On the probabilistic solution of ODEs by Monte Carlo generation of random trees

Qiao Huang[0000-0002-1798-8313] and
Nicolas Privault[0000-0003-4148-8543]

Abstract The main goal of this paper is to provide an algorithm for the random sampling of Butcher trees appearing in the numerical solution of ordinary differential equations (ODEs). This algorithm complements and simplifies a recent approach to the probabilistic representation of ODE solutions, by removing the need to generate random branching times. The random sampling of trees is compared to the finite order truncation of Butcher series in numerical experiments.

Key words: Random trees, branching processes, ordinary differential equations.

1 Introduction

Butcher series [1], [2] are used to represent the solutions of ordinary differential equations (ODEs) by combining rooted tree enumeration with Taylor expansions, see e.g. Chapters 4-6 of [10], and [11] and references therein for applications to geometric numerical integration. Given $f : \mathbb{R}^d \to \mathbb{R}^d$ a smooth function and $t_0 < T$, consider the d-dimensional autonomous ODE problem

$$\begin{cases} \dot{x}(t) = f(x(t)), & t \in (t_0, T], \\ x(t_0) = x_0 \in \mathbb{R}^d. \end{cases} \tag{1}$$

If the solution $x(t)$ is sufficiently smooth at $t = t_0$, Taylor's expansion yields

Qiao Huang
School of Mathematics, Southeast University, Nanjing 211189, P.R. China, e-mail: qiao.huang@seu.edu.cn

Nicolas Privault
School of Physical and Mathematical Sciences, Nanyang Technological University, 21 Nanyang Link, Singapore 637371, e-mail: nprivault@ntu.edu.sg

$$x(t) = x_0 + (t - t_0) f(x(t_0)) + \sum_{k=2}^{\infty} \frac{(t-t_0)^k}{k!} \frac{d^{k-1}}{ds^{k-1}} f(x(s))_{|s=t_0}, \tag{2}$$

for small time $t - t_0$. The series (2) can be rewritten using the "elementary differentials"

$$f,\ \nabla f(f),\ \nabla^2 f(f,f),\ \nabla f(\nabla f(f)), \ldots,$$

as the expansion

$$\begin{aligned} x(t) &= x_0 + (t-t_0) f(x_0) + \frac{(t-t_0)^2}{2} \big(\nabla f(f)\big)(x_0) \\ &\quad + \frac{(t-t_0)^3}{6} \big(\nabla^2 f(f,f) + \nabla f(\nabla f(f))\big)(x_0) + \cdots, \end{aligned} \tag{3}$$

which is known to admit a graph-theoretical expression as the Butcher series

$$x(t) = \sum_{\tau} \frac{(t-t_0)^{|\tau|}}{\tau! \sigma(\tau)} F(\tau)(x_0), \tag{4}$$

over rooted trees τ, where $F(\tau)$ represents elementary differentials, and $\sigma(\tau)$, $\tau!$ respectively represent the symmetry and factorial of the tree τ, see Section 2 for definitions. The series (4) can be used to estimate ODE solutions by expanding $x(t)$ into a sum over trees up to a finite order. However, the generation of high order trees is computationally expensive.

In this paper, we consider the numerical estimation of the series (4) using Monte Carlo generation of random trees and branching processes, which are classical probabilistic tools that have been the object of extensive studies, see for example [23], [24]. Although Monte Carlo estimators cannot compete with classical Runge-Kutta schemes, they represent an alternative to the truncation of series, and they allow for estimates whose precision improves when the number of iterations increases. This approach is also motivated by related constructions extending the use of the Feynman-Kac formula to the numerical estimation of the solutions of fully nonlinear partial differential equations by stochastic branching mechanisms and stochastic cascades, see [22], [15], [19], [17], [6], [7], [13], [8], [9], [20].

Here, in comparison with [20], we present a direct and simpler approach to the random generation of Butcher trees that does not require the use of random branching times. From a simulation point of view, this amounts to estimating (4) as an expected value, by generating random trees $\mathcal{T}$ having a conditional probability distribution of the form

$$\mathbb{P}(\mathcal{T} = \tau \mid |\mathcal{T}| = n) = \frac{c_n}{\tau! \sigma(\tau)}$$

over rooted trees τ of size $|\tau| = n \geq 0$, and $(c_n)_{n\geq 0}$ is a sequence of positive numbers. More precisely, we construct a random tree $\mathcal{T}$ whose size complies with a probability distribution $(p_n)_{n\geq 0}$ on $\mathbb{N}$, such that the solution of the ODE (1) admits the following probabilistic representation:

$$x(t) = \mathbb{E}\left[\frac{(t-t_0)^{|\mathcal{T}|} F(\mathcal{T})(x_0)}{(|\mathcal{T}| \vee 1) p_{|\mathcal{T}|}} \right], \tag{5}$$

see Theorem 1. The Monte Carlo implementation of (5) allows us to estimate $x(t)$ at a given $t > t_0$ within a (possibly finite) time interval. A major difference with the expansion (4) is that the Monte Carlo method proceeds iteratively by randomly sampling trees of arbitrary orders, therefore avoiding the evaluation of (4) at a fixed order. In comparison to the probabilistic representation of ODE solutions proposed in [21], the present algorithm does not require the generation of sequences of random branching times.

In Section 5 we extend our approach to semilinear ODEs of the form

$$\begin{cases} \dot{x}(t) = Ax(t) + f(x(t)), & t \in (t_0, T], \\ x(t_0) = x_0 \in \mathbb{R}^d, \end{cases}$$

where A is a linear operator on $\mathbb{R}^d$. Here, tree sizes are generated with the Poisson distribution of mean $t > 0$, and the impact of the operator A is taken into account via an independent continuous-time Markov chain with generator A. In this case, our approach extends the construction presented in [6] for linear PDEs, by replacing the use of linear chains (or paths) with general random trees in the setting of nonlinear ODEs. This construction can also be seen as a randomization of exponential Butcher series, see [14], [18].

Numerical examples are presented in Section 6, using Mathematica, by comparing the Monte Carlo evaluation of (5) to the truncation

$$x(t) = \sum_{\substack{\tau \in \mathbf{T} \\ |\tau| \le n}} \frac{(t-t_0)^{|\tau|}}{\tau! \sigma(\tau)} F(\tau)(x_0), \qquad t > t_0, \tag{6}$$

of (8) at different orders $n \geq 1$. The Mathematica codes presented in this paper are available at

https://github.com/nprivaul/mc-odes/blob/main/mc-odes.nb

We refer the reader to [16] for a complete implementation of Butcher series computations in Julia.

We proceed as follows. In Section 2 we review the construction of Butcher trees for the representation of ODE solutions. Section 3 reviews and proves additional statements needed for labelled trees. In Section 4 we present the algorithm for the random generation of Butcher trees by the random attachment of vertices. Section 5 deals with semilinear ODEs using Poisson distributed tree sizes and a continuous-time Markov chain. Numerical examples are presented in Section 6, and multidimensional versions of the codes are listed in Section 7.

2 Butcher trees

In this section we review the construction of Butcher trees for the series representation (3) of the solution of the ODE (1). A rooted tree $\tau = (V, E, \bullet)$ is a nonempty set V of vertices and a set of edges E between some of the pairs of vertices, with a specific vertex called the root and denoted by "$\bullet$", such that the graph (V, E) is connected with no loops. We denote by "$\emptyset$" and "$\bullet$" the empty tree and the single node tree, respectively. The next definition uses the B^+ operation, see [3, pages 44-45], namely if $\tau_1, \ldots, \tau_m$ are trees, then $[\tau_1, \ldots, \tau_m]$ denotes the tree τ formed by introducing a new vertex, which becomes the root of τ, and m new edges from the root of τ to each of the roots of τ_i, $i = 1, 2, \ldots, m$. We also use the notation

$$[\tau_1^{k_1}, \ldots, \tau_n^{k_n}] = [\underbrace{\tau_1, \ldots, \tau_1}_{k_1 \text{ terms}}, \ldots, \underbrace{\tau_n, \ldots, \tau_n}_{k_n \text{ terms}}], \quad k_1, \ldots, k_n \in \mathbb{N}.$$

Definition 1 [11, Definition III.1.1]. The set of rooted trees is denoted by $\mathbf{T}$, and can be defined as the closure of $\emptyset$ and $\bullet$ under the B^+ operation, i.e.:

(i) $\emptyset \in \mathbf{T}$, $\bullet \in \mathbf{T}$,
(ii) $[\tau_1, \ldots, \tau_m] \in \mathbf{T}$ if $\tau_1, \ldots, \tau_m \in \mathbf{T}$.

