to achieve a convergence rate of $1 - \varepsilon$.
The above mentioned methods were the classical Monte Carlo method, the Multilevel Monte Carlo method and the Difference Monte Carlo method. The last one is a scheme that gives us directly the weak approximation error. It has two advantages over the other two methods. On the one hand a faster calculation is possible. On the other hand we do not need to calculate an exact solution or a reference solution to get the weak approximation error.
In summary we can conclude from these first numerical simulations that the Monte Carlo method is computationally very costly, but gives us good results. With the Multilevel Monte Carlo method we do not need that much computational time, but it was not possible for us to get as good results as with the classical Monte Carlo scheme. The Difference Monte Carlo scheme gives a little different results. There the results vary only a little. But the convergence rate is not as clear. It starts with a rate of about 1. But then the convergence gets faster and approaches a rate of 2. This could be due to the fact that the difference between different grids gets less and less the finer the grid.
Therefore, it was possible to confirm the previously known results.

Afterwards we looked at other diffusion coefficients. We used the functions

$$b(x,y) = 1 - \frac{y}{5}$$

and

$$b(x,y) = \frac{1 - \frac{y}{5}}{1 + y^2}.$$

Again the three different method Monte Carlo, Multilevel Monte Carlo and Difference Monte Carlo were used. As an additional obstacle we had to face the fact that we no longer where able to calculate a exact solution as reference for the weak approximation error. Therefore, we always used the solution approximated on the finest grid as a reference solution. This leads sometimes to a slower convergence rate for higher N as we are getting in the proximity of the reference solution.
In the figures illustrating the results for Monte Carlo and Multilevel Monte Carlo scheme we see a result very similar to the former ones. We have a convergence rate of about 1 with a high variance for larger N. For the Difference Monte Carlo scheme we again note different results. Looking at the linear diffusion coefficient $b(x,y) = 1 - \frac{y}{5}$ we see a convergence rate that stabilizes very fast, but does not reach the expected result of 1. It is more at about 0.7. The result of the nonlinear diffusion coefficient $b(x,y) = \frac{1-\frac{y}{5}}{1+y^2}$ differs again. We see first a very slow convergence and then reach the expected rate of $1 - \varepsilon$ only for quite large N.
All in all we conclude that we were able to numerically show that a convergence rate is possible for these more complex diffusion coefficients.
Further steps would now be to look at these results from a more theoretical point of view. Especially the Difference Monte Carlo scheme would be a meaningful area for further research. The behaviour of this scheme is not totally clear at this moment.
Additionally it would make sense to rerun some of the simulations above with more time and for a larger number of grid points N. This would give a clearer picture of the convergence rates evaluated.

Bibliography

[AL12] Adam Andersson and Stig Larsson, *Weak convergence for a spatial approximation of the nonlinear stochastic heat equation*, arXiv preprint arXiv:1212.5564 (2012), 18 pages.

[ANZ98] E.J. Allen, S.J. Novosel, and Z. Zhang, *Finite element and difference approximation of some linear stochastic partial differential equations*, Stochastics: An International Journal of Probability and Stochastic Processes **64** (1998), no. 1-2, 117–142.

[BJLS12] Andrea Barth, Arnulf Jentzen, Annika Lang, and Christoph Schwab, *Numerical analysis of stochastic ordinary differential equations*, lecture notes, 2012.

[BL12] Andrea Barth and Annika Lang, *Multilevel monte carlo method with applications to stochastic partial differential equations*, International Journal of Computer Mathematics **89** (2012), no. 18, 2479–2498.

[Bré12] Charles-Edouard Bréhier, *Approximation of the invariant measure with an euler scheme for stochastic pde's driven by space-time white noise*, arXiv preprint arXiv:1202.2707 (2012), 31 pages.

[BTZ04] Ivo Babuska, Raúl Tempone, and Georgios E Zouraris, *Galerkin finite element approximations of stochastic elliptic partial differential equations*, SIAM Journal on Numerical Analysis **42** (2004), no. 2, 800–825.

[DBD06] Anne De Bouard and Arnaud Debussche, *Weak and strong order of convergence of a semidiscrete scheme for the stochastic nonlinear schrodinger equation*, Applied Mathematics and Optimization **54** (2006), no. 3, 369–399.

[Deb11] Arnaud Debussche, *Weak approximation of stochastic partial differential equations: the nonlinear case*, Mathematics of Computation **80** (2011), no. 273, 89–117.

[DP09] Arnaud Debussche and Jacques Printems, *Weak order for the discretization of the stochastic heat equation*, Mathematics of Computation **78** (2009), no. 266, 845–863.

[DPJR10] Giuseppe Da Prato, Arnulf Jentzen, and Michael Roeckner, *A mild itô formula for spdes*, arXiv preprint arXiv:1009.3526 (2010), 39 pages.

