Journal of Computational and Applied Mathematics 296 (2016) 593–612

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Efficient numerical Fourier methods for coupled forward–backward SDEs

T.P. Huijskens [a,*], M.J. Ruijter [b], C.W. Oosterlee [b,c]

[a] University of Oxford, Oxford, United Kingdom
[b] Centrum Wiskunde & Informatica, Amsterdam, The Netherlands
[c] Delft University of Technology, Delft, The Netherlands

ARTICLE INFO

Article history:
Received 23 December 2014
Received in revised form 19 October 2015

MSC:
91G60
60H35
65C30
65T50
60E10
65B05

Keywords:
Fourier-cosine expansion method
Characteristic function
Coupled forward–backward stochastic differential equations
Richardson extrapolation
Second-order convergence
Cross-hedging

ABSTRACT

We develop three numerical methods to solve coupled forward–backward stochastic differential equations. We propose three different discretization techniques for the forward stochastic differential equation. A theta-discretization of the time-integrands is used to arrive at schemes with conditional expectations. These conditional expectations are approximated by using the COS method, which relies on the availability of the conditional characteristic function of the discrete forward process. The numerical methods are applied to different problems, including a financial problem. Richardson extrapolation is used to obtain more accurate results, resulting in the observation of second-order convergence in the number of time steps. Advantages and disadvantages of each method are compared against each other.

1. Introduction

In recent years, the theory and applicability of forward–backward stochastic differential equations (FBSDEs) has grown. Such equations essentially consist of two parts, a forward stochastic differential equation (FSDE) which has to satisfy an initial condition, and a backward stochastic differential equation (BSDE), which has to satisfy a terminal condition:

$$\begin{aligned} X_t &= X_0 + \int_0^t \mu(s, X_s, Y_s, Z_s)ds + \int_0^t \sigma(s, X_s, Y_s)dW_s, \\ Y_t &= g(X_T) + \int_t^T f(s, X_s, Y_s, Z_s)ds - \int_t^T Z_s dW_s, \end{aligned} \tag{1.1}$$

where X_0 is the initial condition of the forward SDE, T is the (finite) time horizon, W_t is a Brownian motion process, μ and σ are the drift and diffusion of the forward SDE, f is the generator of the backward SDE and g is the terminal condition. Further

* Corresponding author.
E-mail addresses: thomas_huijskens@hotmail.com (T.P. Huijskens), marjonruijter@gmail.com (M.J. Ruijter), c.w.oosterlee@cwi.nl (C.W. Oosterlee).

http://dx.doi.org/10.1016/j.cam.2015.10.019
0377-0427/

technical details will be discussed in Section 2. A solution to (1.1) consists of three processes, a process X_t from the forward component, and two processes Y_t and Z_t from the backward component. Process Z_t can be interpreted as a *control process* such that Z_t steers Y_t towards the terminal condition and such that all processes are adapted.

In Eq. (1.1), the processes X_t, Y_t and Z_t are *coupled*. The forward component X_t is allowed to couple into the generator f and the backward components Y_t and Z_t are allowed to couple in the drift μ and/or the diffusion σ. In certain problems the functions μ and σ do not depend on the processes Y_t and Z_t and in this case (1.1) reduces to a *decoupled* FBSDE. Such equations are less complex than coupled FBSDEs but still form a concept with a wide range of applicability, especially in financial mathematics. Decoupled FBSDEs have been applied for the pricing of American options in [1]. A jump diffusion model for decoupled FBSDEs was treated in [2]. Other market imperfections, such as markets with frictions or a market with short selling constraints can also be incorporated with decoupled FBSDEs.

However, *coupled* FBSDEs form a concept with an even wider range of applicability, encompassing decoupled FSDEs, and can be used to approach nonlinear problems. For example, coupled FBSDEs were used to model financial markets where a large investor can influence the stock price in [3]. The solution of a coupled FBSDE is related (see [4]) to the solution of a second-order *quasilinear* partial differential equation (PDE), widening the range of applications. Nonlinear pricing models, such as a market model where we can only hedge a financial option with a correlated asset, but not the underlying itself, as discussed in [5], can also be modelled easily with coupled FBSDEs due to this connection with PDEs.

In general, we do not have the explicit solution of the coupled FBSDE (1.1) available to us. Therefore, there has been a lot of interest in the development of efficient numerical methods for coupled FBSDEs. However, the coupling of the forward and backward components in Eq. (1.1) makes the construction of numerical methods more involved. For decoupled FBSDEs, that do not have the added difficulty of the coupling, efficient algorithms already exist. Probabilistic methods rely on the dynamic programming equations developed in [6] and [7] and subsequently spawned a number of efficient methods, such as the BCOS method developed in [8], the convolution method from [9] or the integration methods from [10]. The research on efficient numerical methods for coupled FBSDEs is still ongoing. This paper is a step in that direction. Traditionally, there are two main problems to tackle when approximating coupled FBSDEs. Firstly, we need a method to deal with the coupling of the backward processes Y_t and Z_t into the forward process and secondly, we need to approximate the resulting conditional expectations.

We will use two techniques available in the literature, one from [11] and one from [12], and also introduce a new technique to approach the first problem. We will use a theta-method, first applied to decoupled FBSDEs in [13], to obtain flexibility in the computational effort of our methods. For the second problem, we use the COS method, developed in [14], and this will result in three different numerical methods for coupled FBSDEs.

Furthermore, we discuss two numerical experiments, to test the numerical methods. We will see that not all methods can cope with these experiments, and that each numerical method has its own advantages and disadvantages. To increase the speed of convergence, we will use Richardson extrapolation to obtain second-order accurate, in the number of timesteps, numerical approximations. Finally, we discuss a financial problem which can be modelled by coupled FBSDEs.

2. Forward–backward stochastic differential equations

We first introduce notation that will be used throughout the paper, and introduce a discretization of the general coupled FBSDE that forms the basis of the three schemes discussed in the paper.

2.1. Definitions

Let W_t be a standard one-dimensional Brownian motion process on the probability space $\left(\Omega, \mathcal{F}, (\mathcal{F}_t)_{0\leq t\leq T}, \mathbb{P}\right)$, where $T > 0$ is a fixed finite time horizon and $\mathcal{F}$ denotes the natural filtration generated by the process W_t, with the usual $\mathbb{P}$-augmentation so that the probability space is complete. We define the following spaces

1. $\mathbb{S}^2(0, T)$ denotes the set of $\mathbb{R}$-valued predictable measurable processes Y_t such that
$$\mathbb{E}\left[\sup_{0\leq t\leq T} |Y_t|^2\right] < \infty.$$
2. $\mathbb{H}^2(0, T)$ denotes the set of $\mathbb{R}$-valued predictable measurable processes Z_t such that
$$\mathbb{E}\left[\int_0^T |Z_t|^2 dt\right] < \infty.$$
3. $\mathbb{L}_T^2(\mathbb{R})$ denotes the set of $\mathcal{F}_T$-measurable random variables X that are square integrable.

Throughout this paper, we consider the *coupled forward–backward stochastic differential equation*:

$$X_t = X_0 + \int_0^t \mu(s, X_s, Y_s, Z_s)ds + \int_0^t \sigma(s, X_s, Y_s)dW_s, \tag{2.1a}$$

$$Y_t = g(X_T) + \int_t^T f(s, X_s, Y_s, Z_s)ds - \int_t^T Z_s dW_s, \tag{2.1b}$$

where $f : \Omega \times [0,T] \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is $\mathcal{P} \otimes \mathcal{B} \otimes \mathcal{B}$-measurable and continuous. Here, $\mathcal{P}$ denotes the set of $\mathcal{F}_t$-progressively measurable scalar processes on $\Omega \times [0,T]$. The function f is commonly referred to as the *generator* of the coupled FBSDE. Furthermore, $g : \Omega \to \mathbb{R}$ is assumed to be in $\mathbb{L}_T^2(\mathbb{R})$. The *drift* $\mu : \Omega \times [0,T] \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ and *diffusion* $\sigma : \Omega \times [0,T] \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ are also assumed to be continuous and respectively $\mathcal{P} \otimes \mathcal{B} \otimes \mathcal{B} \otimes \mathcal{B}$-measurable and $\mathcal{P} \otimes \mathcal{B} \otimes \mathcal{B}$-measurable.

For a solution of the coupled FBSDE to exist, we need the following *standing assumptions* (cf. [15])

1. The functions μ, σ, f and g are uniformly Lipschitz continuous in the space variables, for all $t \in [0,T]$. For example, f should satisfy
$$|f(t,x,y,z) - f(t,x',y',z')| \leq K(|x-x'| + |y-y'| + |z-z'|),$$
for all $t \in [0,T]$. Moreover, we assume these functions are continuously differentiable in all variables.
2. The functions μ, f and g satisfy the growth conditions:
$$\begin{aligned} |f(t,x,y,z)| &\leq K(1 + |y| + |z|), \\ |\mu(t,x,y,z)| &\leq K, \\ |g(x)| &\leq K, \end{aligned}$$
for some $K > 0$.
3. The diffusion σ has bounded second derivatives, and has positive lower and upper bound:
$$c \leq \sigma(t,x,y) \leq C,$$
where $0 < c < C$.

In particular, assumption (A2) entails that the function f is uniformly bounded in x. The solution to the coupled FBSDE is given by the triplet (X, Y, Z), where X and Y are continuous, real-valued and adapted processes and where Z is a continuous, real-valued and predictable process such that the triplet solves (2.1a)–(2.1b). Under assumptions (A1)–(A3) it can be proven that a unique solution $(X, Y, Z) \in \mathbb{S}^2(0,T) \times \mathbb{S}^\infty(0,T) \times \mathbb{H}^2(0,T)$ to the coupled FBSDE (1.1) exists.

Under the standing assumptions, a 'nonlinear Feynman–Kac' theorem can also be proven. Consider the *quasilinear* partial differential equation (PDE):

$$\begin{aligned} &D_t v(t,x) + \mu(t,x,y,z) D_x v(t,x) + \frac{1}{2} D_{xx} v(t,x) \sigma(t,x,v(t,x))^2 \\ &\quad - f(t,x,v(t,x), D_x v(t,x) \sigma(t,x,v(t,x))) = 0, \\ &v(T,x) = g(x), \end{aligned} \tag{2.2}$$

where D_t denotes the time derivative and D_x and D_{xx} denote the first and second order partial derivative to x, respectively. In the following, let $X_s^{t,x}$ denote the solution to the forward component (2.1a), starting in x at time t and where $s \in [t,T]$. The solution to the corresponding backward component (2.1b) is denoted by $(Y_s^{t,x}, Z_s^{t,x})$. The following theorem holds:

Theorem 2.1 (*Nonlinear Feynman–Kac Theorem [16,4]*). *If the PDE (2.2) has a solution $v(t,x) \in C^{1,2}$, the solution (X_t, Y_t, Z_t) of the (Markovian) coupled FBSDE (2.1a)–(2.1b) can be represented as*

$$\begin{aligned} Y_s^{t,x} &= v(s, X_s^{t,x}), \\ Z_s^{t,x} &= D_x v(s, X_s^{t,x}) \sigma(s, X_s^{t,x}, v(s, X_s^{t,x})), \end{aligned} \tag{2.3}$$

for all $s \in [t,T]$.

The converse statement also holds in the viscosity sense. Suppose $(X_s^{t,x}, Y_s^{t,x}, Z_s^{t,x})$ is a solution to the coupled FBSDE, then the function defined by $v(t,x) = Y_t^{t,x}$ is a viscosity solution to the PDE (2.2) (see [4]).

As a consequence of Theorem 2.1, solving a coupled FBSDE is the same as solving the corresponding quasilinear PDE. Therefore, numerical discretization schemes for PDEs may be used to approximate the solution of a coupled FBSDE.

In this paper we will focus on the probabilistic point of view, while borrowing some results from Theorem 2.1. We will discretize the forward component by the simple Euler scheme and the backward component by a theta-scheme. The use of a theta-scheme introduces two parameters θ_1 and θ_2, which are used to generate multiple discretizations. These different discretizations can then be compared later on in terms of efficiency and order of convergence. In the next subsection we will discuss the discretization of the forward and backward components of the coupled FBSDE.

2.2. Discretization of the coupled FBSDE

The Euler method is a well-known scheme to approximate the solution of (2.1a). Let Π be a partition of time points $0 = t_0 < t_1 < \cdots < t_{\mathcal{M}} = T$ of $[0,T]$ with a *fixed* time step $\Delta t := t_{m+1} - t_m$ for $m = \mathcal{M} - 1, \ldots, 0$. Throughout

this paper, we use the notation X_m to denote the value of any arbitrary stochastic process X_t at time point t_m and we set $\Delta W_{m+1} = W_{m+1} - W_m$, where W_m is $\mathcal{N}(0, t_m)$-distributed.

