Krylov-subspace approach for the efficient control of quantum many-body dynamics

Martín Larocca[*] and Diego Wisniacki

Departamento de Física "J. J. Giambiagi" and IFIBA, FCEyN, Universidad de Buenos Aires, 1428 Buenos Aires, Argentina

The gradient ascent pulse engineering (GRAPE) algorithm is a celebrated control algorithm with excellent converging rates, owing to a piecewise-constant ansatz for the control function that allows for cheap objective gradients. However, the computational effort involved in the exact simulation of quantum dynamics quickly becomes a bottleneck limiting the control of large systems. In this paper, we experiment with a modified version of GRAPE that uses Krylov approximations (K-GRAPE) to deal efficiently with high-dimensional state spaces. Even though the number of parameters required by an arbitrary control task scale linearly with the dimension of the system, we find a constant elementary computational effort (the effort per parameter). Since the elementary effort of GRAPE is superquadratic, this speed up allows us to reach dimensions far beyond. The performance of the K-GRAPE algorithm is benchmarked in the paradigmatic XXZ spin-chain model.

I. INTRODUCTION

At the core of quantum technology is our ability to control quantum dynamics. In the last decades, we have evidenced unprecedented advances in the manipulation of dynamical processes at the atomic and molecular scale. The control is usually enforced by applying properly tailored external electromagnetic fields. One prosperous framework for producing these control fields is quantum optimal control (QOC) [1–3]. QOC methods have thrived in a range of emerging quantum technologies, e.g., communication, computation, simulation, and sensing, until now, at the level of a few qubits [4–8].

A notable member of QOC's "zoo" of algorithms [9–14] is the gradient ascent pulse engineering (GRAPE), first introduced in the context of NMR spectroscopy [15]. As its name suggests, it proposed a gradient-based optimization of the control protocols, as opposed to the derivative-free (finite-difference) approaches that were commonly used at the time. The key to GRAPE's success was to propose a piecewise-constant (PWC) ansatz for the control that in turn allowed for cheap gradients of the objective. Gradient-based algorithms usually have much better convergence than the gradient-free algorithms [16]. Its ability to produce high-quality optimal controls in an inexpensive and fast fashion made it the state-of-the-art algorithm in quantum control.

Despite its successful application in small systems, QOC methods encounter severe limits when applied to many-body quantum systems. Due to the exponential complexity of simulating the latter, control algorithms fail to yield a desired final state within an acceptable computational time. There are basically two approaches for the efficient simulation of quantum evolution. The first kind assumes that entanglement will be small during the whole evolution and uses a truncated representation of the state vector. This is the realm of tensor network methods, such as the density-matrix renormalization group evolution or tDMRG [17]. The second alternatives are Krylov-subspace methods. These circumvent the computationally impracticable task of diagonalizing the full Hamiltonian, a key step in the computation of matrix exponentials arising in the treatment of time evolution, by considering only a reduced number of effective energy levels [18–20]. Just to illustrate the power of the method, Ref. [21] reports having been able to simulate time evolution in Hilbert subspaces of dimensions up to 9 billion, using parallel supercomputers.

A number of proposals for controlling many-body dynamics following the first approach were made [22–24]. References [22,23] used a matrix-product-state ansatz with a derivative-free approach to drive a superfluid–Mott-insulator transition in an optical lattice. The latter [24] revisited the problem using gradient-based optimization and achieved much better fidelities. Let us note that this task, the connection of ground states on both sides of a phase transition, is perfectly suited for such a low-entanglement ansatz. Nevertheless, more general control scenarios may require full state descriptions.

In this paper, we explore the possibility of using the second alternative in a control context. We apply a modified GRAPE algorithm that uses Krylov approximations instead of the full Hamiltonian eigendecomposition. As a particular example, we try to control pure-state transitions on an XXZ spin-chain model. Fixing the number of effective levels, we are able to locate optimal protocols with a computational effort per parameter roughly independent of the size of the system. Its dimension only affects the search effort through the number of parameters that are required for control solutions to exist, a quantity that grows linearly with such. In a D-dimensional Hilbert space, at least $2D - 2$ parameters are needed to control pure-state transfers [25–27]. This *minimum number of parameters* is related to the informational content of the control field (basically, $2D - 2$ is the number of real numbers needed to completely specify an arbitrary pure state in a quantum

---
[*]Corresponding author: larocca@df.uba.ar

system with D energy levels). Because the method uses the entire representation of quantum states, we are able to manipulate arbitrary entangled states. Let us note that a recent work proposed the use of Trotter-Suzuki approximations of the propagator in order to deal efficiently with the propagation problem [28].

The paper is organized as follows. In Sec. II we introduce the modified algorithm, hereafter referred to as K-GRAPE. In Sec. III, the algorithm is carefully tested to control an *XXZ* spin-chain model. In this section we describe the control task and the numerical results. Finally, in Sec. IV, we draw conclusions on the results obtained. To make the paper self-contained, we have a included Appendixes on Krylov's approximation and on the GRAPE algorithm.

II. THE K-GRAPE ALGORITHM

Krylov-subspace methods are well-known linear algebra techniques usually used to approximate the action of the function of a matrix on a vector [18,19,29]. The Krylov subspace of a Hamiltonian H and a state $|\psi\rangle$ is defined as

$$\mathcal{K}_N = \{|\psi\rangle, H|\psi\rangle, \ldots, H^{N-1}|\psi\rangle\}. \quad (1)$$

In the following, we will use Krylov subspaces of dimension $N \ll D$, but let us stress that the case $N = D$ (the so-called Krylov space) does not necessarily correspond to the full Hilbert space of the system. Only systems with a nondegenerate energy spectrum explore the entire state space and have Krylov spaces that coincide with the full Hilbert space. An orthonormal basis spanning $\mathcal{K}_N$ can be used to perform extremely cheap and accurate approximations to the time evolution of a state. The basic mechanism first maps the initial state into the ground state of an effective N-level system [see Fig. 1(a)]. Time evolution is computed in this reduced space (and thus very efficiently) using T_N, the projection of the Hamiltonian in the Krylov basis $|\phi\rangle_N = e^{-iT_N\Delta t}|0\rangle_N$. Finally, the map is inverted: The actual "full-dimensional" evolved state is recovered as a linear combination of the Krylov basis vectors, using the amplitudes of the subspace-evolved $|\phi\rangle_N$ as coefficients. For details on this procedure, see Appendix A. We will proceed to introduce the control scenario.

