Efficient algorithms for the laboratory discovery of optimal quantum controls

Gabriel Turinici*
INRIA Rocquencourt, MICMAC project, Domaine de Voluceau, Rocquencourt B.P. 105, 78153 Le Chesnay Cedex, France

Claude Le Bris†
CERMICS-ENPC, Champs sur Marne, 77455 Marne la Vallée Cedex, France

Herschel Rabitz‡
Department of Chemistry, Princeton University, Princeton, New Jersey 08544-1009, USA

(Received 18 December 2003; published 20 July 2004)

The laboratory closed-loop optimal control of quantum phenomena, expressed as minimizing a suitable cost functional, is currently implemented through an optimization algorithm coupled to the experimental apparatus. In practice, the most commonly used search algorithms are variants of genetic algorithms. As an alternative choice, a direct search deterministic algorithm is proposed in this paper. For the simple simulations studied here, it outperforms the existing approaches. An additional algorithm is introduced in order to reveal some properties of the cost functional landscape.

DOI: 10.1103/PhysRevE.70.016704 PACS number(s): 02.70.−c, 32.80.Qk, 42.50.Vk

I. INTRODUCTION

Laser manipulation of quantum dynamics involves the tailoring of a control field, often from a laser, to optimally steer the system toward a desired target outcome. The underlying phenomenon utilized to achieve such goals is the manipulation of quantum interferences in the evolving dynamics driven by the laser field. The success of any particular laser field is measured in terms of a cost functional that incorporates the achieved metric distance to the target, and possibly other auxiliary costs such as the laser pulse energy. The subject of quantum control has well-defined theoretical foundations and an extensive literature [1–4]. In addition to the theoretical developments, numerous optimal control experiments have demonstrated the feasibility of the approach [5–11]. In particular, the most dramatic experimental advances, especially over control of complex systems, have utilized the so-called closed-loop paradigm, introduced a decade ago [12]. This paradigm will be the focus of the present article. The laser field is updated from one experiment to the next, on the basis of a measure of the distance to the desired target goal. The quantum system subjected to control is used in the laboratory, within the optimization loop, as an analog computer to integrate its own evolution equation. The control terminology of "closed-loop", as opposed to "open-loop", means that the distance to the target is evaluated on-the-fly and serves as a guideline to modify the control, in order to further optimize the result upon traversing the loop once again, etc. The loop is closed from one experiment to the next with a *new* sample of the quantum system, as it is both difficult to react in the extremely small time window between experiments, and almost impossible, in the quantum framework, to observe the state of a system without modifying the system itself in some unknown way.

When carrying out the closed-loop optimal control paradigm, a natural consideration is which type of optimization algorithm is preferable to guide the evolving experiments. Following the original work [12], practitioners of the field have used stochastic-like algorithms, and more precisely genetic algorithms (GA), that have proved to be surprisingly efficient in this context [13]. To justify the use of GAs, it is often argued that other types of algorithms, mainly deterministic algorithms, present major drawbacks that prevent them from being efficient in this context involving the optimization of a nonconvex cost function in a high-dimensional search space with a significant degree of laboratory noise. The purpose of this article is to show that in some circumstances, appropriate deterministic algorithms (possibly with a small amount of introduced randomness) can perform equally well and even outperform GAs by an order of magnitude in terms of the number of functional evaluations (i.e., experiments) required. A discrete descent algorithm which gives us some insight into the shape of the cost functional surface will also be introduced to provide a possible explanation of why the GAs perform well in this laser control problem and why they can be outperformed by such deterministic algorithms. Before presenting the algorithms and evaluating their efficiency, it is useful to draw up a list of the main features of the optimization problem at hand, and to see how these features impact on the qualities needed for an algorithm to perform well.

As the optimization algorithm will operate with the laboratory experiments, we will only have access to measurements on the system such that the cost functional can be evaluated for each trial laser field. No derivatives with respect to the laser field are directly available, in contrast to the situation arising when the control process is simulated nu-

*Also at CERMICS-ENPC, Champs sur Marne, 77455 Marne la Vallée Cedex, France. Electronic address: gabriel.turinici@inria.fr

†Also at INRIA Rocquencourt, MICMAC project, Domaine de Voluceau, Rocquencourt B.P. 105, 78153 Le Chesnay Cedex, France. Electronic address: lebris@cermics.enpc.fr

‡Electronic address: hrabitz@princeton.edu

merically [14]. So far, this optimization context is the same as in any practical problem in the control engineering sciences (see Ref. [15] for a recent review). However, quantum control distinguishes itself in one important way: the cost function evaluation is unprecedented in its cheapness. Present experiments may be performed at the rate of hundreds or more per second and available technology could in principle operate at a million per second. The need for signal averaging will reduce these numbers, but they still remain far from the standard duty cycle of engineering applications. The quantum control situation suggests that the optimization algorithm should be of order zero (i.e., only function evaluations are involved with no derivatives) or gradient-free to exploit the ability to perform massive numbers of experiments [16]. Such a criterion immediately suggests using Monte Carlo-type methods, including GAs, but this is not the only choice, as will be shown in this paper.

A second key issue is robustness with respect to noise in the observations, and especially in the controls. This matter is of particular concern for the control of quantum systems, where the manipulation of quantum interference may be sensitive to noise. It is therefore of primary interest to have an optimization algorithm that is as robust as possible with respect to noise. A related matter is the presence of slow drift (i.e., slower than the closed-loop cycle time) in the laser or other apparatus components over the time of the experiment. Such drift is often not critical when considering control alone as the goal, but other auxiliary goals can be affected by drift [17]. Thus, significantly reducing the number of experiments with faster algorithms is of prime interest. This point may also be important for control cases where the target is, by some suitable measure, very far from the initial state, thereby calling for extensive searching to find an effective control.

