Journal of Computational Physics **164,** 335–353 (2000)
doi:10.1006/jcph.2000.6599, available online at http://www.idealibrary.com on IDEAL®

A New Algorithm and Worst Case Complexity for Feynman–Kac Path Integration

Leszek Plaskota,[*] Grzegorz W. Wasilkowski,[†] and Henryk Woźniakowski[*,‡]

[*]*Institute of Applied Mathematics and Mechanics, University of Warsaw, ul. Banacha 2, 02-097 Warsaw, Poland;* [†]*Department of Computer Science, University of Kentucky, Lexington, Kentucky 40506; and* [‡]*Department of Computer Science, Columbia University, New York, New York 10027*
E-mail: leszekp@mimuw.edu.pl, greg@cs.uky.edu, henryk@cs.columbia.edu

Received July 5, 2000; revised August 8, 2000

We study algorithms for approximation of Feynman–Kac path integrals in the worst case setting. The algorithms use a finite number of samples of the initial condition and potential functions. We present a new algorithm and an explicit bound on its cost to compute an ε-approximation to the Feynman–Kac path integral. We also establish bounds on the worst case complexity of Feynman–Kac path integration. The upper bound is equal to the cost of the new algorithm and is given in terms of the complexity of a certain function approximation problem. The lower bound is given in terms of the complexity of a certain weighted integration problem. For some classes of functions, these two bounds coincide modulo a multiplicative factor. In this case, the new algorithm is almost optimal. The new algorithm requires precomputation of some real coefficients that are combinations of multivariate integrals with special weights. This precomputation is difficult and limits the application of the new algorithm. We report the results of the precomputation for specific cases.

Key Words: Feynman–Kac path integration; Wiener measure; multivariate approximation; worst case complexity.

1. INTRODUCTION

Path integrals are defined as integrals over an infinite dimensional space with respect to a probability measure. For the space C of continuous functions and the Wiener measure w, path integrals are called *Wiener integrals*. Wiener integrals play a major role in many areas including quantum physics and chemistry, differential equations, and financial mathematics.

An example of a Wiener integral is given by the famous *Feynman–Kac path integral*, which gives the solution of the heat equation

$$\frac{\partial z}{\partial t}(u,t) = \frac{1}{2}\frac{\partial^2 z}{\partial u^2}(u,t) + V(u)\,z(u,t), \tag{1}$$

$$z(u,0) = v(u), \tag{2}$$

where $u \in \mathbb{R},\ t > 0,\ V$ is a potential function, and v is an initial condition function. Then the solution is the Feynman–Kac path integral of the form

$$z(u,t) = \int_C v(x(t)+u)\exp\left(\int_0^t V(x(s)+u)\,ds\right)w(dx). \tag{3}$$

Approximate evaluations of Wiener integrals was initiated by Cameron [5]. The reader may find many papers on this subject in [7]. Additionally, this subject has been studied in [6, 9, 11]. A typical approach is to replace the continuous function x by $x_1\psi_1 + \cdots + x_n\psi_n$ for some continuous functions ψ_i and real numbers x_i, $i = 1, 2 \ldots, n$. Then the Feynman–Kac path integral is approximated by a multivariate integral over the space $\mathbb{R}^n$. With a proper choice of the functions ψ_i, the error goes to zero as n tends to infinity. For instance, Chorin [6] and Hald [11] proved that the error is of order n^{-2} assuming that $v = 1$ and V is four times continuously differentiable.

In this way, the approximate computation of the Feynman–Kac path integral can be reduced to the approximate computation of an integral over the whole space $\mathbb{R}^n$. The cost of computing the integrand value of the n-dimensional integral is proportional to n. This integral is usually computed by a Monte Carlo algorithm with, if possible, variance reduction. The randomized error of the Monte Carlo algorithm is of order $k^{-1/2}$, where k denotes the number of computed integrand values. The total cost is then proportional to nk.

We add that the order of convergence of the Monte Carlo algorithm cannot be significantly improved. Indeed, Bakhvalov [4] proved that the minimal randomized error is of order $k^{-(r/n+1/2)}$, where r denotes the smoothness of the integrands. Since n is usually much larger than r, the minimal randomized error is roughly of order $k^{-1/2}$.

Suppose we want to compute an approximation with error ε. This can be achieved by using Chorin's algorithm (for $v = 1$ and four times continuously differentiable functions V) with $n = O(\varepsilon^{-1/2})$ and $k = O(\varepsilon^{-2})$. The total cost[1] is nk which is of order $\varepsilon^{-2.5}$.

In this paper, we propose a different approach to approximation of Feynman–Kac path integrals. Instead of reducing (3) to multivariate integrals, we consider algorithms that make the most efficient use of values of v and V at a finite number of points, and of some other real coefficients independent of v and V. Furthermore, we are interested in the *worst case* setting. That is, unlike in the setting of Monte Carlo algorithms, we do not allow randomization, and we compute v and V at deterministically chosen points. In the worst case setting, the error and cost of an algorithm are defined by its *worst* performance over all functions v and V from a given class.

It is natural to ask what is the worst case complexity of Feynman–Kac path integration. The worst case complexity is defined as the minimum of the number of the values of v and V plus the number of arithmetic operations needed to compute and ε-approximation to the

[1] The heat equation (1) can also be approximated by using the classical approach of finite differences. It is known that if the solution z is twice continuously differentiable with respect to the time variable t and four times continuously differentiable with respect to the space variable u (which holds if v and V are four times continuously differentiable) then the solution $z(u,t)$ can be approximated with error ε and with the total cost of order ε^{-2}. Note that the finite difference approach yields a deterministic algorithm and the worst case assurance of its error. Hence, the finite difference algorithm seems a little better than Chorin's algorithm. However, Chorin's algorithm can be applied to Eq. (4) with an arbitrary smooth H, whereas the finite difference algorithm requires $H(u) = \exp(u)$.

Feynman–Kac path integral (3). (The complexity study of general path integrals may be found in [20].)

In particular, we ask whether the bound $\varepsilon^{-2.5}$ for the randomized case setting for four times continuously differentiable function[2] can be improved. As we shall see, even though we switch to the worst case setting, there exists an algorithm which computes an ε-approximation at cost of order roughly $\varepsilon^{-0.25}$ and the worst case complexity is of the same order. Hence, the exponent of ε^{-1} is an order of magnitude smaller; i.e., we gain an exponential reduction in the cost. We stress, however, that our proofs require an additional assumption that the functions v and V decay sufficiently fast.

Before we state our results more precisely, we note that by shifting the initial condition and potential functions, we may, without loss of generality, assume that the space parameter $u = 0$ in (3). Furthermore, as in [6, 11], we consider in this paper a slight generalization of the Feynman–Kac path integrals by changing the exponent function to a function H. Unlike in [6, 11], we assume that H is an entire function.[3] That is, we study approximate computations of

$$S(v, V) = \int_C v(x(t)) H\left(\int_0^t V(x(s))\, ds \right) w(dx) \tag{4}$$

for various functions v and V from a given class, and for a fixed positive t.

Observe that S is a functional which depends *linearly* on the function v, and, in general, *nonlinearly* on the function V. The nonlinear dependence on V is especially important.

We assume that v and V belong to a normed linear space F of functions defined over $\mathbb{R}$. We want to compute an ε-approximation to $S(v, V)$ for all v and V from F that are uniformly bounded in the norm of the space F. We stress that we do not need to specify the space F. That is, specific results depend on F, but our analysis is valid for any space F.

For a given space F, we present a new algorithm A_ε which computes an ε-approximation of the Feynman–Kac path integral S given by (4). It is based on Smolyak's algorithm for multivariate tensor products; see, e.g., [19]. The algorithm A_ε uses the values of v and V at deterministically chosen points derived from a certain weighted approximation problem for the class F. More precisely, we approximate v (and V) by linear combinations $v(t_{1,\varepsilon})g_{1,\varepsilon} + \cdots + v(t_{n,\varepsilon})g_{n,\varepsilon}$, where the sample points $t_{i,\varepsilon}$ and the function $g_{i,\varepsilon}$ are chosen in a special way.