The local evolution, i.e. the evolution on a small time interval $[t_m, t_{m+1})$, of the forward SDE (2.1a) is given by:

$$X_{m+1} = X_m + \int_{t_m}^{t_{m+1}} \mu(s, X_s, Y_s, Z_s)ds + \int_{t_m}^{t_{m+1}} \sigma(s, X_s, Y_s)dW_s. \tag{2.4}$$

The Euler approximation X^{Δ}_{m+1} of X_{m+1} is then given by

$$X^{\Delta}_{m+1} = X^{\Delta}_m + \mu(t_m, X^{\Delta}_m, Y^{\Delta}_m, Z^{\Delta}_m)\Delta t + \sigma(t_m, X^{\Delta}_m, Y^{\Delta}_m)\Delta W_{m+1}, \tag{2.5}$$

which can be interpreted as a left-point approximation rule for the integrals appearing in (2.4). It is known (see [17]) that the classical Euler scheme has strong order of convergence equal to $\frac{1}{2}$ and weak order of convergence equal to 1.

To discretize the backward equation, we look at the local evolution of the backward component (2.1b) of the coupled FBSDE. We then start with

$$Y_m = Y_{m+1} + \int_{t_m}^{t_{m+1}} f(s, X_s, Y_s, Z_s)ds - \int_{t_m}^{t_{m+1}} Z_s dW_s. \tag{2.6}$$

One can obtain an approximation of the process Y_t by taking conditional expectations with respect to the underlying filtration $\mathcal{F}_m$. Using standard techniques in stochastic analysis (Fubini's theorem, properties of the Itô integral and the Itô isometry) and by approximating the integrals that appear with a theta method, as in [8], one obtains the following approximation

$$\begin{aligned} Y_m &= \mathbb{E}_m[Y_{m+1}] + \int_{t_m}^{t_{m+1}} \mathbb{E}_m[f(s, X_s, Y_s, Z_s)]\,ds \\ &\approx \mathbb{E}_m[Y_{m+1}] + \Delta t\theta_1 f(t_m, X_m, Y_m, Z_m) + \Delta t(1-\theta_1)\mathbb{E}_m[f(t_{m+1}, X_{m+1}, Y_{m+1}, Z_{m+1})], \end{aligned} \tag{2.7}$$

where $\theta_1 \in [0, 1]$. Here $\mathbb{E}_m[\cdot]$ denotes $\mathbb{E}[\cdot \mid \mathcal{F}_m]$. Similarly, by first multiplying Eq. (2.6) with ΔW_{m+1} and then taking conditional expectations, one can obtain an approximation for Z_m as well:

$$\begin{aligned} Z_m \approx{}& -\theta_2^{-1}(1-\theta_2)\mathbb{E}_m[Z_{m+1}] + \Delta t^{-1}\theta_2^{-1}\mathbb{E}_m[Y_{m+1}\Delta W_{m+1}] \\ &+ \theta_2^{-1}(1-\theta_2)\mathbb{E}_m[f(t_{m+1}, X_{m+1}, Y_{m+1}, Z_{m+1})\Delta W_{m+1}], \end{aligned} \tag{2.8}$$

where $\theta_2 \in (0, 1]$. As a consequence of Theorem 2.1, the terminal values $Y_{\mathcal{M}}$ and $Z_{\mathcal{M}}$ can be computed by using the terminal condition as follows

$$Y_{\mathcal{M}} = g(X_{\mathcal{M}}), \tag{2.9}$$

$$Z_{\mathcal{M}} = \sigma(t_{\mathcal{M}}, X_{\mathcal{M}}, Y_{\mathcal{M}})D_x g(X_{\mathcal{M}}). \tag{2.10}$$

Eqs. (2.7)–(2.10) will form the basis for all three numerical methods that will follow. Since, for our schemes, the terminal conditions (2.9) and (2.10) are functions of the time t and the Markov process X^{Δ}, we get by an induction argument that also the approximations Y^{Δ}_m and Z^{Δ}_m are of the form:

$$\begin{aligned} Y^{\Delta}_m &= y(t_m, X^{\Delta}_m), \\ Z^{\Delta}_m &= z(t_m, X^{\Delta}_m). \end{aligned} \tag{2.11}$$

Furthermore, since the filtration $\mathcal{F}$ is generated by the Brownian motion W_t, we get that

$$\mathbb{E}_m[\cdot] = \mathbb{E}[\cdot \mid \mathcal{F}_m] = \mathbb{E}[\cdot \mid X_m = x], \tag{2.12}$$

which we will approximate by $\mathbb{E}^x_m[\cdot] := \mathbb{E}\left[\cdot \mid X^{\Delta}_m = x\right]$. We can rephrase scheme (2.7)–(2.10) as follows:

$$y(t_{\mathcal{M}}, x) = g(x), \qquad z(t_{\mathcal{M}}, x) = D_x g(x)\sigma(t_{\mathcal{M}}, x, g(x)), \tag{2.13}$$

$$\begin{aligned} y(t_m, x) = {}& \mathbb{E}^x_m\left[y(t_{m+1}, X^{\Delta}_{m+1})\right] + \Delta t\theta_1 f(t_m, x, y(t_m, x), z(t_m, x)) \\ &+ \Delta t(1-\theta_1)\mathbb{E}^x_m\left[f(t_{m+1}, X^{\Delta}_{m+1}, y(t_{m+1}, X^{\Delta}_{m+1}), z(t_{m+1}, X^{\Delta}_{m+1}))\right], \end{aligned} \tag{2.14}$$

$$\begin{aligned} z(t_m, x) = {}& -\theta_2^{-1}(1-\theta_2)\mathbb{E}^x_m\left[z(t_{m+1}, X^{\Delta}_{m+1})\right] + \Delta t^{-1}\theta_2^{-1}\mathbb{E}^x_m\left[y(t_{m+1}, X^{\Delta}_{m+1})\Delta W_{m+1}\right] \\ &+ \theta_2^{-1}(1-\theta_2)\mathbb{E}^x_m\left[f(t_{m+1}, X^{\Delta}_{m+1}, y(t_{m+1}, X^{\Delta}_{m+1}), z(t_{m+1}, X^{\Delta}_{m+1}))\Delta W_{m+1}\right], \end{aligned} \tag{2.15}$$

for $m = \mathcal{M}-1, \ldots, 0$ and x in a predefined computational domain. The parameters θ_1 and θ_2 give us control of the behaviour and computational effort of our numerical methods. Setting $\theta_1 = 0$ will result in an *explicit* scheme for the backward component, while setting $\theta_1 \neq 0$ results in an *implicit* scheme. In the implicit case, we end up with a fixed-point problem for the function $y(t, x)$, which we will solve by using P_{picard} Picard iterations.

It is important to note that scheme (2.13)–(2.15) does not yet provide a fully implementable scheme. Indeed, for the computation of $y(t_m, x)$ and $z(t_m, x)$ we need the Euler approximation of X^{Δ}_{m+1}, which in turn needs the approximations $y(t_m, x)$ and $z(t_m, x)$ due to the coupling of the backward components in the forward process. In the next section, we will discuss three different methods to tackle this problem, resulting in three different algorithms.

First, however, we introduce the BCOS method, originating from [14], and further developed and successfully applied to *decoupled* FBSDEs in [8,18], for the approximation of the different conditional expectations arising in scheme (2.13)–(2.15).

2.3. Computation of conditional expectations with COS method

In this section we derive the COS approximation for the conditional expectations in (2.13)–(2.15) under the forward dynamics (2.1a). Since the three algorithms we will discuss differ from this scheme, the final COS formulas used in the three algorithms will look different from the ones obtained in this section. However, they are alike and the derivation for each algorithm does not change significantly (in each scheme the forward dynamics are different), so we discuss the derivation first.

It can be seen from Eqs. (2.13)–(2.15) that all conditional expectations we need to approximate are either of the form $\mathbb{E}^x_m\left[h(t_{m+1}, X^{\Delta}_{m+1})\right]$ or $\mathbb{E}^x_m\left[h(t_{m+1}, X^{\Delta}_{m+1})\Delta W_{m+1}\right]$, where $h(t, x)$ is some given function. In the following, let $N > 0$ denote the number of Fourier cosine coefficients we use, and let $[a, b]$ denote our computational domain.

For the first conditional expectation, the COS formula gives us:

$$\begin{aligned}\mathbb{E}^x_m\left[h(t_{m+1}, X^{\Delta}_{m+1})\right] &\approx \sum_{j=0}^{N-1}{}' \mathcal{H}_j(t_{m+1})\mathbb{E}^x_m\left[\cos\left(j\pi\frac{X^{\Delta}_{m+1}-a}{b-a}\right)\right]\\ &= \sum_{j=0}^{N-1}{}' \mathcal{H}_j(t_{m+1})\Re\left\{\phi_{X^{\Delta}_{m+1}}\left(\frac{j\pi}{b-a}\middle| X^{\Delta}_m = x\right)\exp\left(ij\pi\frac{-a}{b-a}\right)\right\},\end{aligned} \tag{2.16}$$

where i is the complex unit, $\sum'$ denotes an ordinary summation where the first term is weighted by one-half, $\mathcal{H}_j(t_{m+1})$ denotes the Fourier-cosine coefficients of $h(t, x)$ at time t_{m+1}, and $\phi_{X^{\Delta}_{m+1}}(u|x)$ is the conditional characteristic function of X^{Δ}_{m+1}, given X^{Δ}_m. The $\approx$ sign in (2.16) is due to the fact that we approximate an infinite sum with a finite number of terms. The computation of the Fourier-cosine coefficients is outlined below, in Eq. (2.21). Since we know the evolution of X^{Δ}_{m+1} from Eq. (2.5), we can compute the conditional characteristic function:

$$\begin{aligned}\phi_{X^{\Delta}_{m+1}}(u|X^{\Delta}_m = x) &= \mathbb{E}\left[\exp\left(iuX^{\Delta}_{m+1}\right) \mid X^{\Delta}_m = x\right]\\ &= \exp\left(iux + iu\mu(t_m, x, y(t_m, x), z(t_m, x))\Delta t - \frac{1}{2}u^2\sigma^2(t_m, x, y(t_m, x))\Delta t\right).\end{aligned} \tag{2.17}$$

It is interesting to note that the conditional characteristic function is not only space-dependent, but time-dependent as well.

Applying the COS formulas to the conditional expectation $\mathbb{E}^x_m\left[h(t_{m+1}, X^{\Delta}_{m+1})\Delta W_{m+1}\right]$, we get:

$$\begin{aligned}\mathbb{E}^x_m\left[h(t_{m+1}, X^{\Delta}_{m+1})\Delta W_{m+1}\right] &\approx \sum_{j=0}^{N-1}{}' \mathcal{H}_j(t_{m+1})\mathbb{E}^x_m\left[\cos\left(j\pi\frac{X^{\Delta}_{m+1}-a}{b-a}\right)\Delta W_{m+1}\right]\\ &= \sum_{j=0}^{N-1}{}' \mathcal{H}_j(t_{m+1})\Re\left\{\mathbb{E}^x_m\left[\exp\left(ij\pi\frac{X^{\Delta}_{m+1}}{b-a}\right)\Delta W_{m+1}\right]\exp\left(ij\pi\frac{-a}{b-a}\right)\right\}.\end{aligned} \tag{2.18}$$

Applying integration by parts, one obtains for the expectation term in Eq. (2.18), where we substitute $u = \frac{j\pi}{b-a}$, to simplify representation:

$$\begin{aligned}&\mathbb{E}^x_m\left[\exp\left(iuX^{\Delta}_{m+1}\right)\Delta W_{m+1}\right]\\ &= \mathbb{E}^x_m\left[\exp\left(iux + iu\underbrace{\mu(t_m, x, y(t_m, x), z(t_m, x))}_{=:m(t_m,x)}\Delta t + iu\underbrace{\sigma(t_m, x, y(t_m, x))}_{=:s(t_m,x)}\Delta W_{m+1}\right)\Delta W_{m+1}\right]\\ &= \frac{1}{\sqrt{2\pi}\sqrt{\Delta t}}\int_{-\infty}^{\infty}\exp\left(iux + ium(t_m, x)\Delta t + ius(t_m, x)\xi\right)\xi\exp\left(-\frac{1}{2}\left(\frac{\xi}{\sqrt{\Delta t}}\right)^2\right)d\xi\\ &= \frac{s(t_m, x)\Delta t}{\sqrt{2\pi}\sqrt{\Delta t}}\int_{-\infty}^{\infty}D_x\exp\left(iux + ium(t_m, x)\Delta t + ius(t_m, x)\xi\right)\exp\left(-\frac{1}{2}\left(\frac{\xi}{\sqrt{\Delta t}}\right)^2\right)d\xi\\ &= \Delta t\sigma(t_m, x, y(t_m, x))\mathbb{E}^x_m\left[D_x\exp\left(iuX^{\Delta}_{m+1}\right)\right].\end{aligned} \tag{2.19}$$

This derivation can be found in [18] in the case of decoupled FBSDEs and for more general discretizations of the forward SDE. Substituting Eq. (2.19) in (2.18) gives us

$$\begin{aligned}
&\mathbb{E}_m^x\left[h(t_{m+1}, X_{m+1}^{\Delta})\Delta W_{m+1}\right] \\
&\quad\approx \Delta t\sigma(t_m, x, y(t_m, x))\sum_{j=0}^{N-1}{}'\mathcal{H}_j(t_{m+1})\Re\left\{\mathbb{E}_m^x\left[D_x\exp\left(ij\pi\frac{X_{m+1}^{\Delta}}{b-a}\right)\right]\exp\left(ij\pi\frac{-a}{b-a}\right)\right\} \\
&\quad= \Delta t\sigma(t_m, x, y(t_m, x))\sum_{j=0}^{N-1}{}'\mathcal{H}_j(t_{m+1})\Re\left\{i\frac{j\pi}{b-a}\phi_{X_{m+1}^{\Delta}}\left(\frac{j\pi}{b-a}\middle| X_m^{\Delta}=x\right)\exp\left(ij\pi\frac{-a}{b-a}\right)\right\}. \qquad (2.20)
\end{aligned}$$

Since the conditional characteristic function is space-dependent, the computations of the conditional expectations in (2.18) and (2.20) include a dense matrix–vector multiplication, as we cannot employ a Fast Fourier Transform (FFT) algorithm here.