A third feature of the quantum control problem is the expected general nonconvexity of the cost functional. Although this feature is not peculiar in comparison with other problems in the engineering sciences, it can be particularly vexing in quantum control. A consequence is the risk that the search algorithm will get trapped in a local poor-quality extremum. It is therefore crucial to ensure that the algorithm have good *global search* properties (i.e., the ability to search for an extremum by efficiently scanning over a large range of control parameter values) as opposed to a *local search*, which optimizes in the vicinity of the current iteration. Again, this would seem to suggest approaches using random variations in one way or another in executing the search, but this choice is not the only class of algorithms as argued in this work.

A final main feature of quantum control experiments is the accessible high dimensionality of the search space with hundreds of phases and amplitude parameters (knobs) that define the laser field. Typically, each of these variables is discretized to about 50 values over its domain of variation. Thus, an enormous space is available for exploration in the experiments. Such a circumstance is within the standard range encountered in other nonlinear nonconvex optimization.

Given the criteria above, we seek a control algorithm with the following properties:

(a) Utilizes only function evaluations;

(b) Exploits the high duty cycle of the experiments;

(c) Assures robustness with respect to noise as best as possible;

(d) Avoids being trapped in an uninteresting local extremum; and

(e) Works effectively in high-dimensional searches.

The next section will review some good candidates to fulfill the above list of requirements. All of these candidates are well-established strategies that have proved their efficiency in other contexts. Section III will then present comparisons between one of these algorithms and the GA on a simple model quantum control problem that is physically relevant and that embodies all the main difficulties of the generic problem. The conclusion is that GAs are not the only way to proceed and other algorithms may be significantly more efficient. In Sec. IV, we describe an (apparently new) quasideterministic algorithm that is used to obtain some insight into the cost function landscapes. This result suggests one possible explanation of the success of GAs in this context, and also permitted us to further tune the algorithms for even better efficiency. Section V summarizes the paper and gives some suggestions for future work.

II. ALGORITHMS

Before presenting the particular algorithms tested in this paper, we will briefly survey some algorithms from the optimization literature that only use point value information and not the gradients (i.e., so-called “direct search” or “gradient-free” optimization). Different trends can be identified.

(1) **Genetic or evolutionary algorithms.** These form an important example of stochastic algorithms that have proved to be useful in a wide variety of engineering contexts. Later in this section, additional details will be presented on the algorithm we have tested.

(2) **Hierarchical algorithms.** The pattern search algorithms formalized by Torczon *et al.* [20–24] organize the search on hierarchical lattices in the parameter space. The solution iteratively wanders from one point of the lattice system to another according to rules that take into account cost function information in the form of a finite difference scheme. Convergence theory for these algorithms is well established and the presence of constraints may be accounted for, but the case of noisy cost functions is not completely analyzed yet. A related approach is the work of Anderson [25], which treats noise and establishes convergence properties. An important assumption used in the proof is that the noise amplitude is weaker around the extrema, which is an open question for quantum control cost functionals.

(3) **Simplex algorithms.** The so-called simplex search procedure is based on using a set of $N+1$ points forming a nondegenerate geometrical object in the N-dimensional space (e.g., triangle in 2D, tetrahedron in 3D, etc.). A common form of this procedure is the Nelder-Mead algorithm (described in *Numerical Recipes* [26] for noiseless functions and used in a related laser control context in Refs. [27,28]) and its modifications to treat noisy functions [29] which is further analyzed in Ref. [30]. A short description of the

Nelder-Mead simplex algorithm is as follows: the simplex can be viewed as "rolling" on the cost function surface. Depending on the outcome of the current "roll" step, the simplex is either *expanded* when it successfully improves the cost function (i.e., minimum) value or *contracted* (presumably to get through a narrow valley) when it cannot improve the minimal value. Further details are provided later in this section.

(4) **Ordinal optimization.** The "ordinal optimization" framework of Ho *et al.* [31–34] accommodates high noise amplitudes. It relies on an ordering approach that aims to find designs that belong to a particular percentile of all designs (but whose cost functional values are hopefully not significantly inferior to the true optimum) [35].

Additional approaches are reviewed in Ref. [25]. Starting from the list above, we will test the performance of two algorithms for noisy cost functionals arising in quantum control: (A) a modified simplex algorithm from Ref. [29], and (B) an evolutionary strategy (GA) algorithm.

Below, the goal is minimization of the cost functional, $J(x_1,\ldots,x_N):\mathbb{R}^N\to\mathbb{R}$, which characterizes the control problem and where $x_1,\ldots,x_N$ are the control "knobs" that can be adjusted in practical laboratory experiments.

A. Modified simplex algorithm

The following discussion presents the *modified simplex algorithm* as introduced by Barton and Ivey [29]. The algorithmic rules are given below.

(1) **Initialization.** In this step $N+1$ points $x^1,\ldots,x^{N+1}$ are randomly chosen to form the vertices of a nondegenerate simplex in the N-dimensional search space (i.e., the geometrical simplex figure has strictly positive volume; for example, in 2D the three points cannot be collinear, in 3D the four points cannot be on the same plane, etc.). For each point x^k the cost functional $J(x^k)$ is evaluated, $k=1,\ldots,N+1$.

(2) **Reflection.** The simplex vertices are ordered by the computed value of the cost functional. The highest J^{high}, second highest J^{sechi}, and lowest J^{low} values are identified and the corresponding points are denoted as x^{high}, x^{sechi}, and x^{low}, respectively. The centroid x^{cent} of all the vertices except x^{high} is computed, and a new vertex x^{refl} is generated by reflecting x^{high} through x^{cent} by the following formula:

$$x^{\text{refl}}=(1+\alpha)x^{\text{cent}}-\alpha x^{\text{high}}. \tag{1}$$

The value that we have used in this work is $\alpha=1$. Depending on the value of J^{refl}, the algorithm branches to one of the following alternatives:

(a) **Accepted reflection.** If $J^{\text{low}}\leqslant J^{\text{refl}}\leqslant J^{\text{sechi}}$, then x^{refl} replaces x^{high} in the simplex; the current iteration finishes and the control is passed to step 3.

