CNMAI Congresso Nacional de Matemática Aplicada à Indústria 2014
18 a 21 de novembro de 2014, Caldas Novas - Goiás

A higher order and stable method for the numerical integration of Random Differential Equations

H. de la Cruz, hugo.delacruz@fgv.br [1]
J.C. Jimenez, jcarlos@icimaf.cu [2]

[1]Fundacão Getúlio Vargas - Escola de Matemática Aplicada. Praia de Botafogo 190. Botafogo. Rio de Janeiro

[2]Instituto de Cibernética, Matemática y Física - ICIMAF. Calle 15 No. 551, Vedado, C. Habana

Abstract. *Over the last few years there has been a growing and renovated interest in the numerical study of Random Differential Equations (RDEs). On one hand it is motivated by the fact that RDEs have played an important role in the modeling of physical, biological, neurological and engineering phenomena, and on the other hand motivated by the usefulness of RDEs for the numerical analysis of Ito-stochastic differential equations (SDEs) -via the extant conjugacy property between RDEs and SDEs-, which allows to study stronger pathwise properties of SDEs driven by different kind of noises others than the Brownian. Since in most common cases no explicit solution of the equations is known, the construction of computational methods for the treatment and simulation of RDEs has become an important need. In this direction the Local Linearization (LL) approach is a successful technique that has been applied for defining numerical integrators for RDEs. However, a major drawback of the obtained methods is its relative low order of convergence; in fact it is only twice the order of the moduli of continuity of the driven stochastic process. The present work overcomes this limitation by introducing a new, exponential-based, high order and stable numerical integrator for RDEs. For this, a suitable approximation of the stochastic processes present in the random equation, together with the local linearization technique and an adapted Padé method with scaling and squaring strategy are conveniently combined. In this way a higher order of convergence can be achieved (independent of the moduli of continuity of the stochastic processes) while retaining the dynamical and numerical stability properties of the low order LL methods. Results on the convergence and stability of the suggested method and details on its efficient implementation are discussed. The performance of the introduced method is illustrated through computer simulations.*

Palavras-chave: *random differential equations, numerical integrators, exponential methods, Local linearization methods*

1. INTRODUCTION

In several physical, chemical, industrial and biological phenomena, noise plays a significant role. This is the case, for example, in turbulent diffusion, epidemiology, genetic regulation, chemical kinetic, biological waste treatment, polymer dynamics, large scale integrated (VLSI) circuit design, finance, neurosciences, to mention just a few. When the evolution of such noisy phenomena has to be studied stochastic effects need to be taken into account, thus the mathematical modeling of such situations is not well matched by deterministic differential equations. There are, of course, many ways to introduce randomness into a mathematical models. In particular, during the last decades, in order to construct more realistic models, Random Differential Equations (RDEs) of the form

$$\mathbf{x}^{'}(t) = \mathbf{f}(t, \mathbf{x}(t), \xi(t)),\ t \in [t_0, T],$$

(which are pathwise Ordinary Differential Equations (ODEs) containing a multidimensional stochastic process ξ in their vector field function), have been used in a wide range of applications, see e.g. [1], [12], [11], [10], [14], [15], [16]. Since, unfortunately, closed-form solutions of these equations are rarely available, the construction of computational methods for the treatment and simulation of RDEs has become an important need.

At a first look one could think that some of the existing numerical schemes for ODEs can be used pathwise for RDEs, but typically the driving stochastic process $\xi(t)$ in the random equation has at most Hölder continuous sample paths, so the

sample paths of the solutions are certainly continuously differentiable, but their derivatives are at most Hölder continuous in time. The resulting vector field $\mathbf{f}(t, \mathbf{x}(t), \xi(t))$ is, thus, at most Hölder continuous in time, no matter how smooth the vector field is in its original variables. Consequently, since the usual estimates of the discretization error of such schemes require sufficient smoothness of the vector field function, numerical schemes for ODEs when applied to RDEs are not convergent or rarely attain their traditional order.

On the other hand, similar to the deterministic scenario, there exists a variety of important issues in designing practical numerical integrators for RDEs. In particular high accuracy, computational efficiency, and stability of the numerical schemes, are very desirable properties. Taking all this into consideration, some numerical integrator have been proposed in literature e.g., [2], [6], [8], [3]. However, these methods or are of implicit nature (involving the numerical solution of a system of nonlinear algebraic equations at each integration step, that typically increase the computational effort of these numerical integrators) or are explicit integrators, having the appealing feature of retaining the standard order of convergence of the classical deterministic schemes, but at the expense of high computational cost and low stability.

The aim of this paper is overcome these limitations by introducing a new, exponential-based, higher order, stable and explicit integrator for RDEs. For this, a suitable approximation of the stochastic processes present in the random equation, together with the local linearization technique and an adapted Padé method with scaling and squaring strategy are conveniently combined. In this way a higher order of convergence can be achieved (independent of the moduli of continuity of the stochastic processes) while retaining the dynamical and numerical stability properties of the low order LL methods proposed in [3] and with a suitable computational effort.

