Hessian-based optimization of constrained quantum control

Mogens Dalgaard,[1] Felix Motzoi,[2] Jesper Hasseriis Mohr Jensen,[1] and Jacob Sherson[1,*]
[1]*Department of Physics and Astronomy, Aarhus University, Ny Munkegade 120, 8000 Arhus C, Denmark*
[2]*Forschungszentrum Jülich, Institute of Quantum Control (PGI-8), D-52425 Jülich, Germany*

(Received 1 June 2020; revised 16 September 2020; accepted 29 September 2020; published 21 October 2020)

Efficient optimization of quantum systems is a necessity for reaching fault-tolerant thresholds. A standard tool for optimizing simulated quantum dynamics is the gradient-based GRAPE algorithm, which has been successfully applied in a wide range of different branches of quantum physics. In this work, we derive and implement exact second-order analytical derivatives of the coherent dynamics and find improvements compared to the standard of optimizing with the approximate second-order Broyden-Fletcher-Goldfarb-Shanno algorithm. We demonstrate performance improvements for both the best and the average errors of constrained unitary gate synthesis on a circuit-QED system over a broad range of different gate durations.

DOI: 10.1103/PhysRevA.102.042612

I. INTRODUCTION

Nowadays, there exists a wide range of proposed technologies that utilize the potential of quantum mechanics in order to achieve improvements over their classical counterparts. These include quantum variational eigensolvers [1,2], annealers [3], simulators [4,5], Boltzman machines [6], and, perhaps most promising, the quantum computer [7]. We may reach a point in time where the majority of these quantum-based technologies outperform their classical counterparts.

Reaching this *quantum advantage* requires, among other things, substantial improvements in our ability to control the underlying quantum systems. On the theoretical side, quantum optimal control theory addresses this issue [8–10]. Here optimization methods with respect to a chopped random basis (CRAB) [11,12] and individual pulse amplitudes (Krotov) [13,14] have been successful especially with respect to unconstrained quantum optimization, where trapping has been shown to rarely occur [15,16].

One of the most used, and widely successful, algorithms within quantum optimal control theory is the gradient ascent pulse engineering (GRAPE) algorithm [17,18]. The original GRAPE algorithm used first-order approximated gradients in combination with the steepest descent [17]. Later, significant improvements were obtained when the analytical gradients were calculated and combined with Hessian approximation methods such as the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm [18–20]. GRAPE has been widely successful providing its use in nuclear magnetic resonance [21–25], superconducting qubit circuits [26–30], spin chains [31–34], nitrogen-vacancy centers [35,36], and ultracold atoms [37,38]. It has also become a standard integrated tool in many numerical packages aimed at quantum physicists [19,39–41]. Further, one could mention the many extensions of GRAPE, which treat filtering [42], robustness [17,43], chopped random basis [44], experiment design [45], and open quantum systems [46].

In addition, there exists many hybrid algorithms that combine GRAPE with, e.g., sequential updates [19] and global optimization algorithms [47,48]. Machine-learning-based control [49–55] could also improve the solution exploration of local quantum optimal control methods such as GRAPE, where initial steps towards a hybrid algorithm have already been taken in Ref. [55].

In this paper we show how GRAPE can further be enhanced to significantly increase its ability to reach high-fidelity solutions, by incorporating the exact Hessian in the numerical optimization. We present here an efficient calculation of the Hessian and apply both gradient- and Hessian-based optimization to unitary gate synthesis for superconducting circuits. A previous calculation of the Hessian made use of the auxiliary matrix method [56,57]. This approach requires the exponentiation of block matrices having three times the size of the Hilbert space, which leads to an unfavorable scaling with respect to both the Hilbert space size and the number of controls. In this work, we present a derivation that only requires exponentiation of matrices with the same size as the Hilbert space. In doing so, we show that every component needed to evaluate the gradient can be recycled to evaluate the Hessian and thus that calculating an element of the Hessian is not much more expensive than evaluating the gradient. These results enable us to demonstrate improvements with Hessian-based GRAPE in equal wall-time simulations without the explicit need for any code parallelization. In addition to enhancing optimization, the Hessian can also be used to characterize the quantum optimal control landscape, which in Ref. [58] was done in the presence of noise.

Besides calculating the Hessian, we further seek to benchmark analytical gradient- and Hessian-based numerical optimization within constrained physical settings. Physically realizable pulses are always constrained in amplitude due to limitations of experimental equipment, but constraints are more often imposed to avoid undesirable processes such as

[*]sherson@phys.au.dk

leakage, ionization, heating, decoherence, and break down of theoretical models. For many quantum control problems, optimal solutions contain segments that lie on the boundary of the admissible region (see, e.g., Refs. [59,60]). Under certain assumptions, this has even been proven to more generally occur [61]. Therefore, it is important how the optimization algorithms handle constraints.

For the work presented here, we demonstrate our methods by synthesizing the unitary dynamics of a superconducting transmon circuit [62]. The circuit consists of two fixed frequency transmon qubits dispersively coupled through a microwave resonator [45,63,64]. This setup could be used as a subpart of many of the aforementioned quantum technologies.

II. OPTIMAL CONTROL OF UNITARY GATES

Our objective is to generate a target unitary V using a set of piecewise constant pulses. However, it should be mentioned that the calculations given here are applicable to many other control problems as well. These include state-to-state transfer for pure quantum states [65] (see also Appendix A) and density matrices in closed quantum systems, as well as state-to-state transfer of density matrices in open quantum systems [17,19].

The pulses consist of N constant-amplitude segments and have a total duration, T. We assume access to a set of M control Hamiltonians $\{H_k\}_{k=1}^M$ such that a bilinear Hamiltonian, $H(t_j) = H_0 + \sum_{k=1}^M c_{j,k} H_k$, governs the system dynamics at time step j. Here H_0 and $c_{j,k}$ denote the drift Hamiltonian and controls, respectively. We have depicted an example of a control pulse in Fig. 1(a). The system starts from an initial unitary U_0, which is typically chosen to be the identity, and then evolves through the unitary evolution $U_j = \exp[-iH(t_j)\Delta t]$, where $\Delta t = T/N$ and $\hbar = 1$.

Replicating the target unitary V, up to a global phase, is achieved by maximizing the fidelity

$$\mathcal{F} = \left| \frac{1}{\text{dim}} \text{Tr}[\mathbb{P} U \mathbb{P} V^\dagger] \right|^2, \tag{1}$$

where $U = U_N U_{N-1} \dots U_0$ denotes the final unitary and $\mathbb{P}$ is a projection into the subspace of interest, whose dimension is denoted as dim. This approach exploits that quantum propagation can always be decomposed smoothly into shorter, differential time segments. Equivalently, we seek to minimize the infidelity $J = 1 - \mathcal{F}$, which serves as our cost function. We express a control vector as $\mathbf{c} = (c_{1,1}, c_{2,1}, \dots, c_{N,1}, c_{1,2}, \dots, c_{N,M})$, where the first index denotes the discretized time and the second denotes different controls. Starting from an initial guess, $\mathbf{c}_0$, we seek to make incremental updates such that $J(\mathbf{c}_{n+1}) < J(\mathbf{c}_n)$, with n denoting the iteration number. The incremental update is of the form

$$\mathbf{c}_{n+1} = \mathbf{c}_n + \alpha_n \mathbf{p}_n. \tag{2}$$

Here $\mathbf{p}_n$ defines a search direction, while α_n defines a step size typically found through a line search. Figure 1(a) illustrates an example of an incremental update depicted with arrows. The initial seed could, e.g., stem from an analytical ansatz or be based on a random guess to explore the optimization landscape. The latter approach is termed multistarting [66].

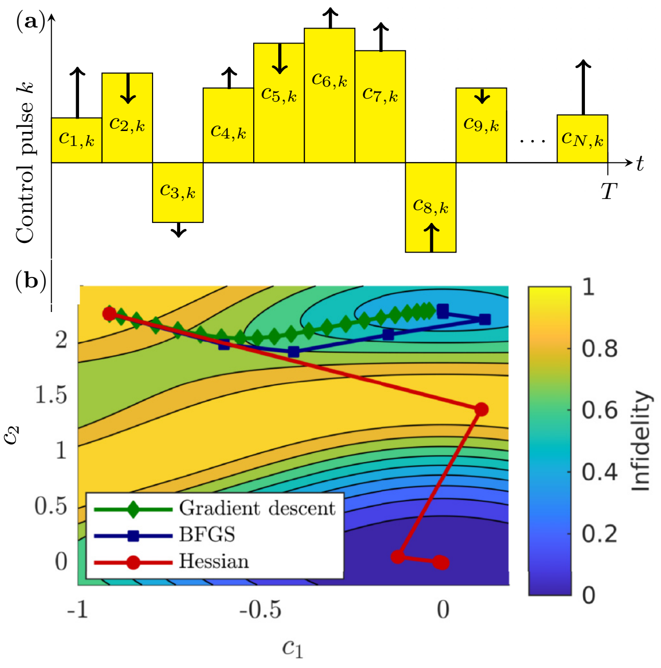

FIG. 1. (a) An example of a piecewise constant control pulse. Here individual updates are depicted as arrows. (b) An example of how Interior-Point with either BFGS or the Hessian searches differently. In addition we also compare to a simple gradient-descent algorithm. We elaborate on this figure in Appendix F.