Finally, we need to approximate the Fourier-cosine coefficients $\mathcal{H}_j(t_{m+1})$ of h at time points t_m, where $m = 0, \ldots, \mathcal{M}$ and $j = 0, \ldots, N-1$. The Fourier-cosine coefficients of h at time t_m are defined by

$$\mathcal{H}_j(t_{m+1}) = \frac{2}{b-a}\int_a^b h(t_{m+1}, x)\cos\left(j\pi\frac{x-a}{b-a}\right)dx. \qquad (2.21)$$

Although this integral may be available to us analytically for some problems, in general it is not and we will approximate the integral with a discrete Fourier-cosine transform. To obtain such an approximation, we take N equidistant grid-points x_n on the grid $[a, b]$ and compute the value of $h(t_{m+1}, x)$ on these grid-points. The midpoint integration rule then gives us the approximation

$$\begin{aligned}
\mathcal{H}_j(t_{m+1}) &= \frac{2}{b-a}\int_a^b h(t_{m+1}, x)\cos\left(j\pi\frac{x-a}{b-a}\right)dx \\
&\approx \frac{2}{b-a}\sum_{n=0}^{N-1}{}'h(t_{m+1}, x_n)\cos\left(j\pi\frac{x_n-a}{b-a}\right)\delta x \\
&= \frac{2}{N}\sum_{n=0}^{N-1}{}'h(t_{m+1}, x_n)\cos\left(j\pi\frac{2n+1}{2N}\right). \qquad (2.22)
\end{aligned}$$

This approximation is known as the discrete Fourier-cosine transform, for which efficient algorithms exist (we will use MATLAB's function `dct`).

We now have the machinery we need to approximate the conditional expectations in scheme (2.13)–(2.15). Convergence of this scheme, with the COS formulas (2.16) and (2.20), is established for the decoupled case in [18]. However, as noted before, this scheme still does not provide an implementable scheme in our situation due to the coupling of the backward components in the forward component (and vice versa).

3. Numerical schemes

In this section, we will describe three different approaches to deal with the coupling in scheme (2.13)–(2.15), which will result in three different algorithms. These algorithms will all use Eqs. (2.16) and (2.20) in a slightly different fashion. Where necessary, we will point out the main differences in their derivations.

Recall that N denotes the number of Fourier-cosine coefficients used in the COS method and that $[a, b]$ denotes the integration interval[1] on which we approximate the forward process X_t.

3.1. Explicit method

Firstly, one can use a technique from [11] to obtain an *explicit* scheme for coupled FBSDEs. Suppose we are working on the time interval $[t_m, t_{m+1})$ and that approximations $y(t_{m+1}, \cdot)$ and $z(t_{m+1}, \cdot)$ are available on $[a, b]$. Assuming the time step $\Delta t = t_{m+1} - t_m$ is small, these approximations can be interpreted as *predictors* of the solution at time point t_m. The following approximation of the forward process then appears:

$$X_{m+1} = x + \mu(t_{m+1}, x, y(t_{m+1}, x), z(t_{m+1}, x))\Delta t + \sigma(t_{m+1}, x, y(t_{m+1}, x))\Delta W_{m+1}, \quad X_m = x. \qquad (3.1)$$

[1] As the characteristic function changes on each interval, one could choose $[a_m, b_m]$ as the computational domain for each $m = 0, \ldots, \mathcal{M}$. A choice for a_m and b_m could be based on the cumulants of the forward dynamics, as in [14,8,18], with corresponding minimum (for a) and maximum (for b) values of the processes Y_t and Z_t. In our examples however, we fix a and b for all time intervals.

This equation differs from (2.5) in the sense that we now use the *right-point approximation* for the y, z and time components in the drift and diffusion. It can be interpreted as a sort of *backward Euler* approximation of the forward SDE.

Since we use a different approximation of the forward process X_t, the conditional characteristic function of X_m^{Δ} will be different as well. The difference is small however, since we only have to replace t_m in (2.17) with t_{m+1}:

$$\phi_{X_{m+1}^{\Delta}}(u|X_m^{\Delta}=x)=\exp\left(iux+iu\mu(t_{m+1},x,y(t_{m+1},x),z(t_{m+1},x))\Delta t-\frac{1}{2}u^2\sigma^2(t_{m+1},x,y(t_{m+1},x))\Delta t\right). \tag{3.2}$$

Furthermore, we have to replace t_m with t_{m+1} in Eqs. (2.16) and (2.20), as their derivations do not change. Again, to simplify notation, we define

$$\Phi_j(x)=\Re\left\{\phi_{X_{m+1}^{\Delta}}\left(\frac{j\pi}{b-a}\middle|X_m^{\Delta}=x\right)\exp\left(ij\pi\frac{-a}{b-a}\right)\right\} \tag{3.3}$$

$$\bar{\Phi}_j(x)=\Re\left\{i\frac{j\pi}{b-a}\phi_{X_{m+1}^{\Delta}}\left(\frac{j\pi}{b-a}\middle|X_m^{\Delta}=x\right)\exp\left(ij\pi\frac{-a}{b-a}\right)\right\}. \tag{3.4}$$

In the rest of this subsection we denote the Fourier cosine-coefficients of $z(t_m,x)$, $y(t_m,x)$ and $f(t_m,x,y(t_m,x),z(t_m,x))$ respectively by $\mathcal{Z}_j(t_m)$, $\mathcal{Y}_j(t_m)$ and $\mathcal{F}_j(t_m)$ (for $j=0,\ldots,N-1$). Since the approximation of the forward process does not depend anymore on the approximations of the backward processes on time t_m, we obtain an explicit, fully implementable scheme.

Assume we want to compute $z(t_m,x)$ and $y(t_m,x)$ for arbitrary m. By using the COS formulas (2.16) and (2.20), adapted by substituting t_{m+1} instead of t_m, we obtain the approximations

$$\begin{aligned}z(t_m,x)\approx{}&\sum_{j=0}^{N-1}{}'-\frac{1-\theta_2}{\theta_2}\mathcal{Z}_j(t_{m+1})\Phi_j(x)\\&+\left(\frac{1}{\Delta t\theta_2}\mathcal{Y}_j(t_{m+1})+\frac{1-\theta_2}{\theta_2}\mathcal{F}_j(t_{m+1})\right)\sigma(t_{m+1},x,y(t_{m+1},x))\Delta t\bar{\Phi}_j(x).\end{aligned} \tag{3.5}$$

$$\begin{aligned}y(t_m,x)&\approx\Delta t\theta_1 f(t_m,x,y(t_m,x),z(t_m,x))+\sum_{j=0}^{N-1}{}'\Phi_j(x)\left(\mathcal{Y}_j(t_{m+1})+\Delta t(1-\theta_1)\mathcal{F}_j(t_{m+1})\right)\\&=\Delta t\theta_1 f(t_m,x,y(t_m,x),z(t_m,x))+p(t_{m+1},x).\end{aligned} \tag{3.6}$$

In the case $\theta_1>0$, we obtain an implicit equation for $y(t_m,x)$. We use P_{picard} Picard iterations to determine $y(t_m,x)$, with initial guess $\mathbb{E}_m^x\left[y(t_{m+1},X_{m+1}^{\Delta})\right]$. In the case $\theta_1=0$, we obtain an explicit approximation for $y(t_m,x)$, resulting in a computationally cheaper scheme (while still maintaining the same accuracy).

The *explicit* algorithm is summarized in Algorithm 1.

Algorithm 1: Explicit method

1: Compute the terminal functions $y(t_{\mathcal{M}},x)$ and $z(t_{\mathcal{M}},x)$ by using the terminal condition.
2: Approximate the terminal coefficients $\mathcal{Y}_j(t_{\mathcal{M}})$, $\mathcal{Z}_j(t_{\mathcal{M}})$ and $\mathcal{F}_j(t_{\mathcal{M}})$, by using a discrete Fourier-cosine transform.
3: **for** $m=\mathcal{M}-1$ to 0 **do**
4: Compute the functions $z(t_m,x)$, $y(t_m,x)$ and $f(t_m,x,y(t_m,x),z(t_m,x))$ by using
5: formulas (3.6) and (3.5).
6: Subsequently, approximate the corresponding coefficients $\mathcal{Z}_j(t_m)$, $\mathcal{Y}_j(t_m)$ and $\mathcal{F}_j(t_m)$ by
7: using a discrete Fourier-cosine transform.
8: **end for**

At the terminal time $t_{\mathcal{M}}$ we compute the Fourier-cosine coefficients by using the terminal condition

$$\mathcal{Y}_j(t_{\mathcal{M}})=\frac{2}{b-a}\int_a^b g(x)\cos\left(j\pi\frac{x-a}{b-a}\right)dx, \tag{3.7}$$

$$\mathcal{Z}_j(t_{\mathcal{M}})=\frac{2}{b-a}\int_a^b D_x g(x)\sigma(t_{\mathcal{M}},x,g(x))\cos\left(j\pi\frac{x-a}{b-a}\right)dx, \tag{3.8}$$

$$\mathcal{F}_j(t_{\mathcal{M}})=\frac{2}{b-a}\int_a^b f(t_{\mathcal{M}},x,g(x),D_x g(x)\sigma(t_{\mathcal{M}},x,g(x)))\cos\left(j\pi\frac{x-a}{b-a}\right)dx. \tag{3.9}$$

It is clear that the explicit method is computationally simple. Furthermore, for any arbitrary value of m, we need to store the values $y(t_{m+1},x)$ and $z(t_{m+1},x)$ for Eqs. (3.6) and (3.5). When we move on to the next iteration, these values are not needed anymore and can be removed.

The computation time for the explicit method is linear in the number of time steps M. The following computations are done when computing the approximations for time t_m:

- Computation of the conditional characteristic function on an x-grid in $\mathcal{O}(N)$ operations.
- Computation of $z(t_m, x)$ and $p(t_m, x)$ on a x-grid in $\mathcal{O}(N^2)$ operations.
- Initialization of the Picard method: Computation of $\mathbb{E}_m^x\left[y(t_{m+1}, X_{m+1}^{\Delta})\right]$ in $\mathcal{O}(N^2)$ operations.
- Computation of the P Picard approximations for $y(t_m, x)$ in $\mathcal{O}(P_{\text{picard}}N)$ operations.
- Computation of the Fourier coefficients $\mathcal{Z}_j(t_m)$, $\mathcal{Y}_j(t_m)$ and $\mathcal{F}_j(t_m)$ with the DCT in $\mathcal{O}(N\log(N))$ operations.

The functions $z(t_m, x)$, $p(t_m, x)$ and the initial estimate for the Picard method are computed on a x-grid with N equidistant points, and is of order of complexity $\mathcal{O}(N^2)$. It is the most time-consuming computation in the algorithm, although the computational time can be improved by computing $z(t_m, x)$ and $p(t_m, x)$ in parallel. The total complexity of the explicit method is $\mathcal{O}(\mathcal{M}(N + N^2 + P_{\text{picard}}N + N\log(N)))$.[2]

3.2. Local method

As an alternative to the explicit method, one may consider whether it were possible to iterate over each time interval $[t_m, t_{m+1})$ individually for all $m = 0, \ldots, \mathcal{M} - 1$. In this way, one obtains an iterative, *local* method. The method employed in this section has not been seen in the literature, although the method in [10] shows resemblance.

Suppose we work on a local interval $[t_m, t_{m+1})$ for some $m \in \{0, \ldots, \mathcal{M} - 1\}$ and that approximations of the Fourier-cosine coefficients of $y(t_{m+1}, x)$ and $z(t_{m+1}, x)$ are available to us. Again, we start with an initial guess for the functions $y^0(t_m, x)$ and $z^0(t_m, x)$.