Consider a typical controlled system, where the Hamiltonian

$$H(t) = H_d + \epsilon(t)H_c \quad (2)$$

is tunable through the time-dependent control $\epsilon(t)$. Here, H_d and H_c are usually addressed as the *drift* and *control* terms, respectively, and their nested commutators determine the degree of *controllability* of the system [1]. In short, how much of Hilbert space can be dynamically explored by arbitrary choices of $\epsilon(t)$. The shaping of this function, also referred to as the *protocol* or *control field* (experimentally, control is usually enforced through dipole couplings with electromagnetic fields), will allow us to govern the evolution of the system. For example, consider the situation where the system is initially in a given state $|i\rangle$ and we are interested in a dynamics that prepares the target state $|f\rangle$ at time $t = T$. In order to search for controls that accomplish the task, one has to introduce a figure of merit quantifying the degree of fulfillment. In terms of the overlap $\beta = \langle f| U(T)|i\rangle = \langle f|\psi(T)\rangle$, we define the
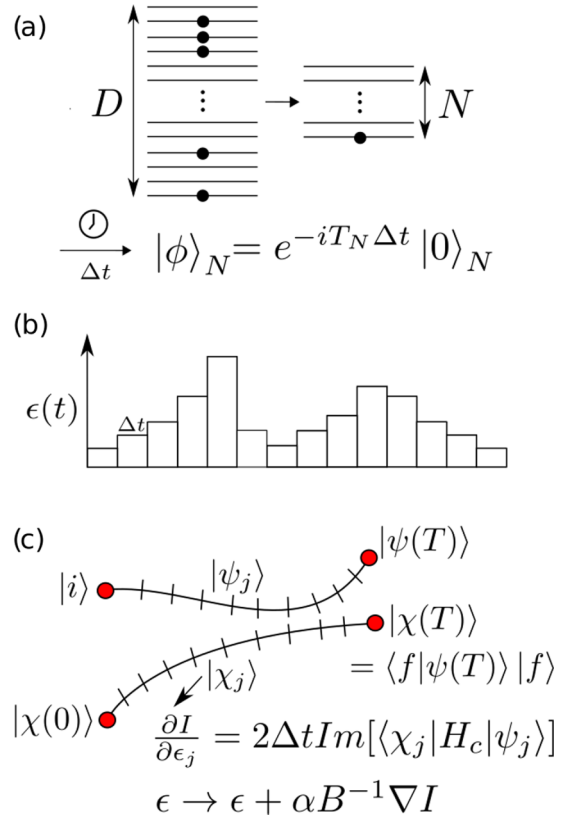

FIG. 1. Artist's impression. (a) Krylov's approximation: An initial state with an arbitrary distribution on the full D-dimensional state space is mapped into the ground state of an effective N-level system. This ground state evolves under the action of T_N, the Hamiltonian in the reduced Krylov basis. Finally, the evolved state on the full Hilbert space is recovered with the inverse mapping. (b) GRAPE's ansatz: A piecewise-constant control field. (c) The zeroth-order approximation of the gradient is equivalent to the bracket of the control Hamiltonian between the forward evolved initial state and the backward evolved target-projected final state at each time slot. At each iteration, the control field ϵ is updated using the gradient ∇I, the inverse of an approximated Hessian B, and a backtracked step length α.

infidelity

$$I = 1 - |\beta|^2. \quad (3)$$

Here (and throughout), we have avoided stating explicitly the dependence of I, β and U on the protocol $\epsilon(t)$, given by the fact that the latter is the solution of

$$i\frac{dU(t)}{dt} = H[\epsilon(t)]U(t), \quad U(0) = \mathbb{1}, \quad (4)$$

evaluated at $t = T$. The functional of Eq. (3) is a map from the space of protocols to the interval [0,1] of real numbers. This map is usually known as the quantum control landscape [30,31]. Solutions to the control problem are global minima of the landscape. Thus, the problem of finding suitable controls is equal to a minimization process in this infinite-dimensional space. In practice, optimal control techniques introduce a parametrization on the field and thus constrain the problem to finite-dimensional search spaces. For example, the GRAPE

Algorithm 1. Basic GRAPE algorithm.

1: Guess initial control amplitudes $\{\epsilon_j\}$
2: **while** not converged **do**
3:     compute propagators $U_j \leftarrow e^{-iH_j \Delta t}$
4:     forward evolve $|\psi_j\rangle \leftarrow U_j |\psi_{j-1}\rangle$
5:     set $|\chi_M\rangle \leftarrow \langle f| \psi_M |f\rangle$
6:     backward evolve $|\chi_j\rangle \leftarrow U_j^\dagger |\chi_{j+1}\rangle$
7:     evaluate gradient $\frac{\partial I}{\partial \epsilon_j} \approx -2\Delta t\, \mathrm{Im}[\langle \chi_j| H_c |\psi_j\rangle]$
8:     update amplitudes $\epsilon \leftarrow \epsilon + \alpha B^{-1} \nabla I$ and go to 3

---

algorithm uses a PWC parametrization of the control, namely,

$$
\epsilon(t) = \begin{cases} \epsilon_1 & \text{if } 0 < t < \Delta t, \\ \vdots & \\ \epsilon_M & \text{if } (M-1)\Delta t < t < T. \end{cases} \tag{5}
$$

Here, the protocol duration was divided in M uniform intervals $\Delta t = T/M$ at which the control is constant [see Fig. 1(b)]. This ansatz has several advantages. First, the propagator factorizes into a product of individual subpropagators, each of which is generated by a constant Hamiltonian and thus has a simple matrix exponential form. More importantly, the derivatives of the objective with respect to the controls are also extremely simple, in particular to first order in Δt (see Appendix B for a detailed derivation),

$$
\frac{\partial I}{\partial \epsilon_j} \approx -2\Delta t\, \mathrm{Im}[\langle \chi_j| H_c |\psi_j\rangle], \tag{6}
$$

where $|\chi_j\rangle = \beta U_{j+1}^\dagger \cdots U_M^\dagger |f\rangle$ and $|\psi_j\rangle = U_j \cdots U_1 |i\rangle$ can be pictured as forward and backward propagated states. More precisely, $|\chi_j\rangle$ is the final time-evolved state $|\psi(T)\rangle = U|i\rangle = |\psi_M\rangle$, projected into the target state and backward evolved [see Fig. 1(c)]. A sketch of GRAPE's framework is presented in Algorithm 1. A general update rule uses a positive-definite matrix B and the gradient ∇I to generate a search direction $p = B^{-1}\nabla I$ and then takes an appropriate step of magnitude α in that direction. In the simplest situation (steepest descent) B is the identity matrix and thus the step is taken in the direction of the gradient. Newton's method instead uses the Hessian: $B = \nabla^2 I$. There are two problems with this: (i) Computing the Hessian is expensive, and (ii) the Hessian may not be positive definite and thus have no inverse. The Broyden-Fletcher-Goldfarb-Shanno (BFGS) method [32] uses the gradient to build very cheap approximations of the Hessian (the approximation is not built entirely at each iteration, but instead only updated where relevant) that in turn are positive definite. Modern implementations of GRAPE use the L-BFGS method [33], a limited-memory variant that achieves superlinear convergence in a time- and memory-efficient manner. At each iteration, the step α is chosen with a backtracking routine such that it satisfies Wolfe's conditions [16].