The algorithm A_ε depends polynomially on the computed values of v and V. The degree of this polynomial depends on ε, and slowly goes to infinity as ε tends to zero. The need to use a nonlinear algorithm is not surprising since the original problem is nonlinear in V.

We provide an explicit bound on the cost of the algorithm A_ε. By the cost we mean the total number of the computed function values of v and V plus all arithmetic operations needed to compute $A_\varepsilon(v, V)$. The essence of this bound is that the cost of A_ε is roughly the same as the cost of approximating the functions v and V from the class F to within ε. In this way, we prove that the Feynman–Kac path integration problem is not essentially harder than the corresponding approximation problem.

We illustrate the behavior of the algorithm A_ε for various subclasses of $C^r(\mathbb{R})$. For these subclasses, the cost of A_ε is roughly of order $\varepsilon^{-1/r}$. Furthermore, the algorithm A_ε is now almost optimal. This should be compared to the cost of order $n\varepsilon^{-2}$ when the

[2] Or the bound ε^{-2} for the worst case setting with the finite difference approach and H restricted to exp.

[3] If H is not entire but smooth, one may replace H by a polynomial P which approximates H such that $|H(u) - P(u)| \leq \varepsilon\rho(u)$, $\forall u \in \mathbb{R}$, where the weight ρ is, for example, equal to $\rho(u) = \exp(u)$.

classical Monte Carlo algorithm is used, where n, as before, denotes the dimension of the reduced multivariate integral. Hence, despite the use of the worst case setting, we have a big improvement.

The algorithm A_ε requires the precomputation of certain real coefficients which are given as some weighted multivariate integrals. These coefficients are independent of the functions $v, V \in F$, but they depend on the global parameters of the Feynman–Kac path integration problem such as ε, H, t, and the class F. The total number of them is roughly equal to the cost of the algorithm A_ε, and it goes to infinity as ε tends to zero. Observe that any implementation of the algorithm A_ε can use these coefficients as built in elements (constants), hence the precomputing does not count in the cost analysis of A_ε.

The precomputing relies on calculation of a large number of multivariate integrals. Hence it is not surprising that it itself is a difficult task. We did three different precomputations for classes F of functions of regularities $r = 1$, 2, and 4. We used Monte Carlo to calculate the integrals. The precomputed coefficients were evaluated with error of order 10^{-4}, see Section 5. This obviously limits the application of the algorithm A_ε to a relatively large ε. We believe that the precomputing of these coefficients requires the design of a special algorithm that takes a full advantage of the specific form of the weighted multivariate integrals to be calculated. Unless we have such an efficient algorithm, the practical use of A_ε will be most likely restricted to moderate accuracy ε. However, if one is interested in rapid approximation of many path integrals with a moderate accuracy then the precomputation is not an issue and the algorithm A_ε can be used very efficiently.

In this paper we also study the worst case complexity of Feynman–Kac path integration. The upper bound on the complexity is obviously supplied by the cost of the algorithm A_ε. As already mentioned, this upper bound is given in terms of a certain approximation problem. We establish a lower bound by showing that Feynman–Kac path integration is not easier than a certain weighted scalar integration problem. The weight is now a one-dimensional Gaussian, $(2\pi t)^{-1/2} \exp(-u^2/(2t))$, and the weighted integration problem is considered again for the class F.

Hence, if the weighted approximation and integration problems for the class F have essentially the same complexity then we have tight bounds on the complexity of Feynman–Kac path integration. This is the case, modulo multiplicative factors, for various subclasses of $C^r(\mathbb{R})$ for any r. For $r = 0$, the weighted integration problem has infinte complexity for small ε, and so does Feynman–Kac path integration. For $r \geq 1$, all three complexities are essentially of order $\varepsilon^{-1/r}$. In this case, the algorithm A_ε is almost optimal.

Finally we stress that, although Feynman–Kac path integration is nonlinear, we bound its complexity by the complexities of two *linear* problems. From below, it is bounded by weighted integration, and from above by approximation. This is essential since it allows us to use the complexity theory of linear problems; see, for example, [13–15, 17] for integration and [14, 17] for approximation. What is perhaps even more striking is that both weighted integration and approximation are *one-dimensional* problems although Feynman–Kac path integration involves integrals over an infinite-dimensional space.

2. FORMULATION OF THE PROBLEM

Let w be the classical Wiener measure defined on the space $C = C([0, +\infty])$ of continuous functions $x : [0, +\infty) \to \mathbb{R}$. For the definition and basic properties of the Wiener measure we refer the reader to, e.g., [18]. Here, we only recall that w is a zero-mean Gaussian measure with the covariance function

$$R_w(s,t) = \int_C x(s)x(t)w(dx) = \min\{s,t\}.$$

Let H be an entire real function, $H : \mathbb{R} \to \mathbb{R}$, normalized such that $H(0) = 1$. We assume that the successive derivatives of H at zero can be easily computed. A primary example of such a function is $H(t) = \exp(t)$. For a given function H, we define the function H^*: $\mathbb{R} \to \mathbb{R}_+$ by

$$H^*(t) = \sum_{d=1}^{\infty} \frac{\left|H^{(d)}(0)\right|}{d!} |t|^d.$$

Clearly, $|H(u)| \le H^*(u)$ and H^* is an increasing function over $[0,+\infty)$. For $H(t) = \exp(t)$ we have $H^* = H$ over $[0,+\infty)$.

We need an upper bound on the coefficients of H^*. For simplicity, we assume that there exists a nonnegative constant $\mathcal{H}$ such that

$$\left|H^{(d)}(0)\right| \le \mathcal{H}^d, \quad \forall d = 0,1,\ldots. \tag{5}$$

Then (5) implies that

$$|H(u)| \le H^*(u) \le \exp(\mathcal{H}|u|).$$

Let F be a normed linear space of functions defined over $\mathbb{R}$. The norm of F will be denoted by $\|\cdot\|_F$. We assume that for every $u \in \mathbb{R}$ the function evaluation functional $L_u(f) = f(u)$ is continuous; i.e., it has a finite (operator) norm $\|L_u\|_F$.

We need an additional assumption that relates the Wiener measure w, the function H, and the space F. Namely, we assume that for any positive t and a we have

$$\int_C \left\|L_{x(t)}\right\|_F H^*\left(a \int_0^t \left\|L_{x(s)}\right\|_F ds\right) w(dx) < \infty. \tag{6}$$

Note that (6) holds if $\|L_t\|_F = O(t^\alpha)$ with $\alpha < 2$ as t tends to infinity. Indeed, in this case the integrand in (6) is bounded by $\alpha_1 \|x\|_{[0,t]}^\alpha \exp(\alpha_2 \|x\|_{[0,t]}^\alpha)$ for some α_1 and α_2 which only depend on t, α, and $\mathcal{H}$, with $\|x\|_{[0,t]} = \max_{u\in[0,t]} |x(u)|$. Since $\alpha < 2$, the integrand is bounded by $\alpha_3 \exp(\alpha_4 \|x\|_{[0,t]}^{2-\delta})$ for some positive δ, α_3, and α_4. The latter function is integrable due to Fernique's theorem; see, e.g., [2].

We illustrate these assumptions by the following examples of the space F.

EXAMPLE 1. Let F be the following Banach space of functions $f : \mathbb{R} \to \mathbb{R}$. Each f has absolutely continuous $(r-1)$st derivative, $f(0) = \cdots = f^{(r-1)}(0) = 0$, and its rth derivative is bounded in the following sense:

$$\|f\|_F := \left\|(f \cdot \psi_1)^{(r)} \cdot \psi_2\right\|_{L_p(\mathbb{R})} < \infty.$$

Here $1 \le p \le \infty$ and ψ_i, $i = 1,2$, are given weight functions of the form

$$\psi_i(x) = (|x|+1)^{a_i} \tag{7}$$

with $a_1 > 0$ and $a_2 \in \mathbb{R}$. Since $f \cdot \psi_1$ has only one-sided derivatives at $x = 0$, by the norm $\|\cdot\|_{L_p(\mathbb{R})}$ we formally mean $(\|\cdot\|^p_{L_p(\mathbb{R}_-)} + \|\cdot\|^p_{L_p(\mathbb{R}_+)})^{1/p}$.