In the first local iteration, we use the approximated Fourier-cosine coefficients of $y(t_{m+1}, x)$ and $z(t_{m+1}, x)$ and the conditional characteristic function to obtain a first approximation of $y(t_m, x)$ and $z(t_m, x)$. In the next iteration, we again use the approximated Fourier-cosine coefficients of $y(t_{m+1}, x)$ and $z(t_{m+1}, x)$ and the approximations of $y(t_m, x)$ and $z(t_m, x)$ from the previous iteration to obtain a second approximation.

This procedure is repeated for a predefined number of iterations. When the algorithm is done iterating, we fix $y(t_m, x)$ and $z(t_m, x)$ and compute their Fourier-cosine coefficients. By repeating this procedure for each local time interval, we obtain an iterative scheme which we will call the *local* method.

Let P_{local} denote the number of local iterations per time interval and let $k = 0, \ldots, P_{\text{local}}$ denote the current local iteration. Furthermore, we denote by $X_m^{\Delta,k}$, $y^k(t_m, x)$ and $z^k(t_m, x)$ respectively the value of X_m^{Δ}, $y(t_m, x)$ and $z(t_m, x)$ in iteration k and let $\mathcal{Y}_j^P(t_m)$, $\mathcal{Z}_j^P(t_m)$, $\mathcal{F}_j^P(t_m)$ denote respectively the Fourier-cosine coefficients of $y^P(t_m, x)$, $z^P(t_m, x)$ and $f(t_m, x, y^P(t_m, x), z^P(t_m, x))$, where P represents P_{local}. Finally, the conditional expectations $\mathbb{E}_m^x[\cdot\,]$ in (2.13)–(2.15) have to be replaced by $\mathbb{E}_m^{k,x}[\cdot\,] := \mathbb{E}_m\left[\cdot | X_m^{\Delta,k} = x\right]$.

The conditional characteristic function of $X_{m+1}^{\Delta,k}$ is now given by

$$\begin{aligned}\phi_{X_{m+1}^{\Delta,k}}(u|X_m^{\Delta,k} = x) &= \mathbb{E}\left[\exp\left(iuX_{m+1}^{\Delta,k}\right) \mid X_m^{\Delta,k} = x\right] \\ &= \exp\left(iux + iu\mu(t_m, x, y^k(t_m, x), z^k(t_m, x))\Delta t - \frac{1}{2}u^2\sigma^2(t_m, x, y^k(t_m, x))\Delta t\right). \end{aligned} \tag{3.10}$$

However, we do not have the functions $y^k(t_m, x)$ and $z^k(t_m, x)$ available to us during iteration k. Therefore, we use the approximation

$$\begin{aligned}\phi_{X_{m+1}^{\Delta,k}}(u|X_m^{\Delta,k} = x) &= \mathbb{E}\left[\exp\left(iuX_{m+1}^{\Delta,k}\right) \mid X_m^{\Delta,k} = x\right] \\ &\approx \mathbb{E}\left[\exp\left(iuX_{m+1}^{\Delta,k-1}\right) \mid X_m^{\Delta,k-1} = x\right] \\ &= \exp\left(iux + iu\mu(t_m, x, y^{k-1}(t_m, x), z^{k-1}(t_m, x))\Delta t - \frac{1}{2}u^2\sigma^2(t_m, x, y^{k-1}(t_m, x))\Delta t\right) \\ &= \phi_{X_{m+1}^{\Delta,k-1}}(u|X_m^{\Delta,k-1} = x). \end{aligned} \tag{3.11}$$

In the local method, when we are finished iterating over a time interval $[t_{m+1}, t_{m+2})$, we fix the approximated functions $y(t_{m+1}, x)$ and $z(t_{m+1}, x)$ and as a consequence their Fourier-cosine coefficients $\mathcal{Y}(t_{m+1})$ and $\mathcal{Z}(t_{m+1})$, by setting them equal to $y^P(t_{m+1}, x)$, $z^P(t_{m+1}, x)$, $\mathcal{Y}_j^P(t_{m+1})$ and $\mathcal{Z}_j^P(t_{m+1})$ respectively. During the local iterations on the next interval, $[t_m, t_{m+1})$, these values remain fixed.

[2] The COS method can be generalized to higher dimensions, for example as in [19]. In this case the complexity analysis is slightly more involved and the method becomes more costly, although some of the computations can be done in parallel to obtain an increase in speed.

Therefore, we use the Fourier-cosine coefficients $\mathcal{Y}^P(t_{m+1})$ and $\mathcal{Z}^P(t_{m+1})$ in the COS approximations, i.e.

$$\mathbb{E}_m^{k,x}\left[h^P(t_{m+1}, X_{m+1}^{\Delta,k})\right] \approx \sum_{j=0}^{N-1}{}' \mathcal{H}_j^P(t_{m+1})\Re\left\{\phi_{X_{m+1}^{\Delta,k-1}}\left(\frac{j\pi}{b-a}\middle| X_m^{\Delta,k-1}=x\right)\exp\left(ij\pi\frac{-a}{b-a}\right)\right\}, \tag{3.12}$$

and

$$\begin{aligned}&\mathbb{E}_m^{k,x}\left[h^P(t_{m+1}, X_{m+1}^{\Delta,k})\Delta W_{m+1}\right]\\ &\quad\approx \Delta t\sigma(t_m, x, y^{k-1}(t_m,x))\sum_{j=0}^{N-1}{}' \mathcal{H}_j^P(t_{m+1})\Re\left\{i\frac{j\pi}{b-a}\phi_{X_{m+1}^{\Delta,k-1}}\left(\frac{j\pi}{b-a}\middle| X_m^{\Delta,k-1}=x\right)\exp\left(ij\pi\frac{-a}{b-a}\right)\right\},\end{aligned} \tag{3.13}$$

where $h^P(t_{m+1}, x)$ denotes the fixed approximation of the function $h(t_{m+1}, x)$ and $\mathcal{H}_j^P(t_{m+1})$ its Fourier-cosine coefficients. The final COS approximations of $z^k(t_m, x)$ and $y^k(t_m, x)$ are equal to

$$\begin{aligned}z^k(t_m,x) &\approx \sum_{j=0}^{N-1}{}' -\frac{1-\theta_2}{\theta_2}\mathcal{Z}_j^P(t_{m+1})\Phi_j^{k-1}(x)\\ &\quad+\left(\frac{1}{\Delta t\theta_2}\mathcal{Y}_j^P(t_{m+1})+\frac{1-\theta_2}{\theta_2}\mathcal{F}_j^P(t_{m+1})\right)\sigma(t_m,x,y^{k-1}(t_m,x))\Delta t\bar{\Phi}_j^{k-1}(x).\end{aligned} \tag{3.14}$$

$$\begin{aligned}y^k(t_m,x) &\approx \Delta t\theta_1 f(t_m,x,y^k(t_m,x),z^k(t_m,x)) + \sum_{j=0}^{N-1}{}' \Phi_j^{k-1}(x)\left(\mathcal{Y}_j^P(t_{m+1})+\Delta t(1-\theta_1)\mathcal{F}_j^P(t_{m+1})\right),\\ &= f(t_m,x,y^k(t_m,x),z^k(t_m,x)) + p^k(t_{m+1},x),\end{aligned} \tag{3.15}$$

where $\Phi_j^{k-1}(x)$ and $\bar{\Phi}_j^{k-1}(x)$ (compared to (3.22) and (3.23)) denote

$$\Phi_j^{k-1}(x) = \Re\left\{\phi_{X_{m+1}^{\Delta,k-1}}\left(\frac{j\pi}{b-a}\middle| X_m^{\Delta,k-1}=x\right)\exp\left(ij\pi\frac{-a}{b-a}\right)\right\} \tag{3.16}$$

$$\bar{\Phi}_j^{k-1}(x) = \Re\left\{i\frac{j\pi}{b-a}\phi_{X_{m+1}^{\Delta,k-1}}\left(\frac{j\pi}{b-a}\middle| X_m^{\Delta,k-1}=x\right)\exp\left(ij\pi\frac{-a}{b-a}\right)\right\}. \tag{3.17}$$

In the case $\theta_1 > 0$, we have an implicit equation for $y^k(t_m, x)$ which we will again solve by using P_{picard} Picard iterations with initial guess $\mathbb{E}_m^{k,x}\left[y^P(t_{m+1}, X_{m+1}^{\Delta,k})\right]$.

The local method is summarized in Algorithm 2. At the terminal time, we set $y^P(t_{\mathcal{M}}, x) = g(x)$ and $z^P(t_{\mathcal{M}}, x) = D_x g(x)\sigma(t_{\mathcal{M}}, x, g(x))$ and compute the Fourier-cosine coefficients $\mathcal{Y}_j^P(t_{\mathcal{M}})$, $\mathcal{Z}_j^P(t_{\mathcal{M}})$ and $\mathcal{F}_j^P(t_{\mathcal{M}})$ by using the discrete Fourier-cosine transform as in Eqs. (3.7)–(3.9). For a small enough timestep, we can expect $y(t_{m-1}, x)$ to be reasonably close to $y(t_m, x)$. Hence, we set the initial estimates $y^0(t_{m-1}, x)$ and $z^0(t_{m-1}, x)$ equal to the final approximations for the previous time point t_m.

Algorithm 2: Local method

Compute the terminal functions $y^P(t_{\mathcal{M}}, x)$ and $z^P(t_{\mathcal{M}}, x)$ by using the terminal condition.
Approximate the terminal coefficients $\mathcal{Y}_j^P(t_{\mathcal{M}})$, $\mathcal{Z}_j^P(t_{\mathcal{M}})$ and $\mathcal{F}_j^P(t_{\mathcal{M}})$, by using a discrete Fourier-cosine transform.
for all $x \in [a, b]$ **do**
    $y^0(t_{\mathcal{M}-1}, x) \leftarrow g(x)$
    $z^0(t_{\mathcal{M}-1}, x) \leftarrow D_x g(x)\sigma(t_{\mathcal{M}}, x, g(x))$
end for
for $m = \mathcal{M} - 1$ to 0 **do**
    for $k = 1$ to P_{local} **do**
        Compute the functions $z^k(t_m, x)$, $y^k(t_m, x)$ and $f(t_m, x, y^k(t_m, x), z^k(t_m, x))$ by using
        formulas (3.15) and (3.14) and functions $z^{k-1}(t_m, x)$, $y^{k-1}(t_m, x)$ from the previous
        local iteration.
    end for
    Approximate the corresponding coefficients $\mathcal{Z}_j^P(t_m)$, $\mathcal{Y}_j^P(t_m)$ and $\mathcal{F}_j^P(t_m)$ by using a
    discrete Fourier-cosine transform.
    $y^0(t_{m-1}, x) \leftarrow y^P(t_m, x)$
    $z^0(t_{m-1}, x) \leftarrow z^P(t_m, x)$
end for

The algorithm then computes a first approximation $y^1(t_{\mathcal{M}-1}, x)$ and $z^1(t_{\mathcal{M}-1}, x)$ according to Eqs. (3.14) and (3.15). In the next iteration, we use these approximations in the conditional characteristic function to obtain a more accurate approximation $y^2(t_{\mathcal{M}-1}, x)$ and $z^2(t_{\mathcal{M}-1}, x)$. We do this P_{local} times until we obtain approximations $y^P(t_{\mathcal{M}-1}, x)$ and $z^P(t_{\mathcal{M}-1}, x)$. Finally, we compute the Fourier-cosine coefficients $\mathcal{Y}_j^P(t_{\mathcal{M}-1})$, $\mathcal{Z}_j^P(t_{\mathcal{M}-1})$ and $\mathcal{F}_j^P(t_{\mathcal{M}-1})$.

This procedure is repeated for all time intervals $[t_m, t_{m+1})$, $m = 0, \ldots, \mathcal{M}-1$. Since the conditional characteristic functions at time point t_m and iteration k depend on $y^{k-1}(t_m, x)$ and $z^{k-1}(t_m, x)$, we need to store these values in computer memory, as well as the Fourier-cosine coefficients $\mathcal{Y}_j^P(t_{m+1})$, $\mathcal{Z}_j^P(t_{m+1})$ and $\mathcal{F}_j^P(t_{m+1})$. Therefore, the local method needs to store fewer values than the global method and equally as many values as the explicit method. However, the local method does require more computational effort than the explicit method.

The explicit method performs the same operations as the explicit method in each time step, but these operations are performed repeatedly for every local iteration. Hence, the total complexity of the algorithm is equal to $\mathcal{O}(\mathcal{M}P_{\text{local}}(N + N^2 + P_{\text{picard}}N + N\log(N)))$.

3.3. Global method

Lastly, one can use a technique proposed in [12], where an iterative procedure is used over the complete time domain. In the first iteration, we start with an initial guess for the functions $y(t_m, x)$ and $z(t_m, x)$ for all time points t_m, which is then substituted in the characteristic function (2.17) of X_{m+1}^{Δ}. As a result, the characteristic function at time t_m does not depend on the functions $y(t_m, x)$ and $z(t_m, x)$ anymore, and Eqs. (2.13)–(2.15) in combination with the COS formulas are fully explicit and a next estimate of the functions $y(t_m, x)$ and $z(t_m, x)$ can be computed. By substituting these new estimates in the conditional characteristic function in the next iteration and repeating this procedure, we arrive at an iterative scheme which we will call the *global* method here.