(b) **Attempt expansion.** If $J^{\text{refl}}<J^{\text{low}}$, then the reflection is expanded to further exploit the current "search direction." The expanded point is given by the formula

$$x^{\text{exp}}=\gamma x^{\text{refl}}+(1-\gamma)x^{\text{cent}}. \tag{2}$$

The expansion coefficient γ will be taken as 2 in the present work. The value J^{exp} of the cost functional at x^{exp} is computed and the iteration utilizes one of the following alternatives:

(i) **Accept expansion.** If $J^{\text{exp}}<J^{\text{low}}$, then x^{exp} replaces x^{high} in the simplex.

(ii) **Reject expansion.** If $J^{\text{exp}}\geqslant J^{\text{low}}$, then the expansion is rejected and x^{refl} replaces x^{high}.

The current iteration finishes and the control is passed to step 3.

(c) **Attempt contraction.** If the newly constructed vertex x^{refl} would not improve the cost functional in the new simplex, i.e., if $J^{\text{refl}}>J^{\text{sechi}}$, then the simplex contracts. The contraction begins by first selecting the lowest cost functional associated with x^{high} and x^{refl} (i.e., if $J^{\text{refl}}\leqslant J^{\text{high}}$, then x^{refl} replaces x^{high} and J^{refl} replaces J^{high}). The contraction vertex is given by

$$x^{\text{cont}}=\beta x^{\text{high}}+(1-\beta)x^{\text{cent}}. \tag{3}$$

The *contraction coefficient* is chosen as $\beta=0.5$ in this work. The value J^{cont} corresponding to x^{cont} is computed and the following alternatives arise:

(i) **Accept contraction.** If $J^{\text{cont}}\leqslant J^{\text{high}}$, then the contraction is accepted; the current iteration finishes and the control is passed to step 3.

(ii) **Shrink.** If $J^{\text{cont}}>J^{\text{high}}$, then the whole simplex shrinks around x^{low} such that each point except x^{low} is modified by the formula

$$x^k=\delta x^k+(1-\delta)x^{\text{low}}. \tag{4}$$

We use here the shrinkage factor $\delta=0.9$. The algorithm then evaluates J at each point (including x^{low}); the current iteration finishes and the control is passed to step 3.

(3) **Stopping criterion.** Iterations continue until the stopping criterion is met; in the present work the stopping criterion is based on the volume of the simplex, which is not allowed to drop below a certain dimension-normalized value. This logic is based on a shrinking simplex, implying that the search is converging in a local minimum of the cost functional. Another useful criterion is to set a bound on the total number of iteration steps. If the stopping criterion is not met, then the algorithm begins a new iteration at step 2.

B. Evolutionary algorithm

The second algorithm used is an evolutionary strategy optimization algorithm. The evolutionary strategies are one of the most efficient descendants of the genetic algorithms that appeared in the 1980's [36]. The implementation we used is the EO Evolutionary Computation Framework [37]. Some of the settings are given below.

(i) Nonisotropic mutation.

(ii) Population size was taken to be 10, with the percentage number of offspring being 1000%.

(iii) Initialization bounds for the pulse amplitudes are $[-0.10,0.10]$ and for the phases $[-\pi,\pi]$; during a run the final amplitudes were bounded over $[-0.5,0.5]$ and no restrictions were imposed on the phases [see Eq. (5)].

Note that with these parameters each generation requires 100 cost functional evaluations (i.e., experiments), except for the initial step that requires 10 evaluations. As a stopping criterion a maximum number of 300 generations was imposed.

With a slight abuse of language in order to generally conform to the control literature, we will denote this procedure as a “GA,” although it is actually an evolutionary strategy.

III. SIMULATIONS OF QUANTUM CONTROL

The algorithms in Sec. II were successfully tested on typical optimization benchmarks extracted from the literature [29,38], in addition to noise-free quantum control problems. In order to better model the experimental conditions, this section will describe quantum control simulations where both control field and measurement noise is present.

The modified simplex algorithm and the evolutionary learning strategy were applied to a model 10-level quantum system presented earlier in Ref. [39]. The numerical values below have the units of fs for time, rad/fs for frequency and energy, and V/Å for the electric field. The field is expressed in terms of fixed frequencies ω_l, with the amplitudes a_l and phases θ_l being the control variables for optimization

$$\epsilon_c(t) = \epsilon_0 \exp[-((t-T/2)/\sigma)^2]\sum_{l=1}^{L} a_l \cos(\omega_l t + \theta_l). \qquad (5)$$

A total of 32 variables (i.e., $L=16$) was optimized corresponding to selected single, double, and triple quantum transitions [39].

