
Article:
Banchi, Leonardo, Pereira, Jason, Lloyd, Seth et al. (1 more author) (2020) Convex optimization of programmable quantum computers. npj Quantum Information. 42 (2020). ISSN 2056-6387

https://doi.org/10.1038/s41534-020-0268-2

---

npj | Quantum Information

www.nature.com/npjqi

ARTICLE OPEN

Convex optimization of programmable quantum computers

Leonardo Banchi[1,2]✉, Jason Pereira[3], Seth Lloyd[4,5] and Stefano Pirandola[3,5]

A fundamental model of quantum computation is the programmable quantum gate array. This is a quantum processor that is fed by a program state that induces a corresponding quantum operation on input states. While being programmable, any finite-dimensional design of this model is known to be nonuniversal, meaning that the processor cannot perfectly simulate an arbitrary quantum channel over the input. Characterizing how close the simulation is and finding the optimal program state have been open questions for the past 20 years. Here, we answer these questions by showing that the search for the optimal program state is a convex optimization problem that can be solved via semidefinite programming and gradient-based methods commonly employed for machine learning. We apply this general result to different types of processors, from a shallow design based on quantum teleportation, to deeper schemes relying on port-based teleportation and parametric quantum circuits.

npj Quantum Information (2020)6:42 ; https://doi.org/10.1038/s41534-020-0268-2

INTRODUCTION

Back in 1997 a seminal work by Nielsen and Chuang[1] proposed a quantum version of the programmable gate array that has become a fundamental model for quantum computation[2]. This is a quantum processor where a fixed quantum operation is applied to an input state together with a program state. The aim of the program state is to induce the processor to apply some target quantum gate or channel[3] to the input state. Such a desired feature of quantum programmability comes with a cost: the model cannot be universal, unless the program state is allowed to have an infinite dimension, i.e., infinite qubits[1,4]. Even though this limitation has been known for many years, there is still no exact characterization on how well a finite-dimensional programmable quantum processor can generate or simulate an arbitrary quantum channel. Also there is no literature on how to find the corresponding optimal program state or even to show that this state can indeed be found by some optimization procedure. Here, we show the solutions to these long-standing open problems.

Here, we show that the optimization of programmable quantum computers is a convex problem for which the solution can always be found by means of classical semidefinite programming (SDP), and classical gradient-based methods that are commonly employed for machine learning (ML) applications. ML methods have found wide applicability across many disciplines[5], and we are currently witnessing the development of new hybrid areas of investigation, where ML methods are interconnected with quantum information theory, such as quantum-enhanced ML (refs. [6–10]; e.g., quantum neural networks, quantum annealing, etc.), protocols of quantum-inspired ML (e.g., for recommendation systems[11] or component analysis and supervised clustering[12]), and classical learning methods applied to quantum computers, as explored here in this manuscript.

In our work, we quantify the error between an arbitrary target channel and its programmable simulation in terms of the diamond distance[3,13], and other suitable cost functions, including the trace distance and the quantum fidelity. For all the considered cost functions, we are able to show that the minimization of the simulation error is a convex optimization problem in the space of the program states. This already solves an outstanding problem that affects various models of quantum computers (e.g., variational quantum circuits), where the optimization over classical parameters is non-convex and therefore not guaranteed to converge to a global optimum. By contrast, because our problem is proven to be convex, we can use SDP to minimize the diamond distance and always find the optimal program state for the simulation of a target channel, therefore optimizing the programmable quantum processor. Similarly, we may find suboptimal solutions by minimizing the trace distance or the quantum fidelity by means of gradient-based techniques adapted from the ML literature, such as the projected subgradient method[14] and the conjugate gradient method[15,16]. We note indeed that the minimization of the ℓ_1-norm, mathematically related to the quantum trace distance, is widely employed in many ML tasks[17,18], so many of those techniques can be adapted for learning program states.

With these general results in our hands, we first discuss the optimal learning of arbitrary unitaries with a generic programmable quantum processor. Then, we consider specific designs of the processor, from a shallow scheme based on the teleportation protocol, to higher-depth designs based on port-based teleportation (PBT)[19–21] and parametric quantum circuits (PQCs)[22], introducing a suitable convex reformulation of the latter. In the various cases, we benchmark the processors for the simulation of basic unitary gates (qubit rotations) and various basic channels, including the amplitude damping channel that is known to be the most difficult to simulate[23,24]. For the deeper designs, we find that the optimal program states do not correspond to the Choi matrices of the target channels, which is rather counterintuitive and unexpected.

RESULTS

We first present our main theoretical results on how to train the program state of programmable quantum processors, either via convex optimization or first-order gradient-based algorithms. We then apply our general methods to study the learning of arbitrary unitaries, and the simulation of different channels via processors

[1]Department of Physics and Astronomy, University of Florence, via G. Sansone 1, I-50019 Sesto Fiorentino (FI), Italy. [2]INFN Sezione di Firenze, via G.Sansone 1, I-50019 Sesto Fiorentino (FI), Italy. [3]Department of Computer Science, University of York, York YO10 5GH, UK. [4]Department of Mechanical Engineering, Massachusetts Institute of Technology (MIT), Cambridge, MA 02139, USA. [5]Research Laboratory of Electronics, Massachusetts Institute of Technology (MIT), Cambridge, MA 02139, USA. ✉email: leonardo.banchi@unifi.it

Published in partnership with The University of New South Wales

npj nature partner journals

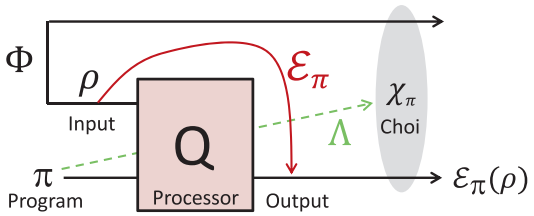

Fig. 1 Quantum processor *Q* with program state π that simulates a quantum channel $\mathcal{E}_\pi$ from input to output. We also show the CPTP map Λ of the processor, from the program state π to the output Choi matrix χ_π (generated by partial transmission of the maximally entangled state Φ).

built either from quantum teleportation and its generalization, or from PQCs.

Programmable quantum computing

Let us consider an arbitrary mapping from d-dimensional input states into d'-dimensional output states, where $d' \neq d$ in the general case. This is described by a quantum channel $\mathcal{E}$ that may represent the overall action of a quantum computation and does not need to be a unitary transformation. Any channel $\mathcal{E}$ can be simulated by means of a programmable quantum processor, which is modeled in general by a fixed completely positive trace-preserving (CPTP) map Q that is applied to both the input state and a variable program state π. In this way, the processor transforms the input state by means of an approximate channel $\mathcal{E}_\pi$ as

$$\mathcal{E}_\pi(\rho) = \mathrm{Tr}_2[Q(\rho \otimes \pi)], \tag{1}$$

where Tr_2 is the partial trace over the program state. A fundamental result[1] is that there is no fixed quantum "processor" Q that is able to exactly simulate any quantum channel $\mathcal{E}$. In other terms, given $\mathcal{E}$, we cannot find the corresponding program π such that $\mathcal{E} \equiv \mathcal{E}_\pi$. Yet simulation can be achieved in an approximate sense, where the quality of the simulation may increase for larger program dimension. In general, the open problem is to determine the optimal program state $\tilde{\pi}$ that minimizes the simulation error, that can be quantified by the cost function

$$C_\diamond(\pi) := \|\mathcal{E} - \mathcal{E}_\pi\|_\diamond, \tag{2}$$

namely the diamond distance[3,13] between the target channel $\mathcal{E}$ and its simulation $\mathcal{E}_\pi$. In other words,

$$\text{Find } \tilde{\pi} \text{ such that } C_\diamond(\tilde{\pi}) = \min_\pi C_\diamond(\pi). \tag{3}$$

From theory[1,4], we know that we cannot achieve $C_\diamond = 0$ for arbitrary $\mathcal{E}$ unless π and Q have infinite dimensions. As a result, for any finite-dimensional realistic design of the quantum processor, finding the optimal program state $\tilde{\pi}$ is an open problem. Recall that the diamond distance is defined by $\|\mathcal{E} - \mathcal{E}_\pi\|_\diamond := \max_\varphi \|\mathcal{I} \otimes \mathcal{E}(\varphi) - \mathcal{I} \otimes \mathcal{E}_\pi(\varphi)\|_1$, where $\mathcal{I}$ is the identity map and $\|O\|_1 := \mathrm{Tr}\sqrt{O^\dagger O}$ is the trace norm[2].