Obviously, $\|\cdot\|_F$ is a norm. Furthermore, function evaluations, $L_x(f) = f(x)$, are continuous functionals. Indeed, for a positive x, we have

$$\begin{aligned} |f(x)| &= (\psi_1(x))^{-1}\left| \int_0^\infty \frac{(x-t)_+^{r-1}}{(r-1)!} \cdot (f \cdot \psi_1)^{(r)}(t) \cdot \frac{\psi_2(t)}{\psi_2(t)}\, dt \right| \\ &\leq (\psi_1(x))^{-1} \cdot \|f\|_F \cdot A(x) \end{aligned}$$

with

$$A(x) = \left(\int_0^x \left(\frac{(x-t)^{r-1}}{(r-1)!} \cdot (t+1)^{-a_2} \right)^q dt \right)^{1/q}$$

and q such that $1/p + 1/q = 1$. It is easy to see that $A(x) = O(x^{r-1-a_2+1/q}) = O(x^{r-1/p-a_2})$. Since the same can be shown for negative x, we conclude that

$$\|L_x\|_F = O\big(|x|^{r-1/p-a_1-a_2}\big) \quad \text{as } |x| \to \infty. \tag{8}$$

Hence (6) is satisfied if $r - 1/p - a_1 - a_2 < 2$.

EXAMPLE 2. Let F be the Banach space of functions $f : \mathbb{R} \to \mathbb{R}$ with continuous and bounded derivatives up to order r, and

$$\|f\|_F = \max\big\{\|f \cdot \psi\|_\infty, \big\|f^{(i)}\big\|_\infty, 1 \leq i \leq r\big\}.$$

Here $\psi \in C(\mathbb{R})$ is a suitably chosen function with values at least 1. For instance, $\psi \equiv 1$, or $\psi(u) = (1 + |u|)^a$ with $a \geq 0$, or $\psi(u) = \exp(au^2)$ with $a \geq 0$. Then $\|L_x\|_F \leq 1/\psi(x) \leq 1$ for any $x \in \mathbb{R}$, and (6) holds.

EXAMPLE 3. Let F be a reproducing kernel Hilbert space whose kernel is denoted by $K(\cdot,\cdot)$. For the definition and basic properties of reproducing kernel Hilbert spaces see, e.g., [3]. We only recall that for any $f \in F$ and any $t \in \mathbb{R}$ we have $K(t,\cdot) \in F$ and $f(t) = \langle f, K(t,\cdot)\rangle_F$, $\forall t \in \mathbb{R}$. In particular, $K(t,t) \geq 0$ and $\|L_t\|_F = \|K(t,\cdot)\|_F = \sqrt{K(t,t)}$. Hence, (6) holds if $K(t,t) = O(t^\beta)$ with $\beta < 4$ as t tends to infinity.

To be more specific, consider now $K(u,t) = \frac{1}{2}\exp(-|u-t|)$. This kernel corresponds to the space F consisting of functions f with

$$\|f\|_F^2 = \langle f, f\rangle_F = \int_{\mathbb{R}} ((f(t))^2 + (f'(t))^2)\, dt < \infty.$$

Then $K(t,t) = 1/2$ and (6) holds.

We are ready to define our problem. For $v, V \in F$ and a fixed positive t we want to approximate

$$S(v,V) = S(v,V;t) = \int_C v(x(t)) H\left(\int_0^t V(x(s))\, ds\right) w(dx). \tag{9}$$

Observe that the operator $S : F^2 \times \mathbb{R}_+ \to \mathbb{R}$ is well defined. Indeed, due to $|v(u)| \le \|v\|_F \|L_u\|_F$, we have

$$\left| v(x(t)) H\left(\int_0^t V(x(s))\, ds \right) \right| \le |v(x(t))|\, H^*\left(\left| \int_0^t V(x(s))\, ds \right| \right)$$
$$\le \|v\|_F \left\| L_{x(t)} \right\|_F H^*\left(\|V\|_F \int_0^t \left\| L_{x(s)} \right\|_F ds \right).$$

Hence, the integral in (9) exists and is finite due to (6).

The operator S is given by a *path* integral, i.e., by an integral over the space of continuous functions. In general, S is *nonlinear*.

As already mentioned in the introduction, for $H(u) = \exp(u)$, (9) is the famous Feynman–Kac formula for the solution of the heat equation with the initial condition v and the potential function V, see, e.g., [1, 8, 12]. This is why we call (9) a *Feynman–Kac path integral* independent of the choice of the function H.

We wish to approximate S with (worst case) error at most ε for various v and V. In what follows, we assume that we know some upper bounds on their norms. For positive ν and $\mathcal{V}$, let ν and $\mathcal{V}$ denote upper bounds on the norms of v and V, and let

$$F_{\nu,\mathcal{V}} = \{(v, V) \in F^2 : \|v\|_F \le \nu \quad \text{and} \quad \|V\|_F \le \mathcal{V}\}.$$

We wish to find an algorithm A_ε such that

$$\sup_{(v,V)\in F_{\nu,\mathcal{V}}} |S(v, V) - A_\varepsilon(v, V)| \le \varepsilon.$$

We require that the algorithm A_ε can use only *values* of the functions v and V, as well as arithmetic operations and comparisons of real numbers. The cost of the algorithm A_ε, $\mathrm{cost}(A_\varepsilon)$, is the number of such function values and arithmetic operations. The (worst case) complexity $\mathrm{comp}(\varepsilon)$ of Feynman–Kac path integration is defined as the minimal cost of algorithms with error ε; see, e.g., [17] for a precise definition. Obviously, the complexity $\mathrm{comp}(\varepsilon)$ depends on all the parameters of the problem, i.e., on the function H and its bound $\mathcal{H}$ given by (5), on the parameter t, and on the class F, as well as the norm bounds ν and $\mathcal{V}$. To stress the dependence on the class F, we will write $\mathrm{comp}(\varepsilon) = \mathrm{comp}(\varepsilon; F)$.

3. NEW ALGORITHM

In this section, we derive a new algorithm for approximating Feynman–Kac path integrals. We analyze its error and estimate the cost needed to compute an ε-approximation. The optimality properties of this new algorithm will be established in the next section.

We begin with expanding the function H in (9). We have

$$S(v, V) = \int_C v(x(t)) H\left(\int_0^t V(x(s))\, ds \right) w(dx) = \sum_{d=0}^{\infty} S_{d+1}(v, V),$$

where

$$S_{d+1}(v, V) = \frac{H^{(d)}(0)}{d!} \int_C v(x(t)) \left(\int_0^t V(x(s))\, ds \right)^d w(dx)$$
$$= \frac{H^{(d)}(0)}{d!} \int_C v(x(t)) \left(\int_{[0,t]^d} V(x(t_1)) \cdots V(x(t_d))\, d\mathbf{t} \right) w(dx)$$

with $\mathbf{t} = [t_1, t_2, \ldots, t_d]$. The inner integrand is a symmetric function of t_i's and, hence,

$$S_{d+1}(v, V) = H^{(d)}(0) \int_C v(x(t)) \int_{0 \le t_1 \le \cdots \le t_d \le t} V(x(t_1)) \cdots V(x(t_d))\, d\mathbf{t}\, w(dx).$$