The physical objective in the cost functional is to maximally project onto a target state $|\Psi_T\rangle$ at time T balanced with a laser fluence penalty weighted through a parameter $\mu \geqslant 0$

$$J(a_1, \ldots, a_L, \theta_1, \ldots, \theta_L) = J(\epsilon(t)) = \frac{\mu}{2}\int_0^T \epsilon^2(t)dt - \frac{1}{2}|\langle\Psi|\Psi_T\rangle|^2. \qquad (6)$$

Initially, the system is in its ground state and the target $|\Psi_T\rangle$ is chosen to be the fifth excited state. The system is simulated over a total time $T=500$; we set $\sigma=200$ and $e_0=1$ in Eq. (5). The system should be fully controllable and when $\mu=0$ the minimum possible value for J is -0.5, which corresponds to 100% overlap with the target state $|\Psi_T\rangle$; as $\mu>0$ increases, this global optimum value will also increase. The algorithms will be compared for several values of μ. In all cases a noisy cost functional $\tilde{J}$ is optimized which is computed from J by building noise into the field $\epsilon(t)$ through the amplitudes $a_1, \ldots, a_{16}$ and the phases $\theta_1, \ldots, \theta_{16}$. In order to simulate field noise these values are perturbed by $a_k \to a_k \cdot (1+0.02\cdot\eta_k^a)$, where η_k^a are independent, uniform, random variables taking values in $[-1;1]$ (this will be called 2% relative amplitude noise); similarly, the phases are randomly altered by $\theta_k \to \theta_k \cdot (1+0.01\cdot\eta_k^\theta)$ (1% relative phase noise). This produces a noise-contaminated modified field $\epsilon_m(t)$. The terms $\mu/2\int_0^T \epsilon_m^2(t)dt$ and $1/2|\langle\Psi|\Psi_T\rangle|^2$ of the cost functional for this modified field are then computed. The observation of the field fluence and the target were both taken to have a noise level of 0.05 (5% relative noise). Unless $(1+0.05)|\langle\Psi|\Psi_T\rangle|^2>1$ (see below), the two terms in the cost functional were multiplied by $(1+0.05\cdot\eta_1)$ and $(1+0.05\cdot\eta_2)$, respectively, where η_1, η_2 are uniform random variables with values in $[-1;1]$, such that

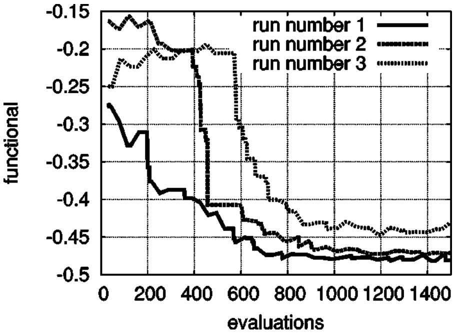

FIG. 1. Results for $\mu=0.0$ (simplex algorithm); the abscissa is the number of cost functional evaluations, and the ordinate is the noisy cost functional values for these typical runs. The relative noise levels are 1% for phases, 2% for amplitudes, and 5% observation noise. Good quality results are obtained with the cost functional converging close to the optimum value of -0.5. Similar results are obtained for other small weights such as $\mu=0.001$.

$$\tilde{J}(a_1, \ldots, a_{16}, \theta_1, \ldots, \theta_{16}) = (1+0.05\cdot\eta_1)\frac{\mu}{2}\int_0^T \epsilon_m^2(t)dt + (1+0.05\cdot\eta_2)\frac{1}{2}|\langle\Psi|\Psi_T\rangle|^2.$$

The value of $\tilde{J}$ is returned to the evaluation routine except when $(1+0.05)|\langle\Psi|\Psi_T\rangle|^2>1$, whereupon the relative noise for the target is modified to $1/|\langle\Psi|\Psi_T\rangle|^2-1$ in order to avoid unphysical noisy values above the maximum target overlap of 1.0. In this case the returned cost functional value is

$$\tilde{J}(a_1, \ldots, a_{16}, \theta_1, \ldots, \theta_{16}) = (1+0.05\cdot\eta_1)\frac{\mu}{2}\int_0^T \epsilon_m^2(t)dt + \left(1+\left(\frac{1}{|\langle\Psi|\Psi_T\rangle|^2}-1\right)\eta_2\right)\frac{1}{2}|\langle\Psi|\Psi_T\rangle|^2.$$

For both tested algorithms three runs are taken starting at random initial guesses; typical sample runs are shown in Figs. 1–3. In the laboratory an ensemble of experiments would be performed with the actual cost being the average over the ensemble.

The parameter μ has to be chosen carefully in order for the minimization of the cost functional to give a good target value. If μ is too large, such as $\mu=0.1$ in Fig. 3, the solution will converge toward the zero field, as too much weight is put on the fluence term $\int_0^T \epsilon(t)^2dt$. In the range $\mu \leqslant 0.001$ that gives satisfactory solutions for the control problem, both algorithms find many local minima that have overlaps with target typically in the range of 60%–90% (if μ is lowered this quality increases). In these cases both algorithms are sometimes trapped in a local minimum of lesser quality, but running several times eventually ensures finding acceptable quality results. When μ is lowered to $\mu=0.0001$ (and even more so for $\mu=0$), then good quality solutions are almost always found. The results for the two algorithms at $\mu=0.0$ are shown in Figs. 1 and 2. Both algorithms can give good

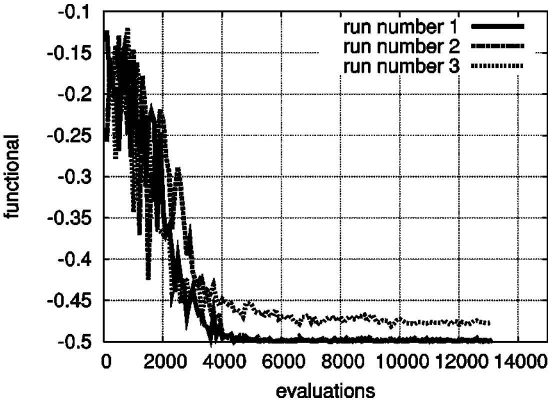

FIG. 2. Results for $\mu=0.0$ (GA); the abscissa is the number of cost functional evaluations, and the ordinate is the noisy cost functional values for a few typical runs. The relative noise levels are 1% for phases, 2% for amplitudes, and 5% for the observations. A good quality result is obtained with the cost functional converging close to the optimum value of -0.5. When compared with the simplex algorithm in Fig. 1, the GA is slower by a factor of between 5 and 10. Similar results are obtained for $\mu=0.001$.

target yields, but the simplex algorithm was generally an order of magnitude more efficient than the GA.