[DPZ92] Guiseppe Da Prato and Jerzy Zabczyk, *Stochastic equations in infinite dimensions*, Cambridge University Press, 1992.

[Gil08] Michael B Giles, *Multilevel monte carlo path simulation*, Operations Research **56** (2008), no. 3, 607–617.

[GK96] W. Grecksch and P.E. Kloeden, *Time-discretised galerkin approximations of parabolic stochastic pde's*, Bulletin of the Australian mathematical society **54** (1996), no. 1, 79–86.

[GK03] István Gyöngy and Nicolai Krylov, *On the splitting-up method and stochastic partial differential equations*, The Annals of Probability **31** (2003), no. 2, 564–591.

[GKL09] Matthias Geissert, Mihály Kovács, and Stig Larsson, *Rate of weak convergence of the finite element method for the stochastic heat equation with additive noise*, BIT Numerical Mathematics **49** (2009), no. 2, 343–356.

[Hau03a] Erika Hausenblas, *Approximation for semilinear stochastic evolution equations*, Potential Analysis **18** (2003), no. 2, 141–186.

[Hau03b] ———, *Weak approximation for semilinear stochastic evolution equations*, Stochastic analysis and related topics VIII, Springer, 2003, pp. 111–128.

[Hau08] ———, *Finite element approximation of stochastic partial differential equations driven by poisson random measures of jump type*, SIAM Journal on Numerical Analysis **46** (2008), no. 1, 437–471.

[Hau10] ———, *Weak approximation of the stochastic wave equation*, Journal of computational and applied mathematics **235** (2010), no. 1, 33–58.

[Hei01] Stefan Heinrich, *Multilevel monte carlo methods*, Large-scale scientific computing, Springer, 2001, pp. 58–67.

[JK09] Arnulf Jentzen and Peter E. Kloeden, *The numerical approximation of stochastic partial differential equations*, Milan Journal of Mathematics **77** (2009), no. 1, 205–244.

[JK11] Arnulf Jentzen and Peter E. Kloeden, *Taylor approximations for stochastic partial differential equations*, CBMS-NSF regional conference series in applied mathematics, no. 83, siam Philadelphia, 2011.

[JKN09] A Jentzen, PE Kloeden, and A Neuenkirch, *Pathwise approximation of stochastic differential equations on domains: higher order convergence rates without global lipschitz coefficients*, Numerische Mathematik **112** (2009), no. 1, 41–64.

[JR10] Arnulf Jentzen and Michael Röckner, *A milstein scheme for spdes*, arXiv preprint arXiv:1001.2751 (2010), 37 pages.

[KLL10] Mihály Kovács, Stig Larsson, and Fredrik Lindgren, *Strong convergence of the finite element method with truncated noise for semilinear parabolic stochastic equations with additive noise*, Numerical Algorithms **53** (2010), no. 2-3, 309–320.

[KLL11] ———, *Weak convergence of finite element approximations of linear stochastic evolution equations with additive noise*, BIT Numerical Mathematics **52** (2011), no. 1, 85–108.

[KLL12] ———, *Weak convergence of finite element approximations of linear stochastic evolution equations with additive noise ii: Fully discrete schemes*, arXiv preprint arXiv:1203.2029v1 (2012), 24 pages.

[Kru12] Raphael Kruse, *Strong and weak approximation of semilinear stochastic evolution equations*, Ph.D. thesis, Universität Bielefeld, Bielefeld, 2012.

[LS13] Felix Lindner and René L Schilling, *Weak order for the discretization of the stochastic heat equation driven by impulsive noise*, Potential Analysis **38** (2013), no. 2, 345–379.

[mat13] *Matlab documentation: Control random number streams*, http://www.mathworks.ch/ch/help/distcomp/control-random-number-streams.html, July 2013.

[PR07] Claudia Prévôt and Michael Röckner, *A concise course on stochastic partial differential equations*, Springer, 2007.

[Rot02] Ch. Roth, *Difference methods for stochastic partial differential equations*, ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik **82** (2002), no. 11-12, 821–830.

[RS72] Michael Reed and Barry Simon, *Methods of modern mathematical physics: Vol.: 1.: Functional analysis*, Academic press, 1972.

[Tho97] Vidar Thomée, *Galerkin finite element methods for parabolic problems*, vol. 25, Springer Verlag, 1997.

[Wal05] John B. Walsh, *Finite element methods for parabolic stochastic pde's*, Potential Analysis **23** (2005), no. 1, 1–43.

[WG12] Xiaojie Wang and Siqing Gan, *Weak convergence analysis of linear implicit euler method for semilinear stochastic partial differential equations with additive noise*, Journal of Mathematical Analysis and Applications (2012), 151—-169.

[wik13] *Brutus wiki*, http://www.clusterwiki.ethz.ch/brutus/Brutus_wiki, June 2013.