This method also shows resemblance with the *iterated optimal stopping* method developed in [20]. Here, a similar iterative procedure is employed to solve a certain type of nonlinear control PDEs. One drawback of this method, which also holds for the global method discussed in this section, is that the approximated solutions at all time steps and all grid points need to be stored in computer memory.

We will essentially use the notation from Section 3.2, but with a slightly different meaning. Let P_{global} denote the number of iterations in the global scheme and let $k = 0, \ldots, P_{\text{global}}$ denote the current iteration. Furthermore, let $y^k(t_m, x)$ and $z^k(t_m, x)$ respectively denote the approximations of $y(t_m, x)$ and $z(t_m, x)$ in the kth iteration. The Fourier-cosine coefficients of $y(t_m, x)$, $z(t_m, x)$ and $f(t_m, x, y(t_m, x), z(t_m, x))$ in iteration k will be denoted by $\mathcal{Y}_j^k(t_m)$, $\mathcal{Z}_j^k(t_m)$ and $\mathcal{F}_j^k(t_m)$ respectively. For the rest of this subsection we again let $\mathbb{E}_m^{k,x}[\cdot] := \mathbb{E}_m\left[\cdot | X_m^{\Delta,k} = x\right]$, where $X_m^{\Delta,k}$ denotes the value of X_m^{Δ} in the kth global iteration.

The same logic as for Eqs. (3.10) and (3.11) also applies here. The conditional characteristic function of $X_{m+1}^{\Delta,k}$ is given by

$$\begin{aligned}\phi_{X_{m+1}^{\Delta,k}}(u|X_m^{\Delta,k} = x) &= \mathbb{E}\left[\exp\left(iuX_{m+1}^{\Delta,k}\right) \mid X_m^{\Delta,k} = x\right] \\ &= \exp\left(iux + iu\mu(t_m, x, y^k(t_m, x), z^k(t_m, x))\Delta t - \frac{1}{2}u^2\sigma^2(t_m, x, y^k(t_m, x))\Delta t\right). \end{aligned} \tag{3.18}$$

However, since we do not have the functions $y^k(t_m, x)$ and $z^k(t_m, x)$ available to us during iteration k and time point t_m, we use the previous iteration to approximate the conditional characteristic function, i.e.

$$\begin{aligned}\phi_{X_{m+1}^{\Delta,k}}(u|X_m^{\Delta,k} = x) &= \mathbb{E}\left[\exp\left(iuX_{m+1}^{\Delta,k}\right) \mid X_m^{\Delta,k} = x\right] \\ &\approx \mathbb{E}\left[\exp\left(iuX_{m+1}^{\Delta,k-1}\right) \mid X_m^{\Delta,k-1} = x\right] \\ &= \exp\left(iux + iu\mu(t_m, x, y^{k-1}(t_m, x), z^{k-1}(t_m, x))\Delta t - \frac{1}{2}u^2\sigma^2(t_m, x, y^{k-1}(t_m, x))\Delta t\right) \\ &= \phi_{X_{m+1}^{\Delta,k-1}}(u|X_m^{\Delta,k-1} = x). \end{aligned} \tag{3.19}$$

The COS formulas (2.16) and (2.20) change to

$$\mathbb{E}_m^{k,x}\left[h(t_{m+1}, X_{m+1}^{\Delta,k})\right] \approx \sum_{j=0}^{N-1}{}' \mathcal{H}_j^k(t_{m+1})\Re\left\{\phi_{X_{m+1}^{\Delta,k-1}}\left(\frac{j\pi}{b-a}\middle| X_m^{\Delta,k-1} = x\right)\exp\left(ij\pi\frac{-a}{b-a}\right)\right\}, \tag{3.20}$$

and

$$\begin{aligned}&\mathbb{E}_m^{k,x}\left[h(t_{m+1}, X_{m+1}^{\Delta,k})\Delta W_{m+1}\right] \\ &\approx \Delta t\sigma(t_m, x, y^{k-1}(t_m, x))\sum_{j=0}^{N-1}{}' \mathcal{H}_j^k(t_{m+1})\Re\left\{i\frac{j\pi}{b-a}\phi_{X_{m+1}^{\Delta,k-1}}\left(\frac{j\pi}{b-a}\middle| X_m^{\Delta,k-1} = x\right)\exp\left(ij\pi\frac{-a}{b-a}\right)\right\}. \end{aligned} \tag{3.21}$$

The derivations of these formulas are identical to the original derivations. Assume we want to compute $z^k(t_m, x)$ and $y^k(t_m, x)$ for arbitrary m and k and assume, for now, that the Fourier-cosine coefficients at time point t_{m+1} in the current iteration are known.

For the computation of $z^k(t_m, x)$, we need to compute the conditional expectations $\mathbb{E}_m^{k,x}\left[z^k(t_{m+1}, X_{m+1}^{k,\Delta})\right]$, $\mathbb{E}_m^{k,x}\left[y^k(t_{m+1}, X_{m+1}^{k,\Delta})\Delta W_{m+1}\right]$ and $\mathbb{E}_m^{k,x}\left[f(t_{m+1}, X_{m+1}^{\Delta,k}, y^k(t_{m+1}, X_{m+1}^{\Delta,k}), z^k(t_{m+1}, X_{m+1}^{\Delta,k}))\Delta W_{m+1}\right]$. To simplify presentation, we define

$$\Phi_j^{k-1}(x) = \Re\left\{\phi_{X_{m+1}^{\Delta,k-1}}\left(\frac{j\pi}{b-a}\middle| X_m^{\Delta,k-1} = x\right)\exp\left(ij\pi\frac{-a}{b-a}\right)\right\}, \tag{3.22}$$

$$\bar{\Phi}_j^{k-1}(x) = \Re\left\{i\frac{j\pi}{b-a}\phi_{X_{m+1}^{\Delta,k-1}}\left(\frac{j\pi}{b-a}\middle| X_m^{\Delta,k-1} = x\right)\exp\left(ij\pi\frac{-a}{b-a}\right)\right\}. \tag{3.23}$$

Using the COS formulas (3.20) and (3.21), we obtain the following approximations:

$$\mathbb{E}_m^{k,x}\left[z^k(t_{m+1}, X_{m+1}^{k,\Delta})\right] \approx \sum_{j=0}^{N-1}{}' \mathcal{Z}_j^k(t_{m+1})\Phi_j^{k-1}(x), \tag{3.24}$$

$$\mathbb{E}_m^{k,x}\left[y^k(t_{m+1}, X_{m+1}^{k,\Delta})\Delta W_{m+1}\right] \approx \sigma(t_m, x, y^{k-1}(t_m, x))\Delta t\sum_{j=0}^{N-1}{}' \mathcal{Y}_j^k(t_{m+1})\bar{\Phi}_j^{k-1}(x), \tag{3.25}$$

$$\begin{aligned}&\mathbb{E}_m^{k,x}\left[f(t_{m+1}, X_{m+1}^{k,\Delta}, y^k(t_{m+1}, X_{m+1}^{k,\Delta}), z^k(t_{m+1}, X_{m+1}^{k,\Delta}))\Delta W_{m+1}\right]\\ &\quad\approx \sigma(t_m, x, y^{k-1}(t_m, x))\Delta t\sum_{j=0}^{N-1}{}' \mathcal{F}_j^k(t_{m+1})\bar{\Phi}_j^{k-1}(x).\end{aligned} \tag{3.26}$$

The final approximation $z^k(t_m, x)$ is then given by substituting the above three equations in (2.15):

$$\begin{aligned}z^k(t_m, x) \approx &\sum_{j=0}^{N-1}{}' -\frac{1-\theta_2}{\theta_2}\mathcal{Z}_j^k(t_{m+1})\Phi_j^{k-1}(x)\\ &+\left(\frac{1}{\Delta t\theta_2}\mathcal{Y}_j^k(t_{m+1}) + \frac{1-\theta_2}{\theta_2}\mathcal{F}_j^k(t_{m+1})\right)\sigma(t_m, x, y^{k-1}(t_m, x))\Delta t\bar{\Phi}_j^{k-1}(x).\end{aligned} \tag{3.27}$$

For the computation of $y^k(t_m, x)$, we need to compute the conditional expectations $\mathbb{E}_m^{k,x}\left[y^k(t_{m+1}, X_{m+1}^{\Delta,k})\right]$ and $\mathbb{E}_m^{k,x}\left[f(t_{m+1}, X_{m+1}^{\Delta,k}, y^k(t_{m+1}, X_{m+1}^{\Delta,k}), z^k(t_{m+1}, X_{m+1}^{\Delta,k}))\right]$. Using the COS formula (3.19), we obtain:

$$\mathbb{E}_m^{k,x}\left[y^k(t_{m+1}, X_{m+1}^{\Delta,k})\right] \approx \sum_{j=0}^{N-1}{}' \mathcal{Y}_j^k(t_{m+1})\Phi_j^{k-1}(x), \tag{3.28}$$

$$\mathbb{E}_m^{k,x}\left[f(t_{m+1}, X_{m+1}^{\Delta,k}, y^k(t_{m+1}, X_{m+1}^{\Delta,k}), z^k(t_{m+1}, X_{m+1}^{\Delta,k}))\right] \approx \sum_{j=0}^{N-1}{}' \mathcal{F}_j^k(t_{m+1})\Phi_j^{k-1}(x). \tag{3.29}$$

The final approximation $y^k(t_m, x)$ is then given by substituting the above two equations in (2.14):

$$\begin{aligned}y^k(t_m, x) &\approx \Delta t\theta_1 f(t_m, x, y^k(t_m, x), z^k(t_m, x)) + \sum_{j=0}^{N-1}{}' \Phi_j^{k-1}(x)\left(\mathcal{Y}_j^k(t_{m+1}) + \Delta t(1-\theta_1)\mathcal{F}_j^k(t_{m+1})\right),\\ &= \Delta t\theta_1 f(t_m, x, y^k(t_m, x), z^k(t_m, x)) + p^k(t_{m+1}, x).\end{aligned} \tag{3.30}$$

In the case $\theta_1 > 0$, we obtain an implicit equation for $y^k(t_m, x)$. We use P_{picard} Picard iterations to determine $y^k(t_m, x)$ with initial guess $\mathbb{E}_m^{k,x}\left[y(t_{m+1}, X_{m+1}^{\Delta,k})\right]$.

Algorithm 3: Global method

for $m = 0$ to $\mathcal{M} - 1$ and all $x \in [a, b]$ **do**
    $y^0(t_m, x) \leftarrow 0$
    $z^0(t_m, x) \leftarrow 0$
end for
for $k = 1$ to P_{global} **do**
    Compute the terminal functions $y^k(t_{\mathcal{M}}, x)$ and $z^k(t_{\mathcal{M}}, x)$ by using the terminal condition.
    Approximate the terminal coefficients $\mathcal{Y}_j^k(t_{\mathcal{M}})$, $\mathcal{Z}_j^k(t_{\mathcal{M}})$ and $\mathcal{F}_j^k(t_{\mathcal{M}})$, by using a discrete
    Fourier-cosine transform.
    for $m = \mathcal{M} - 1$ to 0 **do**
        Compute the functions $z^k(t_m, x)$, $y^k(t_m, x)$ and $f(t_m, x, y^k(t_m, x), z^k(t_m, x))$ by using
        formulas (3.30) and (3.27) and functions $z^{k-1}(t_m, x)$, $y^{k-1}(t_m, x)$ from
        the previous global iteration.
        Subsequently, approximate the corresponding coefficients $\mathcal{Z}_j^k(t_m)$, $\mathcal{Y}_j^k(t_m)$ and $\mathcal{F}_j^k(t_m)$
        by using a discrete Fourier-cosine transform.
    end for
end for

Algorithm 3 summarizes the global method. Initially, we set[3] $y^0(t_m, x) = 0$ and $z^0(t_m, x) = 0$ for all time points t_m, $m = 0, \ldots, \mathcal{M} - 1$ and all $x \in [a, b]$. At the terminal time $t = t_{\mathcal{M}}$ we set $y^k(t_{\mathcal{M}}, x) = g(x)$ and $z^k(t_{\mathcal{M}}, x) = D_x g(x)\sigma(t_{\mathcal{M}}, x, g(x))$ and compute $\mathcal{Z}_j^k(t_{\mathcal{M}})$, $\mathcal{Y}_j^k(t_{\mathcal{M}})$ and $\mathcal{F}_j^k(t_{\mathcal{M}})$ for all $k = 1, \ldots, P_{\text{global}}$ by using the discrete Fourier-cosine transform

$$\mathcal{Y}_j^k(t_{\mathcal{M}}) = \frac{2}{b-a}\int_a^b g(x)\cos\left(j\pi\frac{x-a}{b-a}\right)dx, \tag{3.31}$$

$$\mathcal{Z}_j^k(t_{\mathcal{M}}) = \frac{2}{b-a}\int_a^b D_x g(x)\sigma(t_{\mathcal{M}}, x, g(x))\cos\left(j\pi\frac{x-a}{b-a}\right)dx, \tag{3.32}$$

$$\mathcal{F}_j^k(t_{\mathcal{M}}) = \frac{2}{b-a}\int_a^b f(t_{\mathcal{M}}, x, g(x), D_x g(x)\sigma(t_{\mathcal{M}}, x, g(x)))\cos\left(j\pi\frac{x-a}{b-a}\right)dx. \tag{3.33}$$

We then compute the functions $z^1(t_{\mathcal{M}-1}, x)$ and $y^1(t_{\mathcal{M}-1}, x)$ on the equidistant x-grid $[a, b]$ according to formulas (3.27) and (3.30). Subsequently, the Fourier-cosine coefficients $\mathcal{Z}_j^1(t_{\mathcal{M}-1})$, $\mathcal{Y}_j^1(t_{\mathcal{M}-1})$ and $\mathcal{F}_j^1(t_{\mathcal{M}-1})$ are computed. This procedure is repeated until we reach the starting time t_0. As we have reached t_0, we have a first approximation $y^1(t_m, x)$ and $z^1(t_m, x)$ for all times t_m and the first iteration of the algorithm is completed.