The paper is organized in 4 sections as follows. After this introduction, section 2 presents the deduction of the proposed method, also a Padé algorithm is conveniently adapted and details on the effective implementation of the method is given. In section 3 the proposed method is evaluated by mean of simulations and finally in section 4 some concluding remarks are presented.

2. A HIGHER ORDER METHOD FOR RDES

Let $(\Omega, \mathcal{F}, P)$ be a complete probability space, and $(\mathcal{F}_t)_{t\geq 0}$ be an increasing right continuous family of complete sub σ-algebras of $\mathcal{F}$. Consider the d-dimensional random integral equation (see [7])

$$\begin{aligned} &\mathbf{x}^{'}(t) = \mathbf{f}(t, \mathbf{x}(t), \xi(t)),\ t \in [t_0, T], \\ &\mathbf{x}(t_0) = \mathbf{x}_0, \end{aligned} \tag{1}$$

where ξ_t is a $\mathcal{F}_t$-adapted finite continuous processes.

2.1. Formulation of the method

Let $(t)_h = \{t_n : n = 0, 1, \ldots, N\}$ be a partition of the time interval $[t_0, T]$, with equidistant stepsize $h < 1$, i.e., defined as a sequence of times $t_0 < t_1 < \ldots < t_N = T$ such that $t_n = t_0 + nh$, for $n = 0, 1, \ldots, N$.

Starting from the initial value $\mathbf{x}_0$, the approximations $\{\mathbf{x}_i\}$ to $\{\mathbf{x}(t_i)\}$, $(i = 1, 2, \ldots, N)$ are obtained recursively as follows.

For each time interval $\mathbf{\Lambda}_n = [t_n, t_{n+1}]$ we consider the random local problem

$$\mathbf{x}(t) = \mathbf{x}_n + \int_{t_n}^{t} \mathbf{f}(s, \mathbf{x}(s), \xi(s))ds, \tag{2}$$

Then, the idea is to get an approximation of $\mathbf{x}(t_{n+1})$, through the solution of the auxiliary random equation resulting from approximating $\mathbf{f}$ and the stochastic increment. For this, let´s consider $\bar{h} = h^{\gamma}$ (we need to take $\bar{h}$ in this way in order to guaranty the order of convergence of the method we are constructing here) with $\gamma \geq 2$ and such that $h^{1-\gamma} \in \mathbb{N}$ and let $(t_n)_{\bar{h}} = \{t_n^i : t_n^i = t_n + i\bar{h},\ i = 0, 1, \ldots, \left[h^{1-\gamma}\right] + 1\}$ a partition of $\mathbf{\Lambda}_n$.

For $t \in \mathbf{\Lambda}_n$, define $n_t = \max\{k : t_n^k \leq t < t_n^{k+1}\}$. Then by a linear interpolation to $\xi(t)$ in $[t_n^{n_t}, t_n^{n_t+1}]$

$$\xi(t) \approx \bar{\xi}(t) = \xi(t_n^{n_t}) + \frac{\Delta\xi(t_n^{n_t})}{\bar{h}}(t - t_n^{n_t}), \tag{3}$$

It follows, by using the approximations above and a first order Taylor expansion of $\mathbf{f}$ at $(t_n, \mathbf{x}_n, \xi_n)$, that the solution of (2) in $\mathbf{\Lambda}_n$ can be approximated by the solution of the (pathwise) differential equation

$$\begin{aligned} \mathbf{x}(t) = \mathbf{x}_n + \int_{t_n}^{t} &(\mathbf{f}(t_n, \mathbf{x}_n, \xi_n) + \mathbf{f}_{\mathbf{x}}(t_n, \mathbf{x}_n, \xi_n)(\mathbf{x}(\mathbf{s}) - \mathbf{x}_n) + \mathbf{f}_t(t_n, \mathbf{x}_n, \xi_n)(s - t_n) \\ + \sum_{k=0}^{\left[h^{1-\gamma}\right]} \int_{t_n}^{t} &(\mathbf{f}'_{\xi}(t_n, \mathbf{x}_n, \xi_n)\left((\xi_{t_n^k} - \xi_n) + \frac{\Delta\xi(t_n^k)}{\bar{h}}(s - t_n^k)\right)\mathbf{1}_{[t_n^k\ t_n^{k+1}]}(s))ds \end{aligned} \tag{4}$$

Let's denote

$$\mathbf{A}_n = \mathbf{f}_{\mathbf{x}}(t_n, \mathbf{x}_n, \xi_n)$$
$$\mathbf{b}_n^k = \mathbf{f}_t(t_n, \mathbf{x}_n, \xi_n) + \mathbf{f}'_{\xi}(t_n, \mathbf{x}_n, \xi_n)\frac{\Delta\xi(t_n^k)}{\bar{h}}$$
$$\mathbf{c}_n^k = \mathbf{f}(t_n, \mathbf{x}_n, \xi_n) - \mathbf{A_n x_n} - \mathbf{f}_t(t_n, \mathbf{x}_n, \xi_n)t_n + \mathbf{f}'_{\xi}(t_n, \mathbf{x}_n, \xi_n)((\xi_{t_n^k} - \xi_n) - \frac{\Delta\xi(t_n^k)}{\bar{h}})t_n^k$$

Then, the solution of (4) at $t = t_{n+1}$ is obtained as follows. Starting at $\mathbf{x}(t_n^0) = \mathbf{x}(t_n) = \mathbf{x}_n$, $\mathbf{x}(t_n^{k+1})$ is computed recursively from $\mathbf{x}(t_n^k)$ by solving in $[t_n^k, t_{n+1}^{k+1}]$ the linear differential equation

$$\mathbf{x}^{'}(t) = \mathbf{A}_n\mathbf{x}(t) + \mathbf{b}_n^k t + \mathbf{c}_n^k \tag{5}$$

with initial condition $\mathbf{x}(t_n^k)$ in $t = t_n^k$.