Set $t_0 = z_0 = 0$, $t_{d+1} = t$, and $z_j = x(t_j)$ for $j = 1, \ldots, d+1$. Using a well-known fact on the distribution of the random vector $\mathbf{z} = [z_1, z_2, \ldots, z_{d+1}] \in \mathbb{R}^{d+1}$, we get

$$S_{d+1}(v, V) = H^{(d)}(0) \int_{\mathbb{R}^{d+1}} f_{d+1}(\mathbf{z}) g_{d+1}(\mathbf{z})\, d\mathbf{z}, \tag{10}$$

where

$$f_{d+1}(\mathbf{z}) = v(z_{d+1}) \prod_{k=1}^{d} V(z_k), \tag{11}$$

$$\begin{aligned} g_1(z_1) &= \frac{\exp\left(-z_1^2/(2t)\right)}{(2\pi t)^{1/2}}, \\ g_{d+1}(\mathbf{z}) &= \int_{0 \le t_1 \le t_2 \le \cdots \le t_d \le t_{d+1}} \frac{\exp\left(-\frac{1}{2} \sum_{j=1}^{d+1} (z_j - z_{j-1})^2/(t_j - t_{j-1})\right)}{(2\pi)^{(d+1)/2} \sqrt{(t_1 - t_0) \cdots (t_{d+1} - t_d)}}\, d\mathbf{t}. \end{aligned} \tag{12}$$

Hence, $S_{d+1}(v, V)$ is a $(d+1)$-dimensional weighted integral with the weight g_{d+1}. The weight g_{d+1} is nonnegative, and

$$\int_{\mathbb{R}} g_1(z)\, dz = 1 \quad \text{and} \quad \int_{\mathbb{R}^{d+1}} g_{d+1}(\mathbf{z})\, d\mathbf{z} = \int_{0 \le t_1 \le t_2 \le \cdots \le t_d \le t_{d+1}} d\mathbf{t} = \frac{t^d}{d!}.$$

We will need the following lemma.

LEMMA 1. *We have*

$$\|g_1\|_{L_2(\mathbb{R})}^2 = \frac{1}{2\sqrt{\pi t}} \quad \textit{and} \quad \|g_{d+1}\|_{L_2(\mathbb{R}^{d+1})}^2 \le \frac{t^{(3d-1)/2}}{2\Gamma(1 + (3d-1)/2)}, \quad \forall d \ge 1.$$

Proof. See the Appendix. ■

To define the new algorithm for approximating the Feynman–Kac path integrals, we need efficient algorithms for approximating multivariate functions f_d given in (10) and (11) with error measured in the d-dimensional L_2-norm:

$$\| f_d \|_{L_2(\mathbb{R}^d)} = \sqrt{\int_{\mathbb{R}^d} f_d^2(\mathbf{t})\, d\mathbf{t}}.$$

It follows from [19] that such efficient algorithms exist for a number of classes F.

Indeed, since f_d is a tensor product of scalar functions and $\| \cdot \|_{L_2(\mathbb{R}^d)}$ is a tensor product norm, Lemma 2 of [19] holds. We need only to replace the worst case error in (19) of [19] by the error for the function f_d. Due to linearity of the corresponding algorithms, this will result in increasing the error bound ε by multiplying it by the norm of f_d; i.e., instead of ε, the error will be bounded by $\varepsilon \|v\|_F \|V\|_F^{d-1}$. Furthermore, all remaining worst case results of [19] hold if the worst case errors are replaced by the errors for the functions of the form f_d. In particular, for $\varepsilon > 0$ and $d \geq 1$, there are numbers $n(\varepsilon, d) \in \mathbb{N}_+$, functions $g_{i,\varepsilon,d} \in L_2(\mathbb{R}^d)$ and points $\mathbf{t}_{i,\varepsilon,d} \in \mathbb{R}^d$, $i = 1, \ldots, n(\varepsilon, d)$, such that the algorithms

$$A_{\varepsilon,d}(f_d) = \sum_{i=1}^{n(\varepsilon,d)} f_d(\mathbf{t}_{i,\varepsilon,d}) g_{i,\varepsilon,d} \tag{13}$$

approximate f_d with errors bounded by

$$\| f_d - A_{\varepsilon,d}(f)\|_{L_2(\mathbb{R}^d)} \leq \varepsilon \|v\|_F \|V\|_F^{d-1} \leq \varepsilon \nu \mathcal{V}^{d-1}. \tag{14}$$

The number $n(\varepsilon, d)$ of function evaluations used by $A_{\varepsilon,d}$ is bounded by

$$n(\varepsilon, d) \leq \alpha_0 \left(\alpha_1 + \alpha_2 \frac{\ln 1/\varepsilon}{d-1} \right)_+^{(\alpha+1)(d-1)} \varepsilon^{-\alpha} \tag{15}$$

for some numbers α_i and a positive α; see [19]. We have $a_+ = \max\{a, 0\}$, and by convention, the right-hand side of (15) equals $\alpha_0 \varepsilon^{-\alpha}$ when $d = 1$. We stress that the numbers α_i and α do not depend on ε and d, and they are fully determined by the approximation problem for the class F.

The essence of (15) is that the leading term $\varepsilon^{-\alpha}$ is the same for all d. The only dependence on d is through the logarithmic factors.

We are ready to approximate the weighted integral $S_d(v, V)$. For given $\varepsilon > 0$ and $d \geq 1$, consider the algorithm

$$S_d(v, V) \sim \phi_{\varepsilon,d}(v, V) \quad \text{with} \quad \phi_{\varepsilon,d}(v, V) = H^{(d-1)}(0) \int_{\mathbb{R}^d} A_{\varepsilon,d}(f_d)(\mathbf{z})\, g_d(\mathbf{z})\, d\mathbf{z}, \tag{16}$$

where $A_{\varepsilon,d}$ is the algorithm (13) for approximating f_d. Due to linearity of $A_{\varepsilon,d}$, we have

$$\phi_{\varepsilon,d}(v, V) = H^{(d-1)}(0) \sum_{i=1}^{n(\varepsilon,d)} f_d(\mathbf{t}_{i,\varepsilon,d}) a_{i,\varepsilon,d} \quad \text{with} \quad a_{i,\varepsilon,d} = \int_{\mathbb{R}^d} g_{i,\varepsilon,d}(\mathbf{z})\, g_d(\mathbf{z})\, d\mathbf{z}. \tag{17}$$

From (14) and Lemma 1, we get

$$|S_d(v, V) - \phi_{\varepsilon,d}(v, V)| \leq |H^{(d-1)}(0)|\, \| f_d - A_{\varepsilon,d}(f_{\varepsilon,d})\|_{L_2(\mathbb{R}^d)} \|g_d\|_{L_2(\mathbb{R}^d)} \leq \varepsilon K_d \nu \mathcal{V}^{d-1}, \tag{18}$$

where

$$K_1 = \frac{|H(0)|}{\sqrt{2\sqrt{\pi t}}}$$

and

$$K_d = \frac{t^{(3d-4)/4} |H^{(d-1)}(0)|}{\sqrt{2\sqrt{\pi}\Gamma(1 + (3d-4)/2)}} \quad \text{for } d = 2, 3, \ldots.$$

Since $\Gamma(z) = z^{z-1/2} e^{-z} \sqrt{2\pi}(1 + o(1))$ as $z \to +\infty$, there exists a positive $K = K(t, \mathcal{H})$ such that

$$K_d = O\big(K^d / d^{3d/2}\big).$$

Hence, K_d goes super-exponentially fast to zero as d goes to infinity.

We now define the new algorithm A_ε for approximating the Feynman–Kac path integral $S(v, V)$. For a given ε,

$$A_\varepsilon(v, V) = \sum_{d=1}^{\infty} \phi_{\varepsilon_d, d}(v, V), \tag{19}$$

where

$$\varepsilon_d = \frac{\varepsilon}{K_d 2^d \nu \mathcal{V}^{d-1}}. \tag{20}$$

Since K_d goes super-exponentially fast to zero, ε_d goes super-exponentially to infinity with d. Hence, $n(\varepsilon_d, d)$ and consequently $\phi_{\varepsilon_d,d}$ are zero for large d. Therefore, A_ε consists of only finitely many terms in (19).