In the next iteration, new, more accurate, approximations $z^2(t_m, x)$ and $y^2(t_m, x)$ can be computed by using the functions $z^1(t_m, x)$ and $y^1(t_m, x)$ from the previous iterations in Eqs. (3.27) and (3.30) for all $m = 0, \ldots, \mathcal{M} - 1$. The algorithm is iterated until we have completed P_{global} iterations.

Here, we note that the global method also shows resemblance to the *waveform relaxation* methods discussed in [21]. These methods are characterized by an iteration over the time domain, followed by an iteration over the space domain (the global iteration). Convergence of waveform relaxation methods is fast for a small time duration T, a feature that the global method discussed here also exhibits, as we will see in Section 4.2.

From Algorithm 3, the difference with the local method is immediately clear. Instead of iterating over the individual time intervals, represented by the inner loop in Algorithm 2, the global method iterates over the complete time domain, represented by the global loop in Algorithm 3. The algorithm is computationally more demanding than the explicit and local methods. During global iteration k, we need all values $y^{k-1}(t_m, x)$ and $z^{k-1}(t_m, x)$ for all values $m = 0, \ldots, \mathcal{M} - 1$. Therefore, the global method has to store all values in computer memory.

The global method performs the same operations in each time step as the explicit method. However, we now also need to compute an initial guess for the solution on an x-grid with N equidistant points for each time-point t_m. This computation is of order of complexity $\mathcal{O}(\mathcal{M}N)$. Furthermore, the computations for each separate time-point are performed for every global iteration. Hence, the total complexity of the global method is equal to $\mathcal{O}(\mathcal{M}P_{\text{global}}(N + N^2 + P_{\text{picard}}N + N\log(N)))$.

Establishing the convergence of these iterations is hard, but the authors of [12] provide assumptions under which the iterations converge to the true solution as the time step goes to zero and the number of iterations grows. Most notably, they assume that the drift μ and the diffusion σ are not dependent on Z and that one of the following holds for the FBSDE under consideration:

[3] This initial guess is somewhat arbitrary. Another possibility could be to use one of the other numerical methods discussed in this section with a small number of time steps M, and interpolate this result to obtain an initial guess for the global method.

- Small time duration.
- Weak coupling of Y into the forward SDE or weak coupling of X into the backward SDE.
- Either the driver function $f(t, x, y, z)$ is strongly decreasing in y or the drift function $\mu(t, x, y, z)$ is strongly decreasing in X.

In this case, the theoretical framework presented in Section 2 ensures us that the global iterations converge to the true solution as the time step goes to zero and the number of iterations grows. However, when we are not in one of the above cases, we will see in Section 4.2 that the iterations do not necessarily converge. Furthermore, the speed of convergence of the iterations depends on the strength of the coupling of the FBSDE. Stronger coupling implies slower convergence of the iterations.

4. Numerical experiments

In this section we will apply the numerical methods developed in Sections 3.3–3.1 to several coupled FBSDEs and analyse their convergence behaviour. MATLAB 8.2.0.701 was used for the computations, with an Intel(R) Core(TM) i7-3537U CPU @ 2.00 GHz and 4.0 GB of RAM.

We first test the methods on two different BSDEs, taken from the literature. These problems are used to test the capabilities of our numerical methods. The error in the approximations of $y(0, x_0)$ and $z(0, x_0)$ are studied for each algorithm as the time step gets smaller. The second problem poses more difficulties, since the diffusion also depends on the process Z_t. Richardson extrapolation is applied to this example as well, to accelerate the speed of convergence in the number of timesteps $\mathcal{M}$. Finally, we discuss a FBSDE arising from a financial problem.

For each method, we test the following schemes.

Scheme A:	$\theta_1 = 1$ and $\theta_2 = 1$	Scheme C:	$\theta_1 = \frac{1}{2}$ and $\theta_2 = \frac{1}{2}$
Scheme B:	$\theta_1 = 0$ and $\theta_2 = 1$	Scheme D:	$\theta_1 = 0$ and $\theta_2 = \frac{1}{2}$

For the local method and the global method, we use a stopping criterion to determine when we should stop iterating. For the local method, we keep iterating until

$$\max\left(|y^k(t_m, x_0) - y^{k-1}(t_m, x_0)|, |z^k(t_m, x_0) - z^{k-1}(t_m, x_0)|\right) \leq \epsilon_0, \tag{4.1}$$

and for the global method, we keep iterating until

$$\max_{1\leq m\leq \mathcal{M}}\left(|y^k(t_m, x_0) - y^{k-1}(t_m, x_0)|, |z^k(t_m, x_0) - z^{k-1}(t_m, x_0)|\right) \leq \epsilon_1, \tag{4.2}$$

where k denotes the local or global iteration, and ϵ_0 and ϵ_1 are set to 10^{-3}. In the case $\theta_1 > 0$, we also have to perform a number of Picard iterations for each time point t_m. Here we use the same type of stopping criterion as in (4.1).

As a consequence of the Fourier cosine methods employed in this paper, the local error near the boundaries of the integration interval $[a, b]$ can disrupt our algorithm. A method to deal with these local errors is discussed in [22]. Alternatively, the authors of [23] use a different grid discretization in their Fourier approach that may be applicable to our approach as well. In all of our examples, we take a relatively wide integration range, as to make sure these local errors do not influence the approximations in the areas of our interest.

4.1. Example 1: example from Milstein and Tretyakov [24]

The first problem is taken from [24]. We consider the following coupled FBSDE:

$$X_t = X_0 + \int_0^t \frac{X_s\left(1 + X_s^2\right)}{\left(2 + X_s^2\right)^3} ds + \int_0^t \frac{1 + X_s^2}{2 + X_s^2} \sqrt{\frac{1 + 2Y_s^2}{1 + Y_s^2 + \exp\left(-\frac{2X_s^2}{s+1}\right)}} dW_s, \tag{4.3}$$

$$Y_t = \exp\left(-\frac{X_T^2}{T+1}\right) - \int_t^T a(s, X_s, Y_s) + b(s, X_s, Y_s) Z_s ds + \int_t^T Z_s dW_s, \tag{4.4}$$

where

$$a(t, x, u) = \frac{1}{t+1} \exp\left(-\frac{x^2}{t+1}\right)\left[\frac{4x^2(1+x^2)}{(2+x^2)^3} + \left(\frac{1+x^2}{2+x^2}\right)^2\left(1 - \frac{2x^2}{t+1}\right) - \frac{x^2}{t+1}\right], \tag{4.5}$$

$$b(t, x, u) = \frac{x}{(2+x^2)^2}\sqrt{\frac{1 + u^2 + \exp\left(-\frac{2x^2}{t+1}\right)}{1 + 2u^2}}. \tag{4.6}$$

Table 4.1
Error in the approximations of $y(0, x_0)$ and $z(0, x_0)$ for the explicit method.

$\mathcal{M}$	Error	$N = 8$	$N = 16$	$N = 32$	$N = 64$
$\mathcal{M} = 16$	Y_0	1.167E−01	1.124E−02	1.142E−02	1.142E−02
	Z_0	9.194E−02	1.954E−02	2.181E−02	2.181E−02
$\mathcal{M} = 32$	Y_0	1.169E−01	5.630E−03	5.862E−03	5.862E−03
	Z_0	8.850E−02	8.186E−03	1.062E−02	1.061E−02
$\mathcal{M} = 64$	Y_0	1.170E−01	2.701E−03	2.968E−03	2.968E−03
	Z_0	8.708E−02	2.682E−03	5.198E−03	5.196E−03
$\mathcal{M} = 128$	Y_0	1.167E−01	1.206E−03	1.493E−03	1.493E−03
	Z_0	8.645E−02	8.661E−07	2.566E−03	2.565E−03
$\mathcal{M} = 256$	Y_0	1.170E−01	4.506E−04	7.489E−04	7.490E−04
	Z_0	8.616E−02	1.321E−03	1.274E−03	1.273E−03
$\mathcal{M} = 512$	Y_0	1.170E−01	7.100E−05	3.750E−04	3.751E−04
	Z_0	8.603E−02	1.976E−03	6.345E−04	6.342E−04

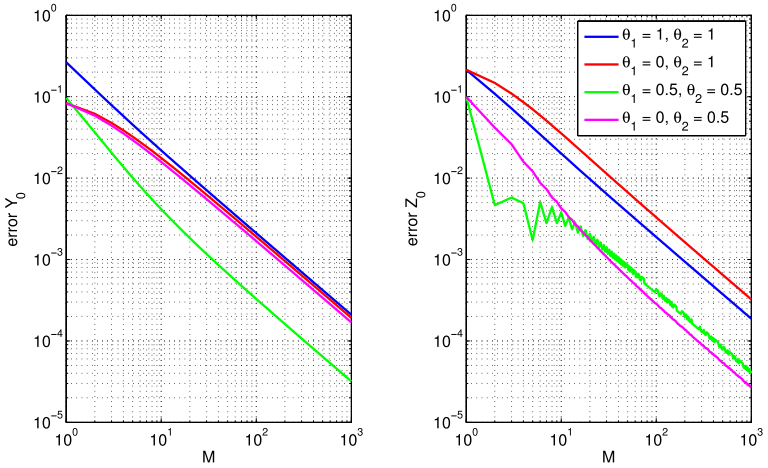

Fig. 4.1. Plot of the error of the explicit method for example 1.

Note that this FBSDE satisfies the assumptions in Section 2. It is known that the solution of (4.3)–(4.4) is given by

$$y(t, x) = \exp\left(-\frac{x^2}{t+1}\right), \quad z(t, x) = -\frac{2x(1+x^2)}{(t+1)(2+x^2)} \exp\left(-\frac{x^2}{t+1}\right). \tag{4.7}$$

For the experiment, we use $T = 1$ and $X_0 = 1$ and set $a = -5$ and $b = 5$, with a and b the interval boundaries of the integration range $[a, b]$. First, we will analyse the error in the approximations of $y(0, x_0)$ and $z(0, x_0)$ for different values of N. Table 4.1 shows the error for these approximations.

Here, the approximations are obtained by using the explicit method and scheme B. Accurate results are quickly obtained for small values of N. Furthermore, we observe that the error in the approximations is dominated by the number of time points $\mathcal{M}$, as the error stops decreasing at some point for increasing values of N. Therefore, we are more concerned with the behaviour of the error as the number of time points varies. For all following numerical tests, we will take $N = 2^9$ to ensure sufficient accuracy.

For the explicit method, the error in the approximation $y(0, x_0)$ and $z(0, x_0)$ can be found in Fig. 4.1. The results for the local method and the global method do not differ significantly from the explicit method, so these will not be shown here. All methods (and all schemes) achieve first-order convergence. Although scheme C, where $\theta_1 = \theta_2 = \frac{1}{2}$, should achieve second-order convergence for the approximation of the integrals in (2.6), the error in the Euler approximation of the forward SDE is too large for the global error to converge quadratically. Furthermore, we observe some oscillating behaviour in the case $\theta_2 = \frac{1}{2}$ for the component Z_t, in accordance with [18].