GRAPE quickly became widely used in the control community. It offered the possibility of optimizing control pulses in much larger control spaces than those allowed by finite-difference gradients. Note that these require at least $M+1$ full time evolutions (in the forward-difference setting), as compared with the two evolutions required by GRAPE. Over the years, several enhancements to the method were proposed

[4,34–36]. In particular, it was realized that the eigenvalues and eigenvectors of each of the constant Hamiltonians, used to compute the matrix exponentials for the objective, could be cached and reused to compute exact gradients [4]. This setting is faster, but for large system dimensions it quickly becomes impractical in terms of memory, considering we have to store M (one for each parameter) D-dimensional matrices and that the number of parameters scales at least linearly with D. In short, it requires storing more and bigger matrices.

In this work, we focus on the possibility of slightly modifying the GRAPE algorithm to perform efficiently in the near many-body regime by replacing the exact forward and backward state propagations with Krylov approximations. We use a centered version of the zeroth-order gradient approximation [Eq. (6)] which we feed to an L-BFGS routine. We note that, although there is plenty of room for playing with more accurate higher-order approximations [see Eq. (B10)] or the numerical integration of the natural gradient [Eq. (B8)], we found the simple centered zeroth-order gradient to be more than enough for an initial demonstration of the virtues of the proposed method. In the following, we define a particular control setting and compare the performance of GRAPE and K-GRAPE algorithms. We use a built-in implementation of GRAPE offered by the python toolbox QuTiP [37,38], that is based on the DYNAMO package [4]. In order to be able to reach large dimensions, we chose not to cache the propagator gradients and instead let the fidelity computer calculate them as needed. Let us point out that the GRAPE implementation offered by the SPINACH package [39] uses a similar approach to deal efficiently with the dynamics of large systems. Instead of a Lanczos procedure, a Taylor expansion is used to propagate the state.

III. EXAMPLE: CONTROLLING AN *XXZ* HEISENBERG SPIN CHAIN

In this section, we test K-GRAPE in the well-known *XXZ* spin chain. We benchmark the procedure comparing its performance with the usual GRAPE algorithm.

A. Spin-chain model and control task

Let us consider a one-dimensional system consisting of L spin-$\frac{1}{2}$ particles whose drift evolution is described by the *XXZ* Hamiltonian [40],

$$
H_d = \frac{J}{2} \sum_{i=1}^{L-1} \sigma_i^x \sigma_{i+1}^x + \sigma_i^y \sigma_{i+1}^y + \alpha_z \sigma_i^z \sigma_{i+1}^z. \tag{7}
$$

Here, $\sigma_i^{x,y,z}$ are the Pauli matrices for the ith particle, and we have set $\hbar = 1$ such that energy is measured in units of the interaction strength J and time is measured in units of J^{-1}. This Hamiltonian has a number of symmetries. First, it conserves the total magnetization in the z direction, $\sigma_z = \sum_{i=1}^{L} \sigma_i^z$, since $[H_d, s_z] = 0$. This allows us to fracture the entire state space into subspaces S_K of fixed number of excitations,

$$
\mathcal{H} = \bigoplus_{K=0}^{L} \mathcal{S}_K. \tag{8}
$$

The dimension of these subspaces is simply the number of ways of picking K excitations out of L sites, and is given by

$$D_K = \binom{L}{K} = \frac{L!}{K!(L-K)!}. \tag{9}$$

Additionally, the Hamiltonian in Eq. (7) conserves parity. The parity operator acts on a given computational state by "mirroring" against the middle of the chain, e.g.,

$$\Pi \left| \downarrow\downarrow\uparrow\downarrow \right\rangle = \left| \downarrow\uparrow\downarrow\downarrow \right\rangle. \tag{10}$$

Because the couplings in H_d are homogeneous, these two operators commute $[H_d, \Pi] = 0$. Thus, each excitation subspace S_k is further broken into two parity-excitation (PE) subspaces, $S_{k,+}$ and $S_{k,-}$, with even and odd parity, respectively. Finally, we will avoid conservation of S^2 by choosing $\alpha_z = 0.5$ and set the coupling strength to $J = 1$.

As a control Hamiltonian, we use

$$H_c = \frac{J}{2}\left(\sigma_1^z + \sigma_L^z\right) \tag{11}$$

such that the total Hamiltonian $H(t)$ in Eq. (2) still commutes with σ_z and Π for any choice of control function. Initial states that are eigenstates of these operators evolve constrained to the PE subspaces, thus allowing us to use a reduced representation in the computations. The Hamiltonian in the reduced PE subspace is constructed in the following way: (i) Write the parity operator in a given excitation subspace S_K and (ii) diagonalize it, (iii) use the eigenvectors associated with the desired parity (we use even parity throughout) to build a rectangular change of basis matrix Q, and (iv) reduce the Hamiltonian, $H_{\text{red}} = QHQ^\dagger$. The computational basis vectors of the reduced excitation subspace are ordered relative to the binary number associated with each sequence of bits (considering spin-ups as zeros and spin-downs as ones), in ascending order.

As a control task, we will attempt to drive the first coordinate vector in a given PE subspace into the last one,

$$e_1 \xrightarrow[\epsilon(t)]{} e_D. \tag{12}$$

In the particular case of an odd number of spins and an even number of excitations, this task corresponds to the transport of a cluster of excitations from the middle of the chain to the edges, for example,

$$
\begin{aligned}
e_1 &= \left| \downarrow \cdots\cdots \downarrow\uparrow \cdots \uparrow\downarrow \cdots\cdots \downarrow \right\rangle, \\
e_D &= \frac{1}{\sqrt{2}}(\left| \uparrow \cdots \uparrow\downarrow \cdots\cdots \downarrow \right\rangle + \left| \downarrow \cdots\cdots \downarrow\uparrow \cdots \uparrow \right\rangle).
\end{aligned}
\tag{13}
$$

Let us mention that we have performed numerical controllability tests (using the nested commutators of the reduced H_0 and H_c, as described in Ref. [1]) and found the PE subspaces to be controllable.