It is important to note that this problem can be reduced to a simpler one by introducing the channel's Choi matrix

$$\begin{aligned} \chi_{\mathcal{E}_\pi} &= \mathcal{I} \otimes \mathcal{E}_\pi(\Phi) \\ &= d^{-1} \sum_{ij} |i\rangle\langle j| \otimes \mathrm{Tr}_2[Q(|i\rangle\langle j| \otimes \pi)], \end{aligned} \tag{4}$$

where $\Phi := |\Phi\rangle\langle\Phi|$ is a d-dimensional maximally entangled state. From this expression, it is clear that the Choi matrix $\chi_{\mathcal{E}_\pi}$ is linear in the program state π. More precisely, the Choi matrix $\chi_{\mathcal{E}_\pi}$ at the output of the processor Q can be directly written as a CPTP linear map Λ acting on the space of the program states π, i.e.,

$$\chi_\pi := \chi_{\mathcal{E}_\pi} = \Lambda(\pi). \tag{5}$$

This map is also depicted in Fig. 1, and fully describes the action of the processor Q. Then, using results from refs. [3,25,26], we may write

$$C_\diamond(\pi) \leq d\, C_1(\pi) \leq 2d\sqrt{C_F(\pi)}, \tag{6}$$

where

$$C_1(\pi) := \|\chi_\mathcal{E} - \chi_\pi\|_1, \tag{7}$$

is the trace distance[2] between target and simulated Choi matrices, and

$$C_F(\pi) = 1 - F(\pi)^2, \tag{8}$$

where $F(\pi)$ is Bures' fidelity between the two Choi matrices $\chi_\mathcal{E}$ and χ_π, i.e.,

$$F(\pi) := \left\|\sqrt{\chi_\mathcal{E}}\sqrt{\chi_\pi}\right\|_1 = \mathrm{Tr}\sqrt{\sqrt{\chi_\mathcal{E}}\chi_\pi\sqrt{\chi_\mathcal{E}}}. \tag{9}$$

Another possible upper bound can be written using the quantum Pinsker's inequality[27,28]. In fact, we may write $C_1(\pi) \leq (2\ln\sqrt{2})\sqrt{C_R(\pi)}$, where

$$C_R(\pi) := \min\{S(\chi_\mathcal{E}\|\chi_\pi), S(\chi_\pi\|\chi_\mathcal{E})\}, \tag{10}$$

and $S(\rho\|\sigma) := \mathrm{Tr}[\rho(\log_2\rho - \log_2\sigma)]$ is the quantum relative entropy between ρ and σ. In Supplementary Note 1.3, we also introduce a cost function $C_p(\pi)$ based on the Schatten p-norm.

Convex optimization

One of the main problems in the optimization of reconfigurable quantum chips is that the relevant cost functions are not convex in the set of classical parameters. This problem is completely solved here thanks to the fact that the optimization of a programmable quantum processor is done with respect to a quantum state. In fact, in the methods section, we prove the following

Theorem 1 *Consider the simulation of a target quantum channel $\mathcal{E}$ by means of a programmable quantum processor Q. The optimization of the cost functions $C_\diamond$, C_1, C_F, C_R, or C_p is a convex problem in the space of program states π. In particular, the global minimum $\tilde{\pi}$ for $C_\diamond$ can always be found as a local minimum.*

This convexity result is generally valid for any cost function that is convex in π. This is the case for any desired norm, not only the trace norm, but also the Frobenius norm, or any Schatten p-norm. It also applies to the relative entropy. Furthermore, the result can also be extended to any convex parametrization of the program states.

When dealing with convex optimization with respect to positive operators, the standard approach is to map the problem to a form that is solvable via SDP (refs. [29,30]). Since the optimal program is the one minimizing the cost function, it is important to write the computation of the cost function itself as a minimization. For the case of the diamond distance, this can be achieved by using the dual formulation[29]. More precisely, consider the linear map $\Omega_\pi := \mathcal{E} - \mathcal{E}_\pi$ with Choi matrix $\chi_{\Omega_\pi} = \chi_\mathcal{E} - \chi_\pi = \chi_\mathcal{E} - \Lambda(\pi)$, and the spectral norm $\|O\|_\infty := \max\{\|Ou\| : u \in \mathbb{C}^d, \|u\| \leq 1\}$, which is the maximum eigenvalue of $\sqrt{O^\dagger O}$. Then, by the strong duality of the diamond norm, $C_\diamond(\pi) = \|\Omega_\pi\|_\diamond$ is given by the SDP (ref. [30])

$$\begin{aligned} &\text{Minimize } 2\|\mathrm{Tr}_2 Z\|_\infty, \\ &\text{Subject to } Z \geq 0 \text{ and } Z \geq d(\chi_\mathcal{E} - \Lambda(\pi)). \end{aligned} \tag{11}$$

The importance of the above dual formulation is that the diamond distance is a minimization, rather than a maximization over a set of matrices. In order to find the optimal program $\tilde{\pi}$, we apply the unique minimization of Eq. (11), where π is variable and satisfies the additional constraints $\pi \geq 0$ and $\mathrm{Tr}(\pi) = 1$.

In the methods section, we show that other cost functions, such as C_1 and C_F can also be written as SDPs. Correspondingly, the optimal programs $\tilde{\pi}$ can be obtained by numerical SDP solvers. Most numerical packages implement second-order algorithms, such as the interior point method[31]. However, second-order

methods tend to be computationally heavy for large problem sizes[32,33], namely when $\tilde{\pi}$ contains many qudits. In the following section, we introduce first-order methods, that are better suited for larger program states. It is important to remark that there also exist zeroth-order (derivative-free) methods, such as the simultaneous perturbation stochastic approximation method[34], which was utilized for a quantum problem in ref. [35]. However, it is known that zeroth-order methods normally have slower convergence times[36] compared to first-order methods.

Gradient-based optimization

In ML applications, where a large amount of data is commonly available, there have been several works that study the minimization of suitable matrix norms for different purposes[17,18,37,38]. First-order methods, are preferred for large dimensional problems, as they are less computationally intensive and require less memory. Here, we show how to apply first-order (gradient-based) algorithms, which are widely employed in ML applications, to find the optimal quantum program.