The search direction taken by GRAPE was originally proposed to be the steepest descent, $\mathbf{p}_n = -\nabla J(\mathbf{c}_n)$, where the gradient was approximated to first order in Δt [17]. The steepest descent only uses information about the first derivative and thus suffers from having no information about the curvature of the optimization landscape. Hence, it is often superior to use the second-order derivatives, i.e., the Hessian matrix

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 J}{\partial c_{1,1}^2} & \frac{\partial^2 J}{\partial c_{1,1} \partial c_{2,1}} & \cdots & \frac{\partial^2 J}{\partial c_{1,1} \partial c_{N,M}} \\ \frac{\partial^2 J}{\partial c_{2,1} \partial c_{1,1}} & \frac{\partial^2 J}{\partial c_{2,1}^2} & \cdots & \frac{\partial^2 J}{\partial c_{2,1} \partial c_{N,M}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 J}{\partial c_{N,M} \partial c_{1,1}} & \frac{\partial^2 J}{\partial c_{M,N} \partial c_{2,1}} & \cdots & \frac{\partial^2 J}{\partial c_{N,M}^2} \end{bmatrix}. \tag{3}$$

If one has access to the Hessian, a standard search direction is given by Newton"s method $\mathbf{p}_n = -[\mathbf{H}(\mathbf{c}_n)]^{-1} \nabla J(\mathbf{c}_n)$. If the second-order derivatives are not available, one can approximate the Hessian $\mathbf{B} \approx \mathbf{H}$ via the BFGS Hessian-approximation scheme [20] that gradually builds $\mathbf{B}$ using only the gradient. The iterative BFGS update is of the form

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{\mathbf{y}_k \mathbf{y}_k^T}{\mathbf{y}_k^T \mathbf{s}_k} - \frac{\mathbf{B}_k \mathbf{s}_k \mathbf{s}_k^T \mathbf{B}_k^T}{\mathbf{s}_k^T \mathbf{B}_k \mathbf{s}_k}, \tag{4}$$

where $\mathbf{y}_k = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k)$ and $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$. The derivation and use of an exact gradient enabled the precision needed for BFGS to outclass its first-order counterpart [18]. It is now the standard within quantum optimal control theory [19,40,41].

Since physically realizable pulses are always constrained in amplitude, we optimize via a constrained optimization algorithm that uses either the actual Hessian or the gradient-based BFGS (see Appendix D). In Fig. 1(b) we illustrate the difference between these methods (Hessian-based and gradient-only) for a simple two-dimensional optimization problem (see Appendix F for a technical explanation). The problem has a global minimum at $(c_1, c_2) = (0, 0)$. In addition, there is also a local one above. We start the optimization near the two solutions and plot how each method moves through the optimization landscape. The two gradient-based approaches converge toward the local solution, while the Hessian-based method manages to avoid this solution and converge towards the global optimum. A possible explanation for this behavior is that Hessian-based optimization generally has more information about the landscape curvature. This can allow it to avoid suboptimal nearby solutions like the one depicted in Fig. 1(b). As we demonstrate later, the two different approaches (gradient-only and Hessian based optimization) often find different solutions although starting from the same initial pulse. However, we would like to stress that the Hessian is not guaranteed to find the best solution of the two.

Moreover, Fig. 1(b) also illustrates how Hessian-based optimization generally requires fewer iterations to converge [67]. If the cost function is sufficiently simple near an optimal solution, $\mathbf{x}^*$, a nearby initial guess (seed), $\mathbf{x}_0$, will converge as $||\mathbf{x}_{k+1} - \mathbf{x}^*|| \leqslant C||\mathbf{x}_k - \mathbf{x}^*||^q$, where $C \geqslant 0$ for the optimization methods discussed here, with the Hessian being quadratic ($q = 2$) and BFGS being superlinear ($1 < q < 2$) [67].

In the following we derive exact analytical expressions for both the gradient and the Hessian of Eq. (1). The gradient has previously been derived (see, e.g., Refs. [18,19]), but is reproduced here for completeness.

A. Gradient

The first step is to express the derivative of the fidelity, Eq. (1), in terms of the derivative of the unitary:

$$\frac{\partial \mathcal{F}}{\partial c_{j,k}} = \frac{2}{\dim^2}\mathrm{Re}\left[\mathrm{Tr}\left(\mathbb{P}\frac{\partial U}{\partial c_{j,k}}\mathbb{P}V^\dagger\right)\mathrm{Tr}(V\mathbb{P}U^\dagger\mathbb{P})\right]. \tag{5}$$

It is only the unitary at the jth time step, U_j, that depends on $c_{j,k}$ and hence the derivative is

$$\frac{\partial U}{\partial c_{j,k}} = U_N \dots \frac{\partial U_j}{\partial c_{j,k}} \dots U_1 U_0 = U^L_{j+1}\frac{\partial U_j}{\partial c_{j,k}}U^R_{j-1}, \tag{6}$$

where we have defined the left and right unitaries as $U^L_j = U_N U_{N-1}\dots U_j$ and $U^R_j = U_j U_{j-1}\dots U_0$, respectively. The left and right unitaries must be calculated for each time step, but this can be done efficiently since $U^L_j = U^L_{j-1}U_j$ and $U^R_j = U_j U^R_{j-1}$. Thus, the gradient calculations scale as $O(N)$ in terms of the number of matrix multiplications rather than as the intuitive $O(N^2)$.

In order to evaluate the derivative $\frac{\partial U_j}{\partial c_{j,k}}$ we use the following expression for a general η-dependent matrix $\chi(\eta)$ [68]:

$$\frac{d}{d\eta}e^{\chi(\eta)} = \int_0^1 e^{\alpha\chi(\eta)}\frac{d\chi(\eta)}{d\eta}e^{(1-\alpha)\chi(\eta)}d\alpha. \tag{7}$$

There are many known ways to approximate this integral (see Appendix C), especially relevant when state transfer or sparse matrices are optimized (see Appendix A). Here, however, we consider an exact method, which is to solve it explicitly in the eigenbasis $\{|n\rangle\}$ of the Hamiltonian $H(t_j)$. A direct calculation reveals

$$\langle m|\frac{\partial U_j}{\partial c_{j,k}}|n\rangle = \langle m|H_k|n\rangle I(m,n). \tag{8}$$

Here we have defined

$$I(m,n) = \begin{cases} -i\Delta t e^{-iE_m\Delta t}, & \text{if} E_m = E_n, \\ \frac{e^{-iE_m\Delta t} - e^{-iE_n\Delta t}}{E_m - E_n}, & \text{if} E_m \neq E_n, \end{cases} \tag{9}$$

where E_n denotes the eigenenergy of $|n\rangle$. This method requires diagonalization of the Hamiltonian at each instance of time, i.e., $H(t_j) = RDR^\dagger$, with R being the transformation matrix whose columns are the eigenvectors of $H(t_j)$ and D containing the eigenenergies $D = \mathrm{Diag}(E_1, E_2, \dots)$. One can use the diagonalization efficiently to first evaluate the matrix exponential $e^{-iH(t_j)\Delta t} = R\,\mathrm{Diag}(-iE_1\Delta t, -iE_2\Delta t, \dots)R^\dagger$ and subsequently use it to switch from one basis to another. If we define the matrix I with $I_{m,n} = I(m,n)$, Eq. (8) can be written in the original basis as

$$\frac{\partial U_j}{\partial c_{j,k}} = R[(R^\dagger H_k R)\odot I]R^\dagger, \tag{10}$$

where $\odot$ denotes the Hadamard (elementwise) product. For multiple controls $\{H_k\}^M_{k=1}$, each derivative $\{\partial U_j/\partial c_{j,k}\}^M_{k=1}$ may be evaluated efficiently since R, $R^\dagger$, and I need only be calculated once. Note one can also exploit the symmetry $I_{m,n} = I_{n,m}$ when evaluating I.