In order to test the robustness of the algorithms with respect to control and observation noise levels, further numerical experiments were carried out. The parameter μ was set to zero, and the relative noise was increased by a factor of 5 to become 5% phase noise, 10% amplitude noise, and 25% observation noise. The results are given in Figs. 4 and 5. Additional scenarios were also tested (e.g., noise levels ten times as large at 10% phase noise, 20% amplitude noise, and 50% observation noise), and the qualitative behavior was the same. The results are generally very robust to large degrees of noise, which appears consistent with recent analyses [40,41].

Signal averaging would generally be done in the laboratory to reduce the influence of noise. Such averaging would be effective for the observation noise, but the nonlinear influence of the control noise will still have a residual effect. As a conservative worst case scenario, no signal averaging was done in the present simulations [42]. One interesting conclusion drawn from these simulations is that recomputing the cost functional value at an apparently good quality point is important in order to detect faulty situations where the "good quality" is in fact due to random noise deviations. Note that the simplex algorithm takes provisions against this phenomenon: for any shrink step the *whole* simplex is reevaluated. These arguments were at the core of the modification introduced by Barton and Ivey [29] to the standard Nelder-Mead algorithm. The GA also implicitly takes into account this phenomenon if an appropriate replacement rule is chosen, for example requiring that the parents are necessarily replaced after one generation. The influence of noise needs to be considered in examining the fluctuating values of any single functional trajectory versus the number of evaluations in the figures, expecially at high noise levels.

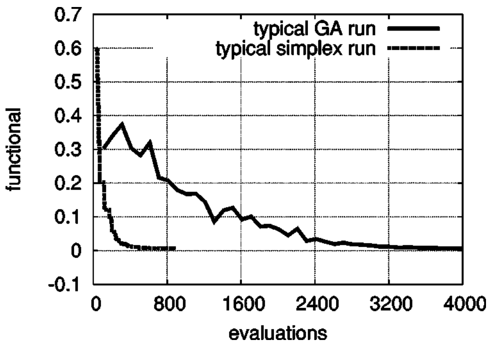

FIG. 3. Results for $\mu=0.1$; the abscissa is the number of functional evaluations and the ordinate is the noisy cost functional values. Relative noise levels: 1% for phases, 2% for amplitudes, and 5% output noise. Because μ is high no effective control is exhibited (i.e., there is no overlap with the target and all the algorithms converge to nearly the constant zero field).

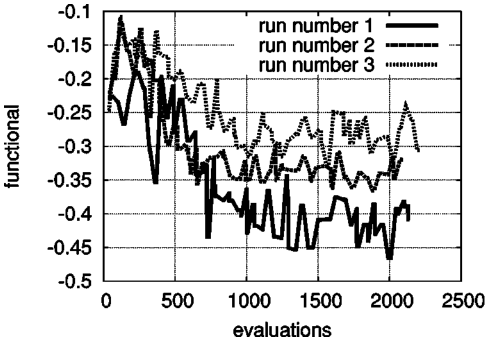

FIG. 4. Results for $\mu=0.0$ (simplex). The abscissa is the number of functional evaluations, and the ordinate is the noisy cost functional values. The relative noise levels are 5% for phases, 10% for amplitudes, and 25% for observation.

IV. CONSIDERATIONS REGARDING THE COST FUNCTIONAL SURFACE

As a follow-up of the numerical results of Sec. III, we were also able to attain insight into the cost functional surface from a new algorithm that has been designed to test the feasibility of generic gradient-free algorithms for optimization of quantum control cost functionals. This algorithm, called the Monte Carlo discrete gradient algorithm, has not been optimized for performance but rather is used to reveal some properties of the cost functional surface. The features of this search procedure are presented in the Appendix. This algorithm will operate by converging to the robust solution closest to the initial point as it does not have global search properties other than those brought by the random selection of the initial point (i.e., the Monte Carlo aspect of the algorithm). The Monte Carlo discrete descent algorithm was tested for $\mu=0$ and zero noise level. The results are given in Fig. 6. It was found that the Monte Carlo discrete descent algorithm converges to a good solution *for any randomly chosen initial point*. The conclusion that may be drawn from

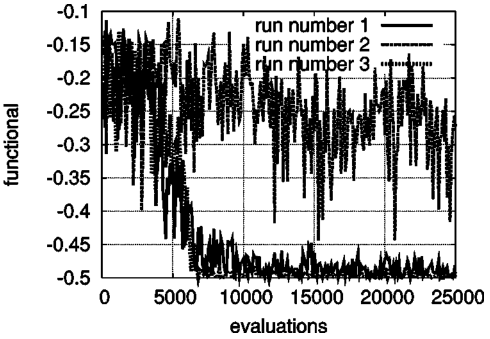

FIG. 5. Results for $\mu=0.0$ (GA). The abscissa is the number of functional evaluations, and the ordinate is the noisy cost functional values. The relative noise levels are 5% for phases, 10% for amplitudes, and 25% for observation noise.

this result is that many local minima exist with values close to that of the global optimum. This property has also been observed and analyzed in Ref. [43], and this behavior may not hold for general observables beyond simple projections to a single target state. This conclusion provides a possible basis to understand why the GA is less efficient than the simplex algorithm in this case: since the cost functional has many good quality minima, the global exploration properties of the GA are not needed. The GA does not have foreknowledge of this behavior and tends to explore the whole cost functional surface in the hope of finding yet a better solution, which does not exist. In contrast, the simplex algorithm less thoroughly explores the cost functional surface and still finds good results because close to any initial point there is a good quality solution.