For this purpose, we need to introduce the subgradient of the cost function C at any point $\pi \in \mathcal{S}$, which is the set

$$\partial C(\pi) = \{Z : C(\sigma) - C(\pi) \geq \mathrm{Tr}\,[Z(\sigma - \pi)],\ \forall \sigma \in \mathcal{S}\}, \tag{12}$$

where Z is Hermitian [39,40]. If C is differentiable, then $\partial C(\pi)$ contains a single element: its gradient $\nabla\, C(\pi)$. We explicitly compute this gradient for an arbitrary programmable quantum processor (1) whose Choi matrix $\chi_{\mathcal{E}_\pi} \equiv \chi_\pi = \Lambda(\pi)$, can be written as a quantum channel Λ that maps a generic program state to the processor's Choi matrix. This map can be defined by its Kraus decomposition $\Lambda(\pi) = \sum_k A_k \pi A_k^\dagger$ for some operators A_k. In fact, let us call $\Lambda^*(\rho) = \sum_k A_k^\dagger \rho A_k$ the dual map, then in the methods section we prove the following

Theorem 2 *Consider an arbitrary quantum channel $\mathcal{E}$ with Choi matrix $\chi_\mathcal{E}$ that is simulated by a quantum processor Q with map $\Lambda(\pi) = \chi_\pi$ (and dual map Λ^*). Then, we may write the following gradients for the trace distance cost $C_1(\pi)$ and the infidelity cost $C_F(\pi)$*

$$\nabla C_1(\pi) = \sum_k \mathrm{sign}(\lambda_k)\Lambda^*(P_k), \tag{13}$$

$$\nabla C_F(\pi) = -2\sqrt{1 - C_F(\pi)}\nabla F(\pi), \tag{14}$$

$$\nabla F(\pi) = \frac{1}{2}\Lambda^*\left[\sqrt{\chi_\mathcal{E}}\left(\sqrt{\chi_\mathcal{E}}\Lambda(\pi)\sqrt{\chi_\mathcal{E}}\right)^{-\frac{1}{2}}\sqrt{\chi_\mathcal{E}}\right], \tag{15}$$

where λ_k (P_k) are the eigenvalues (eigenprojectors) of the Hermitian operator $\chi_\pi - \chi_\mathcal{E}$. When $C_1(\pi)$ or $C_F(\pi)$ are not differentiable in π, then the above expressions provide an element of the subgradient $\partial C(\pi)$.

Once we have the (sub)gradient of the cost function C, we can solve the optimization $\min_{\pi\in\mathcal{S}} C(\pi)$, using the projected subgradient method[14,39]. Let $\mathcal{P}_\mathcal{S}$ be the projection onto the set of program states $\mathcal{S}$, namely $\mathcal{P}_\mathcal{S}(X) = \mathrm{argmin}_{\pi\in\mathcal{S}} \| X - \pi\|_2$, that we show to be computable from the spectral decomposition of any Hermitian X (see Theorem 3 in the "Methods" section). Then, we iteratively apply the steps

$$\begin{aligned} &1)\ \text{Select an operator } g_i \text{ from } \partial C(\pi_i),\\ &2)\ \pi_{i+1} = \mathcal{P}_\mathcal{S}(\pi_i - a_i g_i), \end{aligned} \tag{16}$$

where i is the iteration index, a_i is what is called "learning rate", and Theorem 2 can be employed to find g_i at each step. It is simple to show that π_i converges to the optimal program state $\tilde{\pi}$ in $\mathcal{O}(\epsilon^{-2})$ steps, for any desired precision ϵ such that $|C(\pi) - C(\tilde{\pi})| \leq \epsilon$. Another approach is the conjugate gradient method[15,39], sometimes called Frank–Wolfe algorithm. Here, we apply

$$\begin{aligned} &1)\ \text{Find the smallest eigenvalue } |\sigma_i\rangle \text{ of } \nabla C(\pi_i),\\ &2)\ \pi_{i+1} = \frac{i}{i+2}\pi_i + \frac{2}{i+2}|\sigma_i\rangle\langle\sigma_i|. \end{aligned} \tag{17}$$

When the gradient of f is Lipschitz continuous with constant L, the method converges after $\mathcal{O}(L/\epsilon)$ steps[16,41]. To justify the applicability of this method, a suitable smoothening of the cost function must be employed. The downside of the conjugate gradient method is that it necessarily requires a differentiable cost function C, with gradient ∇C. Specifically, this may create problems for the trace distance cost C_1 that is generally non-smooth. A solution to this problem is to define the cost function in terms of the smooth trace distance $C_\mu(\pi) = \mathrm{Tr}\left[h_\mu(\chi_\pi - \chi_\mathcal{E})\right]$, where h_μ is the so-called Huber penalty function $h_\mu(x) := x^2/(2\mu)$ if $|x| < \mu$ and $|x| - \mu/2$ if $|x| \geq \mu$. This quantity satisfies $C_\mu(\pi) \leq C_1(\pi) \leq C_\mu(\pi) + \mu d/2$ and is a convex function over program states, with gradient $\nabla C_\mu(\pi) = \Lambda^*[h'_\mu(\chi_\pi - \chi_\mathcal{E})])$.

Learning of arbitrary unitaries

One specific application is the simulation of quantum gates or, more generally, unitary transformations[22,42–45]. Here, the infidelity provides the most convenient cost function, as the optimal program can be found analytically. In fact, suppose we use a quantum processor with map Λ to simulate a target unitary U. Because the Choi matrix of U is pure $|\chi_U\rangle\langle\chi_U|$, we first note that $F(\pi)^2 = \langle\chi_U|\Lambda(\pi)|\chi_U\rangle$ and then we see that Eq. (15) drastically simplifies to $\nabla F(\pi) = \Lambda^*(|\chi_U\rangle\langle\chi_U|)/\sqrt{4F(\pi)^2}$. As a result, we find

$$\nabla C_F(\pi) = -\Lambda^*[|\chi_U\rangle\langle\chi_U|], \tag{18}$$

where there is no dependence on π. Therefore, using the conjugate gradient method in Eq. (17), we see that the optimal program state $\tilde{\pi}$ for the infidelity cost function C_F is a fixed point of the iteration and is equal to the maximum eigenvector of $\Lambda^*[|\chi_U\rangle\langle\chi_U|]$.

Teleportation processor

Once we have shown how to optimize a generic programmable quantum processor, we discuss some specific designs, over which we will test the optimization procedure. One possible (shallow) design for the quantum processor Q is a generalized teleportation protocol[46] over an arbitrary program state π. In dimension d, the protocol involves a basis of d^2 maximally entangled states $|\Phi_i\rangle$ and a basis $\{U_i\}$ of teleportation unitaries such that $\mathrm{Tr}(U_i^\dagger U_j) = d\delta_{ij}$ (ref. [47]). An input d-dimensional state ρ and the A part of the program π_{AB} are subject to the projector $|\Phi_i\rangle\langle\Phi_i|$. The classical outcome i is communicated to the B part of π_{AB}, where the correction U_i^{-1} is applied.

The above procedure defines the teleportation channel $\mathcal{E}_\pi$ over ρ

$$\mathcal{E}_\pi^{\mathrm{tele}}(\rho) = \sum_i U_i^B \langle\Phi_i^{SA}|\rho^S \otimes \pi^{AB}|\Phi_i^{SA}\rangle U_i^{B\dagger}. \tag{19}$$

Its Choi matrix can be written as $\chi_\pi = \Lambda_{\mathrm{tele}}(\pi)$, where the map of the teleportation processor is equal to

$$\Lambda_{\mathrm{tele}}(\pi) = d^{-2}\sum_i \left(U_i^* \otimes U_i\right)\pi\left(U_i^* \otimes U_i\right)^\dagger, \tag{20}$$

which is clearly self-dual $\Lambda^* = \Lambda$. Given a target quantum channel $\mathcal{E}$ which is teleportation covariant[23,24], namely when $[\pi, U_i^* \otimes U_i] = 0$, then we know that its simulation is perfect and the optimal program $\tilde{\pi}$ is the channel's Choi matrix, i.e., one of the fixed points of the map Λ_{tele}. For a general channel, the optimal program $\tilde{\pi}$ can be approximated by using the cost functions in our Theorem 2 with Λ being given in Eq. (20), or directly found by optimizing $C_\diamond(\pi)$.