THEOREM 1. *The algorithm A_ε defined by* (19) *and* (20) *approximates the Feynman–Kac path integral $S(v, V)$ with error not exceeding ε and cost bounded by*

$$\mathrm{cost}(A_\varepsilon) \le K(\varepsilon)\varepsilon^{-\alpha}, \tag{21}$$

where $K(\varepsilon) = 2\alpha_0(2\nu)^\alpha/(\sqrt{2\sqrt{\pi t}})^\alpha C(\varepsilon)$,

$$\begin{aligned} C(\varepsilon) = 2 + \sum_{d=2}^{\infty} &(d+1) C_1^{\alpha(d-1)} \\ &\times \left(C_2 + \alpha_2 \frac{\frac{1}{2}\ln(2\sqrt{\pi t}) + \ln\varepsilon + \ln\Gamma(1+(3d-4)/2)}{d-1} \right)_+^{(\alpha+1)(d+1)} \Bigg/ \\ &\Gamma(1+(3d-4)/2)^\alpha, \end{aligned} \tag{22}$$

and $C_1 = 2\mathcal{H}\mathcal{V}t^{3/4}$, $C_2 = \alpha_1 + \alpha_2 \ln C_1$. Furthermore,

$$K(\varepsilon) = O(\varepsilon^{-\delta}), \quad \forall \delta > 0.$$

Proof. We first prove that the error of the algorithm A_ε is at most ε. We have

$$S(v, V) - A_\varepsilon(v, V) = \sum_{d=1}^{\infty} (S_d(v, V) - \phi_{\varepsilon_d, d}(v, V)).$$

Due to (18) and (20) we conclude

$$\|S(v, V) - A_\varepsilon(v, V)\| \le \sum_{d=1}^{\infty} \varepsilon_d K_d \nu \mathcal{V}^{d-1} = \varepsilon,$$

as claimed.

We now estimate the cost of the algorithm A_ε. We need to compute $\phi_{\varepsilon_d,d}(v, V)$ given by (17) and (11) for $d = 1, 2, \ldots$. Assuming that the numbers $H^{(d-1)}(0)a_{i,\varepsilon,d}$ are precomputed, we need to compute $n(\varepsilon_d, d)$ values of f_d and at most $2n(\varepsilon_d, d) - 1$ arithmetic operations.

Each value of f_d requires one function value of v and $d-1$ values of V to be computed as well as $d-1$ multiplications. Hence, the cost of computing $\phi_{\varepsilon_d,d}(u,V)$ is bounded by $2(d+1)n(\varepsilon_d,d)$. The cost of computing $A_\varepsilon(v,V)$ is estimated by

$$\mathrm{cost}(A_\varepsilon) \le \sum_{d=1}^{\infty} 2(d+1)n(\varepsilon_d,d).$$

Due to (15), after some calculations we obtain

$$\mathrm{cost}(A_\varepsilon) \le K(\varepsilon)\varepsilon^{-\alpha},$$

which proves (21).

Finally, we prove that $K(\varepsilon) = O(\varepsilon^{-\delta})$ for arbitrarily small positive δ. Of course, it is enough to prove that $C(\varepsilon) = O(\varepsilon^{-\delta})$. Let $\beta = 1.5(\alpha+1)$. Using once more the fact that $\Gamma(z) = z^{z-1/2}e^{-z}\sqrt{2\pi}(1+o(1))$ as $z \to +\infty$, there exists a positive C such that

$$C(\varepsilon) \le \sum_{d=2}^{\infty} \frac{C^{\beta d}\left(1+\ln(1/\varepsilon)/d\right)^{\beta d}}{d^{1.5\alpha d}}.$$

Clearly,

$$1 + \frac{\ln 1/\varepsilon}{d} \le \frac{\beta}{\delta}\left(1 + \frac{\delta \ln 1/\varepsilon}{\beta d}\right), \quad \forall \delta \le \beta.$$

Hence,

$$\left(1+\frac{\ln 1/\varepsilon}{d}\right)^{\beta d} \le \left(\frac{\beta}{\delta}\right)^{\beta d}\left(1+\frac{\ln(1/\varepsilon)^{\delta}}{\beta d}\right)^{\beta d} \le \left(\frac{\beta}{\delta}\right)^{\beta d}\left(\frac{1}{\varepsilon}\right)^{\delta}.$$

From this we conclude that

$$C(\varepsilon) \le \left(\frac{1}{\varepsilon}\right)^{\delta} \sum_{d=2}^{\infty} \left(\frac{C\beta}{\delta}\right)^{\beta d} d^{-1.5\alpha d}.$$

Since the last series is convergent, $C(\varepsilon) = O(\varepsilon^{-\delta})$, as claimed. ■

4. COMPLEXITY

In this section, we analyze the worst case complexity of Feynman–Kac path integration. We obtain bounds on the complexity by relating Feynman–Kac path integration to some specific *approximation* and *integration* problems over $\mathbb{R}$.

We begin with an upper bound. We assume that the functions $f \in F$ can be approximated in the space $L_2(\mathbb{R})$ with order $p > 0$ by using values of f. That is, there is a sequence $\{A_n\}_n$ of algorithms, each of the form

$$A_n(f) = \sum_{i=1}^{n} f(t_{i,n})g_{i,n} \quad \text{with} \quad t_{i,n} \in \mathbb{R}, \quad \text{and} \quad g_{i,n} \in L_s(\mathbb{R}), \tag{23}$$

such that

$$\frac{\|f - A_n(f)\|_{L_2(\mathbb{R})}}{\|f\|_F} = O(n^{-p}), \quad \forall f \in F. \tag{24}$$

Let $p^* = p^*(F)$ denote the supremum of p satisfying (24). The exponent p^* is closely related to the complexity $\mathrm{comp}^{\mathrm{app}}(\varepsilon, F)$ of the approximation problem. That is, if we want to approximate functions from the unit ball of the class F in the L_2-norm over $\mathbb{R}$ then

$$\Omega\left(\varepsilon^{-1/p^*+\delta}\right) = \mathrm{comp}^{\mathrm{app}}(\varepsilon; F) = O\left(\varepsilon^{-1/p^*-\delta}\right), \quad \forall\delta > 0. \tag{25}$$

The sequence $\{A_n\}_n$ of algorithms may serve as a basic step in constructing the algorithms $A_{\varepsilon,d}$ of (13) as explained in [19]. In fact, the parameter α of (15) can be set to $\alpha = 1/p$ for any p satisfying (24). Theorem 1 states that the cost of the algorithm A_ε is bounded by $K(\varepsilon)\varepsilon^{-\alpha}$ with $K(\varepsilon) = O(\varepsilon^{-\delta})$. Obviously, the cost of the algorithm A_ε is an upper bound on the complexity of Feynman–Kac path integration. Since p can be arbitrarily close to p^*, we get the following upper bound.

$$\mathrm{comp}(\varepsilon; F) = O\left(\varepsilon^{-1/p^*-\delta}\right), \quad \forall\delta > 0.$$

We now turn to a lower bound on the complexity of Feynman–Kac path integration. Consider the Gaussian-weighted integration problem for the class F. That is, suppose we want to approximate the integrals

$$I(f) = (2\pi t)^{-1/2} \int_{\mathbb{R}} f(u) \exp(-u^2/(2t))\, du$$

by using function values only. Assume that there is a sequence $\{B_n\}_n$ of algorithms each of the form

$$B_n(f) = \sum_{i=1}^{n} f(\tau_{i,n}) a_{i,n} \quad \text{with} \quad \tau_{i,n} \in \mathbb{R}, \quad \text{and} \quad a_{i,n} \in \mathbb{R},$$

such that

$$\frac{|I(f) - B_n(f)|}{\|f\|_F} = O(n^{-q}), \quad \forall f \in F. \tag{26}$$

Let $q^* = q^*(F)$ denote the supremum of q satisfying (26). It is known that the complexity $\mathrm{comp}^{\mathrm{int}}(\varepsilon, F)$ of integration in the unit ball of the class F satisfies

$$\Omega\left(\varepsilon^{-1/q^*+\delta}\right) = \mathrm{comp}^{\mathrm{int}}(\varepsilon, F) = O\left(\varepsilon^{-1/q^*-\delta}\right), \quad \forall\delta > 0. \tag{27}$$

It is clear that the complexity of Feynman–Kac path integration is bounded from below by

$$\mathrm{comp}(\varepsilon; F) = \Omega\left(\varepsilon^{-1/q^*+\delta}\right), \quad \forall\delta > 0.$$

Indeed, take $V = 0$. Since $H(0) = 1$ we have $S(v, 0) = \int_C v(x(t))w(dx)$. Changing variables by $u = x(t)$ we get $S(v, 0) = I(u)$. Hence, Feynman–Kac path integration problem is not easier than the integration problem for the ball of radius ν in the space F. Since ν changes only the multiplicative factor of the integration complexity, the lower bound on $\mathrm{comp}(\varepsilon; F)$ is proven. Hence, we have

THEOREM 2. *Let p^* and q^* be the exponents of the approximation and integration problems defined by* (25) *and* (27). *Then the worst case complexity of Feynman–Kac path integration is bounded by*

$$\Omega\left(\varepsilon^{-1/q^*(F)+\delta}\right) = \mathrm{comp}(\varepsilon; F) = O\left(\varepsilon^{-1/p^*(F)-\delta}\right), \quad \forall\delta > 0. \tag{28}$$

Observe that the exponents of the lower and upper bounds in (28) do not depend on the function H. For some functions H, the exponent of the lower bound is sharp. This holds, for example, for $H(t) = 1, \forall t$, independently of the choice of the space F.