That is,

$$\begin{aligned}\mathbf{x}(t_n^{k+1}) &= e^{\mathbf{A}_n(t-t_n^k)}\mathbf{x}(t_n^k) + \int_{t_n^k}^{t_n^{k+1}} e^{\mathbf{A}_n(t-s)}(\mathbf{b}_n^k s + \mathbf{c}_n^k)ds \\ &= \mathbf{x}(t_n^k) + \int_0^{\bar{h}} e^{\mathbf{A}_n(\bar{h}-s)}(\mathbf{b}_n^k s + \mathbf{d}_n^k)ds \\ &:= \varphi(\mathbf{x}(t_n^k))\end{aligned} \tag{6}$$

where

$$\mathbf{d}_n^k = \mathbf{c}_n^k + \mathbf{A}_n\mathbf{x}(t_n^k) + \mathbf{b}_n^k t_n^k$$

Thus, as we want to approximate $\mathbf{x}(t_{n+1})$, we conclude that

$$\begin{aligned}\mathbf{x}_{n+1} &= \varphi^{(\left[h^{1-\gamma}\right]+1)}(\mathbf{x}_n) \\ &= \underbrace{\varphi \circ \varphi \circ ... \circ \varphi}(\mathbf{x}_n) \\ &\quad (\left[h^{1-\gamma}\right]+1) \text{ times}\end{aligned}$$

Summarizing, the numerical integrator for (1) is given by the recursive equation

$$\mathbf{x}_{n+1} = \varphi^{(\left[h^{1-\gamma}\right]+1)}(\mathbf{x}_n) \tag{7}$$

for $n = 0, 1, \ldots, N-1$ with $\mathbf{x}(t_0) = \mathbf{x}_0$ and $\gamma \geq 2$.

An important problem in the evaluation of (7) is the efficient and stable computation of φ. A naive way to do this is through the explicit computation of the integral defining φ. However, this procedure might eventually fail since it is not computationally feasible in case of singular or nearly singular matrices $\mathbf{A}_n$ (see e.g. comments in [4]). In the next section we will propose an efficient algorithm for computing $\mathbf{x}_{n+1}$. Concerning the convergence and velocity of convergence of the proposed method we have the following theorem.

Theorem: Let's suppose that the moduli of continuity of ξ satisfies that $\varpi_{\xi}\left(\bar{h}\right) = O(\bar{h}^{\beta})$ and let $\gamma\beta \geq 2$. Then the numerical integrator (7) is almost surely globally convergent and we have that with probability one $\sup_n \|\mathbf{x}(t_n) - \mathbf{x}_n\| = O(h^2)$. (i.e., the method retaining the standard order of convergence of the classical deterministic schemes, namely 2)

2.2. Implementation details

In this section an efficient computational algorithm to implement the method is provided. At first we will show that $\mathbf{x}(t_n^{k+1})$ in (6) can be represented in terms of a single appropriated exponential of a matrix. Then we will discuss in detail an efficient and accurate alternative to evaluate this matrix exponential defining $\mathbf{x}(t_n^{k+1})$.

The equation (5) can be rewritten

$$\begin{aligned}\mathbf{x}^{'}(t) &= \mathbf{A}_n(\mathbf{x}(t) - \mathbf{x}_n^k) + \mathbf{b}_n^k(t - t_n^k) + \mathbf{d}_n^k \\ \mathbf{x}(t_n^k) &= \mathbf{x}_n^k\end{aligned}$$

where $\mathbf{x}_n^k$ is the approximation to $\mathbf{x}(t_n^k)$.

Let $\mathbf{Z}(t) = (\mathbf{x}(t) - \mathbf{x}_n^k, t - t_n^k, \mathbf{d}_n^k, 1)^{\top} \in \mathbb{R}^{2d+2}$ then

$$\begin{aligned}\mathbf{Z}^{'}(t) &= \mathbf{M}\,\mathbf{Z}(t) \\ \mathbf{Z}(t_n^k) &= \left[\mathbf{0}_{1\times d+1}\ \ \left(\mathbf{d}_n^k\right)^{\top}\ 1\right]^{\top},\end{aligned}$$

where

$$\mathbf{M} = \begin{pmatrix} \mathbf{A}_n & \mathbf{b}_n^k & \mathbf{I}_{d\times d} & \mathbf{0}_{d\times 1} \\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 1 \\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1} \\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 0 \end{pmatrix}.$$