The size (or order) of $\tau \in \mathbf{T}$ is defined as the number of its vertices, and denoted by $|\tau|$. In particular, we have $|\emptyset| = 0$ and $|\bullet| = 1$. For $n \geq 0$, we denote by $\mathbf{T}_n$ the subset of trees of order n in $\mathbf{T}$, and for $a_1, \ldots, a_m \in \mathbb{R}^d$ we let

$$\nabla^m f(a_1, \ldots, a_m) := \left(\sum_{i_1, \ldots, i_m = 1}^{d} \frac{\partial^m f}{\partial x_{i_1} \cdots \partial x_{i_m}} a_{1,i_1} \cdots a_{m,i_m} \right)_{j=1,\ldots,d}.$$

Definition 2 [11, Definition III.1.2]. The elementary differential of $f \in C^\infty(\mathbb{R}^d, \mathbb{R}^d)$ is the mapping $F : \mathbf{T} \to C^\infty(\mathbb{R}^d, \mathbb{R}^d)$ defined recursively by

(i) $F(\emptyset) = \mathbf{Id}$, $F(\bullet) = f$,
(ii) $F(\tau) = \nabla^m f(F(\tau_1), \ldots, F(\tau_m))$ for $\tau = [\tau_1, \ldots, \tau_m]$.

The pair $(\tau, F(\tau))$ is called a Butcher tree, and the map F can be used to express any of the terms involving f in the series (3). Indeed, when $|\tau| = n$, $F(\tau)$ takes the form

$$\prod_{i=1}^{n} \nabla^{m_i} f = \nabla^{m_1} f(\nabla^{m_2} f(\cdots), \ldots, \ldots (\ldots, f) \cdots), \tag{7}$$

for some integer sequence $(m_i)_{i=1,\ldots,n}$ such that $m_n = 0$ and $m_1 + \cdots + m_n = n - 1$.

Given a tree $\tau \in \mathbf{T}$, the map F provides a way to encode each vertex of τ using f or its derivatives: each vertex with no descendants is coded by f, and the vertices with m descendants are coded by $\nabla^m f$, for $m \geq 1$. In order to characterize the coefficients of (3), we need two functionals defined on trees.

Definition 3 a) [2, Section 304], [3, Section 2.5]. The symmetry σ of a tree is defined recursively by

(i) $\sigma(\emptyset) = 1$, $\sigma(\bullet) = 1$,
(ii) $\sigma([\tau_1^{k_1}, \ldots, \tau_m^{k_m}]) = \prod_{i=1}^m k_i!\sigma(\tau_i)^{k_i}$ for $\tau_1, \ldots, \tau_m$ distinct and $k_1, \ldots, k_m \in \mathbb{N}$.

b) The factorial (or density) $\tau!$ of a tree τ is defined by

(i) $\emptyset! = 1$, $\bullet ! = 1$,
(ii) $\tau! = |\tau| \prod_{i=1}^m \tau_i!$ for $\tau = [\tau_1, \ldots, \tau_m]$.

Using the above formalism, we obtain the following result.

Proposition 1. *[3, Definition 3.4B, Theorem 3.5C]. The series (3) can be rewritten as the Butcher series*

$$x(t) = \sum_{\tau \in \mathbf{T}} \frac{(t - t_0)^{|\tau|}}{\tau!\sigma(\tau)} F(\tau)(x_0), \qquad t > t_0. \tag{8}$$

The computation of the Taylor expansion (2) can be implemented in the following Mathematica code, by noting that in order to calculate the term of order $k \geq 2$ in the Taylor expansion (2), the quantity $x(s)$ in $f(x(s))$ can be replaced with its expansion until the order $k-1$, as the $(k-1)$-th derivative d^{k-1}/ds^{k-1} of $(s-t_0)^l$ vanishes at $s = t_0$ when $l \geq k$.

```
 Taylor[f_, t_, x0__, t0_, k_] := (d = Length[x0];
 g[x_] := x0 + f[x0]*(x - t0); h = Array[hh, d];
 For[j = 2, j <= k, j++, For[i = 1, i <= d, i++,
   h[[i]][z_] = g[x][[i]] + (z - t0)^j/j!*D[f[g[s]][[i]], {s, j - 1}] /. {s ->
        t0};]; v = {};
  For[i = 1, i <= d, i++, v = Append[v, h[[i]][x]]]; g[x_] = v];
 Return[g[t]])
f[x__] := {f1[x[[1]], x[[2]]], f2[x[[1]], x[[2]]]}
Taylor[f, t, {x1, x2}, 0, 2]
```

with sample output

$$\left\{ \mathrm{x1} + t\mathrm{f1}(\mathrm{x1},\mathrm{x2}) + \frac{1}{2}t^2 \left(\mathrm{f1}^{(0,1)}(\mathrm{x1},\mathrm{x2})\mathrm{f2}(\mathrm{x1},\mathrm{x2}) + \mathrm{f1}(\mathrm{x1},\mathrm{x2})\mathrm{f1}^{(1,0)}(\mathrm{x1},\mathrm{x2}) \right), \right.$$
$$\left. \mathrm{x2} + t\mathrm{f2}(\mathrm{x1},\mathrm{x2}) + \frac{1}{2}t^2 \left(\mathrm{f1}(\mathrm{x1},\mathrm{x2})\mathrm{f2}^{(1,0)}(\mathrm{x1},\mathrm{x2}) + \mathrm{f2}(\mathrm{x1},\mathrm{x2})\mathrm{f2}^{(0,1)}(\mathrm{x1},\mathrm{x2}) \right) \right\}$$

for $k = 2$, $d = 2$, and $t_0 = 0$. Instead of the above code, we will use the following implementation of the truncated Butcher series (6) up to any tree order $n \geq 1$ in case $d = 1$, which also prints out the corresponding trees.

```
 B[f_, t_, x0_, t0_, n_] := (If[n == 0, Return[x0],
   If[n == 1, Return[x0 + (t - t0)*f[x0]],
    sample = x0 + (t - t0)*f[x0]; g = Graph[{1 -> 2}];
    g = Graph[g, VertexLabels -> {1 -> D[f[ y], y]}];
    g = Graph[g, VertexLabels -> {2 -> f[y]}]; m = 1;
    sample = sample + 1/2*(t - t0)^VertexCount[g]*
```

```
      Product[ff[[2]] , {ff, List @@@ PropertyValue[g, VertexLabels]}] /. {y -> x0};
    list = {g};
    While[m <= (n - 2), temp = list; list = {};
     Do[l = VertexCount[g];
      For[j = 1, j <= l, j++, gg = VertexAdd[g, {l + 1}];
       gg = Graph[gg, VertexLabels -> {l + 1 -> f[ y]}];
       lab = Sort[List @@@ PropertyValue[gg, VertexLabels]][[j]][[2]];
       gg = Graph[gg, VertexLabels -> {j -> D[lab, y]}];
       gg = EdgeAdd[gg, j -> l + 1]; Print[gg];
       sample = sample + (t - t0)^(l + 1)/(l + 1)!*
          Product[ff[[2]] , {ff, List @@@ PropertyValue[gg, VertexLabels]}] /. {y ->
              x0};
        list = Append[list, gg]], {g, temp}]; m = m + 1];
    Return[sample]]]);
```

For example, the command `B[f,t,x0,t0,4]` produces the scalar output

$$\begin{aligned}
&\mathrm{x0} + \mathrm{f(x0)(t-t0)} + \frac{1}{2}\mathrm{f(x0)(t-t0)^2f'(x0)} \\
&+ \frac{1}{6}\mathrm{f(x0)^2(t-t0)^3f''(x0)} + \frac{1}{6}\mathrm{f(x0)(t-t0)^3f'(x0)^2} \\
&+ \frac{1}{24}\mathrm{f(x0)(t-t0)^4f'(x0)^3} + \frac{1}{24}\mathrm{f(x0)^3f^{(3)}(x0)(t-t0)^4} + \frac{1}{6}\mathrm{f(x0)^2(t-t0)^4f'(x0)f''(x0)}
\end{aligned}$$

and enumerates the trees appearing in the series (8).