In the case of larger Hilbert spaces, the exact matrix diagonalization becomes intractable. In this case, there exist other methods in the literature that may be more applicable while still permitting analytical differentiation [65,69,70] and gradient- and Hessian-based optimization.

B. Hessian

We start by evaluating the derivative of Eq. (5), which reveals

$$\frac{\partial^2 \mathcal{F}}{\partial c_{i,k'}\partial c_{j,k}} = \frac{2}{\dim^2}\mathrm{Re}\left[\mathrm{Tr}\left(\mathbb{P}\frac{\partial U^2}{\partial c_{i,k'}\partial c_{j,k}}\mathbb{P}V^\dagger\right)\mathrm{Tr}(V\mathbb{P}U^\dagger\mathbb{P}) + \mathrm{Tr}\left(\mathbb{P}\frac{\partial U}{\partial c_{j,k}}\mathbb{P}V^\dagger\right)\mathrm{Tr}\left(\mathbb{P}\frac{\partial U}{\partial c_{i,k'}}\mathbb{P}V^\dagger\right)^*\right]. \tag{11}$$

The first-order derivatives of the unitary have already been calculated when evaluating the gradient. What remains is hence to calculate the second derivatives of the unitary. Here we distinguish between two cases: when the derivatives are with respect to different time steps ($i \neq j$) or the same time step ($i = j$). For different time steps ($j > i$), the second-order derivatives become

$$\frac{\partial^2 U}{\partial c_{i,k'}\partial c_{j,k}} = U^L_{j+1}\frac{\partial U_j}{\partial c_{j,k}}U^R_{j-1}\left(U^R_i\right)^\dagger\frac{\partial U_i}{\partial c_{i,k'}}U^R_{i-1}. \tag{12}$$

Here we have expressed the middle products of unitaries $U_{j-1}\dots U_{i+1}$ in terms of the right unitaries we defined when calculating the gradient. Evaluating the second-order derivatives of the unitary with respect to the same time step is similar to Eq. (6):

$$\frac{\partial^2 U}{\partial c_{j,k'}\partial c_{j,k}} = U^L_{j+1}\frac{\partial^2 U_j}{\partial c_{j,k'}\partial c_{j,k}}U^R_{j-1}. \tag{13}$$

Equations (12) and (13) imply that the Hessian may be evaluated efficiently by recycling already calculated quantities. All that is left is to calculate the second-order derivatives of U_j. This is done by differentiating Eq. (7):

$$\frac{\partial^2 U_j}{\partial c_{j,k'}c_{j,k}} = -i\Delta t\int_0^1\left(\frac{\partial}{\partial c_{j,k'}}e^{\alpha X}\right)H_k e^{(1-\alpha)X}d\alpha - i\Delta t\int_0^1 e^{\alpha X}H_k\left(\frac{\partial}{\partial c_{j,k'}}e^{(1-\alpha)X}\right)d\alpha, \tag{14}$$

where we have introduced the shorthand notation $X = -iH\Delta t$. In the last integral above, we make the substitution $1-\alpha \rightarrow \alpha$ and then insert Eq. (7) to obtain

$$\frac{\partial^2 U_j}{\partial c_{j,k'}c_{j,k}} = (-i\Delta t)^2\left(\int_0^1 d\alpha\alpha\int_0^1 d\beta e^{\beta\alpha X}H_{k'}e^{(1-\beta)\alpha X}H_k e^{(1-\alpha)X} + \int_0^1 d\alpha\alpha\int_0^1 d\beta e^{(1-\alpha)X}H_k e^{\beta\alpha X}H_{k'}e^{(1-\beta)\alpha X}\right). \tag{15}$$

Similar to before, there are different ways to approximate this integral depending on how the exponential is propagated (see Appendices A and C). We derive here the exact solution by evaluating the elements of the eigenbasis of H. We insert the identity $\mathbf{1} = \sum_{n'}|n'\rangle\langle n'|$ to evaluate the middle part, i.e., between the two Hamiltonians H_k and $H_{k'}$ in the above expression. This reveals

$$\langle m|\frac{\partial^2 U_j}{\partial c_{j,k'}c_{j,k}}|n\rangle = (-i\Delta t)^2\sum_{n'}\left(H_{k'}^{(m,n')}H_k^{(n',n)}\int_0^1 d\alpha\alpha\int_0^1 d\beta e^{\beta\alpha\lambda_m}e^{(1-\beta)\alpha\lambda_{n'}}e^{(1-\alpha)\lambda_n} + H_k^{(m,n')}H_{k'}^{(n',n)}\int_0^1 d\alpha\alpha\int_0^1 d\beta e^{(1-\alpha)\lambda_m}e^{\beta\alpha\lambda_{n'}}e^{(1-\beta)\alpha\lambda_n}\right), \tag{16}$$

where we have defined $H_k^{(m,n)} = \langle m|H_k|n\rangle$ and $\lambda_n = -iE_n\Delta t$. The two double integrals in the above expression turn out to be equivalent, which we calculate in Appendix B. This allows us to write

$$\langle m|\frac{\partial^2 U_j}{\partial c_{j,k'}c_{j,k}}|n\rangle = \sum_{n'}\left(H_{k'}^{(m,n')}H_k^{(n',n)} + H_k^{(m,n')}H_{k'}^{(n',n)}\right)\mathcal{I}(n,n',m). \tag{17}$$

Here $\mathcal{I}(n,n',m)$ is given by Eq. (B9). The intuitive scaling for calculating the analytical Hessian would be $O(N^3)$, since one would have to calculate the left, middle, and right sequence of unitaries for each pair of time steps (i,j). However, having the left and right unitaries in advance reduces this to $O(N^2)$ via Eq. (12). Note that this is also true for other methods for evaluating the time-ordered integral (Appendix C). One can use the transformation matrix R to switch from one basis to another similar to Eq. (10). Furthermore, Hessian-based optimization generally converges in fewer iterations than gradient-only optimization, especially near the optimum where its convergence is quadratic. Last but not least, the higher accuracy of the calculation may help it altogether avoid local traps that can plague gradient-based quantum optimal control methods.

Hence, we expect Hessian-based optimization to be an improvement over gradient-only optimization unless the total number of steps N is significantly large. We emphasize that for larger Hilbert spaces, the exact matrix diagonalization via Eq. (8) becomes expensive, and other alternatives in the literature might be preferable [65,69,70] though the same general conclusions hold.

In the following sections, we test and benchmark gradient-only vs Hessian-based optimization on a standard quantum computational setup.

III. TRANSMON SYSTEM

As a testbed for the Hessian-based optimization, we have chosen two transmon qubits dispersively coupled through a linear microwave resonator activated by a microwave field [45,63,64,71]. This type of setup is currently a frontier within superconducting circuit-based quantum information [72] and could enable the realization of many of the quantum technologies outlined in the Introduction. Such a setup has previously been studied for gradient and machine-learning-based optimal control [29,45,55,73].

Transmons are insensitive to charge noise, but suffer from having relatively low anharmonicity [62,74]. We therefore include a third level for each transmon qutrit. We use an effective Hamiltonian, where we adiabatically eliminate the cavity and replace the qutrit-cavity coupling with an effective qutrit-qutrit coupling [75]. Our starting point is to model each transmon as an anharmonic Duffing oscillator [76] and describe the transmon-cavity coupling via the Jaynes-Cummings model, which in the absence of control is

$$H_0 = \sum_{j=1,2}\omega_j b_j^\dagger b_j + \frac{\delta_j}{2}b_j^\dagger b_j(b_j^\dagger b_j - 1) + \omega_r a^\dagger a + \sum_{j=1,2}g_j(ab_j^\dagger + a^\dagger b_j). \tag{18}$$

Here $b_j(b_j^\dagger)$ denotes the annihilation (creation) operator for the jth transmon in the $\{|00\rangle, |01\rangle, \ldots, |22\rangle\}$ basis. We choose the transmon frequencies to be $\omega_1/2\pi = 5.0$ GHz