The solution of this equation is

$$\mathbf{Z}(t) = e^{\mathbf{M}(t-t_n^k)}\, \mathbf{Z}(t_n^k).$$

Therefore, looking at the first component of $\mathbf{Z}(t)$, it follows that the solution $\mathbf{x}(t)$ of (5) in $[t_n^k\ t_n^{k+1}]$ can be computed by

$$\mathbf{x}(t) = \mathbf{x}_n^k + \left[\mathbf{I}_{d\times d}\ \ \mathbf{0}_{d\times(d+2)}\right] e^{\mathbf{M}(t-t_n^k)} \left[\mathbf{0}_{1\times d+1}\ \ \left(\mathbf{d}_n^k\right)^\top\ \ 1\right]^\top.$$

Thus, in particular

$$\mathbf{x}(t_n^{k+1}) = \mathbf{x}_n^k + \left[\mathbf{I}_{d\times d}\ \ \mathbf{0}_{d\times(d+2)}\right] e^{\mathbf{M}\bar{h}} \left[\mathbf{0}_{1\times d+1}\ \ \left(\mathbf{d}_n^k\right)^\top\ \ 1\right]^\top. \tag{8}$$

Hence, the numerical implementation of $\mathbf{x}(t_n^{k+1})$ is reduced to the use of a algorithm to compute exponential of matrices. In particular, those algorithms based on the rational (p,q)-Padé approximation ($p \le q \le p+2$) combined with the "scaling and squaring" strategy provide stable approximations to the matrix exponential. Nowadays, professional mathematical software, such as MATLAB, provide efficient codes for implementing a number of such algorithms. However, note that the matrix $\mathbf{M}$ changes in each interval $[t_n^k\ t_n^{k+1}]$, so we would need to compute a lot of exponentials (one for each interval). This turns out that a straightforward implementation of the Padé method would be prohibitively expensive.

In the rest of this section we will derive an algorithm that alleviates significantly the computational burden. Our key idea is to exploit the special structure of the matrix $\mathbf{M}$ and to adapt conveniently the "scaling and squaring" strategy (see [5], [13]) in such a way that the computational saving achieved are very significant.

Before this, we first summarize the existing Padé algorithm with "scaling and squaring" strategy on which we will work.

2.2.1. The Padé algorithm for computing the matrix exponential

The (p,q) rational Padé approximation to $e^{\mathbf{C}}$ is defined by

$$\mathbf{P}_{p,q}(\mathbf{C}) = [\mathbf{D}_{p,q}(\mathbf{C})]^{-1}\mathbf{N}_{p,q}(\mathbf{C}),$$

where

$$\mathbf{N}_{p,q}(\mathbf{C}) = \sum_{j=0}^{p} \frac{(p+q-j)!p!}{(p+q)!j!(p-j)!}\mathbf{C}^j$$

and

$$\mathbf{D}_{p,q}(\mathbf{C}) = \sum_{j=0}^{q} \frac{(p+q-j)!q!}{(p+q)!j!(q-j)!}(-\mathbf{C})^j.$$

Diagonal approximation (that is, $p=q$) are preferred, since $\mathbf{P}_{p,q}$ with $p>q$ ($p<q$) is less accurate than $\mathbf{P}_{p,p}$ ($\mathbf{P}_{q,q}$), and $\mathbf{P}_{p,p}$ ($\mathbf{P}_{q,q}$) can be evaluated at the same cost. From now on, we denote $\mathbf{D}_{q,q}(\mathbf{C})$, $\mathbf{N}_{q,q}(\mathbf{C})$, $\mathbf{P}_{q,q}(\mathbf{C})$ by $\mathbf{D}_q(\mathbf{C})$, $\mathbf{N}_q(\mathbf{C})$, $\mathbf{P}_q(\mathbf{C})$ respectively.

$e^{\mathbf{C}}$ can be well approximated by Padé only near the origin, that is, for small $\|\mathbf{C}\|$. For this reason $e^{\mathbf{C}}$ is approximated by $\left(\mathbf{P}_q(\frac{\mathbf{C}}{m})\right)^m$ where m is the minimum integer such that $\left\|\frac{\mathbf{C}}{m}\right\| < \frac{1}{2}$. In order to reduce the number of matrix multiplications, the idea is to choose m to be a power of two. Then $\left(\mathbf{P}_q(\frac{\mathbf{C}}{m})\right)^m$ can be efficiently computed by repeated squaring.

The Padé algorithm with scalling-squaring strategy for computing $e^{\mathbf{C}}$ can be described as follows.