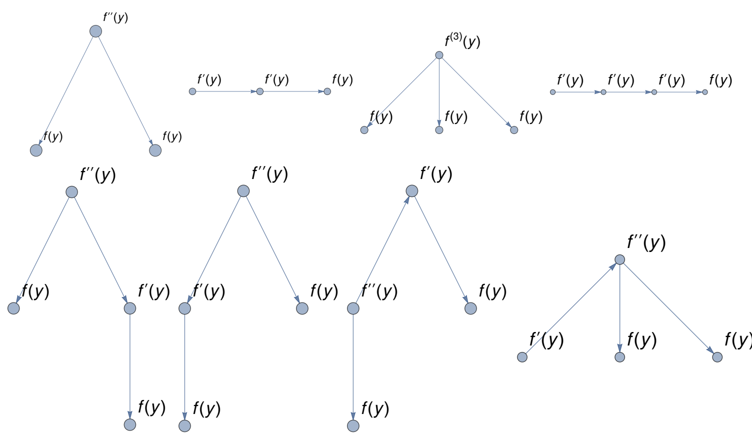


Fig. 1: Generation of Butcher trees.

Figure 1 prints directed rooted trees of orders 3 and 4, each with a natural orientation away from the root. In addition to the result of Proposition 1, the derivatives of the Butcher series (8) can be written using the elementary differentials F introduced in Definition 2, as in the following lemma.

Lemma 1 *[3, Lemma 3.5B] We have*

$$x^{(n)}(t) = \sum_{\tau \in \mathbf{T}_n} \alpha(\tau) F(\tau)(x(t)), \quad n \geq 1, \tag{9}$$

where

$$\alpha(\tau) := \frac{|\tau|!}{\tau!\sigma(\tau)}, \qquad \tau \in \mathbf{T}. \tag{10}$$

Proof. For completeness, we give a proof of (9) by induction using the Faà di Bruno formula [5, Theorem 2.1], which states that for a smooth function $g : \mathbb{R}^d \to \mathbb{R}^{d'}$, the derivatives of $y(t) = g(x(t))$ are given by, for each $n \geq 1$,

$$\begin{aligned} &y^{(n)}(t) \\ &= \sum_{\substack{k_1,\ldots,k_n \geq 0 \\ \sum_{i=1}^n ik_i = n}} \frac{n!}{\prod_{i=1}^n k_i!(i!)^{k_i}} \nabla^{k_1+\cdots+k_n} g(x(t)) \Big(\underbrace{\dot{x}(t), \ldots, \dot{x}(t)}_{k_1 \text{ terms}}, \ldots, \underbrace{x^{(n)}(t), \ldots, x^{(n)}(t)}_{k_n \text{ terms}} \Big), \end{aligned} \tag{11}$$

where the summation is taken over all different nonnegative solutions $(k_1, \ldots, k_n)$ of the linear Diophantine equation $\sum_{i=1}^n ik_i = n$. Now we prove (9). We start with $n = 1$, for which the result holds. Assume that the result is true for lower orders than n. Then we apply Faà di Bruno's formula (11) to derive

$$\begin{aligned} x^{(n)}(t) &= [f(x)]^{(n-1)}(t) \\ &= \sum_{\substack{k_1,\ldots,k_{n-1} \geq 0 \\ \sum_{i=1}^{n-1} ik_i = n-1}} \frac{(n-1)!}{\prod_{i=1}^{n-1} k_i!(i!)^{k_i}} \\ &\qquad \sum_{\tau_i} \prod_{i=1}^{n-1} \left(\frac{i!}{\tau_i!\sigma(\tau_i)} \right)^{k_i} \nabla^{k_1+\cdots+k_{n-1}} f\big(F(\tau_1)^{k_1}, \ldots, F(\tau_{n-1})^{k_{n-1}}\big)(x(t)) \\ &= \sum_{\substack{k_1,\ldots,k_{n-1} \geq 0 \\ \sum_{i=1}^{n-1} ik_i = n-1}} n! \frac{\nabla^{k_1+\cdots+k_{n-1}} f\big(F(\tau_1)^{k_1}, \ldots, F(\tau_{n-1})^{k_{n-1}}\big)(x(t))}{\Big(n \prod_{i=1}^{n-1} (\tau_i!)^{k_i}\Big) \Big(\prod_{i=1}^{n-1} k_i!\sigma(\tau_i)^{k_i}\Big)]} \\ &= \sum_{\tau \in \mathbf{T}_n} \frac{n!}{\tau!\sigma(\tau)} F(\tau)(x(t)), \end{aligned}$$

where in the first line, the first summation is taken over all nonnegative integers $(k_i)_{i=1,\ldots,n-1}$ satisfying $\sum_{i=1}^{n-1} ik_i = n-1$, the second summation is taken over all trees $(\tau_i)_{i=1,\ldots,n-1}$ with $|\tau_i| = i$; in the last line $\tau = [\tau_1^{k_1}, \ldots, \tau_{n-1}^{k_{n-1}}]$ so that $|\tau| = 1 + \sum_{i=1}^{n-1} ik_i = n$. □

The next lemma presents a tree expansion for $g(x(t))$, with $g \in C^\infty(\mathbb{R}^d, \mathbb{R}^{d'})$, $d, d' \geq 1$. For this purpose, we consider the elementary differential $F_g : \mathbf{T} \to C^\infty(\mathbb{R}^d, \mathbb{R}^{d'})$ of g defined by

$$F_g(\emptyset) := \mathbf{Id}, \ F_g(\bullet) := g, \ \text{and} \ F_g(\tau) := \nabla^m g(F(\tau_1), \ldots, F(\tau_m)), \ \tau = [\tau_1, \ldots, \tau_m], \tag{12}$$

with $\nabla^0 g = g$ and $0! = 1$.

Lemma 2 *Let $g \in C^\infty(\mathbb{R}^d, \mathbb{R}^{d'})$. We have*

$$g(x(t)) = \sum_{\tau\in\mathbf{T}\setminus\{\emptyset\}} \frac{(t-t_0)^{|\tau|-1}}{(|\tau|-1)!}\alpha(\tau)F_g(\tau)(x_0). \tag{13}$$

Proof. Letting $x(t) = (x_i(t))_{1\le i\le d}$, $x_0 = (x_i(0))_{1\le i\le d}$, and denoting by $(x(t)-x_0)^{\otimes m}$ the matrix $\big((x_{i_1}(t)-x_{i_1}(0)),\dots,(x_{i_m}(t)-x_{i_m}(0))\big)_{1\le i_1,\dots,i_m\le d}$, by (8) and Definition 3 we have

$$\begin{aligned}
g(x(t)) &= \sum_{m=0}^{\infty} \frac{1}{m!}\nabla^m g(x_0)\left((x(t)-x_0)^{\otimes m}\right)\\
&= \sum_{m=0}^{\infty} \frac{1}{m!} \sum_{\tau_1,\dots\tau_m\in\mathbf{T}\setminus\{\emptyset\}} \frac{(t-t_0)^{|\tau_1|+\dots+|\tau_m|}}{\prod_{i=1}^m(\tau_i!\sigma(\tau_i))}\nabla^m g\left(F(\tau_1),\dots,F(\tau_m)\right)(x_0)\\
&= \sum_{m=0}^{\infty} \sum_{\tau_1,\dots\tau_m\in\mathbf{T}\setminus\{\emptyset\}} \frac{|\tau|\prod_j k_j!}{m!\tau!\sigma(\tau)}(t-t_0)^{|\tau|-1}\nabla^m g\left(F(\tau_1),\dots,F(\tau_m)\right)(x_0),
\end{aligned}$$

where the indexes $k_1, k_2, \dots$ count equal trees among $\tau_1,\dots,\tau_m$. From the fact that there are $\binom{m}{k_1,k_2,\dots}$ possibilities for writing the tree τ in the form $[\tau_1,\dots,\tau_m]$, it then follows that

$$\begin{aligned}
g(x(t)) &= \sum_{\tau\in\mathbf{T}\setminus\{\emptyset\}} |\tau|\frac{(t-t_0)^{|\tau|-1}}{\tau!\sigma(\tau)}\nabla^m g\left(F(\tau_1),\dots,F(\tau_m)\right)(x_0)\\
&= \sum_{\tau\in\mathbf{T}\setminus\{\emptyset\}} \frac{(t-t_0)^{|\tau|-1}}{(|\tau|-1)!}\alpha(\tau)\nabla^m g\left(F(\tau_1),\dots,F(\tau_m)\right)(x_0),
\end{aligned}$$

where $\alpha(\tau)$ is defined in (10). Then, we get (13). □

In particular, when g is the function f in ODE (1), then $F_f = F$ and

$$f(x(t)) = \sum_{\tau\in\mathbf{T}\setminus\{\emptyset\}} \frac{(t-t_0)^{|\tau|-1}}{(|\tau|-1)!}\alpha(\tau)F(\tau)(x_0),$$

which in turn proves that the Butcher series (8) solves (1).