B. Numerical results

In the following we present a numerical study of the performance of K-GRAPE on the spin-chain pure-state transfer task defined above, using different choices of length L and
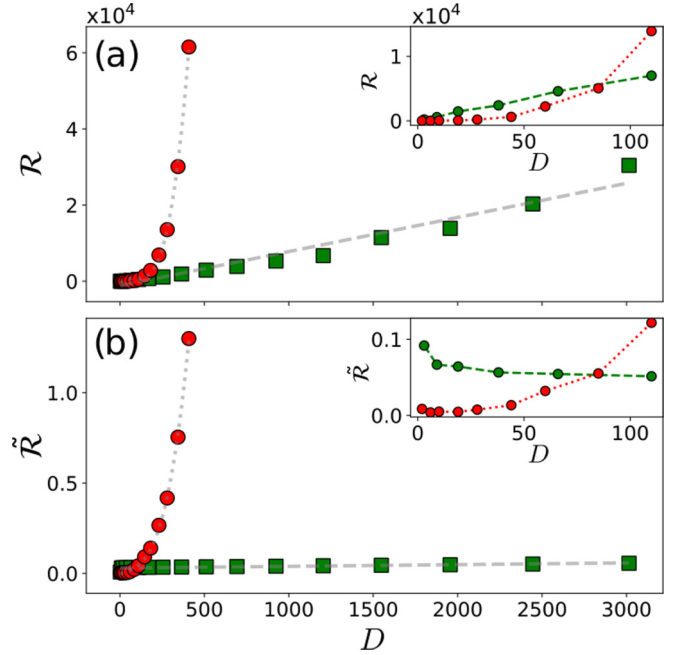

FIG. 2. Algorithm benchmarking: (a) Optimization runtime $\mathcal{R}$ and (b) elementary runtime $\tilde{\mathcal{R}}$ (runtime per iteration per time step) for the state transfer task in Eq. (12) using GRAPE (dotted line and red squares) and K-GRAPE (dashed line and green squares) as a function of D, the dimension of the subspace holding the dynamics. The data points corresponding to GRAPE and K-GRAPE are fitted with linear and a cubic functions, respectively. In the first low-dimensional regime GRAPE outperforms (see the inset), while in the large-dimensional one, K-GRAPE does.

excitations K. The results are properly compared with the control using the GRAPE algorithm. We initialize random fields with $M = 4D$ parameters drawn from a uniform distribution in $[-1, 1]$. This linear scaling of M with dimension is roughly twice the strictly needed.[1] We further fix $\Delta t = .5$ and truncate at $N = 10$ Krylov vectors. A target infidelity of $I_{\text{target}} = 10^{-2}$ is set and as an additional stopping criterion, the minimum change in the objective from one iteration to the next is chosen to be $\Delta I_{\min} = 10\text{EPS}$, with EPS the machine precision. In the following, we present runtime data, which was measured in seconds.[2]

Figure 2(a) shows the total runtime $\mathcal{R}$ consumed by GRAPE (dotted line and red squares) and K-GRAPE (dashed line and green squares) as a function of dimension D. Each data point corresponds to a single successful optimization. We note that almost no traps were observed with these settings. The results correspond to chains with $K = 3$ excitations in the case of GRAPE and $K = 4$ in the case of K-GRAPE. Other values of K were tested and were found to have no influence on the results. On the first low-dimensional regime, GRAPE outplays (see the inset). Instead, for $D \geqslant 100$, K-

---

[1]There is a lower bound on the minimum number of parameters for solutions to arbitrary pure-state transfer problems to exist [27].

[2]The optimizations were carried out on an Intel(R) Core(TM) i7-8550U CPU @ 1.80 GHz with 16GB of RAM.

GRAPE is clearly more efficient.[3] The gray lines connecting the data are linear and cubic fits for K-GRAPE and GRAPE, respectively. Figure 2(b) provides the "elementary" runtime $\tilde{\mathcal{R}}$, defined as the runtime per field evaluation (iteration) and per time step. This elementary runtime is independent of dimension for K-GRAPE and is at least quadratic for GRAPE. This is reasonable since K-GRAPE uses a fixed size effective Hamiltonian while GRAPE has to deal with D-dimensional matrices. GRAPE's effort is well explained in terms of the eigendecompositions at its core [41]. In turn, these elementary efforts account for the total runtimes observed. Since both algorithms perform a number of iterations that is roughly independent of D, the only dependence of the full runtime with dimension is through the number of parameters, that must grow linearly with dimension to satisfy the control constraint. Thus, for example, K-GRAPE's constant elementary effort is translated into a linear scaling. Let us note that we have checked that the error between exact and Krylov propagated final states is orders of magnitude below the target infidelity.

The performance of K-GRAPE depends critically on the choice of time step Δt. On one hand, the algorithm is built upon an approximated gradient that works optimally in the low Δt regime. On the other, there is a minimum time $T_{\min}$ (lower bounded by the quantum speed limit time) such that the control problem has solutions. That is, with a too-small value of Δt (and a fixed number of parameters) the algorithm will converge properly but will not be able to attain good fidelities simply because they do not exist. Finally, the quality of the gradient is further tied to the quality of the Krylov approximation, which depends on a large-enough truncation N and, again, on a small-enough time step Δt.

In order to study the behavior of K-GRAPE with the time step, we initialize and optimize 20 seeds for different values of N and Δt. The number of parameters is still fixed at $M = 4D$ (and will be fixed throughout) and we use $D = 60$. Figure 3(a) shows the minimum infidelity achieved as a function of Δt for $N = 2$, 10, and 18 (blue squares, green circles, and red diamonds, respectively). The yellow pluses correspond to an exact evolution (no Krylov) and a centered zeroth-order gradient. As mentioned above, there is a trade-off in Δt: Small values prohibit the state transfer while large values compromise the gradient and thus convergence. In the middle there is a "control window." Notice how too-small values of truncation (e.g., $N = 2$) also damage the gradient and in consequence no window is observed. Instead, for $N = 18$ we find a broader window than in the case of $N = 10$. Figure 3(d) displays the elementary effort, which is observed to grow with N. The mean number of iterations, shown in Fig. 3(c), is seen to present a "bump" in the region around $\Delta t_{\min} = T_{\min}/M \approx 0.25$. We think this could be related to a blossoming of traps in the control landscape [42]. To the right of this maximum, the iterations decrease and then grow back again, this time
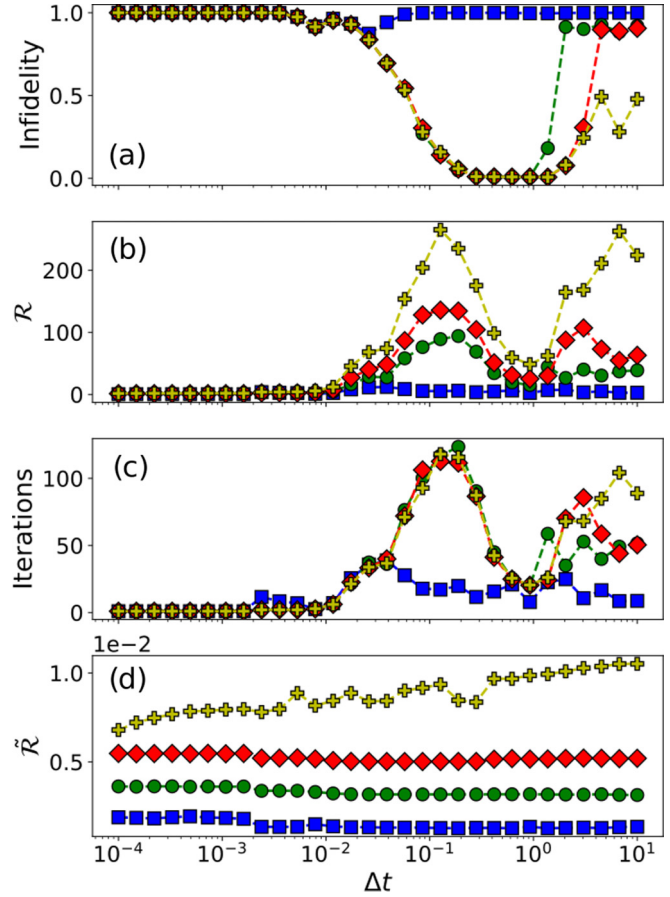
---