1. Determine the minimum integer k such that $\left\|\frac{\mathbf{C}}{2^k}\right\| < \frac{1}{2}$
2. Compute $\mathbf{N}_q(\frac{\mathbf{C}}{2^k})$ and $\mathbf{P}_q(\frac{\mathbf{C}}{2^k})$
3. Compute $\mathbf{P}_q(\frac{\mathbf{C}}{2^k}) = [\mathbf{D}_q(\frac{\mathbf{C}}{2^k})]^{-1}\mathbf{N}_q(\frac{\mathbf{C}}{2^k})$, by solving the system $\mathbf{D}_q(\frac{\mathbf{C}}{2^k})\mathbf{P}_q(\frac{\mathbf{C}}{2^k}) = \mathbf{N}_q(\frac{\mathbf{C}}{2^k})$ (using, for instance, a suitable Gaussian elimination)
4. Compute $[\mathbf{P}_q(\frac{\mathbf{C}}{2^k})]^{2^k}$ by squaring $\mathbf{P}_q(\frac{\mathbf{C}}{2^k})$ k times

2.2.2. The Adapted Padé algorithm for computing $e^{\mathbf{M}\bar{h}}$

Now we apply the (q,q) Padé method above to compute the exponential of the matrix we are interested, namely $\mathbf{C}=\mathbf{M}\bar{h}$. Let us denote, for simplicity, $\mathbf{A}=\mathbf{A}_n$ and $\mathbf{b}=\mathbf{b}_n^k$, then

$$\mathbf{C}=\begin{pmatrix}\mathbf{A} & \mathbf{b} & \mathbf{I}_{d\times d} & \mathbf{0}_{d\times 1}\\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 1\\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1}\\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 0\end{pmatrix}\bar{h}.$$

Let k the minimum integer such that $\left\|\frac{\mathbf{C}}{2^k}\right\|<\frac{1}{2}$ and the coefficients $c_j=\frac{(2q-j)!q!}{(2q)!j!(q-j)!}$ $(j=0,\ldots,q)$, then

$$\left(\frac{\mathbf{C}}{2^k}\right)=\begin{pmatrix}\mathbf{A} & \mathbf{b} & \mathbf{I}_{d\times d} & \mathbf{0}_{d\times 1}\\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 1\\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1}\\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 0\end{pmatrix}\frac{\bar{h}}{2^k},\quad \left(\frac{\mathbf{C}}{2^k}\right)^2=\begin{pmatrix}\mathbf{A}^2 & \mathbf{Ab} & \mathbf{A} & \mathbf{b}\\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 0\\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1}\\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 0\end{pmatrix}\left(\frac{\bar{h}}{2^k}\right)^2,\ldots,$$

and in general, by an induction argument, we have that for any $j\geq 2$, $j\in\mathbb{N}$

$$\left(\frac{\mathbf{C}}{2^k}\right)^j=\begin{pmatrix}\mathbf{A}^j & \mathbf{A}^{j-1}\mathbf{b} & \mathbf{A}^{j-1} & \mathbf{A}^{j-2}\mathbf{b}\\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 0\\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1}\\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 0\end{pmatrix}\left(\frac{\bar{h}}{2^k}\right)^j.$$

Hence,

$$\begin{aligned}\mathbf{N}_q(\frac{\mathbf{C}}{2^k})&=\sum_{j=0}^{q}c_j\left(\frac{\mathbf{C}}{2^k}\right)^j\\ &=\begin{pmatrix}\mathbf{I}+\mathbf{A}\left(\alpha_1\mathbf{I}+\mathbf{AS}\right) & \left(\alpha_1\mathbf{I}+\mathbf{AS}\right)\mathbf{b} & \left(\alpha_1\mathbf{I}+\mathbf{AS}\right) & \mathbf{Sb}\\ \mathbf{0}_{1\times d} & 1 & \mathbf{0}_{1\times d} & \alpha_1\\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1} & \mathbf{I}_{d\times d} & \mathbf{0}_{d\times 1}\\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 1\end{pmatrix},\end{aligned}$$

where,

$$\mathbf{S}=\alpha_2\mathbf{I}+\alpha_3\mathbf{A}+\ldots+\alpha_q\mathbf{A}^{q-2},$$

where $\alpha_j=c_j\left(\frac{\bar{h}}{2^k}\right)^j$.

Similarly,

$$\begin{aligned}\mathbf{D}_q(\frac{\mathbf{C}}{2^k})&=\sum_{j=0}^{q}\alpha_j\left(-\frac{\mathbf{C}}{2^k}\right)^j\\ &=\begin{pmatrix}\mathbf{I}+\mathbf{A}\left(-\alpha_1\mathbf{I}+\mathbf{A}\bar{\mathbf{S}}\right) & \left(-\alpha_1\mathbf{I}+\mathbf{A}\bar{\mathbf{S}}\right)\mathbf{b} & \left(-\alpha_1\mathbf{I}+\mathbf{A}\bar{\mathbf{S}}\right) & \bar{\mathbf{S}}\mathbf{b}\\ \mathbf{0}_{1\times d} & 1 & \mathbf{0}_{1\times d} & -\frac{\alpha_1}{2^k}\\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1} & \mathbf{I}_{d\times d} & \mathbf{0}_{d\times 1}\\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 1\end{pmatrix},\end{aligned}$$

where,

$$\bar{\mathbf{S}}=\alpha_2\mathbf{I}-\alpha_3\mathbf{A}+\ldots+(-1)^{q-2}\alpha_q\mathbf{A}^{q-2}.$$

Now, we will compute $\mathbf{P}_q(\frac{\mathbf{C}}{2^k})=[\mathbf{D}_q(\frac{\mathbf{C}}{2^k})]^{-1}\mathbf{N}_q(\frac{\mathbf{C}}{2^k})$ by solving the matrix linear system $\mathbf{D}_q(\frac{\mathbf{C}}{2^k})\mathbf{P}_q(\frac{\mathbf{C}}{2^k})=\mathbf{N}_q(\frac{\mathbf{C}}{2^k})$.