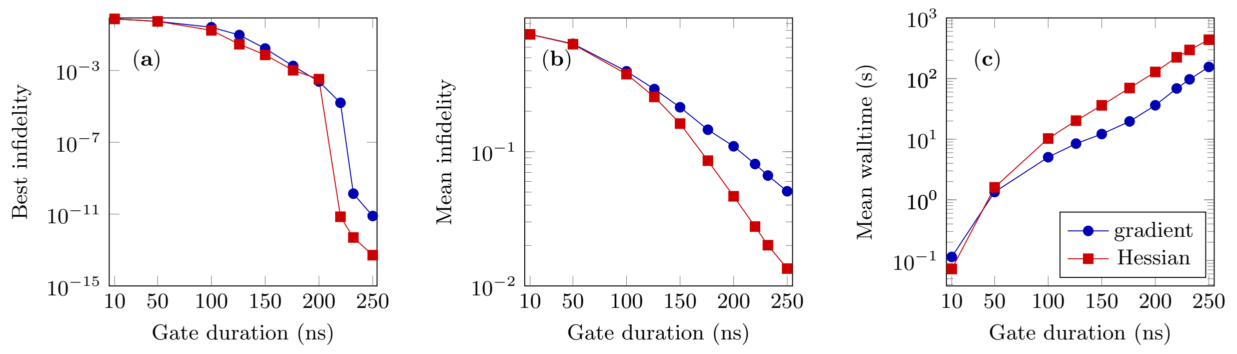

FIG. 2. The results of GRAPE optimizations using Interior-Point with either the Hessian or the gradient-based Hessian-approximation scheme BFGS (denoted gradient) for the same 5000 seeds which we draw uniformly at random for each gate duration. The figure depicts (a) the best infidelity, (b) the mean infidelity per seed, and (c) the mean wall time consumption per seed.

and $\omega_2/2\pi = 5.5$ GHz with equal anharmonicities $\delta_1/2\pi = \delta_2/2\pi = -350$ MHz. For the cavity resonance frequency we choose $\omega_r/2\pi = 7.5$ GHz with equal transmon-cavity couplings $g_1/2\pi = g_2/2\pi = 100$ MHz. These values are within typical experimental ranges (see, e.g., Ref. [77]).

In Appendix E we derive an effective Hamiltonian where we eliminate the cavity and move into a rotating frame. We further drive the first transmon directly, which leads to the effective Hamiltonian

$$H_{\text{eff}}(t) = \Delta b_1^\dagger b_1 + \sum_{j=1,2} \frac{\delta_j}{2} b_j^\dagger b_j (b_j^\dagger b_j - 1) + J\big(b_1^\dagger b_2 + b_1 b_2^\dagger\big) + \Omega(t)(b_1^\dagger + b_1), \tag{19}$$

where Δ denotes the detuning between the transmons and J denotes the effective coupling between the transmons (see Appendix E). Here Ω denotes our control, which we limit to be in the range of $\Omega/2\pi = \pm 200$ MHz. This constrained amplitude, which will be enforced by the optimization, in part determines the minimal time and the quantum speed limit [78] for which the control problem becomes exactly solvable. For many control tasks, optimal solutions contain segments that lie on the constraint boundary; hence, how the optimization algorithm handles constraints is of vast importance. In this work, we attempt to make a $V = \text{CNOT}$ gate starting from the identity $U_0 = \mathbf{1}$ and so the projectors in Eq. (1) are with respect to the qubit subspace $\mathbb{P} = \sum_{i,j=0,1} |i, j\rangle \langle i, j|$. We steer the system via a single piecewise constant control, $c_{j,1} = \Omega(t_j)$, as discussed in the previous section.

IV. HESSIAN- VS GRADIENT-BASED OPTIMIZATION

We consider here the control problem outlined in the previous section, where we use piecewise constant pulses that consist of $\Delta t = 2.0$-ns segments, which is within the bandwidth of standard microwave pulse generators. We compare the two approaches, gradient-only optimization (i.e., BFGS) vs analytical Hessian-based optimization via an Interior-Point algorithm (see Appendix D), over a wide range of different gate durations using multistarting. For each gate duration we use the two approaches to optimize the same 5000 seeds, which we draw uniformly at random. The relative optimality and step tolerance was set to 10^{-9} and 10^{-10}, respectively (see documentation [79]). We plot the results in Figs. 2(a)–2(c), which respectively show the best infidelity over the entire range of gate durations, the mean infidelity per seed, and the mean wall time consumption per seed for each gate duration. The two approaches often find different solutions in the optimization landscape, despite starting from the same initial seed. This stresses the need for selecting the optimization algorithm with care and also motivates the comparison given in Appendix D.

In Fig. 2(a), we see that, when the gate duration increases above 200 ns, the control problem becomes exactly solvable in the sense that the best infidelities become insignificantly low. In the literature the minimal amount of time for which the problem becomes exactly solvable is termed the quantum speed limit [44,45,59,78]. Here we see that the Hessian-based optimization is able to come closer to the (unknown) quantum speed limit relative to gradient-only optimization. Even in the low-infidelity regime (i.e., below 10^{-8}), Hessian-based optimization reaches a few orders of magnitude improvement over gradient-only optimization.

Figure 2(b) depicts the mean infidelity for the two algorithms. The two algorithms perform equally well on average at shorter gate durations. In contrast, Hessian-based optimization performs better at longer gate durations, with mean infidelities being 3–4 times lower than those of gradient-only optimization.

Figure 2(c) depicts the mean wall time consumption for each gate duration. As already elaborated on, the computational cost of the gradient scales linearly as $(O(N))$ while the Hessian scales quadratically as $(O(N^2))$. However, Hessian-based optimization generally requires fewer iterations for convergence than BFGS. For this reason, Hessian-based optimization actually turns out to be comparable with gradient-only optimization at very short gate durations, where the Hessian is cheaper to calculate but the number of iterations needed remains significantly lower. For longer gate durations, however, Hessian-based optimization becomes 2–3 times slower than gradient-only optimization.

In summary, Fig. 2 shows that there are two distinct regimes, where the analytical Hessian of the unitary dynamics is preferable to its BFGS approximation, but for different reasons. For fewer than 50 ns (or 25 control steps), the two approaches perform equally well with respect to infidelity; however, the figure and scaling arguments indicate a potential speed advantage for the Hessian-based approach. Above this

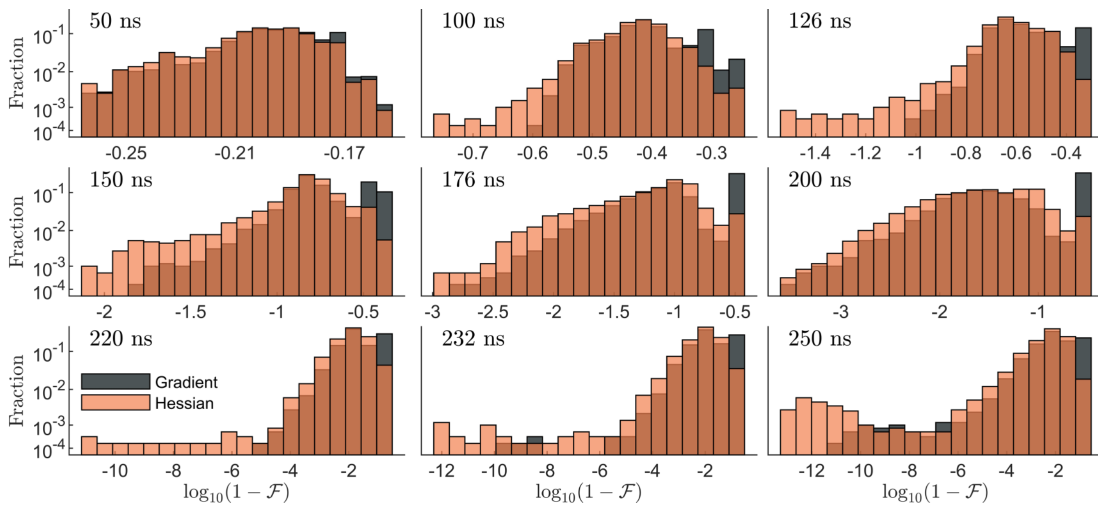

FIG. 3. Distribution of infidelities (lower is better) at different gate durations. Blue depicts gradient-only optimization, while orange depicts Hessian-based. Brown depicts the overlap of the two distributions. Gate durations are depicted in the figure.

time, and especially near the quantum speed limit, we see the exact Hessian will produce better gate infidelities, both on average and for the best case, although at the price of being slower.

We plot in Fig. 3 histograms over the infidelity distributions for the same data presented in Fig. 2. At 50 ns in Fig. 3(a) we see two almost identical distributions. In contrast, at larger gate durations we see that Hessian-based optimization generally performs better than gradient-only optimization in terms of both the quality and the quantity of the best solutions found.

From Figs. 2 and 3 we see that the two approaches converge to different minima although starting from the same initial seed. We also see that the Hessian-based solutions tend to be better, i.e., reach lower infidelity compared to gradient-only optimization, which is clear from the bimodal distributions in Fig. 3. We attribute this to the fact that Hessian-based optimization obtains more information about the curvature of the optimization landscape through the second derivative relative to gradient-only optimization, which only has approximate knowledge of the second derivatives. This can lead gradient-only optimization to slow down, or even be unable to converge at all. It may also cause trapping in local suboptimal solutions as illustrated in Fig. 1(b).