[3]If the caching of propagator gradients is enabled, GRAPE performs much faster but still superquadratically, meaning the intersection with K-GRAPE is only pushed forward. We built a runtime curve (similar to those of Fig. 2) up to $D = 255$ (where the routine collapsed due to memory overloading), fitted the data, and estimated the crossing to be at $D = 400$.
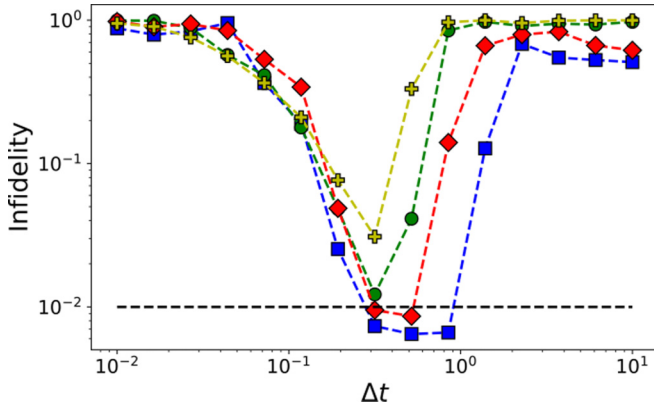

FIG. 3. Time-step study: (a) Minimum infidelity, (b) mean runtime $\mathcal{R}$, (c) mean iterations, and (d) $\tilde{\mathcal{R}}$ runtime per iteration per time step, as a function of Δt. The different curves correspond to different values of the truncation parameter, $N = 2$, 10, and 18, marked with blue squares, green circles, and red diamonds, respectively. The yellow pluses correspond to the centered zeroth-order gradient, evaluated using an exact evolution. See text for details.

due to the growing inaccuracies in the gradient. Finally, these aspects merge in the total run-time observed [Fig. 3(b)]. Note that the observed $\Delta t_{\min}$ corresponds to $T_{\min} \approx D$. We have numerically checked this relation to hold for a wide range of dimensions. In particular, the benchmarking study of Fig. 2 is consistent with this estimation, since we chose $\Delta t = 0.5$ (twice this minimum value) and found solutions every time. We cannot give a precise explanation of why this is the case, but here is an attempt. If one considers the quantum speed limit time associated with the drift Hamiltonian (that in this case is $\tau_{qsl} \approx 1$) as a bound on the time needed to reach an orthogonal state, and one assumes that, in the worst-case scenario, the trajectory explores all the $D - 1$ orthogonal states before reaching the desired target, we can argue that the minimum control time is bounded by $T_{qsl} \approx D - 1$ [43,44].

Let us further characterize the control windows observed in Fig. 3(a). To do so, we repeat the previous study, this time as a function of dimension (see Fig. 4). We plot the minimum achieved infidelity as a function of time step Δt for a fixed truncation of $N = 6$ and different dimensions $D = 10$, 19, 44, and 146 (marked with blue squares, green circles, red

FIG. 4. "Death" of a control window: Minimum infidelity as a function of time step Δt for a fixed truncation of $N = 6$ and different dimensions $D = 10, 19, 44,$ and 146 (blue squares, green circles, red diamonds, and yellow pluses).

diamonds, and yellow pluses). We find that while the truncation is enough to control systems of dimensions $D = 10$ and 19 (see how the blue and red dots do find infidelities below the target, plotted as a black dotted line), the control window "shutters" for greater dimensions. We emphasize that, due to this behavior, we do not expect the linear behavior of Fig. 2 to continue forever. At some large value of dimension, we expect an increase on the number of iterations (owing to the buildup of imprecisions in the gradient) followed by a loss of controllability.

To conclude, we present a study of the truncation parameter N. Figure 5(a) shows curves of total runtime $\mathcal{R}$ involved in achieving an optimal control field (within the desired fidelity), as a function of dimension, for different values of truncation: $N = 8, 10,$ and 12, marked with blue squares, green circles, and red diamonds, respectively. Since the elementary runtime $\tilde{\mathcal{R}}$ [Fig. 5(c)] is flat with D, and since the number of iterations [Fig. 5(b)] is roughly independent of N, the total runtime grows roughly linearly with N [in accordance with Fig. 2(a)]. An exception is the case of $N = 8$. Here, for dimensions $D > 38$, the window starts to close, the iterations grow, and the total runtime becomes superlinear. To avoid this situation, either the number of parameters should be increased (such that $\Delta t_{\min} = T_{\min}/M$ decreases, pulling away the lower edge of the window) or the gradient should be made more precise (kicking forward its upper one). Either choice involves further computations, evidencing a trade-off between window width and effort.

IV. FINAL REMARKS

Krylov-subspace methods have been delivering fruitful insights and advancements in several areas of research that go from optimization theory to the characterization of operator complexity in chaotic many-body quantum systems [45–48]. Important for the context of this paper, several recent works have exploited its extreme efficiency at simulating time evolution on large quantum systems [19,21].

The GRAPE algorithm is an acclaimed quantum optimal control method that has enabled the efficient production of high-quality protocols to actively guide the dynamics
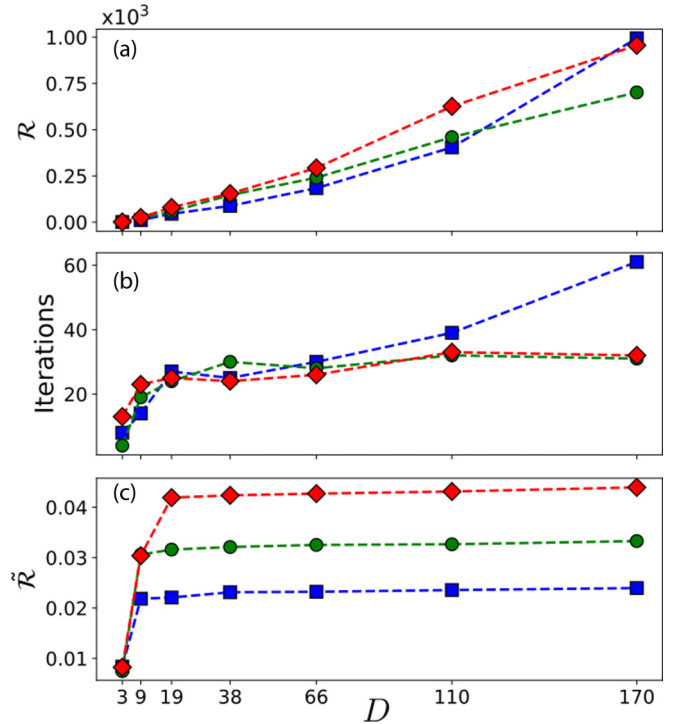