3 Labelled trees

Our random generation of Butcher trees will use labelled trees, which provide a combinatorial interpretation of the coefficients appearing in (8). By convention, the empty tree $\emptyset$ is labelled by 0 and the root of any non-empty tree is labelled by 1.

Definition 4 [3, (2.5e)] For $n\ge 1$ we denote by $\mathbf{T}_n^\sharp$ the set of labelled rooted trees τ of order n with vertex sequence $V=\{1,\dots,n\}$, written as $\tau=(V,E,1)$, such that the label of every vertex is smaller than that of each of its children. We also let

$\mathbf{T}^\sharp := \bigcup_{n\geq 1}\mathbf{T}_n^\sharp$, and define the canonical forgetful map $\iota : \mathbf{T}^\sharp \to \mathbf{T}$ by "forgetting" the labeling information contained in a labeled tree.

We note that the labelling of a tree τ is not necessarily unique. By abuse of notation, we shall omit the map ι when there is no ambiguity, in which case we will simply use τ to denote $\iota(\tau)$ for $\tau \in \mathbf{T}^\sharp$.

Proposition 2. *[3, Theorem 2.5F] The number of all possible labellings of a rooted tree $\tau = (V, E, \bullet)$ is given by the coefficient $\alpha(\tau)$ defined in (10).*

As a consequence of Proposition 2, since the labelling of a tree does not affect its elementary differential, we can rewrite (8) and (9) respectively as

$$x(t) = \sum_{\tau\in\mathbf{T}^\sharp} \frac{(t-t_0)^{|\tau|}}{|\tau|!} F(\tau)(x_0) \quad \text{and} \quad x^{(n)}(t) = \sum_{\tau\in\mathbf{T}_n^\sharp} F(\tau)(x(t)), \quad n \geq 1, \tag{14}$$

see also [12, Theorem II.2.6]. Next, we define a new product on labelled trees that generalizes the beta-product [3, Section 2.1], cf. § 1.5 of [4].

Definition 5 (Grafting product) Let $\tau_1 = (V_1, E_1, 1)$ and $\tau_2 = (V_2, E_2, 1)$ be two labelled trees, and let $l \in V_1 = \{1, \ldots, |\tau_1|\}$.

- The grafting product with label l of τ_1 and τ_2, denoted by $\tau_1 *_l \tau_2$, is the tree of order $|\tau_1| + |\tau_2|$ formed by grafting (attaching) τ_2 from its root to the vertex l of τ_1, so that the vertices of τ_2 become descendants of the vertex l.
- The tree $\tau_1 *_l \tau_2$ is labelled by keeping the labels of τ_1, and by adding $|\tau_1|$ to the labels of τ_2.

For any labelled tree τ, we let $\emptyset *_0 \tau = \tau *_l \emptyset = \tau$ for all $0 \leq l \leq |\tau|$, and keep the labels of τ.

Remark 1 (i) The beta-product is a grafting product with label 1, as the second tree is always attached to the root of the first one.
(ii) The B^+ operation can also be expressed by grafting-products, by forgetting labelling. For example, we have $[\tau_1, \tau_2] = \bullet *_1 \tau_1 *_1 \tau_2 = \bullet *_1 \tau_2 *_1 \tau_1$.

We note that any labelling is equivalent to a sequence of grafting of dots. In the next lemma we let $\triangle_0 := \{0\}$, and

$$\triangle_n := \{(l_1, \ldots, l_n) \ : \ 1 \leq l_i \leq i, \ i = 1, \ldots, n\}, \quad n \geq 1.$$

Lemma 3 *(i) Given τ a labelled tree with $|\tau| \geq 2$, there is a unique sequence $(l_i)_{1\leq i\leq|\tau|-1}$ in $\triangle_{|\tau|-1}$ such that*

$$\tau = \bullet *_{l_1} \bullet *_{l_2} \cdots *_{l_{|\tau|-1}} \bullet. \tag{15}$$

(ii) For any $n \geq 2$, the map which sends $\tau \in \mathbf{T}_n^\sharp$ to the sequence $(l_1, \ldots, l_{n-1})$ determined by (15) is a bijection from $\mathbf{T}_n^\sharp$ to $\triangle_{n-1}$.

Proof. We prove (i) by induction on $|\tau|$. The case $|\tau| = 2$ is verified since $\tau = \bullet *_1 \bullet$. Suppose that (15) holds for all trees τ such that $|\tau| = n$, and let τ be a labelled tree with $|\tau| = n + 1$. Denote by τ_- the subtree obtained by removing the vertex with label $n+1$ from τ. It is clear from Definition 4 that the parent of the vertex $n+1$ has label l not bigger than n, hence $\tau = \tau_- *_l \bullet$ has the form (15). The converse of (i) holds, since for each $n \geq 2$, the sequence $(l_1, \ldots, l_{n-1}) \in \triangle_{n-1}$ determines a unique tree $\bullet *_{l_1} \bullet *_{l_2} \cdots *_{l_{n-1}} \bullet$ in $\mathbf{T}_n^\sharp$. Assertion (ii) follows from (i). □

The next result is a consequence of Lemma 3-(ii).

Corollary 1 *The number of labelled trees of order $n \geq 1$ is given by*

$$|\mathbf{T}_n^\sharp| = \sum_{\tau \in \mathbf{T}_n^\sharp} 1 = \sum_{\tau \in \mathbf{T}_n} \alpha(\tau) = (n-1)!.$$

Proof. For completeness, we provide a proof that does not rely on Lemma 3-(ii). By (9) and (10) we have

$$x^{(n)}(t_0) = \sum_{\tau \in \mathbf{T}_n} \alpha(\tau) F(\tau)(x_0).$$

Letting $f(x) := e^x$, $x_0 := 0$ and $t_0 := 0$, by (7) we have $F(\tau) = f^n$ for all $\tau \in \mathbf{T}_n$. Hence, the solution $x(t) = -\log(1-t)$ of (1) satisfies $x^{(n)}(0) = \sum_{\tau \in \mathbf{T}_n} \alpha(\tau)$, and it remains to note that $x^{(n)}(t) = (n-1)!(1-t)^{-n}$, $n \geq 1$. □

4 Random sampling of Butcher trees

In this section we discuss the representation of solutions to (1) by the random generation of Butcher trees.

Definition 6 Given $(p_n)_{n \geq 0}$ a probability distribution on $\mathbb{N}$ such that $p_n > 0$ for all $n \geq 0$, we generate a random labelled tree $\mathcal{T}$ by uniform attachment, as follows.

i) Choose the order of $\mathcal{T}$ with the distribution $\mathbb{P}(|\mathcal{T}| = n) = p_n$, $n \geq 0$;
ii) Start from a root $\bullet$ with the label 1;
iii) Starting from a tree τ with order l, $1 \leq l \leq n-1$, attach a new vertex with label $l+1$ to an independently and uniformly chosen vertex of τ, and repeat this step inductively until we reach the given order n.

For $n \geq 0$, we let

$$q_n^\sharp(\tau) := \mathbb{P}\left(\mathcal{T} = \tau \mid |\mathcal{T}| = n\right), \quad \tau \in \mathbf{T}_n^\sharp,$$

denote the conditional distribution of $\mathcal{T}$ given its size is $n \geq 0$. We note that $\iota(\mathcal{T})$ is $\mathbf{T}$-valued, and its conditional distribution on $\mathbf{T}$ is given by

$$q_n(\tau) := \mathbb{P}\left(\iota(\mathcal{T}) = \tau \mid |\mathcal{T}| = n\right) = \sum_{\substack{\tau' \in \mathbf{T}_n^\sharp \\ \iota(\tau')=\tau}} q_n^\sharp(\tau'), \quad \tau \in \mathbf{T}_n. \tag{16}$$

By Lemma 3-(ii), the random labelled tree $\mathcal{T}$ generated in Definition 6 takes the form

$$\mathcal{T} = \bullet *_{\eta_1} \bullet *_{\eta_2} \cdots *_{\eta_{|\mathcal{T}|-1}} \bullet,$$

where $(\eta_1, \ldots, \eta_{|\mathcal{T}|-1})$ is a uniform random variable taking values in $\triangle_{|\mathcal{T}|-1}$.