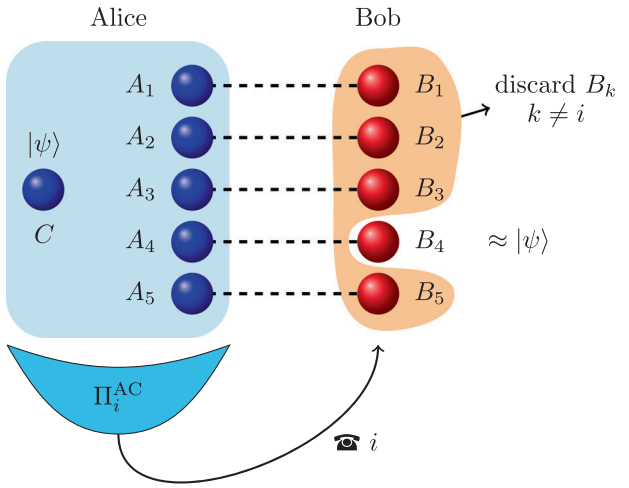

Fig. 2 PBT scheme. Two distant parties, Alice and Bob, share N maximally entangled pairs $\{A_k, B_k\}_{k=1}^N$. Alice also has another system C in the state $|\psi\rangle$. To teleport C, Alice performs the POVM $\{\Pi_i^{\mathbf{AC}}\}$ on all her local systems $\mathbf{A} = \{A_k\}_{k=1}^N$ and C. She then communicates the outcome i to Bob. Bob discards all his systems $\mathbf{B} = \{B_k\}_{k=1}^N$ with the exception of B_i. After these steps, the state $|\psi\rangle$ is approximately teleported to B_i. Similarly, an arbitrary channel $\mathcal{E}$ is simulated with N copies of the Choi matrix $\chi_{\mathcal{E}}^{A_kB_k}$. The figure shows an example with $N = 5$, where $i = 4$ is selected.

Port-based teleportation

A deeper design is provided by a PBT processor, whose overall protocol is illustrated in Fig. 2. Here, we consider a more general formulation of the original PBT protocol[19,20], where the resource entangled pairs are replaced by an arbitrary program state π. In a PBT processor, each party has N systems (or 'ports'), $\mathbf{A} = \{A_1, \ldots, A_N\}$ for Alice and $\mathbf{B} = \{B_1, \ldots, B_N\}$ for Bob. These are prepared in a program state $\pi_{\mathbf{AB}}$. To teleport an input state ρ_C, Alice performs a joint positive operator-value measurement (POVM) $\{\Pi_i\}$ (ref. [19]) on system C and the $\mathbf{A}$-ports. She then communicates the outcome i to Bob, who discards all ports except B_i, which is the output B_{out}. The resulting PBT channel $\mathcal{P}_\pi : \mathcal{H}_C \mapsto \mathcal{H}_{B_{\text{out}}}$ is then

$$\begin{aligned}\mathcal{P}_\pi(\rho) &= \sum_{i=1}^N \mathrm{Tr}_{\mathbf{A}\bar{\mathbf{B}}_i C}\left[\sqrt{\Pi_i}(\pi_{\mathbf{AB}} \otimes \rho_C)\sqrt{\Pi_i}\right]_{B_i \to B_{\text{out}}} \\ &= \sum_{i=1}^N \mathrm{Tr}_{\mathbf{A}\bar{\mathbf{B}}_i C}[\Pi_i(\pi_{\mathbf{AB}} \otimes \rho_C)]_{B_i \to B_{\text{out}}},\end{aligned} \tag{21}$$

where $\bar{\mathbf{B}}_i = \mathbf{B}\backslash B_i = \{B_k : k \neq i\}$.

In the standard PBT protocol[19,20], the program state is fixed as $\pi_{\mathbf{AB}} = \bigotimes_{k=1}^N \Phi_{A_kB_k}$, where $|\Phi_{A_kB_k}\rangle$ are Bell states, and the following POVM is used

$$\Pi_i = \tilde{\Pi}_i + \frac{1}{N}\left(\mathbb{1} - \sum_k \tilde{\Pi}_k\right), \tag{22}$$

where

$$\tilde{\Pi}_i = \sigma_{\mathbf{AC}}^{-1/2}\Phi_{A_iC}\sigma_{\mathbf{AC}}^{-1/2}, \tag{23}$$

$$\sigma_{\mathbf{AC}} := \sum_{i=1}^N \Phi_{A_iC}, \tag{24}$$

and $\sigma^{-1/2}$ is an operator defined only on the support of σ. The PBT protocol is formulated for $N \geq 2$ ports. However, we also include here the trivial case for $N = 1$, corresponding to the process where Alice's input is traced out and the output is the reduced state of Bob's port, i.e., a maximally mixed state. In the limit $N \to \infty$, the standard PBT protocol approximates an identity channel $\mathcal{P}_\pi(\rho) \approx \rho$, with fidelity[19,21] $F_\pi = 1 - \mathcal{O}(\frac{1}{N})$, so perfect simulation is possible only in the limit $N \to \infty$. Since the standard PBT protocol provides an approximation to the identity channel, we call it $\mathcal{I}_N$.

From the PBT simulation of the identity channel, it is possible to approximate any general channel $\mathcal{E}$ by noting that $\mathcal{E}$ can be written as a composition $\mathcal{E} \circ \mathcal{I}$, where $\mathcal{I}$ is the identity channel. This is done by replacing the identity channel $\mathcal{I}$ with its PBT simulation $\mathcal{I}_N$, and then applying $\mathcal{E}$ to B_i. However, since Bob does not perform any post-processing on his systems $\mathbf{B}$, aside from discarding all ports B_k with $k \neq i$, he can also apply first the channel $\mathcal{E}^{\otimes N}$ to all his ports and then discard all the ports B_k with $k \neq i$. In doing so, he changes the program state to

$$\pi_{\mathbf{AB}} = \mathbb{1}_A \otimes \mathcal{E}_B^{\otimes N}\left[\otimes_{k=1}^N \Phi_{A_kB_k}\right] = \otimes_{k=1}^N \chi_{\mathcal{E}}^{A_kB_k}. \tag{25}$$

In other terms, any channel $\mathcal{E}$ can be PBT approximated by N copies of its Choi matrix $\chi_{\mathcal{E}}$ as program state. Since PBT simulation can be decomposed as $\mathcal{E}_\pi = \mathcal{E} \circ \mathcal{I}_N$, the error $C_\diamond^N = \| \mathcal{E} - \mathcal{E}_\pi \|_\diamond$ in simulating the channel $\mathcal{E} \equiv \mathcal{E} \circ \mathcal{I}$ satisfies

$$C_\diamond^N = \| \mathcal{E} \circ \mathcal{I} - \mathcal{E} \circ \mathcal{I}_N \| \leq \| \mathcal{I} - \mathcal{I}_N \|_\diamond \leq 2d(d-1)N^{-1}, \tag{26}$$

where we used the data processing inequality and an upper bound from ref. [48]. While the channel's Choi matrix assures that $C_\diamond^N \to 0$ for large N, for any finite N it does not represent the optimal program state. In general, for any finite N, finding the optimal program state $\pi_{\mathbf{AB}}$ simulating a channel $\mathcal{E}$ with PBT is an open problem, and no explicit solutions or procedures are known.