In Fig. 4(a), we also depict the average number of infidelity evaluations used by either approach. Here we clearly see that the Hessian-based optimization uses fewer infidelity evaluations in comparison to the gradient-only optimization. This illustrates the difference between quadratic convergence (Hessian) vs superlinear convergence (gradient) as elaborated on in Sec. II.

To verify that the two approaches indeed do find different solutions, we plot in Fig. 4(b) a scatter plot of the optimized infidelities for the two approaches at 176 ns. If a given point lies on the dashed diagonal it would imply that the two approaches (most likely) found the same solution, as the figure shows this is most often not the case.

Although the Hessian is more expensive to calculate than the gradient, we showed that Hessian-based optimization can be advantageous over gradient-only optimization for even over 100 optimization parameters, which allows for reasonably complicated but still practical pulses that may easily be generated via modern arbitrary wave-form generators. However, for a much larger number of parameters, Hessian-based optimization may become too slow to use at one point, even if the final result would be better. Hence, to include the Hessian is a question of balancing the quantity and quality of found solutions. When using only a relatively few seeds (e.g., due

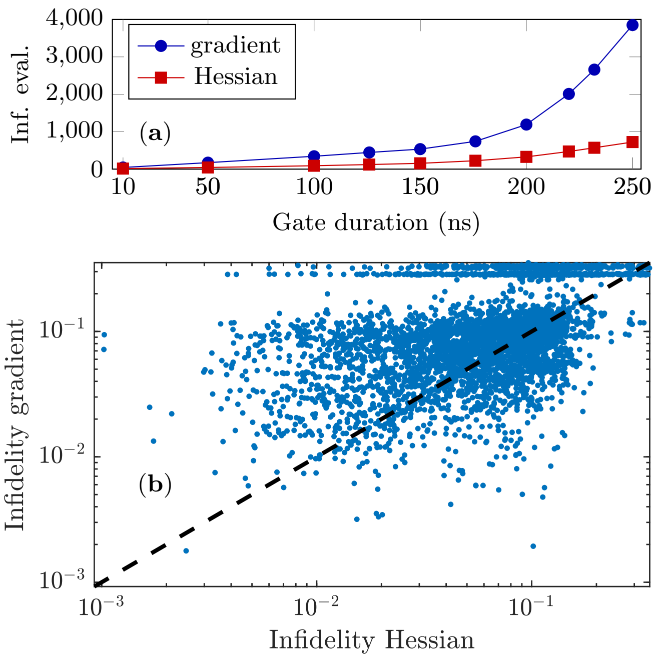

FIG. 4. (a) The average number of infidelity evaluations for gradient-only and Hessian-based optimization. The figure verifies that gradient-only optimization scales worse than Hessian-based optimization with respect to the number of infidelity evaluations. (b) A scatter plot of the optimized infidelities from Fig. 3 at 176 ns. The x axis is the Hessian-based optimized solutions while the y axis is for the gradient-only method. Solutions that lie on the dashed diagonal line corresponds to gradient-only and Hessian-based optimization doing equally well. Above the diagonal is where the Hessian did best and vice versa.

to a large amount of optimization parameters or complicated state spaces that require more exploration), it may be very unlikely to reach near the global optimum for a practical wall time: In this case a gradient-only approach may be expected to produce faster if less reliable results.

Even so, many optimization tasks within quantum control have less than 125 optimization parameters. For some control tasks, like the one considered here, increasing this number would make experimental implementation and calibration infeasible. Hence, we believe that explicitly incorporating the analytical Hessian of the unitary dynamics is advantageous for many quantum control tasks.

Finally, one may link the apparent utility of the Hessian to the nature of the control landscape. Indeed, the smoothness of the landscape [80] has earlier been argued to provide a computational advantage for quantum optimization [12,15,78]. In the present work, we have explored the case where the cost functional is strongly constrained. Although such constraints formally remove the possibility of a global convergence [48,81], in our context the functional smoothness can nonetheless provide a mechanism to greatly speed up convergence alongside a multistarting strategy with analog controls [61].

V. CONCLUSION

In this paper, we have obtained the Hessian of the unitary dynamics as an extension to the widely used gradient-based, quantum optimization algorithm GRAPE. Our calculations, which were based on diagonalization, revealed that the Hessian may be computed efficiently, with a high level of reusability of already-calculated quantities obtained when evaluating the gradient. We believe our efficient calculation is advantageous to previous proposals and it allowed us to demonstrate improvements over gradient-only optimization in equal wall-time simulations without any code parallelization. We optimized a CNOT gate on a circuit-QED system consisting of two coupled transmon qubits. Here we demonstrated that a fast CNOT gate is, in principle, feasible using only a single control on one qubit driven cross-resonantly.

For the numerical optimization, we used an Interior-Point algorithm, with either an analytically exact Hessian or the Hessian-approximation scheme BFGS that only relies on the gradient. We compared the two approaches, Hessian-based or gradient-only, for a wide range of different gate durations. Since the Hessian contains squared the number of elements of the gradient, it is more expensive to calculate per iteration. However, Hessian-based optimization generally uses fewer iterations to converge. Moreover, the convergence occurs with greater accuracy, which can improve the quality as well as the quantity of good solutions.

We have found that, depending on the number of controls, either the wall time or the fidelity of the solutions can be improved compared to the gradient. This appears to be generally true, although for very complex spaces where multistarting is not appropriate the gradient may be the only practical choice. Nonetheless, for over 100 controls, we were still able to collect statistics over many seeds. We found that below 25 controls, the Hessian enabled faster convergence towards extrema. For more controls (and for the experimentally most interesting regime near the quantum speed limit), incorporating the Hessian provided a higher percentage of good solutions, often accompanied by bimodal distributions pointing to avoiding local trapping. Thus, the best-case error was also seen to be improved.

ACKNOWLEDGMENTS

The authors would like to thank Jens Jakob Sørensen for participating in constructive scientific discussions. The numerical results presented in this work were obtained at the Centre for Scientific Computing, Aarhus, phys.au.dk/forskning/cscaa. This work was funded by the ERC, H2020 Grant No. 639560 (MECTRL), the John Templeton Foundation, and the Carlsberg Foundation.

APPENDIX A: STATE TRANSFER

When only multiplication is needed in the propagation (as for typical expansions, see Appendix C), then this can be a preferred method to use for propagation of only one or a few orthogonal states [essentially specified by the projector $\mathbb{P}$ in Eq. (1)]. More concretely, we can rewrite Eq. (1) as

$$\mathcal{F} = \left| \frac{1}{\dim} \sum_{k=1}^{\dim} \langle \psi_k | V^\dagger U | \psi_k \rangle \right|^2 . \tag{A1}$$

In this case, the propagator scales as $O(n^2)$, with n being the linear dimension, rather than exponentiation and matrix-matrix multiplication which typically scale as $O(n^3)$ [or $O(n^{2.8})$] for dense matrices. Our method is easily adaptable by replacing the left U_j^L and right U_j^R unitaries by left $\langle \psi_j^L | = \langle \psi_{\text{targ}} | U_j^L = \langle \psi_0 | V^\dagger U_j^L$ and right $|\psi_j^R\rangle = U_j^R |\psi_0\rangle$ states, respectively. $\langle \psi_j^L |$ and $|\psi_j^R\rangle$ were dynamically calculated, as were U_j^L and U_j^R, using a recursive approach. Similarly, the second derivatives can recursively use the time-propagated first-order derivatives, as needed for Eq. (12), as in

$$\begin{aligned} \frac{\partial^2 |\psi_j^R\rangle}{\partial c_{i,k'} \partial c_{j,k}} &= \frac{\partial U_j}{\partial c_{j,k}} U_{j-1} \dots U_{i+1} \frac{\partial U_i}{\partial c_{i,k'}} |\psi_{i-1}^R\rangle \\ &= \frac{\partial U_j}{\partial c_{j,k}} \left| \frac{\partial \psi_{j-1}^R}{\partial c_{i,k'}} \right\rangle, \end{aligned} \tag{A2}$$

which uses only matrix-vector multiplication. Thanks to the recursive definition $|\frac{\partial \psi_{j+1}^R}{\partial c_{i,k'}}\rangle = U_{j+1} |\frac{\partial \psi_j^R}{\partial c_{i,k'}}\rangle$, this ends up once again reducing the algorithm complexity with respect to the number of time steps (as in the main text) from $O(N^3)$ to $O(N^2)$. Using state propagation thus permits avoiding matrix-matrix multiplication. See also the original Khaneja *et al.* GRAPE paper [17].