4.2. Example 2: diffusion depending on Z_t

The next example is taken from [10], and is a problem where the diffusion coefficient in the forward SDE depends on Z_t. Such equations form a class of problems where an efficient numerical method is not yet available. The theory for this class of FBSDEs is considerably more complicated than the theoretical framework from Section 2. Existence and uniqueness results

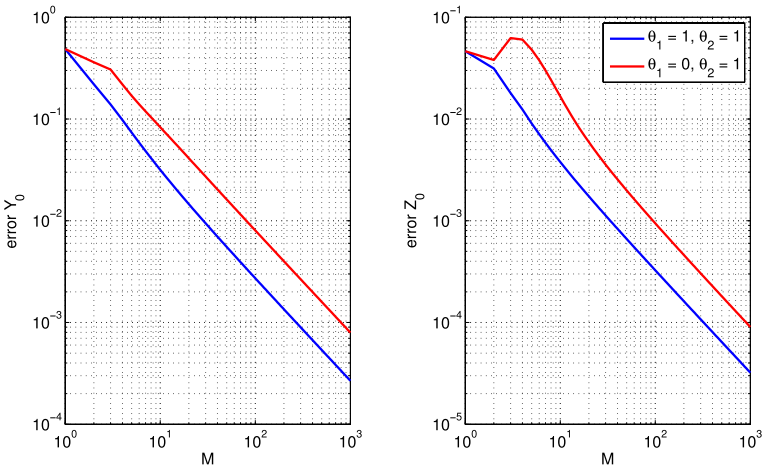

Fig. 4.2. Plot of the error of the local method for example 2.

for this type of problem are proven in [25,26]. The FBSDE used for the numerical test is given by:

$$X_t = X_0 - \int_0^t \frac{1}{2}\sin(s + X_s)\cos(s + X_s)(Y_s^2 + Z_s)ds + \int_0^t \frac{1}{2}\cos(s + X_s)(Y_s \sin(s + X_s) + Z_s + 1)dW_s, \tag{4.8}$$

$$Y_t = \sin(T + X_T) + \int_t^T Y_s Z_s - \cos(s + X_s)ds - \int_t^T Z_s dW_s. \tag{4.9}$$

The exact solutions to Eqs. (4.8)–(4.9) are

$$y(t, x) = \sin(t + x), \qquad z(t, x) = \cos^2(t + x). \tag{4.10}$$

For this experiment, we set $T = 0.1$, $X_0 = 1.5$, $a = -2\pi$ and $b = 2\pi$. As the diffusion coefficient now depends on the process Z_t, Theorem 2.1 does not apply and we do not know the terminal coefficients $\mathcal{Z}_j(t_{\mathcal{M}})$ and $\mathcal{F}_j(t_{\mathcal{M}})$. Therefore, we take $\theta_2 = 1$ in the first time step for all schemes. In this case, the numerical approximation does not depend on $\mathcal{Z}_j(t_{\mathcal{M}})$ and $\mathcal{F}_j(t_{\mathcal{M}})$.

We observe that the global iterations do not converge to the exact solution for any choice of θ_1 and θ_2, regardless of how small the time step is taken.[4] This is not completely surprising, as the authors in [12] assume that the drift and diffusion are not dependent on the process Z. From the footnote, we also see that the assumption of a small time duration might be violated as well.

The results for the local and explicit methods can be found in Figs. 4.2 and 4.3.

We observe that the explicit method converges, for all schemes, to the exact solution and we observe oscillatory behaviour in the case $\theta_2 = \frac{1}{2}$. The local method only converges in the case $\theta_2 = 1$. In the case $\theta_2 = \frac{1}{2}$, the local iterations for the Z-component do not seem to converge as the number of local iterations increases. Therefore, the results for the local method are only displayed for $\theta_2 = 1$.

Table 4.2 shows the CPU times for the local and explicit methods. Here, we observe that the explicit method is significantly faster than the local method. Notice that at each time step the characteristic function is recomputed, and that the computing time grows linearly in the number of time steps M.

Remark. A *generalized* θ-scheme based on four parameters $\tilde{\theta}_1, \ldots, \tilde{\theta}_4$ is proposed in [27], and second-order convergence is proven for $\tilde{\theta}_1 = \tilde{\theta}_2 = \tilde{\theta}_3 = \frac{1}{2}$ and $|\tilde{\theta}_4| < \tilde{\theta}_3$. Schemes C and D are special cases of this generalized scheme where $|\tilde{\theta}_4| = \tilde{\theta}_3$. The authors note that $\tilde{\theta}_4$ controls the stability of the method, and this can explain the observed non-convergence of the local iterations here. An implementation of this generalized scheme where $\tilde{\theta}_1 = \tilde{\theta}_2 = \tilde{\theta}_3 = \frac{1}{2}$ and $\tilde{\theta}_4 = 0$ for the local method does show convergence of the local iterations. Nevertheless, the method is slower than the explicit method and we therefore do not pursue it any further here.

[4] For schemes A and B, convergence of the global iterations is obtained for a smaller value of the time duration, $T = 0.01$, and for an initial guess that is closer to the exact solution, such as $y^0(t, x) = \sin(x)$ and $z^0(t, x) = \cos(x)$. Furthermore, the convergence of the iterations in this case is faster for smaller T, emphasizing the resemblance with the waveform relaxation method.

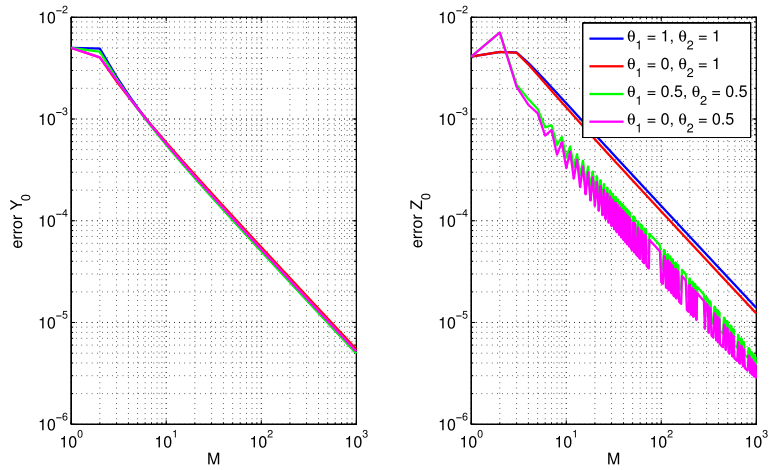

Fig. 4.3. Plot of the error of the explicit method for example 2.

Table 4.2
CPU times of the local and explicit methods (s).

$(N = 2^9)$ M	4	8	16	32	64	128	256
Local method, scheme A	0.5484	1.1212	1.9752	3.8043	7.6687	15.7155	31.3762
Local method, scheme B	0.5897	1.1065	2.0876	4.0386	8.1636	15.7807	31.4967
Explicit method, scheme A	0.1125	0.2036	0.3913	0.7646	1.4978	2.9488	6.2186
Explicit method, scheme B	0.1158	0.2408	0.4511	0.9013	1.6829	3.1934	6.4432

4.3. Example 3: Richardson extrapolation

From the previous examples, we see that in the case $\theta_1 = 1$ the convergence behaviour of our numerical methods is smooth and monotone (cf. Fig. 4.2, for example). In such cases, it is known that *Richardson extrapolation* can be used to obtain a numerical method that has higher order of convergence. In our case, the application of Richardson extrapolation can provide a second-order method for coupled FBDEs.[5]

Suppose we wish to approximate the solution $y(t, x)$ of an arbitrary coupled FBSDE at the point (t_0, x_0). Given all other parameters, our numerical method only depends on the step size $\Delta t = \frac{T}{M}$. Let $\hat{y}(\Delta t)$ denote the approximation of $y(t_0, x_0)$, assuming the error expansion of the method is of the form

$$\hat{y}(\Delta t) = y(t_0, x_0) + K\Delta t + \mathcal{O}(\Delta t^2), \tag{4.11}$$

we can obtain the two equations

$$\hat{y}(\Delta t) = y(t_0, x_0) + K\Delta t + \mathcal{O}(\Delta t^2), \tag{4.12}$$

$$\hat{y}\left(\frac{\Delta t}{2}\right) = y(t_0, x_0) + K\frac{\Delta t}{2} + \mathcal{O}(\Delta t^2). \tag{4.13}$$

By rearranging these equations, we can derive that

$$2\hat{y}\left(\frac{\Delta t}{2}\right) - \hat{y}(\Delta t) = y(t_0, x_0) + \mathcal{O}(\Delta t^2), \tag{4.14}$$

is a second-order approximation. We apply Richardson extrapolation for the explicit and local methods on the problem from Section 4.2

$$X_t = X_0 - \int_0^t \frac{1}{2}\sin(s + X_s)\cos(s + X_s)(Y_s^2 + Z_s)ds \int_0^t \frac{1}{2}\cos(s + X_s)(Y_s \sin(s + X_s) + Z_s + 1)dW_s, \tag{4.8}$$

$$Y_t = \sin(T + X_T) + \int_t^T Y_s Z_s - \cos(s + X_s)ds - \int_t^T Z_s dW_s. \tag{4.9}$$

[5] Another approach is to use a weak second-order discretization of the forward SDE. In this case, one can achieve second-order convergence for $\theta_1 = \theta_2 = \frac{1}{2}$, but the resulting algorithm is computationally expensive, as we have to approximate the first and second spatial and the time derivatives of $y(t, x)$ and $z(t, x)$ as well.

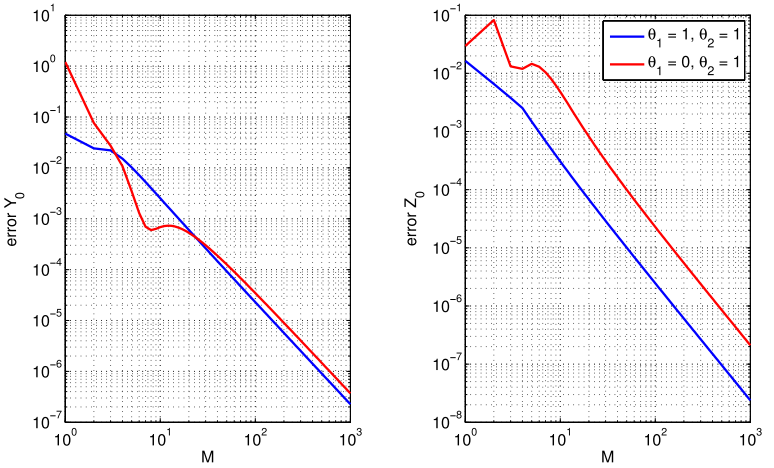

Fig. 4.4. Plot of the error of the local method in combination with Richardson extrapolation.

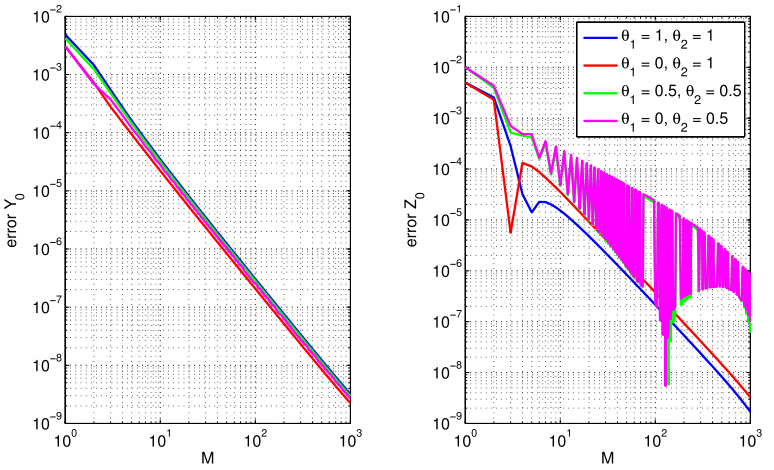

Fig. 4.5. Plot of the error of the explicit method in combination with Richardson extrapolation.

We use the same parameters as in Section 4.2. Since the explicit method converges for $\theta_2 = \frac{1}{2}$, we also test schemes C and D on this problem. The results are displayed in Figs. 4.4 and 4.5.

From Fig. 4.4, we see that we indeed achieve second-order convergence, even for the computationally cheaper schemes where $\theta_1 = 1$. The increase in order of convergence comes at a low price, since we only need to compute the approximation at two different step sizes, and these computations can even be performed in parallel.

For the explicit method, Fig. 4.5 shows that we achieve second-order convergence[6] in the y-component for all schemes. We achieve second-order convergence in the z-component for the schemes where $\theta_2 = 1$. This was to be expected from Fig. 4.3, as we see that the schemes with $\theta_2 = \frac{1}{2}$ show oscillatory behaviour. Therefore, the assumption that the error for these schemes is of the form (4.11) may be invalid and Richardson extrapolation will not work.

Concluding, we can achieve the same accuracy as in Figs. 4.2 and 4.3 in only $\frac{1}{6}$th of the CPU times in Table 4.2, so that CPU times are really competitive with state-of-the-art nonlinear PDE solvers.

4.4. Example 4: hedging with a correlated asset

We discuss a model for hedging an option with a correlated asset, as in [5]. Let $V(t, S_t)$ denote the value of an option on an underlying asset S_t, with dynamics given by

$$S_t = S_0 + \int_0^t \bar{\mu} S_u du + \int_0^t \bar{\sigma} S_u dW_u^1, \tag{4.15}$$

[6] Second-order convergence is also obtained for the global method, when using Richardson extrapolation. However, the global method is only capable of solving the first problem, so we omit the results here.