We employ our convex optimization procedures to find the optimal program state. This can be done either exactly by minimizing the diamond distance cost function $C_\diamond$ via SDP, or approximately, by determining the optimal program state via the minimization of the trace distance cost function C_1 via either SDP or the gradient-based techniques discussed above. For this second approach, we need to derive the map Λ of the PBT processor, between the program state π to output Choi matrix as in Eq. (5). To compute the Choi matrix and CP-map Λ, we consider an input maximally entangled state $|\Phi_{DC}\rangle$ and a basis $|e_j^i\rangle$ of $\mathbf{A}\{\mathbf{B}\backslash B_i\}C$. Then, by using Eq. (21) and the definition $\Lambda(\pi) = \chi_{\mathcal{P}_\pi} = \mathbb{1}_D \otimes \mathcal{P}_\pi[\Phi_{DC}]$, we find the map $\Lambda_{\mathbf{AB} \to DB_{\text{out}}}$ of a PBT processor

$$\Lambda(\pi) = \sum_{ij} K_{ij}\pi K_{ij}^\dagger,\ K_{ij} := \left\langle e_j^i \middle| \sqrt{\Pi_i} \otimes \mathbb{1}_{BD} \middle| \Phi_{DC} \right\rangle. \tag{27}$$

Note that a general program state for PBT consists of $2N$ qudits, and hence the parameter space has exponential size d^{4N}. However, because the PBT protocol is symmetric under permutation of port labels, we show in Supplementary Note 6 that one can exploit this symmetry and reduce the number of free parameters to the binomial coeffficient $\binom{N + d^4 - 1}{d^4 - 1}$, which is polynomial in the number of ports N. Despite this exponential reduction, the scaling in the number of parameters still represents a practical limiting factor, even for qubits for which $\mathcal{O}(N^{15})$. A suboptimal strategy consists in reducing the space of program states to a convex set that we call the "Choi space" $\mathcal{C}$. Consider an arbitrary probability distribution $\{p_k\}$ and then define

$$\mathcal{C} = \{\pi : \pi = \sum_k p_k \rho_{AB}^{k \otimes N},\ \mathrm{Tr}_B(\rho_{AB}^k) = d^{-1}\mathbb{1}\}. \tag{28}$$

One can show (see Supplementary Note 6) that a global minimum in $\mathcal{C}$ is a global minimum in the extremal (non-convex) subspace for $p_k = \delta_{k,1}$, consisting of tensor products of Choi matrices $\rho_{AB}^{\otimes N}$. Among these states, there is the N-copy Choi matrix of the target channel $\chi_{\mathcal{E}}^{\otimes N} = [\mathcal{I} \otimes \mathcal{E}(|\Phi\rangle\langle\Phi|)]^{\otimes N}$, which is not necessarily the optimal program, as we show below.

Parametric quantum circuits

Another deep design of quantum processor is based on PQCs (refs. [22,49]). A PQC is a sequence of unitary matrices

$U(t) = U_N(t_N)\ldots U_2(t_2)U_1(t_1)$, where $U_j(t_j) = \exp(it_jH_j)$ for some Hamiltonian H_j and time interval t_j. The problem with PQCs is that the cost functions in the classical parameters[42] are not convex, so that numerical algorithms are not guaranteed to converge to the global optimum. Here, we fix this issue by introducing a convex formulation of PQCs, where classical parameters are replaced by a quantum program. This results in a programmable PQC processor that is optimizable by our methods.

The universality of PQCs can be employed for universal channel simulation. Indeed, thanks to Stinespring's dilation theorem, any channel can be written as a unitary evolution on a bigger space, $\mathcal{E}(\rho_A) = \mathrm{Tr}_{R_0}[U(\rho_A \otimes \theta_0)U^\dagger]$, where the system is paired to an extra register R_0 and θ_0 belongs to R_0. In the Stinespring representation, U acts on system A and register R_0. In ref. [49], it has been shown that sequences of two unitaries, U_0 and U_1, are almost universal for simulation, i.e., any target unitary U can be approximated as $U \approx \cdots U_1^{m_4}U_0^{m_3}U_1^{m_2}U_0^{m_1}$ for some integers m_j. Under suitable conditions, it takes $\mathcal{O}(d^2\epsilon^{-d})$ steps to approximate U up to precision ϵ. The choice between U_0 and U_1 is done by measuring a classical bit. We may introduce a quantum version, where the two different unitaries $U_0 = e^{iH_0}$ or $U_1 = e^{iH_1}$ are chosen depending on the state of qubit R_j. This results in the conditional gate

$$\hat{U}_j = \exp\Big(iH_0 \otimes |0\rangle_{jj}\langle 0| + iH_1 \otimes |1\rangle_{jj}\langle 1|\Big). \tag{29}$$

Channel simulation is then obtained by replacing the unitary evolution U in the Stinespring dilation via its simulation. The result is illustrated in Fig. 3, where the program state π is defined over $\mathbf{R} = (R_0, \ldots, R_N)$ and each $\hat{H}_j$ acts on the input system A and two ancillary qubits R_0 and R_j. Following the universality construction of ref. [49], we show in the Supplementary Note 3.4 that the channel shown in Fig. 3 provides a universal processor. Moreover, the channel Λ that maps any program π to the processor's Choi matrix is obtained as

$$\Lambda(\pi) = \mathrm{Tr}_{\mathbf{R}}\Big[\hat{U}_{A\mathbf{R}}(\Phi_{BA} \otimes \pi_{\mathbf{R}})\hat{U}_{A\mathbf{R}}^\dagger\Big], \tag{30}$$

where $\hat{U}_{A\mathbf{R}} = \mathbb{1}_B \otimes \prod_{j=1}^N \hat{U}_{j,A,R_0,R_j}$, from which we can identify the optimal program $|\tilde{\pi}\rangle$ via our methods.

PQCs are not inherently monotonic. A deeper (higher N) design may simulate a given channel worse than a more shallow design. We can design a modified PQC that is monotonic by design, which we designate a "monotonic PQC" (mPQC), by replacing the qubits in our program state with qutrits, and modifying Eq. (29) to read

$$\hat{U}_j = \exp\Big(iH_0 \otimes |0\rangle_{jj}\langle 0| + iH_1 \otimes |1\rangle_{jj}\langle 1| + \mathbf{0} \otimes |2\rangle_{jj}\langle 2|\Big), \tag{31}$$

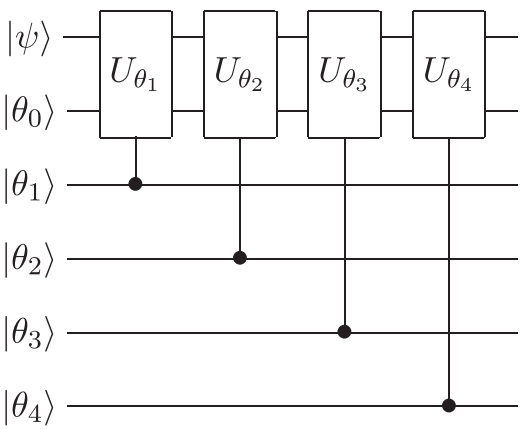

Fig. 3 PQC Scheme. Thanks to the Stinespring decomposition, a quantum channel is written as a unitary evolution in an extended space. The corresponding unitary is then simulated via a parametric quantum circuit that, at each time, applies a certain unitary depending on the state of the quantum register.