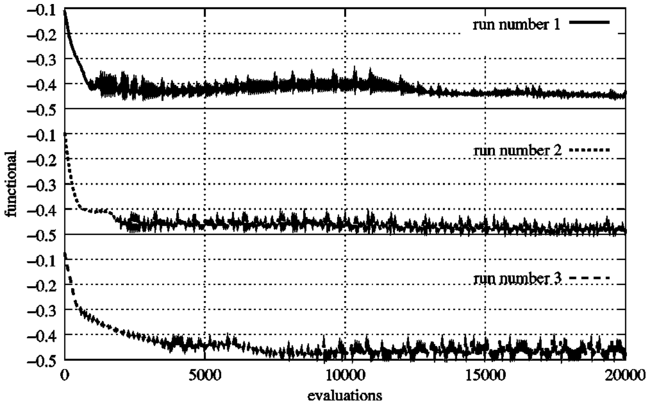

FIG. 6. Results for $\mu=0.0$ and no noise for the discrete descent algorithm $h=\xi=(0.01,\ldots,0.01)$. The abscissa is the number of functional evaluations, and the ordinate is the cost functional value. It was found that *all* random initial points converge to high-quality results of 90% overlap with target (this conclusion was found for many additional runs beyond those presented here). This conclusion remains partially valid as the noise level increases. The value of the cost functional for the first run increases over the period of approximately 5000 to 10 000 evaluations. This behavior arises from using the "reflection at the boundary" procedure (see Remark 3) in order to remain within predefined bounds on the controls. In this case, the minimum most close to initial (random) point is outside the chosen bounds. After a period of oscillations along the boundary the algorithm becomes oriented and converges toward another minimum inside the bounds.

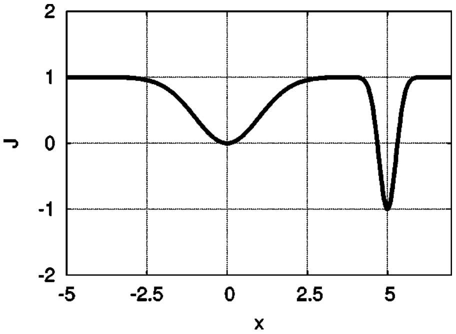

FIG. 7. Generic cost function $J(x)$ behavior with respect to a single variable x. The minimum at $x=5$ is deep, but narrower than that at $x=0$, and may not be robust at larger scales; for example, the $x=0$ minimum is robust for the scale $\|h\|=2.5$ but the one at $x=5$ is not. Both minima are robust for very small values of $\|h\|$, e.g., 0.01.

We also found that when there is no noise for $\mu=0$ both the GA and the modified simplex algorithm converge to high-quality solutions each time. However, as the GA and the simplex algorithms have known global optimization properties, this observation is less definitive about the nature of the functional surface than the conclusion drawn from the fact that the local Monte Carlo discrete descent algorithm converges to a good solution for any initial point.

V. DISCUSSION AND CONCLUSIONS

Optimization algorithms are an important part of the experimental protocols in the laser control of quantum phenomena. The present work demonstrates through numerical simulations on selected cases that genetic algorithms (and their descendants) are not the only algorithms that can perform efficiently in this context. An alternative, that is up to an order of magnitude faster (for the cases studied), is the modified simplex algorithm of Barton and Ivey [29]. One possible explanation of why this latter algorithm is better resides in it exploiting the multiplicity of solutions and only exploring a selected region of the cost functional surface. As a by-product of the search for good optimization algorithms, the present work introduced a discrete descent algorithm whose convergence properties gave additional evidence that indeed the local minima of the cost functional are of very good quality.

ACKNOWLEDGMENTS

The authors thank Marc Schoenauer and Anne Auger (INRIA Rocquencourt, France) for providing a version of the Evolving Objects platform [37]. C.L.B. and G.T. acknowledge financial support from the ACI "Laser control of chemical reactions" of the MENRT (France). H.R. acknowledges support from the U.S. Department of Energy.

APPENDIX: MONTE CARLO DISCRETE DESCENT ALGORITHM

1. Definition of the discrete gradient

Since an exact gradient of the cost functional $J(x_1,\ldots,x_N)$ is not available in practice, we will introduce a **discrete gradient**. Let $\xi=(\xi_1,\ldots,\xi_N)\in\mathbb{R}^N$ be a given vector such that $\xi_k\neq 0, k=1,\ldots,N$. We denote by e_k the vectors of the canonical base of $\mathbb{R}^N$: $e_1=(1,0,\ldots,0),\ldots,e_k=(0,\ldots,1,\ldots,0)$, $e_N=(0,\ldots,1)$, and define the discrete gradient of $J(x)$ at the point x with respect to the increments ξ by the formula

$$\nabla_\xi^d J(x)=\left(\frac{J(x+\xi_1e_1)-J(x)}{\xi_1},\ldots,\frac{J(x+\xi_ke_k)-J(x)}{\xi_k},\ldots,\frac{J(x+\xi_Ne_N)-J(x)}{\xi_N}\right). \tag{A1}$$

Note that in the limit $\xi\to 0$ the quantity thus computed is precisely the gradient of J at the point x, as each component of $\nabla_\xi^d J(x)$ converges to the corresponding partial derivative of $J(x)$. On the other hand, if ξ is fixed we can heuristically argue that the discrete gradient only sees details of $J(x)$ at the scale ξ as the discrete gradient is a finite-difference representation of the true gradient at the given fixed scale ξ. This is an important property which is consistent with the use we make of Eq. (A1), as in practice ξ is not meant to tend to zero.