FIG. 5. Truncation study: (a) Runtime $\mathcal{R}$, (b) iterations, and (c) elementary runtime $\tilde{\mathcal{R}}$ involved in the achievement of controls as a function of dimension D. The different curves correspond to different values of the truncation parameter $N = 8, 10,$ and 12, marked with blue squares, green circles, and red diamonds, respectively.

of quantum systems. Unfortunately, the performance of GRAPE is seriously hindered when reaching out of the small-dimensional regime.

In this work, we have studied the performance of an algorithm combining these two worlds. Powered by efficient Krylov approximations, K-GRAPE is best suited in the near many-body regime, where optimization becomes problematic with traditional methods. We tested the algorithm in an XXZ spin-chain model and demonstrated its ability to find control solutions at dimensions far beyond the capabilities of standard GRAPE. We showed that its intrinsic complexity is independent of the dimension of the problem, as opposed to GRAPE's quadratic scaling. This speed-up allows us to control systems of dimensions, clearly exceeding what was available. We want to emphasize that in no way our algorithm breaks the exponential scaling presented by many-body systems.

To finish, we note the suitability of K-GRAPE's framework to be adapted to operator control. This straightforward generalization is very interesting since, for example, it would grant access to the efficient design of control protocols in large open quantum systems. This investigation will be part of a forthcoming publication.

Algorithm 2. Lanczos algorithm. Receives a Hamiltonian H and a state $|\psi\rangle$ and returns a set of N orthonormal vectors $\{|v_i\rangle\}$, the Krylov basis.

1: $|v_0\rangle \leftarrow |\psi\rangle$
2: **for** $j > 0$ **do**
3:     $|x_j\rangle \leftarrow H |v_{j-1}\rangle$
4:     $|\omega_j\rangle = |x_j\rangle - \sum_{k=j-2}^{j-1} \langle v_k| |x_j\rangle |v_k\rangle$
5:     $b_j = \sqrt{\langle\omega_j| |\omega_j\rangle}$
6:     **if** $b_j > 0$ **then**
7:         $|v_j\rangle \leftarrow \frac{1}{b_j} |\omega_j\rangle.$

APPENDIX A: THE KRYLOV APPROXIMATION

In this Appendix, we describe in detail the mechanism involved in the Krylov approximation. To begin with, consider the construction of $B_N = \{|v_0\rangle, \ldots, |v_{N-1}\rangle\}$, an orthonormal basis spanning the Krylov subspace. This can be done using the Lanczsos method (see Algorithm 2). Notice that we only have to explicitly remove the components of $|x_j\rangle$, the new *candidate* basis vector, on the last two elements of the basis, $|v_{j-1}\rangle$ and $|v_{j-2}\rangle$. The reason for this is that the Hamiltonian, by construction, is tridiagonal in the Krylov basis. Moreover, the coefficients appearing in the orthonormalization procedure, $\langle v_{j-2}| |x_j\rangle$ and $\langle v_{j-1}| |x_j\rangle$, are the off-diagonal and diagonal entries of such a tridiagonal matrix. Of course, it may occur that the new vector is linearly dependent with respect to the previous ones. This is referred to as a *happy breakdown* [29] and indicates the finding of an invariant subspace. The Lanczos algorithm halts, since no new elements are needed to describe the evolution. The dynamics explores only a restricted portion of the state space and thus its description involves only a reduced number elements.

It is important to note that if this procedure is used to build large Krylov bases, round-off errors intrinsic to floating-point arithmetic may cause loss of orthogonality between the basis vectors. An obvious way to handle this problem is to explicitly orthonormalize the new candidate against all previous vectors, as in a standard Gram-Schmidt procedure. This can become fairly expensive time and memory wise. More elaborate alternatives involve keeping track of the orthogonality loss and only perform the reorthonormalizations when needed [49].

Krylov bases have an obvious application in the time evolution of a state,

$$|\psi(t+\Delta t)\rangle = e^{-iH\Delta t}|\psi(t)\rangle \approx |\psi_N(t+\Delta t)\rangle. \quad \text{(A1)}$$

Here, $|\psi_N(t+\Delta t)\rangle \in B_N$ is a cheap yet excellent approximation of the evolved state. Let us explain how to build it. Following Ref. [19], first consider the projector onto the truncated basis, $P_N = \sum_{j=0}^{N-1} |v_j\rangle \langle v_j| = V_N^{\dagger} V_N$, where $V_N^{\dagger}$ is the (D, N) change of basis matrix between the reduced N-dimensional Krylov basis and the original D-dimensional basis,

$$V_N^{\dagger} = \begin{bmatrix} \vdots & \vdots & & \vdots \\ |v_0\rangle, & |v_1\rangle, & & |v_{N-1}\rangle \\ \vdots & \vdots & & \vdots \end{bmatrix}. \quad \text{(A2)}$$

The method proceeds by locating the element $|\psi_N(t+\Delta t)\rangle \in B_N$ that is closest to the exact evolved state. This is carried out by considering the evolution with a projected propagator,

$$|\psi(t+\Delta t)\rangle \approx P_N e^{-iH\Delta t} P_N |\psi(t)\rangle = V_N^{\dagger} e^{-iT_N \Delta t} V_N |\psi(t)\rangle, \quad \text{(A3)}$$

where $T_N = V_N H V_N^{\dagger}$ is the Hamiltonian in the Krylov basis. By construction, V_N maps the initial state into the ground state of an N-dimensional system, $V_N |\psi(t)\rangle = (1, 0, \ldots, 0)^T \equiv |0_N\rangle$. This state evolves subject to T_N, populating these effective levels, and is finally mapped back to the full original space (see Fig. 1).

Numerous estimations and bounds to the error in this approximation exist [18,29,50]. Most of them are based under the assumption of exact arithmetic and are usually too pessimistic to explain the numerically observed error reductions. Nevertheless, we have numerically checked that in the small Δt regime (in particular when $\Delta t < \frac{N^2}{W}$ with W the spectral width of the Hamiltonian) the dependence is $O(\Delta t^N)$ [19].