Theorem 1 *Assume that there exists $C > 0$ such that*

$$|\nabla^m f(x_0)| \le C, \quad \textit{for all } m \ge 0. \tag{17}$$

Then, the solution of the ODE (1) *admits the probabilistic expression*

$$x(t) = \mathbb{E}\left[\frac{(t-t_0)^{|\mathcal{T}|} F(\mathcal{T})(x_0)}{(|\mathcal{T}| \vee 1) p_{|\mathcal{T}|}}\right], \qquad t \in [t_0, t_0 + 1/C). \tag{18}$$

Proof. It follows from Lemma 3-(ii) that given $|\mathcal{T}| = n$, the random tree $\mathcal{T}$ is uniformly distributed in $\mathbf{T}_n^\sharp$, i.e. we have

$$q_n^\sharp(\tau) = \mathbb{P}\left(\mathcal{T} = \tau \mid |\mathcal{T}| = n\right) = \frac{1}{|\mathbf{T}_n^\sharp|} = \frac{1}{((n-1) \vee 0)!},$$

in which case $q_n^\sharp(\tau)$ is independent of $\tau \in \mathbf{T}_n^\sharp$, and the conditional probability (16) is given by

$$q_n(\tau) = \frac{\alpha(\tau)}{((n-1) \vee 0)!}, \qquad \tau \in \mathbf{T}.$$

Hence, we have

$$\begin{aligned}
\mathbb{E}\left[\frac{(t-t_0)^{|\mathcal{T}|} F(\mathcal{T})(x_0)}{(|\mathcal{T}| \vee 1) p_{|\mathcal{T}|}}\right] &= \sum_{n=0}^{\infty} \frac{(t-t_0)^n p_n}{(n \vee 1) p_n} \sum_{\tau \in \mathbf{T}_n^\sharp} q_n^\sharp(\tau) \mathbb{E}\left[F(\mathcal{T})(x_0) \mid |\mathcal{T}| = n, \mathcal{T} = \tau\right] \\
&= \sum_{\tau \in \mathbf{T}^\sharp} \frac{(t-t_0)^{|\tau|}}{|\tau|!} F(\tau)(x_0) \\
&= x(t), \qquad (19)
\end{aligned}$$

by the first equation of (14). From the assumption (17), we have $|F(\tau)(x_0)| \le C^{|\tau|}$ for all $\tau \in \mathbf{T}$ such that $|\tau| \ge 1$. The q-th integrability of (19), $q \ge 1$, can be implied by the bound

$$\mathbb{E}\left[\left|\frac{(t-t_0)^{|\mathcal{T}|} F(\mathcal{T})(x_0)}{(|\mathcal{T}| \vee 1) p_{|\mathcal{T}|}}\right|^q\right] \le \frac{|x_0|^q}{p_0^{q-1}} + \sum_{n=1}^{\infty} \frac{(C(t-t_0))^{nq}}{n^q p_n^{q-1}} \sum_{\tau \in \mathbf{T}_n^\sharp} q_n^\sharp(\tau)$$

$$= \frac{|x_0|^q}{p_0^{q-1}} + \sum_{n=1}^{\infty} \frac{(C(t-t_0))^{nq}}{n^q p_n^{q-1}}, \tag{20}$$

which is finite for $q = 1$, provided that $C(t - t_0) < 1$. □

The random generation of Butcher trees in Theorem 1 is implemented in the following Mathematica code:

```
  MCsample[f_, t_, x0_, dist_] := (n = RandomVariate[dist];
   If[n == 0, Return[x0/PDF[dist, 0]],
    If[n == 1, Return[t*f[x0]/PDF[dist, 1]], g = Graph[{1 -> 2}];
     g = Graph[g, VertexLabels -> {1 -> D[f[ y], y]}];
     g = Graph[g, VertexLabels -> {2 -> f[y]}]; m = 1;
     While[m <= (n - 2), l = VertexCount[g];
      j = RandomVariate[DiscreteUniformDistribution[{1, l}]];
      g = VertexAdd[g, {l + 1}];
      g = Graph[g, VertexLabels -> {l + 1 -> f[ y]}];
      lab = Sort[List @@@ PropertyValue[g, VertexLabels]][[j]][[2]];
      g = Graph[g, VertexLabels -> {j -> D[lab, y]}];
      g = EdgeAdd[g, j -> l + 1]; m++];
     sample = Product[ff[[2]] , {ff,
        List @@@ PropertyValue[g, VertexLabels]}] /. {y -> x0};
      Return[sample*t^n/PDF[dist, n]/n]]]);
f[y_] := Exp[y]
MCsample[f, t, x0, GeometricDistribution[0.5]]
```

5 Connection with semilinear PDEs

In this section, we consider the case where the function f in (1) involves a linear component, i.e. $f(x) = Ax + g(x)$, where A is a linear operator on $\mathbb{R}^d$, in which case the ODE (1) becomes

$$\begin{cases} \dot{x}(t) = Ax(t) + g(x(t)), & t \in (t_0, T], \\ x(t_0) = x_0 \in \mathbb{R}^d, \end{cases} \tag{21}$$

and can be rewritten in integral form as

$$x(t) = e^{(t-t_0)A} x_0 + \int_{t_0}^{t} e^{(t-s)A} g(x(s)) ds, \quad t \in (t_0, T].$$

By [18, Theorem 4.5] we have

$$x(t) = \sum_{\tau \in \mathbf{T}} \alpha(\tau) \phi_{|\tau|}(t, A) F_g(\tau)(x_0), \tag{22}$$

where F_g is defined in (12), with $\phi_0(t, a) := e^{(t-t_0)a}$ and

$$\phi_n(t, a) := \int_{t_0 \le t_1 < \cdots < t_n \le t} e^{(t-t_n)a} dt_n \cdots dt_1 = \int_{t_0}^{t} e^{(t-s)a} \frac{(s-t_0)^{n-1}}{(n-1)!} ds, \tag{23}$$

$n \geq 1$, $t \geq t_0$. In addition, from the fact that labelling does not change elementary differentials, the expansion (22) can be rewritten as the exponential Butcher series

$$x(t) = \sum_{\tau \in \mathbf{T}^\sharp} \phi_{|\tau|}(t, A) F_g(\tau)(x_0). \tag{24}$$

Given $(N_t)_{t \geq t_0}$ a standard Poisson process with

$$\mathbb{P}(N_t = n) = e^{-(t-t_0)} \frac{(t-t_0)^n}{n!}, \quad t \geq t_0,\ n \geq 0,$$

and increasing sequence of jump times $(T_i)_{i \geq 1}$, and let $T_0 = t_0$, let $\mathcal{T}_t$ denote the random tree constructed in Definition 6, using the Poisson distribution $p_n = \mathbb{P}(N_t = n)$, $n \geq 0$. In what follows, we assume that A is a stochastic matrix, that is, a square matrix with non-negative entries where each column sums up to 1, which generates a continuous-time Markov chain $X = (X_t)_{t \geq t_0}$, independent of $(N_t)_{t \geq t_0}$.

In Theorem 2 we propose a canonical way to evaluate the solution to the semilinear equation (21) as an expected value over random trees. It is worth noting that the decomposition $f(x) = Ax + g(x)$ can be used for a generalization to semilinear parabolic PDEs, in which case A is an elliptic operator that can generate a Markov process $X = (X_t)_{t \geq t_0}$, and the discrete $\{1, \ldots, d\}$-valued index i is replaced by the spatial variable of the PDE. This can also be regarded as a randomization of the exponential Butcher series (22), and as a nonlinear extension of the probabilistic representation of [6] which uses linear chains for linear PDEs. In the special case $A = 0$, this probabilistic representation recovers (18) by generating tree sizes via the Poisson distribution $(p_n)_{n \geq 0}$ with parameter $t - t_0$.