For some spaces F, we have $p^* = q^*$, and the exponents of the lower and upper bounds almost coincide. In this case, the algorithm A_ε given by (19) is almost optimal. We now present examples of such spaces F.

EXAMPLE 4. Let F be the Banach space of Example 1 with the functions ψ_1, ψ_2 defined by (7). Let $g(x) = f(x)\psi_1(x)$. Then $\|f\|_F = \|g^{(r)}\psi_2\|_{L_p(\mathbb{R})}$. Suppose we approximate f by U which uses the values $f(x_i)$ for $i = 1, 2, \ldots, n$. We have

$$\int_{\mathbb{R}} |f(x) - U(x)|^2\, dx = \int_{\mathbb{R}} |g(x) - U_1(x)|^2 \psi_1^{-2}(x)\, dx,$$

where $U_1(x) = U(x)\psi_1(x)$. This shows that L_2-approximation for the functions f is equivalent to weighted L_2-approximation with weight $\psi_1^{-2}(x)$ for the functions g. The latter was considered in [21]. Assume that

$$a_1 + \min\{1 - 1/p, a_2\} > r + 1/2 - 1/p.$$

It is proved in [21] that then the exponent for L_2-approximation equals $p^* = r - (1/p - 1/2)_+$. For integration, the exponent equals $q^* = r$. Hence, for $p \geq 2$ we have $p^* = q^* = r$ and, due to Theorem 2, the complexity of Feynman–Kac integration is roughly $\varepsilon^{-1/r}$. Observe also that the last inequality implies (6) due to (8).

EXAMPLE 5. Consider now the Banach space F of Example 2. Since we consider only r times continuously differentiable functions, it is well known that the exponent p^* of L_2-approximation is at most r. To obtain a lower bound for p^* we proceed as follows. Take $T > 0$ and sample points $x_1 = iT/n$, for $i = 0, \pm 1, \pm 2, \ldots, \pm n$. We approximate f by $\tilde{f}$ which is the piecewise polynomial interpolation of order $r - 1$ on the interval $[-T, T]$, and zero outside of this interval. Then we have

$$|\tilde{f}(u) - f(u)| \leq \begin{cases} C(T/n)^r \|f\|_F, & |u| \leq T \\ \psi^{-1}(u)\|f\|_F, & |u| > T \end{cases},$$

which implies

$$\frac{\|\tilde{f} - f\|_{L_2(\mathbb{R})}}{\|f\|_F} = O\left(T^{r+1/2} n^{-r} + \left(\int_{|u|>T} \psi^{-2}(u)\, du\right)^{1/2}\right).$$

Suppose now that $\psi(u) = (1 + |u|)^a$ with $a > 1/2$. Then, for $T = O(n^{r/(r+a)})$, we have that the error

$$\frac{\|\tilde{f} - f\|_{L_2(\mathbb{R})}}{\|f\|_F} = O\big(n^{-r(a-1/2)/(r+a)}\big)$$

and, consequently, the exponent $p^* \geq \left(\frac{r}{r+a}\right)(a - 1/2)$. Hence, for large a, the exponent $p^* \simeq r$. From this it immediately follows that if $\psi(u)$ increases faster than any polynomial as the argument $|u| \to \infty$, then $p^* = r$. This holds, for instance, for the Gaussian weight $\psi(u) = \exp(-u^2/2)/\sqrt{2\pi}$.

Similarly, for integration, we obviously have the exponent $q^* \leq r$ since r is the exact exponent for integration over a finite interval. As before, for ψ increasing faster than polynomials we have $q^* = r$.

Thus, for spaces F with ψ increasing faster than any polynomial, the complexity of Feynman–Kac integration is again roughly $\varepsilon^{-1/r}$.

5. IMPLEMENTATION REMARKS

In this section, we discuss implementation of the Feynman–Kac path integration algorithm A_ϵ of Section 3.

The algorithm A_ε produces an ε-approximation to the solution $S(v, V)$. Since S is nonlinear, it is not surprising that A_ε is also nonlinear. In fact, the algorithm A_ε is a polynomial in the values of v and V at sample points. When we estimate the cost of A_ε, we assume that the coefficients of this polynomial are *precomputed*. For fixed ε, H, t, ν, $\mathcal{V}$, and the class F, this can be done, at least theoretically, since the coefficients of A_ε are independent of the functions v and V which vary through the balls of the class F. Formally, this means that the algorithm A_ε is nonuniform in ε, H, t, ν, and $\mathcal{V}$. The reader is referred to a recent paper [16] where the cost of uniform and nonuniform algorithms is discussed and compared.

The precomputed coefficients of the algorithm A_ε are given as weighted integrals of the form

$$a_{i,\varepsilon,d} = \int_{\mathbb{R}^d} g_{i,\varepsilon,d}(\mathbf{z})g_d(\mathbf{z})\,d\mathbf{z} \tag{29}$$

for some specific (tensor product) functions $g_{i,\varepsilon,d}$, see (13), and weights $g_d(\mathbf{z})$ given by (12). We need $a_{i,\varepsilon,d}$ for all $i = 1, 2, \ldots, n(\varepsilon, d)$ and a few d. The total number of needed $a_{i,\varepsilon,d}$ is $\sum_{d=1}^{\infty} n(\varepsilon, d)$. As already mentioned in Section 3, $n(\varepsilon, d)$ goes super-exponentially fast to zero and only a few terms of the last series are not zero.

Clearly, we do not need the exact values of $a_{i,\varepsilon,d}$. It is enough to know them approximately. However, the required accuracy of the coefficients depends on ε and increases as ε decreases.

The precomputation of the coefficients $a_{i,\varepsilon,d}$ with large precision may be difficult. Indeed, let us first observe that each $a_{i,\varepsilon,d}$ is a linear combination of some multivariate weighted integrals defined over $\mathbb{R}^d$. The integrals are of the form

$$\begin{aligned} I_{i,j} = {} & \int_{\mathbb{R}^{d+1}} \psi_{i_0,j_0}(z_0)\psi_{i_1,j_1}(z_1)\cdots\psi_{i_d,j_d}(z_d) \\ & \times \int_{0\le t_0<\cdots<t_d=t} \frac{\exp\left(-\frac{1}{2}\left(\frac{z_0^2}{t_0} + \frac{(z_1-z_0)^2}{t_1-t_0} + \cdots + \frac{(z_d-z_{d-1})^2}{t_d-t_{d-1}}\right)\right)}{(2\pi)^{(d+1)/2}\sqrt{t_0(t_1-t_0)\cdots(t_d-t_{d-1})}}\,d\mathbf{t}\,d\mathbf{z}, \end{aligned} \tag{30}$$

for different indices $\mathbf{i}$ and $\mathbf{j}$, where ψ_{i_k,j_k} are some basis functions used in the one-dimensional approximation (23). Even for moderate d, the computation of (30) with large precision is a difficult task.