APPENDIX B: THE GRAPE ALGORITHM

Let us review the GRAPE algorithm. The PWC ansatz for the protocols [see Eq. (5)] induces a factorization on the propagator

$$U(T) = U_M \cdots U_1, \quad \text{(B1)}$$

where each of these subpropagators is generated by the constant Hamiltonian $H_j = H_0 + \epsilon_j H_c$, correspondent with a given time slot, and thus has a simple matrix exponential form

$$U_j = e^{-iH_j \Delta t}. \quad \text{(B2)}$$

Moreover, the derivatives of the objective with respect to the controls are also extremely simple. First note that the gradient of the objective is related to the gradient of the propagator through

$$\nabla I = -2 \operatorname{Re}\langle f| \nabla U(T) |i\rangle \langle i| U^{\dagger}(T) |f\rangle. \quad \text{(B3)}$$

Now, the derivatives of the propagator only affect the corresponding subpropagator

$$[\nabla U]_j = \frac{\partial U}{\partial \epsilon_j} = U_M \cdots U_{j+1} \frac{\partial U_j}{\partial \epsilon_j} U_{j-1} \cdots U_1, \quad \text{(B4)}$$

and the problem is reduced to the computation of

$$\frac{\partial U_j}{\partial \epsilon_j} = -i\Delta t \bar{H}_c U_j, \quad \text{(B5)}$$

where

$$\bar{H}_c = \frac{1}{\Delta t} \int_0^{\Delta t} U_j(-\tau) H_c U_j(\tau) d\tau. \quad \text{(B6)}$$

That is, the gradient reads

$$\frac{\partial I}{\partial \epsilon_j} = -2\Delta t \operatorname{Im}[\langle \chi_j| \bar{H}_c |\psi_j\rangle] = \int_{\Delta t_j} g(t) dt$$
$$= \int_0^{\infty} g(t) \sqcap_j (t) dt, \quad \text{(B7)}$$

where in the last lines we have made explicit that the partial derivatives of the objective with respect to the parameters in

the pulse are simply the projections of the so-called *natural gradient* [25,51]

$$g(t) \equiv \frac{\delta I}{\delta \epsilon(t)} = -2 \, \text{Im}[\langle \chi(t) | H_c | \psi(t) \rangle] \quad \text{(B8)}$$

on the basis functions of the PWC parametrization

$$\sqcap_j(t) = \begin{cases} 1 & \text{if } t \in \Delta t_j, \\ 0 & \text{else.} \end{cases}$$

Note that the numerical integration of Eq. (B7) is a straightforward alternative for computing arbitrary precise approximations to the gradient. This option seems particularly suitable for K-GRAPE since, once the Krylov basis has been built for a given time step (and the effective Hamiltonian has been diagonalized), evaluating the evolved state on multiple points on a time grid is virtually free.

Another possibility for improving the quality of the gradient is to Taylor expand the exponentials in Eq. (B6). A hierarchy of approximations to $\bar{H}_c$ unfolds,

$$\bar{H}_c^{(p)} = \frac{(-i\Delta t)^p}{(p+1)!} \mathcal{L}^p H_c, \quad \text{(B9)}$$

where we have introduced the Liouvillian operator $\mathcal{L} \equiv [H, \cdot]$ (Here, of course, the Hamiltonian corresponding to the corresponding time-slot should be used $H \equiv H_j$.) This is ultimately translated into a hierarchy of approximations to the gradient

$$\frac{\partial I^{[P]}}{\partial \epsilon_j} = -2\Delta t \, \text{Im}\big[\langle \chi_j | \bar{H}_c^{[P]} | \psi_j \rangle \big], \quad \text{(B10)}$$

where the notation $[P]$ implies that we sum over $0 \leqslant p \leqslant P$,

$$\bar{H}_c^{[P]} = \sum_{p=0}^{P} \bar{H}_c^{(p)}. \quad \text{(B11)}$$

A common practice for computing the matrix exponentials in Eq. (B2) is to perform an eigendecomposition of the Hamil-

tonian and to invoke the spectral theorem,

$$U_j = \sum_{k=1}^{D} e^{-i\lambda_j \Delta t_j} | \lambda_k \rangle \langle \lambda_k | = QDQ^\dagger, \quad \text{(B12)}$$

where $|\lambda_k\rangle$ are the eigenvectors of H_j, λ_k its eigenvalues, and

$$\begin{aligned} Q^\dagger &= [|\lambda_1\rangle, \ldots, |\lambda_D\rangle], \\ D &= \text{diag}(e^{-i\lambda_1 \Delta t}, \ldots, e^{-i\lambda_M \Delta t}). \end{aligned} \quad \text{(B13)}$$

An alternative way of computing these matrix exponentials is using Padé approximations. Although slightly more expensive, the eigendecomposition has a lot more to offer, since it provides exact derivatives of the propagators

$$\frac{\partial U_j}{\partial c_j} = Q\tilde{H}_c F Q^\dagger, \quad \text{(B14)}$$

where $\tilde{H}_c = Q^\dagger(-iH_c)Q$ and

$$[F]_{jk} = -i\Delta t \, \langle \lambda_j | H_j | \lambda_k \rangle \begin{cases} \omega_1 & \text{if } 0 < t < \Delta t, \\ \vdots \\ \omega_M & \text{if } (M-1)\Delta t < t < T. \end{cases}$$

Finally, let us propose a simple way of improving the quality of the zeroth-order approximation to the gradient [Eq. (6)]. Consider the first component of the standard zeroth-order gradient,

$$\frac{\partial I}{\partial \epsilon_1} \approx \frac{\partial I^{(0)}}{\partial \epsilon_1} = -2\Delta t \, \text{Im}[\langle f | U_M \cdots H_c U_1 | i \rangle \beta^*]. \quad \text{(B15)}$$

This is precisely Δt times the natural gradient of Eq. (B8) evaluated at $t = \Delta t$. By virtue of Eq. (B7) we know that the exact calculation involves the integration of the natural gradient in the whole time domain of the pulse [see Eq. (B7)]. We thus propose a centered version of the approximation, where the natural gradient is still assumed constant in the interval, but evaluated at $t = \Delta t/2$, the center of the pulse,

$$\frac{\partial I}{\partial c_j} \approx \frac{1}{2}\bigg[\frac{\partial I^{(0)}}{\partial c_j} + \frac{\partial I^{(0)}}{\partial c_{j+1}} \bigg]. \quad \text{(B16)}$$

We find this error to be $O(\Delta t^3)$ instead of the characteristic $O(\Delta t^2)$ of the standard zeroth order.

[1] D. D'Alessandro, *Introduction to Quantum Control and Dynamics*, Chapman & Hall/CRC Applied Mathematics & Nonlinear Science (Taylor & Francis, London, 2007).

[2] C. Brif, R. Chakrabarti, and H. Rabitz, New J. Phys. **12**, 075008 (2010).