where $\mathbf{0}$ is a zero operator, so that gate j enacts the identity channel if program qutrit j is in the state $|2\rangle\langle 2|$. Then, if it were the case that a PQC with N program qubits could simulate a given channel better than one with $N+m$, a mPQC with $N+m$ qutrits in the program state could perform at least, as well as the PQC with N program qubits by setting the first m qutrits to $|2\rangle\langle 2|$. This processor design is both universal and monotonic. More precisely, let $C(\mathrm{PQC}_N)$ denote the value of a cost function C for simulating a channel $\mathcal{E}$ with an N-gate PQC, using the optimal program state, and let $C(\mathrm{mPQC}_N)$ denote the value of C for simulating $\mathcal{E}$ with an N-gate mPQC, again using the optimal program state. We are then guaranteed that

$$C(\mathrm{mPQC}_N) \leq \min_{M \leq N} C(\mathrm{PQC}_M). \tag{32}$$

Processor benchmarking

In order to show the performance of the various architectures, we consider the simulation of an amplitude damping channel with probability p. The reason is because this is the most difficult channel to simulate, with a perfect simulation only known for infinite dimension, e.g., using continuous-variable quantum operations[23]. In Figs. 4 and 5, we compare teleportation-based, PBT, PQC, and "mPQC" programmable processors whose program states have been optimized according to the cost functions $C_\diamond$ and C_1. For the PBT processor, the trace distance cost C_1 is remarkably close to $C_\diamond$ and allows us to easily explore high depths. Note that the optimal program states differ from the naive choice of the Choi matrix of the target channel. Note too that PQC processors display non-monotonic behavior when simulating amplitude damping channels, meaning that shallow PQC processors (e.g., for $N = 4$) may perform better than deeper processors. For the PQC processor, we use the universal Hamiltonians $H_0 = \sqrt{2}(X \otimes Y - Y \otimes X)$ and $H_1 = (\sqrt{2}Z + \sqrt{3}Y + \sqrt{5}X) \otimes (Y + \sqrt{2}Z)$, where X, Y, and Z are Pauli operators. mPQC processors guarantee that deeper designs always perform at least, as well as any shallower design. In Fig. 5 perfect simulation is achievable at specific values of p because of our choice of the universal gates U_0 and U_1. More details are provided in the Supplementary Note 3.

Many other numerical simulations are performed in the Supplementary Note 3 where we study the convergence rate in learning a unitary operation, the exact simulation of Pauli

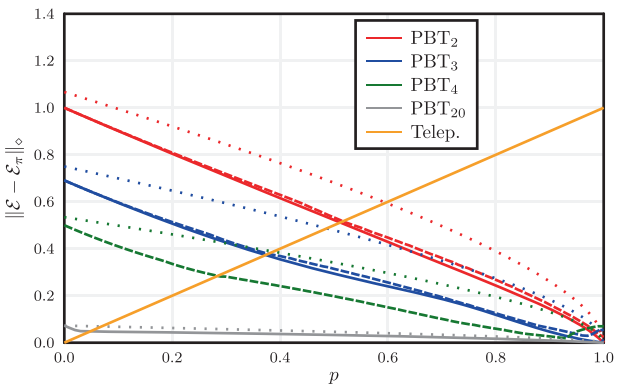

Fig. 4 Diamond distance error $C_\diamond$ in simulating an amplitude damping channel $\mathcal{E}_p$ at various damping rates p. We compare the performance of different designs for the programmable quantum processor: standard teleportation and PBT with N ports (PBT_N). The optimal program $\tilde{\pi}$ is obtained by either minimizing directly the diamond distance $C_\diamond$ (solid lines), or the trace distance C_1 (dashed lines) via the projected subgradient iteration. In both cases, from $\tilde{\pi}$ we then compute $C_\diamond(\tilde{\pi})$. The lowest curves are obtained by optimizing π over the Choi space in Eq. (28). For comparison, we also show the (non-optimal) performance when the program is the channel's Choi matrix (dotted lines).

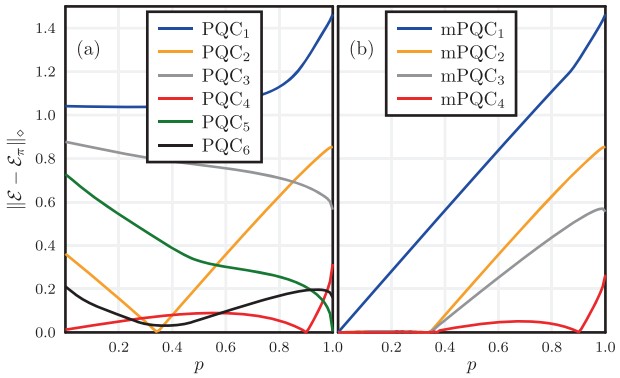

Fig. 5 Diamond distance error $C_\diamond$ in simulating an amplitude damping channel $\mathcal{E}_p$ at various damping rates p. We compare the performance of two different designs for the programmable quantum processor: PQCs with $N+1$ registers (PQC_N) and monotonic parametric quantum circuits with $N+1$ registers (mPQC_N). In both cases the optimal program $|\bar{\pi}\rangle$ is obtained by minimizing the diamond distance $C_\diamond$.

channels, and approximate simulation of both dephasing and amplitude damping channels. In particular, we study the performance of the approximate solution when optimizing over larger, but easier-to-compute, cost functions, such as the trace distance or the infidelity.

DISCUSSION

In this work, we have considered a general finite-dimensional model of a programmable quantum processor, which is a fundamental scheme for quantum computing and also a primitive tool for other areas of quantum information. By introducing suitable cost functions, based on the diamond distance, trace distance and quantum fidelity, we have shown how to characterize the optimal performance of this processor in the simulation of an arbitrary quantum gate or channel. In fact, we have shown that the minimization of these cost functions is a convex optimization problem that can always be solved.

In particular, by minimizing the diamond distance via SDP, we can always determine the optimal program state for the simulation of an arbitrary channel. Alternatively, we may minimize the simpler but larger cost functions in terms of trace distance and quantum fidelity via gradient-based methods adapted from ML, so as to provide a very good approximation of the optimal program state. This other approach can also provide closed analytical solutions, as is the case for the simulation of arbitrary unitaries, for which the minimization of the fidelity cost function corresponds to computing an eigenvector.

We have then applied our results to various designs of programmable quantum processor, from a shallow teleportation-based scheme to deeper and asymptotically universal designs that are based on PBT and PQCs. We have explicitly benchmarked the performances of these quantum processors by considering the simulation of unitary gates, depolarizing and amplitude damping channels, showing that the optimal program states may differ from the naive choice based on the Choi matrix of the target channel. Moreover, our results can be applied also for universal quantum measurements[50].

A potential application of our work may be the development of "programmable" model of cloud-based quantum computation, where a client has an input state to be processed by an online quantum server that is equipped with a programmable quantum processor. The client classically informs the server about what type of computation it needs (e.g., some specified quantum algorithm) and the server generates an optimal program state that closely approximates the overall quantum channel to be applied to the input. The server then accepts the input from the client, processes it, and returns the output together with the value of a cost function quantifying how close the computation was with respect to the client's request.

Our results may also be useful in areas beyond quantum computing, wherever channel simulation is a basic problem. For instance, this is the case when we investigate the ultimate limits of quantum communications[24], design optimal Hamiltonians for one-way quantum repeaters, and for all those areas of quantum sensing, hypothesis testing and metrology that are based on quantum channel simulations [51]. Indeed the study of adaptive protocols of quantum channel discrimination (or estimation) is notoriously difficult, and their optimal performance is not completely understood. Nonetheless, these protocols can be analyzed by using simulation techniques[48,51] where the channel, encoding the unknown parameter, is replaced by an approximate simulating channel, and its parameter is mapped into the label of a program state (therefore reducing the problem from channel to state discrimination/estimation). In this regard, our theory provides the optimal solution to this basic problem, by determining the best simulating channel and the corresponding program state.