Precomputing should be done with higher precision than the desired precision of the final approximation to $S(v, V)$. For instance, if we want the final approximation with precision 10^{-3}, we must have the coefficients $a_{i,\varepsilon,d}$ with precision 10^{-4} or even higher, and hence the number of integrals to be precomputed can be huge. Thus, in addition to the errors occurring when approximating the path integral, we have another source of possible errors which is due to inexact precomputation.

We experienced all these difficulties when implementing the algorithm A_ε. As the one-dimensional approximations we took piecewise polynomials of order 0, 1, and 3, which are (almost) optimal for L_2-approximation for many spaces F including those of Example 2 with regularity $r = 1, 2, 4$, correspondingly (see also Example 5).

In our implementation, the one-dimensional sample points were chosen as the points dividing the real line into the intervals of equal Gaussian (standard) measure. We calculated the integrals $I_{\mathbf{i},\mathbf{j}}$ by using the classical Monte Carlo with respect to $\mathbf{t}$ and $\mathbf{z}$.

We present sample results for one precomputing for each r. We set $t = 1$, the upper bound on d was taken as $d_{\max} = 6$ and the number $n(d)$ of points in approximation of f_d was equal to 511, 769, 1023, 769, 351, 545, 127 for $d = 0, 1, 2, 3, 4, 5, 6$, respectively. This corresponds to precomputation of 10,375 integrals (30) of dimensions 1, 3, 5, 7, 9, 11, 13. For each integral (30), we used Monte Carlo with 10^7 sample points. The calculations were performed on the IBM POWER PC 604 computer, and it took about two weeks of CPU time (for each r) to complete the precomputing.

The precomputing described above was then used in an implementation of A_ε and applied for various functions v and V. We present two series of tests, for v and V with known and unknown exact solutions $S(v, V; t)$. In the latter case, we ran a version of Chorin's algorithm for many sample points—see Chorin [6] or Hald [11]—and treated the results of Chorin's algorithm as an exact solution.

We stress that if the precomputing were exact, Chorin's algorithm would be inferior to our algorithm, at least for functions from classes of Example 2, as explained in the Introduction. In particular, with the precomputing above, A_ε uses only 511 values of v and V, and for $v = 1$ this number even reduces to 127 values of V. Hence, the running time of A_ε is almost negligible. Chorin's algorithm in turn uses nk values, where n is the number of discretization points and k is the number of Monte Carlo points. On the other hand, Chorin's algorithm does *not* require any precomputing and hence it can give more accurate results provided n and k are chosen large enough. The maximal accuracy obtained by Chorin's algorithm was about 10^{-4} (which agrees with theoretical properties of Monte Carlo), and to get this we needed 1 or 2 h of CPU time. With the use of precomputed coefficients, the same accuracy can be obtained by A_ε in less than 1 s.

In all the tests we assume $H(z) = \exp(z)$ and $t = 1$.

Test 1 (Known Solutions). Here v and V satisfy $V(x) = 1 - v''(x)/2v(x))$ for the function v given below. Then the exact solution is $S(v, V; t) = v(0)e^t$.

The results for the first case $v = V = 1$ actually show the accuracy of the precomputing, since if the precomputing were exact we would get the exact solution for all r's. It seems that for $r = 4$ the accuracy of the precomputing is not sufficient and therefore the errors for $r = 4$ are comparable with those for $r = 2$. (This is probably caused by the fact that for $r = 4$ the basis functions in (30) are more complicated than for $r = 1, 2$.)

		$\|A_\varepsilon - S(v, V; 1)\|$		
$v(x)$	$V(x)$	$r = 1$	$r = 2$	$r = 4$
1	1	$5.3_{10} - 3$	$3.4_{10} - 4$	$8.2_{10} - 4$
$\cos(x)$	$3/2$	$1.3_{10} - 1$	$3.4_{10} - 2$	$3.5_{10} - 2$
$\exp(x)$	$1/2$	$8.3_{10} - 2$	$4.5_{10} - 2$	$4.5_{10} - 2$
$\exp(-x^2/2)$	$(3 - x^2)/2$	$6.6_{10} - 2$	$1.1_{10} - 1$	$1.1_{10} - 1$

Test 2 (Unknown Solutions). We set $v(x) = V(x) = \exp(px^2)$, where p is a parameter. The larger p, the larger the norm of v and V, so that the difficulty of the problem increases with p.

As for the previous tests, we do not see any signifficant difference between the errors for $r = 2$ and $r = 4$. Once more, this is an indication that the accuracy of precomputing is not sufficient for $r = 4$.

We want to stress that the results reported below need not be very conclusive due to the fact that Chorin's algorithm is nondeterministic and we only know that its expected error is not smaller than 10^{-4}. Observe that for tests with unknown solutions we use v and V from the class of Example 2, in which case the theoretical complexity is roughly proportional to $\varepsilon^{-1/r}$. However, the bounds on the norms increase with p, and therefore the errors are larger for larger p:

$v(x) = V(x) = \exp(-px)$

		$\lvert A_\varepsilon - \text{Chorin}\rvert$		
p	Chorin	$r = 1$	$r = 2$	$r = 4$
0	2.71828	$5.3_{10} - 3$	$3.4_{10} - 4$	$8.2_{10} - 4$
1	1.30761	$9.6_{10} - 3$	$8.6_{10} - 3$	$8.0_{10} - 3$
2	0.93898	$1.4_{10} - 2$	$1.4_{10} - 2$	$1.5_{10} - 2$
3	0.75257	$2.7_{10} - 2$	$2.7_{10} - 3$	$3.4_{10} - 3$
4	0.63693	$3.9_{10} - 2$	$1.9_{10} - 2$	$2.1_{10} - 2$

We also applied other than Monte Carlo algorithms for computation of integrals (30), including some deterministic algorithms, but we have not obtained qualitative better results. For instance, we used a version of Smolyak's algorithm. This algorithm requires a tensor product form of the weight. It can be achieved by a change of variables, but then the transformed integrand becomes a non-Lipschitz function. Our computations confirmed the difficulty of dealing with non-Lipschitz functions. Another attempt was made by the use of some quasi-Monte Carlo algorithms (QMC) which also did not fully succeed.

It seems to us that the accuracy of precomputing can be reduced only by inventing a special algorithm that makes use of a particular form of the integrals (30). By now, however, the level 10^{-4} has been the maximal accuracy of the precomputing. This establishes an inevitable error when A_ε is applied and restricts, so far, the practical use of A_ε to the cases when one needs fast a solution with a *moderate accuracy* and/or for many different functions v and V.

Observe also that the precomputing for a specific t can be used to obtain an approximation of the path integral for other t's, so that one can easily draw a graph of the solution $S(v, V; t)$ as a function of t. Indeed, this follows from the equality

$$S(v, V; t_1) = S(\tilde{v}, \tilde{V}; t),$$

where $\tilde{v}(x) = v(x\sqrt{t_1/t})$ and $\tilde{V}(x) = (t_1/t)V(x\sqrt{t_1/t})$. Note that if t_1 increases then the norms of $\tilde{v}$ and $\tilde{V}$ also increase. Hence, the bounds ν and $\mathcal{V}$ must be appropriately modified.