It is not hard to see that $\mathbf{P}_q(\frac{\mathbf{C}}{2^k})$ has the form

$$\mathbf{P}_q(\frac{\mathbf{C}}{2^k})=\begin{pmatrix}\mathbf{U1} & \mathbf{U}2 & \mathbf{U3} & (\mathbf{U4})\mathbf{b}\\ \mathbf{0}_{1\times d} & 1 & \mathbf{0}_{1\times d} & 2\alpha_1\\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1} & \mathbf{I}_{d\times d} & \mathbf{0}_{d\times 1}\\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 1\end{pmatrix}$$

where the matrices $\mathbf{U1},\mathbf{U3},\mathbf{U4}$ and the vector $\mathbf{U2}$, satisfy the systems of linear equations

$$\begin{aligned}\left[\mathbf{I}+\mathbf{A}\left(-\alpha_1\mathbf{I}+\mathbf{A}\bar{\mathbf{S}}\right)\right]\mathbf{U}1&=\left[\mathbf{I}+\mathbf{A}\left(\alpha_1\mathbf{I}+\mathbf{AS}\right)\right]\\ \left[\mathbf{I}+\mathbf{A}\left(-\alpha_1\mathbf{I}+\mathbf{A}\bar{\mathbf{S}}\right)\right]\mathbf{U}2&=\left[\left(\alpha_1\mathbf{I}+\mathbf{AS}\right)-\left(-\alpha_1\mathbf{I}+\mathbf{A}\bar{\mathbf{S}}\right)\right]\mathbf{b}\\ \left[\mathbf{I}+\mathbf{A}\left(-\alpha_1\mathbf{I}+\mathbf{A}\bar{\mathbf{S}}\right)\right]\mathbf{U}3&=\left[\left(\alpha_1\mathbf{I}+\mathbf{AS}\right)-\left(-\alpha_1\mathbf{I}+\mathbf{A}\bar{\mathbf{S}}\right)\right]\\ \left[\mathbf{I}+\mathbf{A}\left(-\alpha_1\mathbf{I}+\mathbf{A}\bar{\mathbf{S}}\right)\right]\mathbf{U}4&=\left[\mathbf{S}-\bar{\mathbf{S}}-2\alpha_1\left(-\alpha_1\mathbf{I}+\mathbf{A}\bar{\mathbf{S}}\right)\right],\end{aligned}$$

Since the fundamental matrix of each system above is the same one, we can exploit this, and that $\mathbf{U}2 = (\mathbf{U}3)\,\mathbf{b}$, to yield significant improvements in the computational cost when solving these set of simultaneous equations. That is, we can do the LU descomposition to $\left[\mathbf{I} + \mathbf{A}\left(-\alpha_1\mathbf{I} + \mathbf{A}\bar{\mathbf{S}}\right)\right]$ and then use the standard procedure for obtaining the definitive solution (see for instance [5]).

Once the $\mathbf{U}1$, $\mathbf{U}2$, $\mathbf{U}3$, $\mathbf{U}4$ are obtained, it remains to compute $\left(\mathbf{P}_q(\frac{\mathbf{C}}{2^k})\right)^m$, where $m = 2^k$, to conclude the scalling-squaring Padé method. By an induction argument, it is not hard to prove that

$$\left(\mathbf{P}_q(\frac{\mathbf{C}}{2^k})\right)^m = \begin{pmatrix} \mathbf{U1} & \mathbf{U2} & \mathbf{U3} & (\mathbf{U4})\mathbf{b} \\ \mathbf{0}_{1\times d} & 1 & \mathbf{0}_{1\times d} & 2\alpha_1 \\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1} & \mathbf{I}_{d\times d} & \mathbf{0}_{d\times 1} \\ \mathbf{0}_{1\times d} & 0 & \mathbf{0}_{1\times d} & 1 \end{pmatrix}^m$$

$$= \begin{pmatrix} (\mathbf{U1})^m & \left(\sum_{i=0}^{m-1}(\mathbf{U1})^i\right)\mathbf{U2} & \left(\sum_{i=0}^{m-1}(\mathbf{U1})^i\right)\mathbf{U3} & \left(\sum_{i=0}^{m-1}(\mathbf{U1})^i\right)(\mathbf{U4})\mathbf{b} + \alpha_1\left(\sum_{i=0}^{m-2}(m-1-i)\,(\mathbf{U1})^i\right)\mathbf{U2} \\ 0_{1\times d} & 1 & 0_{1\times d} & m\alpha_1 \\ 0_{d\times d} & 0_{d\times 1} & I_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{1\times d} & 1 \end{pmatrix}.$$