Theorem 2 *Assume that A is a stochastic matrix and there exists $C > 0$ such that*

$$|\nabla^m g(x_0)| + |Ax_0| + |A| \leq C, \quad \text{for all } m \geq 0. \tag{25}$$

Then, for $t \in [t_0, t_0 + 1/C)$ we have

$$x_i(t) = e^{(t-t_0)} \mathbb{E}\Big[((|\mathcal{T}_t| - 1) \vee 0)! \left(F_g(\mathcal{T}_t)(x_0) \right)_{X_{t - T_{|\mathcal{T}_t|}}} \mathbf{1}_{\{T_{|\mathcal{T}_t|} \leq t\}} \Big| X_{t_0} = i \Big], \tag{26}$$

$i = 1, \ldots, d$.

Proof. From the fact that the sequence $(T_i - T_{i-1})_{i=1,\ldots,n}$ is i.i.d. with common exponential distribution, for any integrable function h on the n-dimensional simplex

$$\triangle_t^n := \{(t_1, \ldots, t_n) : t_0 \leq t_1 < \cdots < t_n \leq t\},$$

we have

$$\begin{aligned}
&\mathbb{E}[\mathbf{1}_{\{N_t = n\}} h(T_1, \ldots, T_n)] \\
&= \mathbb{E}\left[\mathbf{1}_{\{t_0 < T_n \leq t < T_{n+1}\}} h\left(T_1, \ldots, t_0 + \sum_{i=1}^n (T_i - T_{i-1}) \right) \right]
\end{aligned}$$

$$
\begin{aligned}
&= \int_{0<r_1+\cdots+r_n\le t-t_0} h\left(t_0+r_1,\dots,t_0+\sum_{i=1}^n r_i\right) e^{-(r_1+\cdots+r_n)} \\
&\qquad\qquad\qquad\qquad \int_{t-t_0-(r_1+\cdots+r_n)}^{\infty} e^{-r_{n+1}} dr_{n+1} dr_n \cdots dr_1 \\
&= e^{-(t-t_0)} \int_{0<r_1+\cdots+r_n\le t-t_0} h\left(t_0+r_1,\dots,t_0+\sum_{i=1}^n r_i\right) dr_n \cdots dr_1 \\
&= e^{-(t-t_0)} \int_{t_0\le t_1<\cdots<t_n\le t} h(t_1,\dots,t_n) dt_n \cdots dt_1,
\end{aligned}
$$

where we applied the change of variables $t_i = t_0 + r_1 + \cdots + r_i$ in the last equality. Taking $h(t_1,\dots,t_n) := e^{(t-t_n)a}$, $t_0 \le t_1 < \cdots < t_n \le t$, it follows that (23) can be rewritten as

$$
\phi_n(t,a) = e^{t-t_0}\mathbb{E}\left[\mathbf{1}_{\{N_t=n\}} e^{(t-T_n)a}\right], \quad n \ge 0.
$$

Next, by construction of the continuous-time Markov chain $(X_t)_{t\ge t_0}$ with generator A, we have

$$
\left(e^{(t-t_0)A}x\right)_i = \mathbb{E}\left[x_{X_t} \middle| X_{t_0}=i\right], \quad i=1,\dots,d,\ x=(x_1,\dots,x_d)\in\mathbb{R}^d.
$$

Finally, as the random tree $\mathcal{T}_t$ is constructed with the Poisson random size N_t and independent uniform attachment, we have

$$
|\mathcal{T}_t| = N_t, \qquad \mathbb{P}\left(\mathcal{T}_t=\tau \mid |\mathcal{T}_t|=n\right) = \frac{1}{|\mathbf{T}_n^\sharp|}, \qquad \tau\in\mathbf{T}_n^\sharp.
$$

Combining the above with (24), we get

$$
\begin{aligned}
x_i(t) &= e^{t-t_0}\sum_{n=0}^\infty \sum_{\tau\in\mathbf{T}_n^\sharp} \mathbb{E}\Big[\mathbf{1}_{\{N_t=n\}}\big(e^{(t-T_n)A}F_g(\tau)(x_0)\big)_i\Big] \\
&= e^{t-t_0}\sum_{n=0}^\infty \sum_{\tau\in\mathbf{T}_n^\sharp} \mathbb{E}\Big[\mathbf{1}_{\{N_t=n\}}\left(F_g(\tau)(x_0)\right)_{X_{t-T_n+t_0}} \Big| X_{t_0}=i\Big] \\
&= e^{t-t_0}\sum_{n=0}^\infty \sum_{\tau\in\mathbf{T}_n^\sharp} \mathbb{E}\Big[\left(F_g(\tau)(x_0)\right)_{X_{t-T_n+t_0}} \Big| N_t=n, X_{t_0}=i\Big]\mathbb{P}(N_t=n) \\
&= e^{t-t_0}\sum_{n=0}^\infty ((n-1)\vee 0)!\,\mathbb{P}(|\mathcal{T}_t|=n) \\
&\qquad \times \sum_{\tau\in\mathbf{T}_n^\sharp} \mathbb{E}\Big[\left(F_g(\tau)(x_0)\right)_{X_{t-T_n+t_0}} \Big| |\mathcal{T}_t|=n, \mathcal{T}_t=\tau, X_{t_0}=i\Big]\mathbb{P}\left(\mathcal{T}_t=\tau \mid |\mathcal{T}_t|=n\right) \\
&= e^{t-t_0}\mathbb{E}\Big[((|\mathcal{T}_t|-1)\vee 0)!\left(F_g(\mathcal{T}_t)(x_0)\right)_{X_{t-T_{|\mathcal{T}_t|}+t_0}} \mathbf{1}_{\{T_{|\mathcal{T}_t|}\le t\}} \Big| X_{t_0}=i\Big].
\end{aligned}
$$

By the definition (12) of F_g and the bound (25), the q-th integrability of (26), $q \geq 1$, can be controlled by the bound

$$\begin{aligned}
&\mathbb{E}\Big[\Big|((|\mathcal{T}_t|-1)\vee 0)!\,\big(F_g(\mathcal{T}_t)(x_0)\big)_{X_{t-T_{|\mathcal{T}_t|}+t_0}}\Big|^q \,\Big|\, X_{t_0}=i\Big] \\
&\qquad \leq e^{-(t-t_0)}|x_0|^q + \mathbb{E}\Big[\mathbf{1}_{\{|\mathcal{T}_t|\geq 1\}}\Big|((|\mathcal{T}_t|-1)!)^q C^{|\mathcal{T}_t|}\Big|^q\Big] \\
&\qquad = e^{-(t-t_0)}|x_0|^q + e^{-(t-t_0)}\sum_{n=1}^{\infty}(n-1)!^q\frac{C^{nq}}{n!}(t-t_0)^n,
\end{aligned}$$

which is finite for $q=1$, provided that $C(t-t_0)<1$. □

6 Numerical examples

In this section we consider numerical implementations of the Monte Carlo generation of Butcher trees for problems of the form (1).

i) Let $f(y) := e^y$, and consider the equation

$$\dot{x}(t) = e^{x(t)}, \quad x(0) = x_0, \quad t_0 = 0, \tag{27}$$

with solution

$$x(t) = -\log(e^{-x_0} - t), \qquad t \in [0, e^{-x_0}).$$

In this case, the moment bound (20) is sharp with $C = e^{x_0}$.

ii) Let $f(t,y) := yt + y^2$, and consider the equation

$$\dot{x}(t) = tx(t) + x^2(t), \qquad x(0) = 1/2, \quad t_0 = 0, \tag{28}$$

with solution

$$x(t) = \frac{e^{t^2/2}}{2 - \int_0^t e^{s^2/2}ds},$$

see Eq. (223a) in [2].

Table 1 displays the growth of computation times for the command `B[f,t,x0,t0,n]` applied to (27) with $x_0 = 1$, and to (28) with $x_0 = 1/2$, $n = 1, \ldots, 8$. For the purpose of benchmarking, all tree generations are performed using Mathematica.