APPENDIX

Proof of Lemma 1. The value of $\|g_1\|^2_{L_2(\mathbb{R})}$ is easy to find. Therefore, we only prove the upper bound on $LHS = \|g_{d+1}\|^2_{L_2(\mathbb{R}^{d+1})}$. Obviously,

$$LHS = \int_{0\le t_1\le\cdots\le t_d\le t} \int_{0\le v_1\le\cdots\le v_d\le t} \int_{\mathbb{R}^{d+1}} h_{d+1}(\mathbf{u},\mathbf{t},\mathbf{v})\,d\mathbf{u}\,d\mathbf{v}\,d\mathbf{t},$$

where

$$h_{d+1}(\mathbf{u},\mathbf{t},\mathbf{v}) = \frac{\exp\Big(-\frac{1}{2}\sum_{j=1}^{d+1}(u_j-u_{j-1})^2(1/(t_j-t_{j-1})+1/(v_j-v_{j-1})\Big)}{\prod_{j=1}^{d+1}\sqrt{4\pi^2(t_j-t_{j-1})(v_j-v_{j-1})}}$$

with $u_0 = v_0 = 0$, $v_{d+1} = t$, and $\mathbf{v} = [v_1,\ldots,v_d]$. Let $B_{d+1} = B_{d+1}(\mathbf{t},\mathbf{v})$ be the inner integral with respect to $\mathbf{u}$. Then

$$B_{d+1} = \int_{\mathbb{R}^d} h_d(\mathbf{u},\mathbf{t},\mathbf{v})I(u_d)\,d\mathbf{u}$$

with

$$I(u_d) = \int_{\mathbb{R}} \frac{\exp(-(u_{d+1}-u_d)^2(1/a+1/b)/2)}{\sqrt{4\pi^2ab}}\,du_{d+1},$$

$a = t_{d+1} - t_d$, and $b = v_{d+1} - v_d$. By a change of variables, $y := (u_{d+1} - u_d)\sqrt{1/a + 1/b}$, we get

$$I(u_d) = \frac{1}{\sqrt{4\pi^2(a+b)}}\int_{\mathbb{R}}\exp(-y^2/2)\,dy = \frac{1}{\sqrt{2\pi(a+b)}}.$$

This proves that

$$B_{d+1} = \frac{B_d}{\sqrt{2\pi(t_{d+1}-t_d+v_{d+1}-v_d)}} = \prod_{j=1}^{d+1}\frac{1}{\sqrt{2\pi(t_j-t_{j-1}+v_j-v_{j-1})}}$$

and, consequently, that

$$\begin{aligned} &LHS \\ &= \frac{1}{(2\pi)^{(d+1)/2}}\int_{0\le t_1\le\cdots\le t_d\le t}\int_{0\le v_1\le\cdots\le v_d\le t}\prod_{j=1}^{d+1}\frac{1}{\sqrt{t_j-t_{j-1}+v_j-v_{j-1}}}\,d\mathbf{v}\,d\mathbf{t} \\ &= \frac{t^{2d-(d+1)/2}}{(2\pi)^{(d+1)/2}}\int_{t_i\ge 0,\sum_{i=1}^d t_i\le 1}\int_{v_i\ge 0,\sum_{i=1}^d v_i\le 1}\left(\left(2-\sum_{k=1}^d(t_k+v_k)\right)\prod_{i=1}^d(t_i+v_i)\right)^{-1/2}\,d\mathbf{v}\,d\mathbf{t}; \end{aligned}$$

with the second equality due to the change of variables: $t_i := (t_i - t_{i-1})/t$ and $v_i := (v_i - v_{i-1})/t$. Let us make another change of variables: $z_i := t_i + v_i$ and $x_i := t_i$. Then the domain of integration is contained in a set of nonnegative z_i's and x_i's such that $x_i \leq z_i$ and $\sum_{i=1}^d z_i \leq 2$. This gives the estimate

$$\begin{aligned}
LHS &\leq \frac{t^{(3d-1)/2}}{(2\pi)^{(d+1)/2}} \int_{0\leq z_i,\sum_{i=1}^d z_i \leq 2} \left(2-\sum_{k=1}^d z_k\right)^{-1/2} \prod_{i=1}^d z_i^{-1/2} \int_{\mathbf{0}\leq \mathbf{x}\leq \mathbf{z}} 1\, d\mathbf{x}\, d\mathbf{z} \\
&= \frac{t^{(3d-1)/2}}{(2\pi)^{(d+1)/2}} \int_{0\leq z_i,\sum_{i=1}^d z_i \leq 2} \left(2-\sum_{k=1}^d z_k\right)^{-1/2} \prod_{i=1}^d \sqrt{z_i}\, d\mathbf{z} \\
&= \frac{(2t)^{(3d-1)/2}}{(2\pi)^{(d+1)/2}} \int_{0\leq z_i,\sum_{i=1}^d z_i \leq 1} \left(1-\sum_{k=1}^d z_k\right)^{-1/2} \prod_{i=1}^d \sqrt{z_i}\, d\mathbf{z} \\
&= \frac{t^{(3d-1)/2} 2^{d-1} (\Gamma(3/2))^d \sqrt{\pi}}{\pi^{(d+1)/2}\Gamma(1+(3d-1)/2)} \\
&= \frac{t^{(3d-1)/2}}{2\Gamma(1+(3d-1)/2)},
\end{aligned}$$

with the second to last equality due to (4.635.4) in [10]. This completes the proof of the lemma.

ACKNOWLEDGMENTS

We are grateful to Joseph F. Traub for valuable comments. We also thank Anargyros Papageorgiou for providing us the FINDER package for multivariate integration. Finally, we thank an anonymous referee who found an error in the proof of the original version of Lemma 1. The first author was partially supported by the State Committee for Scientific Research of Poland under Grant 2 P03A 00913, and the second and third authors were supported by the National Science Foundation under Grants CCR-9729971 and CCR-9731858, respectively.

REFERENCES

1. S. A. Albeverio and R. J. Høegh–Krohn, *Mathematical Theory of Feynman Path Integrals*, Lecture Notes in Math. (Springer–Verlag, Berlin, 1976). Vol. 523.
2. A. Araujo and E. Giné, *The Central Limit Theorem for Real and Banach Valued Random Variables* (Wiley, New York, 1980).
3. N. Aronszajn, Theory of reproducing kernels, *Trans. Amer. Math. Soc.* **68**, 337 (1950).
4. N. S. Bakhavlov, On approximate calculation of integrals, *Vestn. Mosk. Univ. Ser. Mat. Mekh. Astron. Fiz. Khim.* **14**, 3 (1959). [In Russian]
5. R. H. Cameron, A Simpson's rule for the numerical evaluation of Wiener's integrals in function space, *Duke Math. J.* **18**, 111 (1951).
6. A. J. Chorin, Accurate evaluation of Wiener integrals, *Math. Comput.* **27**, 1 (1973).
7. A. D. Egorov, P. I. Sobolevsky, and L. A. Yanovich, *Functional Integrals: Approximate Evaluation and Applications* (Kluwer Academic, Dordrecht, 1993).
8. R. P. Feynman and A. R. Hibbs, *Quantum Mechanics and Path–Integrals* (McGraw–Hill, New York, 1965).
9. L. D. Fosdick, Approximation of a class of Wiener integrals, *Math. Comput.* **19**, 225 (1965).
10. I. S. Gradshteyn and I. Ryzhik, *Table of Integrals, Series and Products* (Academic Press, New York, 1980).
11. O. H. Hald, Approximation of Wiener integrals, *J. Comput. Phys.* **69**, 460 (1987).
12. M. Kac, On distributions of certain Wiener functionals, *Trans. Amer. Math. Soc.* **65**, 1 (1949).
13. H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*, CBMS-NSF Reg. Conf. Series Appl. Math. (SIAM, Philadelphia, 1992), Vol. 63.

14. E. Novak, *Deterministic and Stochastic Error Bounds in Numerical Analysis*, Lecture Notes in Mathematics (Springer–Verlag, Berlin, 1988), Vol. 1349.
15. E. Novak, On adaption, *J. Complexity* **12**, 199 (1996).
16. E. Novak and H. Woźniakowski, On the cost of uniform and nonuniform algorithms, *Theor. Comput. Sci.* **219**, 301 (1999).
17. J. F. Traub, G. W. Wasilkowski, and H. Woźniakowski, *Information-Based Complexity* (Academic Press, New York, 1988).
18. N. N. Vakhania, V. I. Tarieladze, and S. A. Chobanyan, *Probability Distributions on Banach Spaces* (Reidel, Dordrecht, 1987).
19. G. W. Wasilkowski and H. Woźniakowski, Explicit cost bounds of algorithms for multivariate tensor product problems, *J. Complexity* **11**, 1 (1995).
20. G. W. Wasilkowski and H. Woźniakowski, On tractability of path integration, *J. Math. Phys.* **37**(4), 2071 (1996).
21. G. W. Wasilkowski and H. Woźniakowski, Complexity of weighted approximation over $\mathbb{R}^1$, *J. Approx. Theory* **103**, 233 (2000).