2. Definition of the Monte Carlo discrete descent algorithm

Consider the increment vector $h=(h_1,\ldots,h_N)\in\mathbb{R}^N$, $\xi\in\mathbb{R}^N$. The algorithm is prescribed by the following steps:

(1) Choose a random point x^0 in the admissible parameter space $\mathbb{R}^N$; set $n=0$.

(2) Update the current solution by the formula

$$x^{n+1}=x^n-\frac{h*\nabla_\xi^d J(x^n)}{\|h*\nabla_\xi^d J(x^n)\|}\|h\|, \tag{A2}$$

where $h*\nabla_\xi^d J(x^n)$ is the element-wise product of the two vectors

$$h*\nabla_\xi^d J(x^n) = (h_1(\nabla_\xi^d J(x^n))_1,\ldots,h_k(\nabla_\xi^d J(x^n))_k,\ldots,h_N(\nabla_\xi^d J(x^n))_N), \tag{A3}$$

and the "discrete gradient" $\nabla_\xi^d J(\cdot)$ is defined in Eq. (A1).

(3) Unless a predefined stopping criterion is satisfied (e.g., an acceptable value of $J(x^{n+1})$ is reached) set $n=n+1$ and return to step 2 above.

Although the algorithm was defined here for different ξ and h, in practice we chose $\xi=h$.

Remark 1. In practice, in order to increase the performance of the algorithm above we use it on a population of K initial guesses $x_1^0,\ldots,x_K^0$; thus, we obtain K Monte Carlo discrete descent algorithms that run independently in parallel. Note that in our implementation — and in contrast with the GA approach — there is no cross talk between the K different running members. This "Monte Carlo" label characterizes the only stochastic element present in this algorithm.

Remark 2. Except for the random initialization and the constant norm increment properties, similar algorithms have been studied in Refs. [44,45]; related work (local linearization) is pursued in Ref. [46]. With respect to these works, the present study is different in that it is also aimed at exploring the geometry of the cost functional surface by choosing a simpler algorithm which is a particular case of the framework given in Ref. [45], except that there the step size $\|x^{n+1}-x^n\|$ can vary while here it is fixed.

Each of the K members of Monte Carlo discrete descent algorithmic search is expected to "converge" to the "robust" local optimum closest (at the scale $\|h\|$) to its initial point; here, robustness means that in a neighborhood of the local optimum the values of the cost functional are still of high quality. An illustration of the concept is given in Fig. 7.

This algorithm can be viewed as following the path of a ball of diameter ξ and moving at the constant "ground" speed $\|h\|$ on the cost function surface being minimized. The (lower bounded) speed and the nonzero diameter will tend to smooth the cost function surface.

3. Remarks on the algorithm

Note from Eq. (A2) that at each iteration the distance between two consecutive points x^n and x^{n+1} is the constant value $\|h\|$; this choice is designed for the purpose of avoiding long periods of small incremental steps to hopefully accelerate the convergence toward the solution.

Remark 3. If some components x_k of the argument x are required to be within predefined intervals, then these constraints are tested during the update step 2 in the definition of the algorithm. If $(x^{n+1})_k$ would not satisfy the required conditions, then the update formula is modified by a "reflection at the boundary:" $(x^{n+1})_k=(x^n)_k+h_k(\nabla_\xi^d J(x^n))_k\|h\|/\|h*\nabla_\xi^d J(x^n))\|$.

Three important cases can be considered concerning the relationship between ξ and h.

(i) $\|\xi\|<\|h\|$: this corresponds to having iteration steps larger than the steps used to compute the gradient (i.e., having a precise gradient that may eventually result in lower smoothness of the cost functional, for example where the first derivative of the cost functional is either not definite or very big). This corresponds to extrapolation;

(ii) $\|\xi\| > \|h\|$: the opposite circumstance, corresponding to interpolation; and

(iii) $\|\xi\| = \|h\|$: there is a "neighborhood" used both to compute the gradient and to advance to the next step. This is the case that has been used in all the computations of this paper. Note from $h = \xi$ it follows that

$$h * \nabla_h^d J(x) = (J(x + h_1 e_1) - J(x), \dots, J(x + h_k e_k) - J(x), \dots, J(x + h_N e_N) - J(x)).$$

In this case the update step can be written as $x^{n+1} = x^n - \rho_n \Sigma_{k=1}^N [J(x^n + h_k e_k) - J(x^n)] e_k$, where the renormalization factor ρ_n is used to enforce the constant norm increment $\|x^{n+1} - x^n\| = \|h\| = \text{constant}$.

[1] S. Shi, A. Woody, and H. Rabitz, J. Chem. Phys. **88**, 6870 (1988).

[2] M. Shapiro and P. Brumer, Acc. Chem. Res. **22**, 407 (1989).

[3] H. Rabitz, G. Turinici, and E. Brown, in *Computational Chemistry*, edited by C. Le Bris, Special Volume of Handbook of Numerical Analysis, Vol. X, edited by P. G. Ciarlet (Elsevier Science, New York, 2003), pp. 833–887.

[4] S. Rice and M. Zhao, *Optical Control of Quantum Dynamics* (Wiley, New York, 2000).

[5] A. Assion, T. Baumert, M. Bergt, T. Brixner, B. Kiefer, V. Seyfried, M. Strehle, and G. Gerber, Science **282**, 919 (1998).

[6] R. J. Levis, G. Menkir, and H. Rabitz, Science **292**, 709 (2001).

[7] T. Weinacht, J. Ahn, and P. Bucksbaum, Nature (London) **397**, 233 (1999).

[8] C. Bardeen, V. V. Yakovlev, K. R. Wilson, S. D. Carpenter, P. M. Weber, and W. S. Warren, Chem. Phys. Lett. **280**, 151 (1997).