Then, by substituting the above Padé approximation to $e^{\mathbf{M}\bar{h}}$ in (8), the implementation of the approximation $\mathbf{x}_n^{k+1}$ to $\mathbf{x}(t_n^{k+1})$ is

$$\begin{aligned} \mathbf{x}_n^{k+1} &= \mathbf{x}_n^k + \left[\mathbf{I}_{d\times d}\ \mathbf{0}_{d\times(d+2)}\right]\left(\mathbf{P}_q(\frac{\mathbf{C}}{2^k})\right)^m\left[\mathbf{0}_{1\times d+1}\ \left(\mathbf{d}_n^k\right)^\top\ 1\right]^\top \\ &= \mathbf{x}_n^k + \mathbf{L}\mathbf{d}_\mathbf{n}^\mathbf{k} + \mathbf{Q}\mathbf{b}, \end{aligned} \tag{9}$$

where

$$\mathbf{L} = \left(\sum_{i=0}^{m-1}(\mathbf{U1})^i\right)\mathbf{U3},$$

and

$$\mathbf{Q} = \left(\sum_{i=0}^{m-1}(\mathbf{U1})^i\right)\mathbf{U4} + \alpha_1\left(\sum_{i=0}^{m-2}(m-1-i)\,(\mathbf{U1})^i\right)\mathbf{U3}.$$

Note that $\mathbf{L}$ and $\mathbf{Q}$ remain the same in each subinterval $[t_n^k\ t_{n+1}^{k+1}]$, so these values are computed only once in each interval $[t_n\ t_{n+1}]$.

Now that, from (9), we have a computational algorithm to implement $\mathbf{x}_n^{k+1}$ from $\mathbf{x}_n^k$, the integrator (7) can be efficiently implemented by the recursive equation

$$\begin{aligned} \mathbf{x}_{n+1} = &\left(\mathbf{I} + \mathbf{L}\mathbf{A}_n\right)^{h^{1-\gamma}}\mathbf{x}_n + \sum_{i=0}^{h^{1-\gamma}}\left(\mathbf{I} + \mathbf{L}\mathbf{A}_n\right)^{i-1}\left(\mathbf{Q} + \mathbf{L}(\mathbf{f}(t_n, \mathbf{x}_n, \xi_n) - \mathbf{A}_n\mathbf{x}_n)\right) \\ &+ h^{-\gamma}\left(\sum_{i=0}^{h^{1-\gamma}}\left(\mathbf{I} + \mathbf{L}\mathbf{A}_n\right)^{i-1}\Delta\mu^{(k-i)}\right)\mathbf{L} \end{aligned}$$

where $\Delta\xi$ are increments of the process ξ on the partition $(t_n)_{\bar{h}}$.

Note that If we are interested in computing $E(\phi(\mathbf{x}(t)))$, where $\phi(.)$ is a bounded function, we apply Monte Carlo method. For this we need to apply the numerical integrator M times to get approximate samples $\{\mathbf{z}^{[i]}\}_{i=1}^M$ from the distribution of $\mathbf{x}(t)$. Our computed approximation to $E(\phi(\mathbf{x}(t)))$ would then be the sample mean $\mu = \frac{1}{M}\sum_{i=1}^{M}\phi(\mathbf{z}^{[i]})$.

Here is the importance that the numerical evaluation of the proposed scheme can be carried out in an effective, accurate and simple way.

3. A NUMERICAL TEST

In this section a simulation study is carried out to estimate the order of convergency of proposed method, and so, to corroborate that the method introduced in this work retains the standard order of convergence of the classical deterministic schemes.

Let $0 \le t \le 4$, consider the RDE defined by

$$\begin{aligned} \dot{x}_1(t) &= -x_2 + x_1(1 - x_1^2 - x_2^2)\sin(B^H(t))^2 \\ \dot{x}_2(t) &= x_1 + x_2(1 - x_1^2 - x_2^2)\sin(B^H(t))^2 \\ x_1(0) &= 0.5 \\ x_2(0) &= 0.5, \end{aligned} \tag{10}$$

where $B^H(t)$ denotes a fractional Brownian process with Hurst exponent H.

The quantity

$$e(h) = E(\sup_{t_0 \leq t_n \leq T} \|\mathbf{x}(t_n) - \mathbf{y}(t_n)\|)$$

is used to estimate the order β of strong convergence of the proposed scheme, where the simulated trajectory $\mathbf{y} = (y_1, y_2)$ of $\mathbf{x} = (x_1, x_2)$ is computed with step size h. The estimated order or convergence $\widehat{\beta}$ is obtained from the slope of the straight line fitted to the set of points $\{\log_2(h_i), \log_2(\widehat{e}(h_i))\}_{i=1,...p}$, where $\widehat{e}(h_i)$ denotes the estimate of $e(h)$ computed as in [9]. That is, the simulations are arranged into $M = 20$ batches with $K = 100$ trajectories $\mathbf{y}(t)$ in each. The error for the j-th trajectory of the i-th batch is given by

$$\widehat{e}_{i,j}(h) = \sup_{t_0 \leq t_n \leq T} \left\|\mathbf{x}(t_n) - \mathbf{y}^{i,j}(t_n)\right\|,$$

and the sample mean error of the i-th batch and of all batches by

$$\widehat{e}_i(h) = \frac{1}{K}\sum_{j=1}^{K} \widehat{e}_{i,j}(h), \text{ and } \widehat{e}(h) = \frac{1}{M}\sum_{i=1}^{M} \widehat{e}_i(h)$$

respectively.