APPENDIX B: HESSIAN CALCULATION USING EIGENBASIS

The starting point of these calculations is Eq. (16), which can be written as

$$\begin{aligned} \langle m | \frac{\partial^2 U_j}{\partial c_{j,k'} c_{j,k}} | n \rangle = \sum_{n'} \big(& H_{k'}^{(m,n')} H_k^{(n',n)} \mathcal{I}(n, n', m) \\ & + H_k^{(m,n')} H_{k'}^{(n',n)} \mathcal{I}(m, n, n') \big). \end{aligned} \tag{B1}$$

Here we have defined

$$\mathcal{I}(n,n',m) = (-i\Delta t)^2 e^{\lambda_n}\int_0^1 d\alpha\alpha e^{\alpha(\lambda_{n'}-\lambda_n)}\int_0^1 d\beta e^{\beta\alpha(\lambda_m-\lambda_{n'})}. \tag{B2}$$

In order to evaluate the above we must consider the the five different scenarios: $m\neq n\neq n'$, $m=n=n'$, $m=n'\neq n$, $m\neq n=n'$, and $m=n\neq n'$. We start with the first one, which is $m\neq n\neq n'$. A direct calculation reveals

$$\mathcal{I}(n,n',m) = \frac{1}{E_m-E_{n'}}\left[\frac{e^{-iE_m\Delta t}-e^{-iE_n\Delta t}}{E_m-E_n}-\frac{e^{-iE_{n'}\Delta t}-e^{-iE_n\Delta t}}{E_{n'}-E_n}\right]. \tag{B3}$$

Note that the above expression is invariant under any permutation of the indices. Similarly, the second one, which is $m=n=n'$, gives

$$\mathcal{I}(n,n,n) = \frac{(-i\Delta t)^2}{2}e^{-iE_n\Delta t}. \tag{B4}$$

The third one, which is $m=n'\neq n$, gives

$$\mathcal{I}(n,m,m) = (-i\Delta t)^2 e^{\lambda_n}\int_0^1 d\alpha\alpha e^{\alpha(\lambda_m-\lambda_n)}. \tag{B5}$$

The antiderivative of xe^{kx} is $\frac{(kx-1)}{k^2}e^{kx}$, which can be used to evaluate the above:

$$\mathcal{I}(n,m,m) = \frac{[-i\Delta t(E_m-E_n)-1]e^{-iE_m\Delta t}+e^{-iE_n\Delta t}}{(E_m-E_n)^2}. \tag{B6}$$

The fourth one is $m\neq n'=n$, which gives

$$\mathcal{I}(n,n,m) = \frac{e^{-iE_m\Delta t}+[-i\Delta t(E_m-E_n)-1]e^{-iE_n\Delta t}}{(E_m-E_n)^2}. \tag{B7}$$

And the last one, which is $m=n\neq n'$, gives

$$\mathcal{I}(n,n',n) = \frac{[-i\Delta t(E_n-E_{n'})-1]e^{-iE_n\Delta t}+e^{-iE_{n'}\Delta t}}{(E_n-E_{n'})^2}. \tag{B8}$$

We can summarize the above results by writing

$$\mathcal{I}(n_1,n_2,n_3) = \begin{cases} \frac{1}{E_{n_3}-E_{n_2}}[I(n_3,n_2)-I(n_2,n_1)], & \text{if } n_1\neq n_2\neq n_3,\\ \frac{-i\Delta t}{2}I(n_1,n_1), & \text{if } n_1=n_2=n_3,\\ \frac{1}{E_n-E_m}[I(n,m)-I(m,m)], & \begin{cases}\text{if } n=n_1\neq n_2=n_3=m,\\ \text{or if } n=n_2\neq n_1=n_3=m,\\ \text{or if } n=n_3\neq n_1=n_2=m.\end{cases}\end{cases} \tag{B9}$$

We may evaluate the above integral efficiently by storing the already calculated integrals $I(m,n)$ from Eq. (9). Also note that the above integral is independent of the order of coefficients n_1, n_2, and n_3. This implies that we may calculate the above efficiently and in advance of evaluating the matrix elements of the second derivative. Since $\mathcal{I}(n_1,n_2,n_3)$ is independent of the order of the indices, we may also take the integral out of parentheses in Eq. (B1) and hence we obtain Eq. (17).

APPENDIX C: HESSIAN CALCULATION USING (TAYLOR) EXPANSION

The analytical gradient and Hessian of the propagation are intricately linked to how the evolution is calculated. In the main text we used eigenbasis decomposition, which is exact to numerical precision. In certain cases with, e.g., large and/or sparse Hamiltonians, approximate expansions such as Taylor, BCH, Pade, Suzuki-Trotter, or Chebychev may be preferable (especially in combination with matrix-vector algebra, Appendix A) [65,70,82].

As a supplementary example to the diagonalization-based approach, we consider here also the case using Taylor expansion of the propagation [70]. Let such an expansion as $\tilde{U}_j$ be given by (with $c_{j,0}=1$)

$$U_j \approx \tilde{U}_j = \sum_{l=0}^{L}\frac{1}{l!}[-iH(t_j)\Delta t]^l = \sum_{l=0}^{L}\frac{1}{l!}\left(\sum_{k=0}^{M}-ic_{j,k}H_k\Delta t\right)^l \tag{C1}$$

$$= \sum_{l=0}^{L}\frac{1}{l!}\sum_{l_0+l_1+\dots+l_M=l}\frac{l!}{l_0!l_1!\dots l_M!}\prod_{k=0}^{M}(-ic_{j,k}H_k\Delta t)^{l_k}, \tag{C2}$$

where we have used the multinomial theorem for the last equality. Here L denotes a truncation parameter. The gradients and Hessians are calculated exactly as before in the main text, i.e., Eqs. (12) and (13), where only the first and second derivatives of the single-step unitaries need to be calculated differently. Then,

$$\frac{\partial^r\tilde{U}_j}{(\partial c_{j,k'})^r} = \sum_{l=0}^{L}\frac{1}{l!}\sum_{l_1+l_2+\dots+l_M=l}\frac{l!}{l_1!l_2!\dots l_M!}\frac{l_{k'}(l_{k'}-1)^{r-1}}{(c_{j,k'})^r}\left[\prod_{k=0}^{M}(-ic_{j,k}H_k\Delta t)^{l_k}\right], \tag{C3}$$

where $r = 1$ or $r = 2$. Meanwhile

$$\frac{\partial^2 \tilde{U}_j}{\partial c_{j,k'} \partial c_{j,k''}} = \sum_{l=0}^{L} \frac{1}{l!} \sum_{l_1+l_2+\cdots+l_M=l} \frac{l!}{l_1! l_2! \ldots l_M!} \frac{l_{k'} l_{k''}}{c_{j,k'} c_{j,k''}} \left[\prod_{k=0}^{M} (-i c_{j,k} H_k \Delta t)^{l_k} \right], \tag{C4}$$

when $k'' \neq k'$.

Any such expansion for the unitary evolution has its own exact analytical formulas for the gradient and the Hessian of the evolutions with respect to the controls. Note also that even for a very complex method one can always use automatic differentiation as a substitute, which is also exactly as above, though we believe this is not strictly necessary or faster in any particular case. Thus in both cases, regardless of how the Hamiltonian exponential is calculated, one can still benefit from the derivatives (12) and (13).

Note that, while the analytical gradient and Hessian are exact, the expansions themselves are not, which means fixed expansions must be used if monotonic convergence is to be guaranteed (i.e., keeping L fixed above) or else machine-precision-level error tolerance must be enforced. Otherwise, the directions of the gradient and the Hessian may change when, e.g., an extra term is added to the expansion or a term is modified (as would be needed to satisfy adaptive error tolerance criteria for the expansion).

APPENDIX D: BENCHMARKING DIFFERENT OPTIMIZATION ALGORITHMS

In the main text we considered synthesizing a CNOT gate using piecewise constant pulses with $\Delta t = 2.0$ ns via the GRAPE algorithm. GRAPE relies on a numerical optimization algorithm at its back end, for instance, in the main text we used Interior-Point [83,84] implemented in MATLAB's library fmincon [79]. Interior-Point can be supplied either with the gradient and the Hessian-approximation scheme BFGS or with the exact Hessian.

To justify this specific choice we benchmark Interior-Point with other conventional optimization algorithms. Another choice, also implemented in MATLAB's fmincon [79] for constrained optimization, is the Trust-Region-Reflective algorithm [85,86]. Similar to Interior-Point, Trust-Region-Reflective can be supplied with either the gradient or the Hessian. We also benchmark against an unconstrained optimization algorithm quasi-Newton [20] implemented in MATLAB's fminunc, where we instead impose constraint bounds by adding a penalty term to the cost function. We choose a quadratic penalty function that is zero inside the admissible region and grows quadratically outside $J_{\text{penalty}} = \sigma(\Omega - \Omega_{\min/\max})^2$, where we set the penalty factor to $\sigma = 10^5$.