Table 4.3
Parameter values for problem 5.

r	ρ	$\bar{\sigma}$	$\bar{\mu}$	$\bar{\sigma}'$	λ	K	T	X_0	M	a	b
0.05	0.9	0.2	0.07	0.3	0.2	100	1	100	200	0	400

where $\bar{\mu}$ is the drift rate, $\bar{\sigma}$ is the volatility and W_t^1 is a Brownian motion. Suppose that we cannot trade in this asset, but we can trade in an asset H_t, that is correlated with S_t, with dynamics given by

$$H_t = H_0 + \int_0^t \bar{\mu}' H_u du + \int_0^t \bar{\sigma}' H_u dW_u^2, \tag{4.16}$$

where W_t^2 is another Brownian motion correlated with W_t^1 with correlation parameter ρ. By setting up a trading portfolio Π_t consisting of shorting the contract V and holding a risk-free bond and π_t assets H_t, where π_t is chosen to minimize the (real-world) variance of the portfolio at time t, one can arrive at the following dynamics of the portfolio

$$\begin{aligned}\Pi_t = \Pi_0 &- \int_0^t D_t V(u, S_u) + r' S_u D_x V(u, S_u) - rV(u, S_u) + \frac{\bar{\sigma}^2 S_u^2}{2} D_{xx} V(u, S_u) du \\ &+ \int_0^t S_u D_x V(u, S_u) \bar{\sigma} \rho dW_u^1 - \int_0^t S_u D_x V(u, S_u) dW_u^2,\end{aligned} \tag{4.17}$$

where r denotes the risk-free interest rate and $r' = \bar{\mu} - (\bar{\mu}' - r)\frac{\bar{\sigma}\rho}{\bar{\sigma}'}$. Eq. (4.17) contains the drift rates $\bar{\mu}$ and $\bar{\mu}'$. In the complete market setting these drift rates vanish from the pricing PDE, but in this case we are required to estimate them. Since these are difficult to estimate accurately in practice, we assume we can only estimate a range of values $[r'_{\min}, r'_{\max}]$ for r'.

It can then be shown [5], assuming the unhedgeable residual risk is not diversifiable, that the worst case price for a *short* position in the option must satisfy the nonlinear PDE

$$D_t V(t, x) + \left[r^* + \lambda^* \operatorname{sgn}(D_x V(t, x))\right] x D_x V(t, x) + \frac{\bar{\sigma}^2 x^2}{2} D_{xx} V(t, x) - rV(t, x) = 0, \tag{4.18}$$

where $r^* = \frac{r'_{\max} + r'_{\min}}{2}$ and $\lambda^* = \frac{r'_{\max} - r'_{\min}}{2}$. We will price a *straddle* option in this model, within the FBSDE framework.

By Theorem 2.1, the solution $V(t, x)$ to this PDE is equal to the solution Y_t of the following coupled FBSDE

$$X_t = X_0 + \int_0^t r^* + \lambda^* \operatorname{sgn}\left(\frac{Z_s}{\bar{\sigma} X_s}\right) X_s ds + \int_0^t \bar{\sigma} X_s ds, \tag{4.19}$$

$$Y_t = g(X_T) + \int_t^T r Y_s ds - \int_t^T Z_s dW_s, \tag{4.20}$$

where $g(x)$ is the payoff of a straddle option

$$g(x) = \max(K - x, 0) + \max(x - K, 0), \tag{4.21}$$

and K is the strike of the straddle. Analogously, one can derive that the worst case price for a *long* position in the option must satisfy the PDE

$$D_t V(t, x) + \left[r^* - \lambda^* \operatorname{sgn}(D_x V(t, x))\right] x D_x V(t, x) + \frac{\bar{\sigma}^2 x^2}{2} D_x x V(t, x) - rV(t, x) = 0. \tag{4.22}$$

In this case, the solution $V(t, x)$ satisfies the same FBSDE, but with the plus sign changed to a minus sign in the drift of the forward SDE.

As the explicit method is the fastest of the three methods discussed in this paper, we test it on this problem. We set the following parameters, taken from [5] (see Table 4.3).

The parameters $\bar{\mu}'$, r^* and λ^* can be computed from the relations

$$\bar{\mu}' = r + (\bar{\mu} - r)\frac{\bar{\sigma}'\rho}{\bar{\sigma}}, \tag{4.23}$$

$$r^* = \bar{\mu} - (\bar{\mu}' - r)\frac{\bar{\sigma}\rho}{\bar{\sigma}'}, \tag{4.24}$$

$$\lambda^* = \lambda \bar{\sigma} \sqrt{1 - \rho^2}. \tag{4.25}$$

The derivations for these relationships can be found in [5]. The solution $V(t, x)$ at the initial time $t = 0$ is displayed in Fig. 4.6.

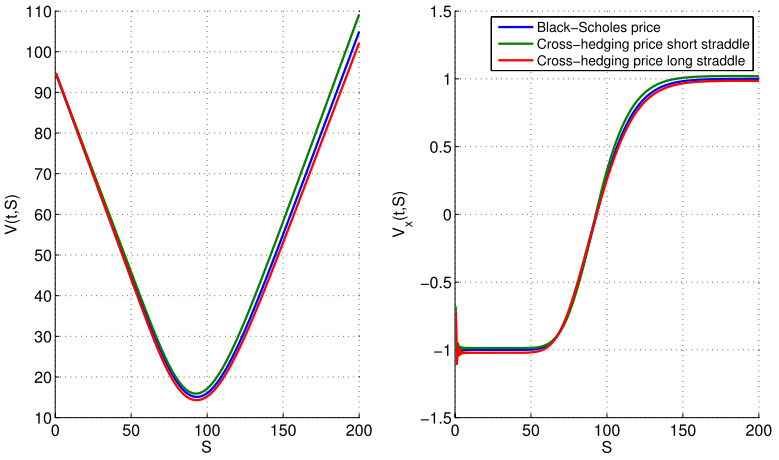

Fig. 4.6. Plot of the cross-hedging price and the Black–Scholes price at time $t = 0$.

From Fig. 4.6, we see that a short position in the contract costs more than in the standard Black–Scholes setting. Since the correlation is not perfect, we have to buy more shares of H_t for the replicating portfolio to hedge the movements of S_t in the case of a short position. Analogously, we have to buy fewer shares of H_t in the case of a long position, hence the replicating portfolio is cheaper in that case.

In [5] it is reported that the option value at the initial time is equal to 17.13 for a short position, and equal to 15.19 for a long position in the option. For $M = 5000$, we obtain the following values, after applying Richardson extrapolation:

Short position:	17.13137897
Long position:	15.18871568

Establishing the fact that our method converges to the true solution in only a few seconds.

5. Conclusion

In this paper we discussed three different numerical methods, based on a probabilistic view and the BCOS method, for solving coupled forward–backward stochastic differential equations. A theta-method was used for approximating the integrands in the time discretization. While it is known theoretically that the approximation with $\theta_1 = \theta_2 = \frac{1}{2}$ converges quadratically, the global error of all three methods converges only linearly due to the approximation error of the forward SDE by an Euler SDE discretization.

We have discussed three methods for dealing with the coupling between the forward and backward processes, with significant differences among them, the global, the local and the explicit methods. The explicit method appears to perform best, as it is fastest and it is accurate, compared against the global and local methods. The global method only converges for relatively simple problems, while the local method is computationally somewhat more expensive than the explicit model, and can be unstable for $\theta_2 = \frac{1}{2}$.

Richardson extrapolation was used to accelerate the speed of convergence, resulting in second-order convergence. As expected, we do not observe second-order convergence for the schemes where $\theta_2 = \frac{1}{2}$, due to oscillatory behaviour in the Z_t-component. Resulting is a highly efficient numerical technique based on Fourier cosines series for the approximation of quasilinear PDEs that are cast in the form of coupled FBSDEs.

References

[1] N. El Karoui, E. Pardoux, M.C. Quenez, Reflected backward SDEs and American options, in: L. Robers, D. Talay (Eds.), Numerical Methods in Finance, Cambridge University Press, 1997, pp. 215–231.
[2] B. Bouchard, R. Elie, Discrete-time approximation of decoupled forward–backward SDE with jumps, Stochastic Process. Appl. 118 (1) (2008) 53–75.
[3] J. Cvitanić, J. Ma, Hedging options for a large investor and forward–backward SDEs, Ann. Appl. Probab. 6 (2) (1996) 370–398.
[4] E. Pardoux, S. Tang, Forward–backward stochastic differential equations and quasilinear parabolic PDEs, Probab. Theory Related Fields 114 (2) (1999) 123–150.
[5] H. Windcliff, J. Wang, P.A. Forsyth, K.R. Vetzal, Hedging with a correlated asset: Solution of a nonlinear pricing PDE, J. Comput. Appl. Math. 200 (1) (2007) 86–115.
[6] B. Bouchard, N. Touzi, Discrete-time approximation and monte-carlo simulation of backward stochastic differential equations, Stochastic Process. Appl. 111 (2) (2004) 175–206.
[7] J. Zhang, A numerical scheme for BSDEs, Ann. Appl. Probab. 14 (1) (2004) 459–488.
[8] M.J. Ruijter, C.W. Oosterlee, A Fourier cosine method for an efficient computation of solutions to BSDEs, SIAM J. Sci. Comput. 37 (2) (2015) A859–A889.
[9] C.B. Hyndman, P. Oyono Ngou, A convolution method for numerical solution of backward stochastic differential equations, Methodol. Comput. Appl. Probab. (2015).

[10] W. Zhao, Y. Fu, T. Zhou, A new kind of high-order multi-step schemes for forward backward stochastic differential equations, 2013. ArXiv Preprint arXiv:1310.5307.
[11] F. Delarue, S. Menozzi, A forward–backward stochastic algorithm for quasi-linear PDEs, Ann. Appl. Probab. 16 (1) (2006) 140–184.
[12] C. Bender, J. Zhang, Time discretization and Markovian iteration for coupled FBSDEs, Ann. Appl. Probab. 18 (1) (2008) 143–177.
[13] W. Zhao, J. Wang, S. Peng, Error estimates of the theta-scheme for backward stochastic differential equations, Discrete Contin. Dyn. Syst. Ser. B 12 (4) (2009) 905–924.
[14] F. Fang, C.W. Oosterlee, A novel pricing method for European options based on Fourier-cosine series expansions, SIAM J. Sci. Comput. 31 (2) (2008) 826–848.
[15] J. Ma, J. Shen, Y. Zhao, On numerical approximations of forward–backward stochastic differential equations, SIAM J. Numer. Anal. 46 (5) (2008) 2636–2661.
[16] J. Ma, P. Protter, J. Yong, Solving forward–backward stochastic differential equations explicitly—a four step scheme, Probab. Theory Related Fields 98 (3) (1994) 339–359.
[17] P.E. Kloeden, E. Platen, Numerical Solution of Stochastic Differential Equations, Springer, 1992.
[18] M.J. Ruijter, C.W. Oosterlee, Numerical Fourier method and second-order Taylor scheme for backward SDEs in finance, 2014. Available at SSRN 2501686.
[19] M.J. Ruijter, C.W. Oosterlee, Two-dimensional Fourier cosine series expansion method for pricing financial options, SIAM J. Sci. Comput. 34 (5) (2012) B642–B671.
[20] E. Bayraktar, H. Xing, Pricing American options for jump diffusions by iterating optimal stopping problems for diffusions, Math. Methods Oper. Res. 70 (3) (2009) 505–525.
[21] Z. Bartoszewski, M. Kwapisz, On the convergence of waveform relaxation methods for differential-functional systems of equations, J. Math. Anal. Appl. 235 (2) (1999) 478–496.
[22] M.J. Ruijter, C.W. Oosterlee, R.F.T. Aalbers, On the Fourier cosine series expansion method for stochastic control problems, Numer. Linear Algebra Appl. 20 (4) (2013) 598–625.
[23] C.B. Hyndman, P. Oyono Ngou, Global convergence and stability of a convolution method for numerical solution of BSDEs, 2014. ArXiv Preprint arXiv:1410.8595.
[24] G.N. Milstein, M.V. Tretyakov, Numerical algorithms for forward–backward stochastic differential equations, SIAM J. Sci. Comput. 28 (2) (2006) 561–582.
[25] S. Peng, Z. Wu, Fully coupled forward–backward stochastic differential equations and applications to optimal control, SIAM J. Control Optim. 37 (3) (1999) 825–843.
[26] Y. Hu, S. Peng, Solution of forward–backward stochastic differential equations, Probab. Theory Related Fields 103 (2) (1995) 273–283.
[27] W. Zhao, Y. Li, G. Zhang, A generalized θ-scheme for solving backward stochastic differential equations, Discrete Contin. Dyn. Syst. Ser. B 17 (5) (2012) 1585–1603.