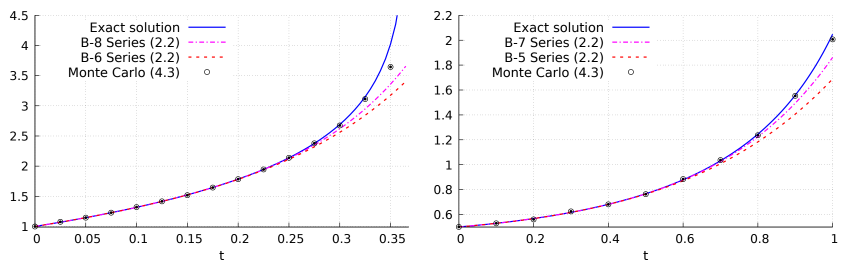
n	1	2	3	4	5	6	7	8	MC (Geometric)
Eq. (27), $d = 1$	0s	0s	0.1s	0.1s	0.4	0.5s	3s	21s	22s (70K samples)
Eq. (28), $d = 2$	0s	0s	0s	0.2s	1s	13s	222s	$>$ 1h	164s (10K samples)

Table 1: Computation times in seconds for (8) applied to (27) and (28).

Figure 2 compares the numerical solutions of (27) and (28) by the truncated Butcher series expansion

$$x(t) = \sum_{\substack{\tau \in \mathbf{T} \\ |\tau| \leq n}} \frac{(t-t_0)^{|\tau|}}{\tau! \sigma(\tau)} F(\tau)(x_0), \qquad t > t_0,$$

denoted by B-n, to the probabilistic representation (18), for different orders $n \geq 1$. The Monte Carlo estimations of (18) use the geometric distribution with respectively $70,000$ and $10,000$ samples, see Table 1, so that their runtimes are comparable to those of the Butcher series estimates. The solution of (27) is estimated using the above codes for one-dimensional ODEs, and the solution of (28) is estimated using the multidimensional codes presented in Section 7, after rewriting the non-autonomous ODE (28) as a two-dimensional autonomous system.



(a) Numerical solutions of (27) with $x_0 = 1$. (b) Numerical solutions of (28) with $x_0 = 1/2$.

Fig. 2: Comparisons of (8) *vs.* (18).

Next, we compare the performance of various probability distributions $(p_n)_{n\in\mathbb{N}}$ in terms of variance.

Variance analysis

(i) *Poisson distribution*. Taking $p_n := \lambda^n e^{-\lambda}/n!$, $n \geq 0$, to be the Poisson distribution with parameter $\lambda > 0$, the variance bound (20) is given by the series

$$\frac{x_0^2}{p_0} + \sum_{n=1}^{\infty} \frac{(C(t-t_0))^{2n}}{n^2 p_n} = \frac{x_0^2}{p} + e^{\lambda} \sum_{n=1}^{\infty} \left(\frac{C^2(t-t_0)}{\lambda} \right)^n \frac{(n-1)!}{n},$$

which diverges for all $t > t_0$.
(ii) *Geometric distribution*. Taking $p_n := (1-p)p^n$, $n \geq 0$, to be the geometric distribution with success probability $1-p$ for some $p \in [0,1)$, the variance bound (20) is given by the series

$$\frac{x_0^2}{p_0}+\sum_{n=1}^{\infty}\frac{(C(t-t_0))^{2n}}{n^2 p_n}=\frac{x_0^2}{1-p}+\frac{1}{1-p}\sum_{n=1}^{\infty}\frac{(C^2(t-t_0)^2/p)^n}{n^2}, \quad t\in[t_0,t_0+\sqrt{p}/C),$$

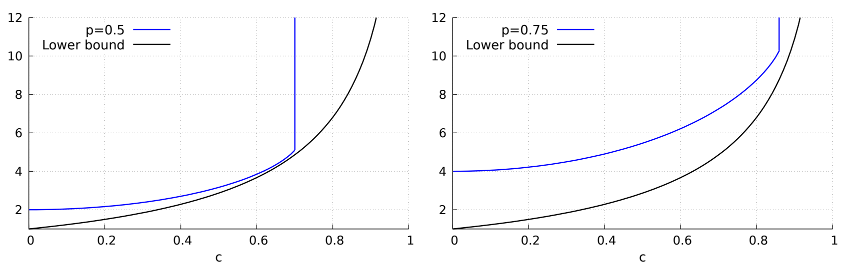
in which case the variance is finite.
(iii) *Optimal distribution*. Using the Lagrangian

$$\frac{x_0^2}{p_0}+\sum_{n=1}^{\infty}\frac{(Ct)^{2n}}{n^2 p_n}+\zeta\left(1-\sum_{n=0}^{\infty}p_n\right)$$

with multiplier ζ, we find that the distribution that minimizes the second moment bound (20) has the form

$$p_0=c_0x_0, \qquad p_n=c_0\frac{(Ct)^n}{n}, \qquad n\geq 1, \tag{29}$$

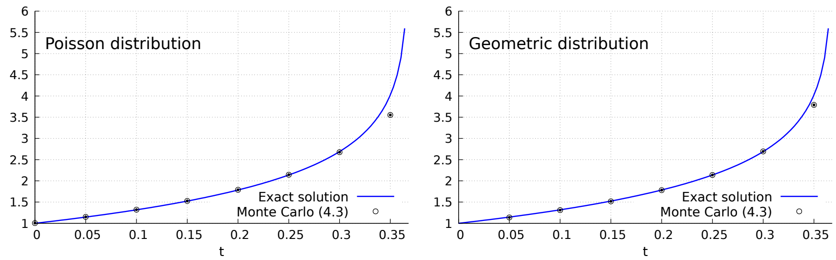
where $c_0=(x_0-\log(1-Ct))^{-1}$ is a normalization constant, see Figure 3 in which the moment bound (20) is plotted as a function of $C\in[0,\sqrt{p}]$ with $t=1$ for the distribution (29) (lower bound) and for the geometric distributions with parameters $p=0.5,0.75$, and $x_0=1$.



(a) $p=0.5$. (b) $p=0.75$.

Fig. 3: Second moment lower bound.

The graphs of Figure 4 are plotted using the Poisson and geometric distributions with respectively 100,000 and 70,000 Monte Carlo samples, in order to match the 22 seconds computation time of Figure 2-(a) for (27), see Table 1.



(a) Poisson tree size. (b) Geometric tree size.

Fig. 4: Numerical solution of (27) by the Monte Carlo method (18).

7 Multidimensional codes

The next Mathematica code estimates the Butcher series (8) up to a given order $n \geq 1$ in the multidimensional case. The second component in the output of `B[f,t,x0,t0,n]` counts the number of trees involved in the Butcher series truncated up to the order $n \geq 1$.

```
B[f_, t_, x0__, t0_, n_] := (d = Length[x0];
 If[n == 0, Return[{x0, 1}],
  If[n == 1, Return[{x0 + (t - t0)*f[x0], 2}], count = 2;
   sample = x0 + (t - t0)*f[x0];
   g = ConstantArray[Graph[{1 -> 2}], d]; ii = Array[i, n];
   For[ii[[1]] = 1, ii[[1]] <= d, ii[[1]]++,
    g[[ii[[1]]]] =
     Graph[g[[ii[[1]]]],
      VertexLabels -> {1 -> D[f[yy], yy[[ii[[1]]]]]}];
    g[[ii[[1]]]] =
     Graph[g[[ii[[1]]]], VertexLabels -> {2 -> f[yy][[ii[[1]]]]}];
    m = 1; count += 1;
    sample += 1/2*(t - t0)^VertexCount[g[[ii[[1]]]]]*
       Product[ff[[2]] , {ff,
         List @@@
          PropertyValue[g[[ii[[1]]]], VertexLabels]}] /. {yy ->
         x0}]; list = g;
   While[m <= (n - 2), temp = list; list = {};
    Do[l = VertexCount[g];
     For[j = 1, j <= l, j++, gg = VertexAdd[g, {l + 1}];
      lab = Sort[List @@@ PropertyValue[gg, VertexLabels]][[j]][[
        2]]; For[ii[[l]] = 1, ii[[l]] <= d, ii[[l]]++,
       gg = Graph[gg,
         VertexLabels -> {l + 1 -> f[ yy][[ii[[l]]]]}];
       gg = Graph[gg, VertexLabels -> {j -> D[lab, yy[[ii[[l]]]]]}];
        gg = EdgeAdd[gg, j -> l + 1];
       GraphPlot[gg,
        PlotStyle -> {FontSize -> 20, FontColor -> Red}];
       count += 1;
       sample += (t - t0)^(l + 1)/(l + 1)!*
          Product[ff[[2]] , {ff,
            List @@@ PropertyValue[gg, VertexLabels]}] /. {yy ->
           x0}; list = Append[list, gg]]], {g, temp}]; m++];
   Return[{sample, count}]]]);
```

```
x0 = {0, 0.5}; t0 = 0; t1 = 1.3885;
f[y__] := {1, y[[1]]*y[[2]] + y[[2]]^2}
B[f, t, x0, t0, 4]
```