[3] S. J. Glaser, U. Boscain, T. Calarco, C. P. Koch, W. Köckenberger, R. Kosloff, I. Kuprov, B. Luy, S. Schirmer, T. Schulte-Herbrüggen, D. Sugny, and F. K. Wilhelm, Eur. Phys. J. D **69**, 279 (2015).

[4] S. Machnes, U. Sander, S. J. Glaser, P. de Fouquières, A. Gruslys, S. Schirmer, and T. Schulte-Herbrüggen, Phys. Rev. A **84**, 022305 (2011).

[5] R. Schutjens, F. A. Dagga, D. J. Egger, and F. K. Wilhelm, Phys. Rev. A **88**, 052330 (2013).

[6] P. Rembold, N. Oshnik, M. M. Müller, S. Montangero, T. Calarco, and E. Neu, AVS Quantum Sci. **2**, 024701 (2020).

[7] N. Wittler, F. Roy, K. Pack, M. Werninghaus, A. S. Roy, D. J. Egger, S. Filipp, F. K. Wilhelm, and S. Machnes, arXiv:2009.09866.

[8] M. Abdelhafez, B. Baker, A. Gyenis, P. Mundada, A. A. Houck, D. Schuster, and J. Koch, Phys. Rev. A **101**, 022321 (2020).

[9] T. Caneva, T. Calarco, and S. Montangero, Phys. Rev. A **84**, 022326 (2011).

[10] S. Machnes, E. Assémat, D. Tannor, and F. K. Wilhelm, Phys. Rev. Lett. **120**, 150401 (2018).

[11] J. J. W. H. Sørensen, M. O. Aranburu, T. Heinzel, and J. F. Sherson, Phys. Rev. A **98**, 022119 (2018).

[12] J. P. Palao and R. Kosloff, Phys. Rev. A **68**, 062308 (2003).

[13] M. Bukov, A. G. R. Day, D. Sels, P. Weinberg, A. Polkovnikov, and P. Mehta, Phys. Rev. X **8**, 031086 (2018).

[14] J.-S. Li, J. Ruths, and D. Stefanatos, J. Chem. Phys. **131**, 164110 (2009).

[15] N. Khaneja, T. Reiss, C. Kehlet, T. Schulte-Herbrüggen, and S. J. Glaser, J. Magn. Reson. **172**, 296 (2005).

[16] J. Nocedal and S. Wright, *Numerical Optimization* (Springer, Berlin, 2006).

[17] U. Schollwöck, Ann. Phys. **326**, 96 (2011).

[18] M. Hochbruck and C. Lubich, SIAM J. Numer. Anal. **34**, 1911 (1997).

[19] S. Paeckel, T. Köhler, A. Swoboda, S. R. Manmana, U. Schollwöck, and C. Hubig, Ann. Phys. **411**, 167998 (2019).

[20] I. Kuprov, J. Magn. Reson. **195**, 45 (2008).

[21] M. Brenes, V. K. Varma, A. Scardicchio, and I. Girotto, Comput. Phys. Commun. **235**, 477 (2019).

[22] P. Doria, T. Calarco, and S. Montangero, Phys. Rev. Lett. **106**, 190501 (2011).

[23] S. van Frank, M. Bonneau, J. Schmiedmayer, S. Hild, C. Gross, M. Cheneau, I. Bloch, T. Pichler, A. Negretti, T. Calarco, and S. Montangero, Sci. Rep. **6**, 34187 (2016).

[24] J. H. M. Jensen, F. S. Møller, J. J. Sørensen, and J. F. Sherson, arXiv:2008.06076.

[25] M. Hsieh, T.-S. Ho, and H. Rabitz, Chem. Phys. **352**, 77 (2008).

[26] V. Beltrani, J. Dominy, T.-S. Ho, and H. Rabitz, J. Chem. Phys. **134**, 194106 (2011).

[27] K. W. Moore and H. Rabitz, J. Chem. Phys. **137**, 134113 (2012).

[28] J. H. M. Jensen, F. S. Møller, J. J. Sørensen, and J. F. Sherson, arXiv:2005.09943.

[29] R. B. Sidje, ACM Trans. Math. Softw. **24**, 130 (1998).

[30] R. Chakrabarti and H. Rabitz, Int. Rev. Phys. Chem. **26**, 671 (2007).

[31] M. Larocca, E. Calzetta, and D. A. Wisniacki, Phys. Rev. A **101**, 023410 (2020).

[32] W. C. Davidon, SIAM J. Optim. **1**, 1 (1991).

[33] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, SIAM J. Sci. Comput. **16**, 1190 (1995).

[34] G. Ciaramella, A. Borzì, G. Dirr, and D. Wachsmuth, SIAM J. Sci. Comput. **37**, A319 (2015).

[35] P. de Fouquieres, S. Schirmer, S. Glaser, and I. Kuprov, J. Magn. Reson. **212**, 412 (2011).

[36] D. L. Goodwin and I. Kuprov, J. Chem. Phys. **143**, 084113 (2015).

[37] J. Johansson, P. Nation, and F. Nori, Comput. Phys. Commun. **184**, 1234 (2013).

[38] J. Johansson, P. Nation, and F. Nori, Comput. Phys. Commun. **183**, 1760 (2012).

[39] H. Hogben, M. Krzystyniak, G. Charnock, P. Hore, and I. Kuprov, J. Magn. Reson. **208**, 179 (2011).

[40] P. M. Poggi and D. A. Wisniacki, Phys. Rev. A **94**, 033406 (2016).

[41] V. Y. Pan and Z. Q. Chen, in *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, STOC '99* (ACM, New York, 1999), pp. 507–516.

[42] M. Larocca, P. M. Poggi, and D. A. Wisniacki, J. Phys. A: Math. Theor. **51**, 385305 (2018).

[43] P. M. Poggi, F. C. Lombardo, and D. A. Wisniacki, Europhys. Lett. **104**, 40005 (2013).

[44] M. Murphy, S. Montangero, V. Giovannetti, and T. Calarco, Phys. Rev. A **82**, 022318 (2010).

[45] N. Gould, S. Lucidi, M. Roma, and P. Toint, SIAM J. Optim. **9**, 504 (1999).

[46] G. von Winckel and A. Borzi', Comput. Phys. Commun. **181**, 2158 (2010).

[47] D. E. Parker, X. Cao, A. Avdoshkin, T. Scaffidi, and E. Altman, Phys. Rev. X **9**, 041017 (2019).

[48] E. Rabinovici, A. Sánchez-Garrido, R. Shir, and J. Sonner, arXiv:2009.01862.

[49] H. D. Simon, Math. Comput. **42**, 115 (1984).

[50] Y. Saad, SIAM J. Numer. Anal. **29**, 209 (1992).

[51] N. Rach, M. M. Müller, T. Calarco, and S. Montangero, Phys. Rev. A **92**, 062343 (2015).