For the comparison we consider the same control problem as in the main text with a gate duration of $T = 200$ ns. We let each optimization algorithm optimize the same 300 random seeds, which are drawn uniformly at random. The results are plotted in Fig. 5 where we plot the infidelity for each seed as a function of wall time consumption, which we limit to 1000 s. From the figure we see that Interior-Point generally converges faster and at lower infidelity solutions. This justifies our choice of using Interior-Point for the results presented in the main text.

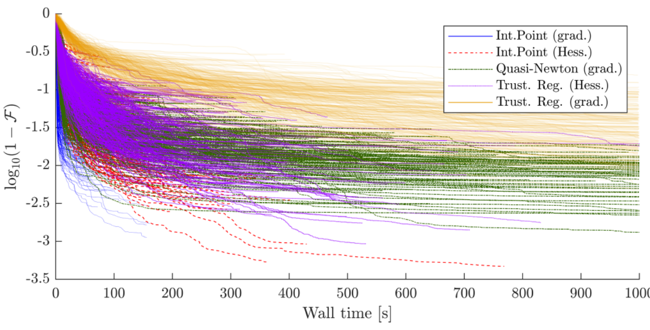

FIG. 5. A comparison between different optimization algorithms with and without the Hessian (see text).

APPENDIX E: CIRCUIT QED CALCULATIONS

The following derivation to some extent resembles the one given in Ref. [75]. The starting point is the drift Hamiltonian given by Eq. (18). Here we eliminate the cavity by going to a frame rotating at ω_r via $R = \exp[-i\omega_r(b_1^\dagger b_1 + b_2^\dagger b_2 + a^\dagger a)]$. This gives

$$H = \sum_{j=1,2} \Delta_j b_j^\dagger b_j + \frac{\delta_j}{2} b_j^\dagger b_j (b_j^\dagger b_j - 1) + \sum_{j=1,2} g_j (a b_j^\dagger + a^\dagger b_j), \tag{E1}$$

where $\Delta_j = \omega_j - \omega_r$. Then we perform a Schrieffer-Wolff transformation [87] using $S = \sum_{j=1,2} \frac{g_j}{\Delta_j}(a b_j^\dagger - a^\dagger b_j)$ in order to eliminate the cavity. The resulting Hamiltonian, when any constant energy shifts have been removed and the cavity neglected, is

$$H = \sum_{j=1,2} \tilde{\omega}_j b_j^\dagger b_j + \frac{\delta_j}{2} b_j^\dagger b_j (b_j^\dagger b_j - 1) + J(b_1^\dagger b_2 + b_1 b_2^\dagger). \tag{E2}$$

Here we see that the transmon-cavity coupling has been replaced with an effective transmon-transmon coupling where $J = \frac{g_1 g_2(\Delta_1+\Delta_2)}{\Delta_1 \Delta_2}$ and $\tilde{\omega}_j = \omega_j + \frac{g_j^2}{\Delta_j}$ is now the dressed transmon state. The last step when transforming Eq. (E2) into Eq. (19) is doing a second rotation, $R' = \exp[-i\tilde{\omega}_2(b_1^\dagger b_1 + b_2^\dagger b_2)]$, such that the detuning becomes $\Delta = \tilde{\omega}_2 - \tilde{\omega}_2$. We also add a direct drive on the first transmon, $H_c(t) = \Omega(t)(b_1^\dagger + b_1)$ (see, e.g., Ref. [75]).

APPENDIX F: TWO-LEVEL EXAMPLE

In Fig. 1(b), we illustrated three optimization methods based on gradient descent and Interior-Point with either BFGS or the Hessian for a two-dimensional optimization problem. Here we have implemented a simple gradient descent algorithm using a fixed step size, where the step size is chosen to illustrate typical differences between the gradient descent and BFGS. We briefly elaborate on what Fig. 1(b) depicts. We consider the two-level Hamiltonian $H(t) = \sigma_x + c(t)\sigma_z$, with the goal of synthesizing a X gate. We limit the control to two steps $N = 2$, which reveals an analytical solution at $T_{\rm QSL} = \pi/2$ with $c_{1,1} = c_1 = 0$ and $c_{2,1} = c_2 = 0$. At $T = 3T_{\rm QSL}$ the same solution is still optimal, but now several other solutions emerge in the optimization landscape, an effect also studied in Ref. [88]. For instance, the figure depicts a suboptimal solution at $(c_1, c_2) = (0.000, 2.285)$. We start the optimization near the two optima at $(c_1, c_2) = (-0.915, 2.251)$ and plot the subsequent optimization. The two gradient-based optimization approaches fall into the nearest trap (i.e., suboptimal solution), while the Hessian optimization manages to avoid this solution. We attribute this result to the fact that the Hessian-based optimization has more information about the landscape curvature, which enables it to avoid the local trap and instead reach the optimal solution. The reader should of course keep in mind that this is but one example.

[1] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'brien, Nat. Commun. **5**, 4213 (2014).
[2] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Nature (London) **549**, 242 (2017).
[3] M. W. Johnson, M. H. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk *et al.*, Nature (London) **473**, 194 (2011).
[4] A. A. Houck, H. E. Türeci, and J. Koch, Nat. Phys. **8**, 292 (2012).
[5] I. Bloch, J. Dalibard, and S. Nascimbene, Nat. Phys. **8**, 267 (2012).
[6] M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, and R. Melko, Phys. Rev. X **8**, 021050 (2018).
[7] M. A. Nielsen and I. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, England, 2002).
[8] J. Werschnik and E. Gross, J. Phys. B: At. Mol. Opt. Phys. **40**, R175 (2007).
[9] C. Brif, R. Chakrabarti, and H. Rabitz, New J. Phys. **12**, 075008 (2010).
[10] S. J. Glaser, U. Boscain, T. Calarco, C. P. Koch, W. Köckenberger, R. Kosloff, I. Kuprov, B. Luy, S. Schirmer, T. Schulte-Herbrüggen *et al.*, Eur. Phys. J. D **69**, 279 (2015).
[11] P. Doria, T. Calarco, and S. Montangero, Phys. Rev. Lett. **106**, 190501 (2011).
[12] T. Caneva, T. Calarco, and S. Montangero, Phys. Rev. A **84**, 022326 (2011).
[13] J. P. Palao and R. Kosloff, Phys. Rev. Lett. **89**, 188301 (2002).
[14] J. P. Palao and R. Kosloff, Phys. Rev. A **68**, 062308 (2003).
[15] H. A. Rabitz, M. M. Hsieh, and C. M. Rosenthal, Science **303**, 1998 (2004).
[16] N. Rach, M. M. Müller, T. Calarco, and S. Montangero, Phys. Rev. A **92**, 062343 (2015).
[17] N. Khaneja, T. Reiss, C. Kehlet, T. Schulte-Herbrüggen, and S. J. Glaser, J. Magn. Reson. **172**, 296 (2005).
[18] P. De Fouquieres, S. Schirmer, S. Glaser, and I. Kuprov, J. Magn. Reson. **212**, 412 (2011).
[19] S. Machnes, U. Sander, S. J. Glaser, P. de Fouquieres, A. Gruslys, S. Schirmer, and T. Schulte-Herbrüggen, Phys. Rev. A **84**, 022305 (2011).
[20] R. Fletcher, *Practical Methods of Optimization* (Wiley & Sons, New York, 2013).
[21] Z. Tošner, S. J. Glaser, N. Khaneja, and N. C. Nielsen, J. Chem. Phys. **125**, 184502 (2006).
[22] J. Zhang, T.-C. Wei, and R. Laflamme, Phys. Rev. Lett. **107**, 010501 (2011).
[23] K. Kobzar, S. Ehni, T. E. Skinner, S. J. Glaser, and B. Luy, J. Magn. Reson. **225**, 142 (2012).
[24] K. R. K. Rao, T. S. Mahesh, and A. Kumar, Phys. Rev. A **90**, 012306 (2014).
[25] J. J. Sørensen, J. S. Nyemann, F. Motzoi, J. Sherson, and T. Vosegaard, J. Chem. Phys. **152**, 054104 (2020).
[26] F. Motzoi, J. M. Gambetta, P. Rebentrost, and F. K. Wilhelm, Phys. Rev. Lett. **103**, 110501 (2009).
[27] P. Groszkowski, A. G. Fowler, F. Motzoi, and F. K. Wilhelm, Phys. Rev. B **84**, 144516 (2011).
[28] D. J. Egger and F. K. Wilhelm, Supercond. Sci. Technol. **27**, 014001 (2013).
[29] S. Kirchhoff, T. Keßler, P. J. Liebermann, E. Assémat, S. Machnes, F. Motzoi, and F. K. Wilhelm, Phys. Rev. A **97**, 042348 (2018).
[30] M. Abdelhafez, B. Baker, A. Gyenis, P. Mundada, A. A. Houck, D. Schuster, and J. Koch, Phys. Rev. A **101**, 022321 (2020).
[31] G. A. Paz-Silva, S. Rebić, J. Twamley, and T. Duty, Phys. Rev. Lett. **102**, 020503 (2009).
[32] X. Wang, A. Bayat, S. G. Schirmer, and S. Bose, Phys. Rev. A **81**, 032312 (2010).
[33] M. Nimbalkar, R. Zeier, J. L. Neves, S. B. Elavarasi, H. Yuan, N. Khaneja, K. Dorai, and S. J. Glaser, Phys. Rev. A **85**, 012325 (2012).
[34] S. Ashhab, Phys. Rev. A **92**, 062305 (2015).
[35] R. S. Said and J. Twamley, Phys. Rev. A **80**, 032303 (2009).
[36] F. Dolde, V. Bergholm, Y. Wang, I. Jakobi, B. Naydenov, S. Pezzagna, J. Meijer, F. Jelezko, P. Neumann, T. Schulte-Herbrüggen *et al.*, Nat. Commun. **5**, 3371 (2014).
[37] B. Khani, S. T. Merkel, F. Motzoi, J. M. Gambetta, and F. K. Wilhelm, Phys. Rev. A **85**, 022306 (2012).
[38] J. H. M. Jensen, J. J. Sørensen, K. Mølmer, and J. F. Sherson, Phys. Rev. A **100**, 052314 (2019).
[39] Z. Tošner, T. Vosegaard, C. Kehlet, N. Khaneja, S. J. Glaser, and N. C. Nielsen, J. Magn. Reson. **197**, 120 (2009).
[40] J. R. Johansson, P. D. Nation, and F. Nori, Comput. Phys. Commun. **184**, 1234 (2013).
[41] J. Sørensen, J. Jensen, T. Heinzel, and J. Sherson, Comput. Phys. Commun. **243**, 135 (2019).
[42] F. Motzoi, J. M. Gambetta, S. T. Merkel, and F. K. Wilhelm, Phys. Rev. A **84**, 022307 (2011).