METHODS

Convexity proofs

In this section, we provide a proof of Theorem 1, namely we show that the minimization of the main cost functions $C_\diamond$, C_1, and C_F is a convex optimization problem in the space of the program states π. This means that we can find the optimal program state $\bar{\pi}$ by minimizing $C_\diamond$ or, alternatively, suboptimal program states can be found by minimizing either C_1 or C_F. For the sake of generality, we prove the result for all of the cost functions discussed in the previous sections. We restate Theorem 1 below for completeness:

Theorem *The minimization of the generic cost function $C = C_\diamond, C_1, C_F, C_R$ or C_p for any $p > 1$ is a convex optimization problem in the space of program states.*

Proof Let us start to show the result for the diamond distance $C_\diamond$. In this case, we can write the following

$$\begin{aligned} &C_\diamond[p\pi+(1-p)\pi'] \\ &:= \left\|\mathcal{E}-\mathcal{E}_{p\pi+(1-p)\pi'}\right\|_\diamond \\ &\overset{(1)}{=} \|(p+1-p)\mathcal{E}-p\mathcal{E}_\pi-(1-p)\mathcal{E}_{\pi'}\|_\diamond \\ &\overset{(2)}{\leq} \|p\mathcal{E}-p\mathcal{E}_\pi\|_\diamond+\|(1-p)\mathcal{E}-(1-p)\mathcal{E}_{\pi'}\|_\diamond \\ &\overset{(3)}{\leq} p\|\mathcal{E}-\mathcal{E}_\pi\|_\diamond+(1-p)\|\mathcal{E}-\mathcal{E}_{\pi'}\|_\diamond \\ &= pC_\diamond(\pi)+(1-p)C_\diamond(\pi'), \end{aligned} \tag{33}$$

where we use (1) the linearity of $\mathcal{E}$, (2) the triangle inequality, and (3) the property $\|xA\|_1 = |x|\|A\|_1$, valid for any operator A and coefficient x.

For any Schatten p-norm C_p with $p \geq 1$, we may prove convexity following a similar reasoning. Since for any combination $\overline{\pi} := p_0\pi_0 + p_1\pi_1$, with $p_0 + p_1 = 1$, we have $\Lambda(\overline{\pi}) = p_0\Lambda(\pi_0) + p_1\Lambda(\pi_1)$, then by exploiting the triangle inequality, and the property $\|xA\|_p = |x|\|A\|_p$, we can show that

$$\begin{aligned} C_p(p_0\pi_0+p_1\pi_1) &:= \|\chi_\mathcal{E}-\Lambda(p_0\pi_0+p_1\pi_1)\|_p \\ &\leq p_0\|\chi_\mathcal{E}-\Lambda(\pi_0)\|_p+p_1\|\chi_\mathcal{E}-\Lambda(\pi_1)\|_p \\ &= p_0C_p(\pi_0)+p_1C_p(\pi_1). \end{aligned} \tag{34}$$

To show the convexity of C_F, defined in Eq. (8), we note that the fidelity function $F(\rho, \sigma)$ satisfies the following concavity relation[52]

$$F\left(\sum_k p_k\rho_k,\sigma\right)^2 \geq \sum_k p_kF(\rho_k,\sigma)^2. \tag{35}$$

Due to the linearity of $\chi_\pi = \Lambda(\pi)$, the fidelity in Eq. (9) satisfies $F_{\overline{\pi}}^2 \geq \sum_k p_kF_{\pi_k}^2$ for $\overline{\pi} := \sum_k p_k\pi_k$. Accordingly, we get the following convexity

result

$$C_F\left(\sum_k p_k\pi_k\right) \leq \sum_k p_k C_F(\pi_k)\,. \tag{36}$$

For the cost function C_R, the result comes from the linearity of $\Lambda(\pi)$ and the joint convexity of the relative entropy. In fact, for $\overline{\pi} := p_0\pi_0 + p_1\pi_1$, we may write

$$\begin{aligned} S[\Lambda(\overline{\pi})||\chi_{\mathcal{E}}] &= S[p_0\Lambda(\pi_0) + p_1\Lambda(\pi_1)||\chi_{\mathcal{E}}] \\ &= S[p_0\Lambda(\pi_0) + p_1\Lambda(\pi_1)||p_0\chi_{\mathcal{E}} + p_1\chi_{\mathcal{E}}] \\ &\leq p_0 S[\Lambda(\pi_0),\chi_{\mathcal{E}}] + p_1 S[\Lambda(\pi_1),\chi_{\mathcal{E}}], \end{aligned} \tag{37}$$

with a symmetric proof for $S[\chi_{\mathcal{E}}||\Lambda(\overline{\pi})]$. This implies the convexity of $C_R(\pi)$ in Eq. (10).■

Convex classical parametrizations

The result of the Theorem 1 can certainly be extended to any convex parametrization of program states. For instance, assume that $\pi = \pi(\boldsymbol{\lambda})$, where $\boldsymbol{\lambda} = \{\lambda_i\}$ is a probability distribution. This means that, for $0 \leq p \leq 1$ and any two parametrizations, $\boldsymbol{\lambda}$ and $\boldsymbol{\lambda}'$, we may write

$$\pi[p\boldsymbol{\lambda} + (1-p)\boldsymbol{\lambda}'] = p\pi(\boldsymbol{\lambda}) + (1-p)\pi(\boldsymbol{\lambda}'). \tag{38}$$

Then the problem remains convex in $\boldsymbol{\lambda}$ and we may therefore find the global minimum in these parameters. It is clear that this global minimum $\tilde{\boldsymbol{\lambda}}$ identifies a program state $\pi(\tilde{\boldsymbol{\lambda}})$ that is not generally the optimal state $\tilde{\pi}$ in the entire program space $\mathcal{S}$, even though the solution may be a convenient solution for experimental applications.

Note that a possible classical parametrization consists of using classical program states, of the form

$$\pi(\boldsymbol{\lambda}) = \sum_i \lambda_i |\varphi_i\rangle\langle\varphi_i|, \tag{39}$$

where $\{|\varphi_i\rangle\}$ is an orthonormal basis in the program space. Convex combinations of probability distributions therefore define a convex set of classical program states

$$\mathcal{S}_{\text{class}} = \{\pi : \pi = \sum_i \lambda_i |\varphi_i\rangle\langle\varphi_i|,\ \langle\varphi_i|\varphi_j\rangle = \delta_{ij}\}. \tag{40}$$

Optimizing over this specific subspace corresponds to optimizing the programmable quantum processor over classical programs. It is clear that global minima in $\mathcal{S}_{\text{class}}$ and $\mathcal{S}$ are expected to be very different. For instance, $\mathcal{S}_{\text{class}}$ cannot certainly include Choi matrices that are usually very good quantum programs.

Gradient-based optimization

As discussed in the main text, the SDP formulation allows the use of powerful and accurate numerical methods, such as the interior point method. However, these algorithms are not suitable for high-dimensional problems, due to their higher computational and memory requirements. Therefore, an alternative approach (useful for larger program states) consists of the optimization of the larger but easier-to-compute cost function $C = C_1$ (trace distance) or C_F (infidelity), for which we can use first-order methods. Indeed, according to Theorem 1, all of the proposed cost functions $C : \mathcal{S} \to \mathbb{R}$ are convex over the program space $\mathcal{S}$ and, therefore, we can solve the optimization $\min_{\pi\in\mathcal{S}} C(\pi)$ by using gradient-based algorithms.

Gradient-based convex optimization is at the heart of many popular ML techniques, such as online learning in a high-dimensional feature space[17], missing value estimation problems[18], text classification, image ranking, and optical character recognition[53], to name a few. In all of the above applications, "learning" corresponds to the following minimization problem: $\min_{x\in\mathcal{S}} f(x)$, where $f(x)$ is a convex function and $\mathcal{S}$ is a convex set. Quantum learning falls into this category, as the space of program states is convex due to the linearity of quantum mechanics and the fact that cost functions are typically convex in this space (see Theorem 1). Gradient-based approaches are among the most applied methods for convex optimization of non-linear, possibly non-smooth functions[39].