[9] C. J. Bardeen, V. V. Yakovlev, J. A. Squier, and K. R. Wilson, J. Am. Chem. Soc. **120**, 13023 (1998).

[10] T. Hornung, R. Meier, and M. Motzkus, Chem. Phys. Lett. **326**, 445 (2000).

[11] J. Kunde, B. Baumann, S. Arlt, F. Morier-Genoud, U. Siegner, and U. Keller, Appl. Phys. Lett. **77**, 924 (2000).

[12] R. S. Judson and H. Rabitz, Phys. Rev. Lett. **68**, 1500 (1992).

[13] A generally accepted view is that GAs are efficient only when the stochastic operators have been adapted to the particular problem under study. It is thus surprising that GAs perform very well in this case generally *without* any tuning of the stochastic operators.

[14] There exist situations when the derivatives are so expensive, either to determine or to implement, that one prefers to avoid them.

[15] T. G. Kolda, R. M. Lewis, and V. Torczon, SIAM Rev. **45**(3), 385 (2003).

[16] Since the function evaluations are so cheap, one may ask why not use them to determine first derivatives through finite difference schemes. To some extent, the strategies we suggest reflect this remark, but only in a loose way. Indeed, calculating gradients often is not attractive in the optimization of highly nonconvex functionals such as the ones expected to exist in general quantum control problems.

[17] Inversion of quantum control data is one circumstance in which reducing the number of experiments is of interest. The inversion operations can be far more time consuming than the experiments, and a premium would be placed on the efficiency of the inversion algorithm. In this case the control field would be chosen to reduce the uncertainty in the extracted Hamiltonian information [18,19].

[18] J. M. Geremia and H. Rabitz, Phys. Rev. Lett. **89**, 263902 (2002).

[19] J. M. Geremia and H. Rabitz, J. Chem. Phys. **118**, 5369 (2003).

[20] R. M. Lewis and V. Torczon, SIAM J. Optim. **12**, 1075 (2002).

[21] P. D. Hough, T. G. Kolda, and V. J. Torczon, SIAM J. Sci. Comput. (USA) **23**, 134 (2001).

[22] R. M. Lewis and V. Torczon, SIAM J. Optim. **10**, 917 (2000).

[23] R. M. Lewis and V. Torczon, SIAM J. Optim. **9**, 1082 (1999).

[24] V. Torczon, SIAM J. Optim. **7**, 1 (1997).

[25] E. Anderson and M. Ferris, SIAM J. Optim. **11**, 837 (2001).

[26] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C++*, 2nd ed. (Cambridge University Press, Cambridge, 2002).

[27] C. Léonard, F. Le Quéré, P. Rosmus, C. Puzzarini, and M. de Lara Castells, Phys. Chem. Chem. Phys. **2**, 1117 (2000).

[28] C. Léonard, G. Chambaud, P. Rosmus, S. Carter, and N. Handy, Phys. Chem. Chem. Phys. **3**, 503 (2001).

[29] R. R. Barton and J. S. J. Ivey, Manage. Sci. **42**, 954 (1996). Following the notations of the paper, the modified simplex algorithm we used corresponds to the variants labeled "S9" and "RS."

[30] D. G. Humphrey and J. R. Wilson, INFORMS J. Comput. **12**, 272 (2000).

[31] Y. C. Ho, R. S. Sreenivas, and P. Vakili, Discrete Event Dyn. Syst. **2**, 61 (1992); URL: http://citeseer.nj.nec.com/ho96ordinal.html

[32] Y.-C. Ho and M. Larson, Discrete Event Dyn. Syst. **5**, 281 (1995); URL: http://citeseer.nj.nec.com/ho95ordinal.html

[33] M. Deng and Y.-C. Ho, Automatica J. IFAC **35**, 331 (1999).

[34] Y.-C. Ho, Inform. Sci. **113**, 169 (1999).

[35] This approach is based on the fact that the estimation of the relative ordering between two noisy values converges much more rapidly than the estimation of their difference. This remark is exploited by looking for ordinal rather than cardinal optimization, i.e., concentrating on finding good designs—relative to all possible designs—rather than on accurately estimating (by optimizing directly) the performance *value* of these designs. The solution obtained will therefore score conveniently in terms of percentile rank among all designs, but its precise performance with respect to the global optimum value is not known in advance.

[36] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning* (Addison-Wesley, Reading, MA, 1989).

[37] M. Keijzer, J. Merelo, G. G. Romero, and M. Schoenauer,

Evolving Objects: A General Purpose Evolutionary Computation Library, Proceedings of the 5th International Conference on Artificial Evolution, 29–31 October 2001, Université de Bourgogne, France; URL: http://eodev.sourceforge.net

[38] A. Törn and A. Zilinskas, *Global Optimization*, Lecture Notes in Computer Science, Vol. 350 (Springer, Berlin, 1989).

[39] B. Li, G. Turinici, V. Ramakhrishna, and H. Rabitz, J. Phys. Chem. B **106**, 8125 (2002).

[40] H. Rabitz, Phys. Rev. A **66**, 063405 (2002).

[41] J. M. Geremia, W. Zhu, and H. Rabitz, J. Chem. Phys. **113**, 10841 (2000).

[42] We also used the modified simplex algorithm on signal averaged surfaces, and we confirmed that it does converge to good quality solutions.

[43] H. Rabitz, M. Hsieh, and C. Rosental, Science **303**, 1998 (2004).

[44] G. Tóth, A. Lörincz, and H. Rabitz, J. Chem. Phys. **101**, 3715 (1994).

[45] P. Gross, D. Neuhauser, and H. Rabitz, J. Chem. Phys. **98**, 4557 (1993).

[46] M. Q. Phan and H. Rabitz, J. Chem. Phys. **110**, 34 (1999).