The next Mathematica code generates a single random Butcher tree sample in (18) for a multidimensional ODE.

```
 MCsample[f_, t_, x0_, t0_, dist_] := (d = Length[x0];
  n = RandomVariate[dist];
  If[n == 0, Return[x0/PDF[dist, 0]],
   If[n == 1, Return[(t - t0)*f[x0]/PDF[dist, 1]],
    g = ConstantArray[Graph[{1 -> 2}], d]; ii = Array[i, n];
    sample = 0;
    For[ii[[1]] = 1, ii[[1]] <= d, ii[[1]]++,
     g[[ii[[1]]]] =
      Graph[g[[ii[[1]]]],
       VertexLabels -> {1 -> D[f[yy], yy[[ii[[1]]]]]}];
     g[[ii[[1]]]] =
      Graph[g[[ii[[1]]]], VertexLabels -> {2 -> f[yy][[ii[[1]]]]}];
     sample += (t - t0)^2/PDF[dist, 2]/2*
        Product[ff[[2]] , {ff,
          List @@@
           PropertyValue[g[[ii[[1]]]], VertexLabels]}] /. {yy ->
         x0}]; If[n == 2, Return[sample]]; sample = 0; list = g;
    m = 1; While[m <= (n - 2), temp = list; list = {};
     Do[l = VertexCount[g];
      j = RandomVariate[DiscreteUniformDistribution[{1, l}]];
      gg = VertexAdd[g, {l + 1}];
      lab = Sort[List @@@ PropertyValue[gg, VertexLabels]][[j]][[2]];
       For[ii[[l]] = 1, ii[[l]] <= d, ii[[l]]++,
       gg = Graph[gg, VertexLabels -> {l + 1 -> f[ yy][[ii[[l]]]]}];
       gg = Graph[gg, VertexLabels -> {j -> D[lab, yy[[ii[[l]]]]]}];
       gg = EdgeAdd[gg, j -> l + 1];
       GraphPlot[gg,
        PlotStyle -> {FontSize -> 20, FontColor -> Red}];
       If[m == (n - 2),
        sample +=
         Product[ff[[2]] , {ff,
            List @@@ PropertyValue[gg, VertexLabels]}] /. {yy ->
            x0}]; list = Append[list, gg]], {g, temp}]; m++];
    Return[sample*(t - t0)^n/PDF[dist, n]/n]]]);
x0 = {0, 0.5}; t0 = 0; t1 = 1.3885;
f[y__] := {1, y[[1]]*y[[2]] + y[[2]]^2}
MCsample[f, t, x0, t0, GeometricDistribution[0.5]]
```

Acknowledgements This research is supported by the Ministry of Education, Singapore, under its Tier 1 Grant RG103/23. The work of Q. Huang is supported by the Start-Up Research Fund of Southeast University under Grant No. RF1028624194 and the Jiangsu Provincial Scientific Research Center of Applied Mathematics under Grant No. BK20233002.

References

1. J.C. Butcher. Coefficients for the study of Runge-Kutta integration processes. *J. Austral. Math. Soc.*, 3:185–201, 1963.
2. J.C. Butcher. *Numerical methods for ordinary differential equations*. John Wiley & Sons, Ltd., Chichester, third edition, 2016.
3. J.C. Butcher. *B-Series: Algebraic Analysis of Numerical Methods*, volume 55 of *Springer Series in Computational Mathematics*. Springer, Cham, 2021.

4. F. Chapoton and M. Livernet. Pre-Lie algebras and the rooted trees operad. *Internat. Math. Res. Notices*, 8:395–408, 2001.
5. G.M. Constantine and T.H. Savits. A multivariate Faa di Bruno formula with applications. *Trans. Amer. Math. Soc.*, 348(2):503–520, 1996.
6. R.C. Dalang, C. Mueller, and R. Tribe. A Feynman-Kac-type formula for the deterministic and stochastic wave equations and other P.D.E.'s. *Trans. Amer. Math. Soc.*, 360(9):4681–4703, 2008.
7. R. Dascaliuc, N. Michalowski, E. Thomann, and E.C. Waymire. Complex Burgers equation: a probabilistic perspective. In *Sojourns in probability theory and statistical physics. I. Spin glasses and statistical mechanics, a Festschrift for Charles M. Newman*, volume 298 of *Springer Proc. Math. Stat.*, pages 138–170. Springer, Singapore, 2019.
8. R. Dascaliuc, T.N. Pham, E. Thomann, and E.C. Waymire. Doubly stochastic Yule cascades (Part I): The explosion problem in the time-reversible case. *J. Funct. Anal.*, 284(1):Paper No. 109722, 25, 2023.
9. R. Dascaliuc, T.N. Pham, E. Thomann, and E.C. Waymire. Doubly stochastic Yule cascades (part II): The explosion problem in the non-reversible case. *Ann. Inst. Henri Poincaré Probab. Stat.*, 59(4):1907–1933, 2023.
10. P. Deuflhard and F. Bornemann. *Scientific Computing with Ordinary Differential Equations*, volume 42 of *Texts in Applied Mathematics*. Springer-Verlag, New York, 2002.
11. E. Hairer, C. Lubich, and G. Wanner. *Geometric numerical integration*, volume 31 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, second edition, 2006.
12. E. Hairer, S.P. Nørsett, and G. Wanner. *Solving ordinary differential equations. I*, volume 8 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, second edition, 1993. Nonstiff problems.
13. P. Henry-Labordère, N. Oudjane, X. Tan, N. Touzi, and X. Warin. Branching diffusion representation of semilinear PDEs and Monte Carlo approximation. *Ann. Inst. H. Poincaré Probab. Statist.*, 55(1):184–210, 2019.
14. M. Hochbruck and A. Ostermann. Exponential integrators. *Acta Numer.*, 19:209–286, 2010.
15. N. Ikeda, M. Nagasawa, and S. Watanabe. Branching Markov processes I, II, III. *J. Math. Kyoto Univ.*, 8-9:233–278, 365–410, 95–160, 1968-1969.
16. D.I. Ketcheson and H. Ranocha. Computing with B-series. *ACM Transactions on Mathematical Software*, 12 2022.
17. Y. Le Jan and A. S. Sznitman. Stochastic cascades and 3-dimensional Navier-Stokes equations. *Probab. Theory Related Fields*, 109(3):343–366, 1997.
18. V.T. Luan and A. Ostermann. Exponential B-series: The stiff case. *SIAM Journal on Numerical Analysis*, 51(6):3431–3445, 2013.
19. H.P. McKean. Application of Brownian motion to the equation of Kolmogorov-Petrovskii-Piskunov. *Comm. Pure Appl. Math.*, 28(3):323–331, 1975.
20. J.Y. Nguwi, G. Penent, and N. Privault. A fully nonlinear Feynman-Kac formula with derivatives of arbitrary orders. *Journal of Evolution Equations*, 23:Paper No. 22, 29pp., 2023.
21. G. Penent and N. Privault. Numerical evaluation of ODE solutions by Monte Carlo enumeration of Butcher series. *BIT Numerical Mathematics*, 62:1921–1944, 2022.
22. A.V. Skorokhod. Branching diffusion processes. *Teor. Verojatnost. i. Primenen.*, 9:492–497, 1964.
23. V.A. Vatutin. Branching processes and random trees. In H. Niemi, G. Högnas, A. N. Shiryaev, and A. V. Melnikov, editors, *Proceedings of the Third Finnish-Soviet symposium on probability theory and mathematical statistics. Turku, Finland, August 13-16, 1991*, volume 1 of *Frontiers in Pure and Applied Probability*, pages 256–268. De Gruyter, Utrecht: VSP; Moscow: TVP, 1993.
24. V.A. Vatutin and A.M. Zubkov. Branching processes. II. *J. Soviet Math.*, 67(6):3407–3485, 1993.