Specifically, the simulations were arranged into $M = 20$ batches of $K = 100$ trajectories for each step size $h_i = 2^{-(i+3)}$, with $i = 1, ..., 6$. The significance level was taken $\alpha = 0.1$. Table I shows the estimated values of $e(h_i)$ and their respective 90% confidence interval.

h	$\widehat{e}(h)$	
	$H = 0.25$	$H = 0.5$
2^{-5}	0.0001856	0.0000673
2^{-6}	0.0001091	0.0000238
2^{-7}	0.0000647	0.0000084
2^{-8}	0.0000381	0.0000029
2^{-9}	0.0000233	0.0000011

Table I: Uniform discretization errors for the method (7) applied to (10).

Table II show the estimated slope $\widehat{\beta}$ (with its 95% confidence interval). This corroborate the usefulness of (7) for the achievement of a desirable deterministic order of convergence.

$\widehat{\beta}$	
$H = 0.25$	$H = 0.5$
1.9802 ± 0.0197	1.9938 ± 0.0064

Table II: estimated slope $\widehat{\beta}$

4. CONCLUSIONS

In this work we introduce an effective numerical integrator for the computer simulation of the RDE (7). For this, a suitable approximation of the stochastic processes present in the random equation, together with the local linearization technique and an adapted Padé method with scaling and squaring strategy are conveniently combined. In this way we avoid order reduction an consequently deterministic order of convergence can be achieved independent of the moduli of continuity of the driven stochastic processes. It is important to note that thanks to the particular structure of the involved matrix in the proposed method, the computational burden is considerably reduced (from dimension $2d + 2$ to dimension d, via an adapted Padé method with scaling-squaring strategy) to the same computational load of solving simultaneous linear systems through a $\mathbf{LU}$ decomposition of a unique matrix. It is worth to note that the proposed method is well suited for parallel computing and since the computational cost scales with the dimension of the underlined original equation the approach has great potential even for very large simulation models.

REFERENCES

[1] Arnold, L., *Random Dynamical Systems*, Springer-Verlag, Heidelberg, 1998.

[2] Bharucha-Reid, A.T., *Approximate Solution of Random Equations*, North- Holland, New York and Oxford, 1979.

[3] Carbonell F., Jimenez J.C, Biscay R.J, de la Cruz H., *The Local Linearization method for numerical integration of random differential equations*, BIT Num. Math, 45 (2005), 1-14.

[4] De la Cruz H., Biscay R.J., Jimenez J.C, Carbonell F., Ozaki T., *High Order Local Linearization methods: an approach for constructing A-stable high order explicit schemes for stochastic differential equations with additive noise*, BIT Num. Math, 50 (2010), 509-539.

[5] Golub G. H. and Van Loan C. F., *Matrix Computations*, The Johns Hopkins University Press, 2nd Edition, 1989.

[6] Grune, L. and Kloeden, P. E., *Pathwise approximation of random ordinary differential equation*, BIT, 41 (2001) 711–721.

[7] Hasminskii, R. Z., *Stochastic Stability of Differential Equations*, Sijthoff Noordhoff, Alphen aan den Rijn, The Netherlands, 1980.

[8] Jentzen, A. , Kloeden P., *Pathwise Taylor schemes for random ordinary differential equations*. BIT Numer. Math., 49 (2009) 113-140.

[9] Kloeden, P. E. and E. Platen, E., *Numerical Solution of Stochastic Differential Equations*, Springer-Verlag, Berlin, 1992.

[10] Sobczyk, K., *Stochastic differential equations with applications to Physics and Engineering*, Kluwer, Dordrecht, 1991.

[11] Soong, T. T., *Random Differential Equations in Science and Engineering*, Academic Press, New York, 1974. 16 (1979) 1019-1035.

[12] Vom Scheidt, J., Starkloff, H. J. and Wunderlich, R., *Random transverse vibrations of a one-sided fixed beam and model reduction*, ZAMM Z. Angew. Math. Mech., 82 (2002) 847-859.

[13] Sidje, R. B., *EXPOKIT: software package for computing matrix exponentials*, AMC Trans. Math. Software, 24 (1998) 130-156.

[14] Tiwari, J. L. and Hobbie, J. E., *Random differential equations as models of ecosystems: Monte Carlo simulation approach*, Math. Biosci., 28 (1976) 25-44.

[15] Tsokos, C. P. and Padgett, W. J., *Random Integral Equations with Applications in Sciences and Engineering*, Academic Press, New York, (1974).

[16] Wonham, W.M., *Random Differential Equations in Control Theory*, In: Probabilistic Methods in Applied Mathematics, Vol. 2, A.T. Bharucha-Reid (Ed.), Academic Press, N.Y., (1971), 31-212.

RESPONSABILIDADE AUTORAL

"O(s) autor(es) (so) o(s) nico(s) responsvel(is) pelo contedo deste trabalho".