[43] X. Ge, H. Ding, H. Rabitz, and R.-B. Wu, Phys. Rev. A **101**, 052317 (2020).
[44] J. J. W. H. Sørensen, M. O. Aranburu, T. Heinzel, and J. F. Sherson, Phys. Rev. A **98**, 022119 (2018).
[45] M. H. Goerz, F. Motzoi, K. B. Whaley, and C. P. Koch, npj Quantum Inf. **3**, 1 (2017).
[46] T. Schulte-Herbrüggen, A. Spörl, N. Khaneja, and S. Glaser, J. Phys. B: At. Mol. Opt. Phys. **44**, 154013 (2011).
[47] J. J. Sørensen, M. Aranburu, T. Heinzel, and J. Sherson, arXiv:1802.07521.
[48] E. Zahedinejad, S. Schirmer, and B. C. Sanders, Phys. Rev. A **90**, 032310 (2014).
[49] P. Palittapongarnpim, P. Wittek, E. Zahedinejad, S. Vedaie, and B. C. Sanders, Neurocomputing **268**, 116 (2017).
[50] M. Bukov, A. G. R. Day, D. Sels, P. Weinberg, A. Polkovnikov, and P. Mehta, Phys. Rev. X **8**, 031086 (2018).
[51] D. Fitzek, M. Eliasson, A. F. Kockum, and M. Granath, Phys. Rev. Res. **2**, 023230 (2020).
[52] M. Y. Niu, S. Boixo, V. N. Smelyanskiy, and H. Neven, npj Quantum Inf. **5**, 1 (2019).
[53] Z. An and D. Zhou, Europhys. Lett. **126**, 60002 (2019).
[54] J. Yao, M. Bukov, and L. Lin, arXiv:2002.01068.
[55] M. Dalgaard, F. Motzoi, J. J. Sørensen, and S. Jacob, npj Quantum Inf. **6**, 6 (2020).
[56] D. Goodwin and I. Kuprov, J. Chem. Phys. **143**, 084113 (2015).
[57] D. Goodwin and I. Kuprov, J. Chem. Phys. **144**, 204107 (2016).
[58] D. Hocker, C. Brif, M. D. Grace, A. Donovan, T.-S. Ho, K. M. Tibbetts, R. Wu, and H. Rabitz, Phys. Rev. A **90**, 062309 (2014).
[59] G. C. Hegerfeldt, Phys. Rev. Lett. **111**, 260501 (2013).
[60] C. Lin, D. Sels, and Y. Wang, Phys. Rev. A **101**, 022320 (2020).
[61] D. V. Zhdanov and T. Seideman, arXiv:1709.09423.
[62] J. Koch, T. M. Yu, J. Gambetta, A. A. Houck, D. I. Schuster, J. Majer, A. Blais, M. H. Devoret, S. M. Girvin, and R. J. Schoelkopf, Phys. Rev. A **76**, 042319 (2007).
[63] C. Rigetti and M. Devoret, Phys. Rev. B **81**, 134507 (2010).
[64] J. M. Chow, A. D. Córcoles, J. M. Gambetta, C. Rigetti, B. R. Johnson, J. A. Smolin, J. R. Rozen, G. A. Keefe, M. B. Rothwell, M. B. Ketchen, and M. Steffen, Phys. Rev. Lett. **107**, 080502 (2011).
[65] J. H. M. Jensen, F. S. Møller, J. J. Sørensen, and J. F. Sherson, arXiv:2005.09943.
[66] Z. Ugray, L. Lasdon, J. Plummer, F. Glover, J. Kelly, and R. Martí, INFORMS J. Comput. **19**, 328 (2007).
[67] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. (Springer Science & Business Media, New York, 2006).
[68] R. Wilcox, J. Math. Phys. **8**, 962 (1967).
[69] F. F. Floether, P. De Fouquieres, and S. G. Schirmer, New J. Phys. **14**, 073023 (2012).
[70] D. L. Goodwin, Advanced Optimal Control Methods for Spin Systems, Ph.D. thesis, University of Southampton (2017).
[71] G. S. Paraoanu, Phys. Rev. B **74**, 140504(R) (2006).
[72] G. Wendin, Rep. Prog. Phys. **80**, 106001 (2017).
[73] J. L. Allen, R. Kosut, and E. Ginossar, arXiv:1902.08056.
[74] L. Theis, F. Motzoi, S. Machnes, and F. Wilhelm, Europhys. Lett. **123**, 60001 (2018).
[75] E. Magesan and J. M. Gambetta, Phys. Rev. A **101**, 052308 (2020).
[76] B. Khani, J. Gambetta, F. Motzoi, and F. K. Wilhelm, Phys. Scr. **2009**, 014021 (2009).
[77] D. C. McKay, S. Filipp, A. Mezzacapo, E. Magesan, J. M. Chow, and J. M. Gambetta, Phys. Rev. Appl. **6**, 064007 (2016).
[78] T. Caneva, M. Murphy, T. Calarco, R. Fazio, S. Montangero, V. Giovannetti, and G. E. Santoro, Phys. Rev. Lett. **103**, 240501 (2009).
[79] See documentation at https://se.mathworks.com/help/optim/ug/fmincon.html.
[80] L. Hardy, arXiv:quant-ph/0101012.
[81] D. V. Zhdanov and T. Seideman, Phys. Rev. A **92**, 052109 (2015).
[82] M. Ndong, H. Tal-Ezer, R. Kosloff, and C. P. Koch, J. Chem. Phys. **130**, 124108 (2009).
[83] R. H. Byrd, M. E. Hribar, and J. Nocedal, SIAM J. Optim. **9**, 877 (1999).
[84] R. A. Waltz, J. L. Morales, J. Nocedal, and D. Orban, Math. Program. **107**, 391 (2006).
[85] T. F. Coleman and Y. Li, Math. Program. **67**, 189 (1994).
[86] T. F. Coleman and Y. Li, SIAM J. Optim. **6**, 418 (1996).
[87] J. R. Schrieffer and P. A. Wolff, Phys. Rev. **149**, 491 (1966).
[88] M. Larocca, P. M. Poggi, and D. A. Wisniacki, J. Phys. A: Math. Theor. **51**, 385305 (2018).