When the cost function is not differentiable, we cannot formally define its gradient. Nonetheless, we can always define the subgradient ∂C of C as in Eq. (12), which in principle contains many points. When C is not only convex but also differentiable, then $\partial C(\pi) = \{\nabla C(\pi)\}$, i.e., the subgradient contains a single element, the gradient ∇C, that can be obtained via the Fréchet derivative of C (for more details see Supplementary Note 4). When C is not differentiable, the gradient still provides an element of the subgradient that can be used in the minimization algorithm.

In order to compute the gradient ∇C, it is convenient to consider the Kraus decomposition of the processor map Λ. Let us write

$$\Lambda(\pi) = \sum_k A_k \pi A_k^\dagger, \tag{41}$$

with Kraus operators A_k. We then define the dual map Λ^* of the processor as the one (generally non-trace-preserving), which is given by the following decomposition

$$\Lambda^*(\rho) = \sum_k A_k^\dagger \rho A_k. \tag{42}$$

With these definitions in hands, we can now prove Theorem 2, which we rewrite here for convenience.

Theorem *Suppose we use a quantum processor Q with map $\Lambda(\pi) = \chi_\pi$ in order to approximate the Choi matrix $\chi_{\mathcal{E}}$ of an arbitrary channel $\mathcal{E}$. Then, the gradients of the trace distance $C_1(\pi)$ and the infidelity $C_F(\pi)$ are given by the following analytical formulas*

$$\nabla C_1(\pi) = \sum_k \text{sign}(\lambda_k)\Lambda^*(P_k), \tag{43}$$

$$\nabla C_F(\pi) = -2\sqrt{1 - C_F(\pi)}\nabla F(\pi), \tag{44}$$

$$\nabla F(\pi) = \frac{1}{2}\Lambda^*\left[\sqrt{\chi_{\mathcal{E}}}\left(\sqrt{\chi_{\mathcal{E}}}\Lambda(\pi)\sqrt{\chi_{\mathcal{E}}}\right)^{-\frac{1}{2}}\sqrt{\chi_{\mathcal{E}}}\right], \tag{45}$$

where λ_k (P_k) are the eigenvalues (eigenprojectors) of the Hermitian operator $\chi_\pi - \chi_{\mathcal{E}}$. When $C_1(\pi)$ or $C_F(\pi)$ are not differentiable at π, then the above expressions provide an element of the subgradient $\partial C(\pi)$.

Proof We prove the above theorem assuming that the functions are differentiable for program π. For non-differentiable points, the only difference is that the above analytical expressions are not unique and provide only one of the possibly infinite elements of the subgradient. Further details of this mathematical proof are given in Supplementary Note 4. Following matrix differentiation, for any function $f(A) = \text{Tr}[g(A)]$ of a matrix A, we may write

$$d\,\text{Tr}\,[g(A)] = \text{Tr}\,[g'(A)dA], \tag{46}$$

and the gradient is $\nabla f(A) = g'(A)$. Both the trace distance and fidelity cost functions can be written in this form. To find the explicit gradient of the fidelity function, we first note that, by linearity, we may write

$$\Lambda(\pi + \delta\pi) = \Lambda(\pi) + \Lambda(\delta\pi)\,, \tag{47}$$

and therefore the following expansion

$$\sqrt{\chi_{\mathcal{E}}}\Lambda(\pi + \delta\pi)\sqrt{\chi_{\mathcal{E}}} = \sqrt{\chi_{\mathcal{E}}}\Lambda(\pi)\sqrt{\chi_{\mathcal{E}}} + \sqrt{\chi_{\mathcal{E}}}\Lambda(\delta\pi)\sqrt{\chi_{\mathcal{E}}}\,. \tag{48}$$

From this equation and differential calculations of the fidelity (see Supplementary Note 4.2 for details), we find

$$dF = \frac{1}{2}\,\text{Tr}\left[\left(\sqrt{\chi_{\mathcal{E}}}\Lambda(\pi)\sqrt{\chi_{\mathcal{E}}}\right)^{-\frac{1}{2}}\sqrt{\chi_{\mathcal{E}}}\Lambda(\delta\pi)\sqrt{\chi_{\mathcal{E}}}\right], \tag{49}$$

where $dF = F(\pi + \delta\pi) - F(\pi)$. Then, using the cyclic property of the trace, we get

$$dF = \frac{1}{2}\,\text{Tr}\left[\Lambda^*\left[\sqrt{\chi_{\mathcal{E}}}\left(\sqrt{\chi_{\mathcal{E}}}\Lambda(\pi)\sqrt{\chi_{\mathcal{E}}}\right)^{-\frac{1}{2}}\sqrt{\chi_{\mathcal{E}}}\right]\delta\pi\right]. \tag{50}$$

Exploiting this expression in Eq. (46), we get the gradient $\nabla F(\pi)$ as in Eq. (45). The other Eq. (44) simply follows from applying the definition in Eq. (8).

For the trace distance, let us write the eigenvalue decomposition

$$\chi_\pi - \chi_{\mathcal{E}} = \sum_k \lambda_k P_k\,. \tag{51}$$

Then using the linearity of Eq. (47), the definition of a processor map of Eq. (5) and differential calculations of the trace distance (see Supplementary Note 4.3 for details), we can write

$$\begin{aligned} dC_1(\pi) &= \sum_k \text{sign}(\lambda_k)\,\text{Tr}\,[P_k\Lambda(d\pi)] \\ &= \sum_k \text{sign}(\lambda_k)\,\text{Tr}\,[\Lambda^*(P_k)d\pi] \\ &= \text{Tr}\,\{\Lambda^*[\text{sign}(\chi_\pi - \chi_{\mathcal{E}})]d\pi\}\,. \end{aligned} \tag{52}$$

From the definition of the gradient in Eq. (46), we finally get

$$\nabla C_1(\pi) = \Lambda^*[\mathrm{sign}(\chi_\pi - \chi_\mathcal{E})], \tag{53}$$

which leads to the result in Eq. (43). ■

The above results in Eqs. (44) and (43) can be used together with the projected subgradient method[14] or conjugate gradient algorithm[15,16] to iteratively find the optimal program state in the minimization of $\min_{\pi\in\mathcal{S}} C(\pi)$ for $C = C_1$ or C_F. In the following sections, we present two algorithms, the projected subgradient method and the conjugate gradient method, and show how they can be adapted to our problem.

Projected subgradient methods have the advantage of simplicity and the ability to optimize non-smooth functions, but can be slower, with a convergence rate $\mathcal{O}(\epsilon^{-2})$ for a desired accuracy ϵ. Conjugate gradient methods[15,16] have a faster convergence rate $\mathcal{O}(\epsilon^{-1})$, provided that the cost function is smooth. This convergence rate can be improved even further to $\mathcal{O}(\epsilon^{-1/2})$ for strongly convex functions[54] or using Nesterov's accelerated gradient method[41]. The technical difficulty in the adaptation of these methods for learning program states comes because the latter is a constrained optimization problem, namely at each iteration step the optimal program must be a proper quantum state, and the cost functions coming from quantum information theory are, generally, non-smooth.

Projected subgradient method

Given the space $\mathcal{S}$ of program states, let us define the projection $\mathcal{P}_\mathcal{S}$ onto $\mathcal{S}$ as

$$\mathcal{P}_\mathcal{S}(X) = \operatorname*{argmin}_{\pi\in\mathcal{S}} \|X - \pi\|_2, \tag{54}$$

where argmin is the argument of the minimum, namely the closest state $\pi \in \mathcal{S}$ to the operator X. Then, a first-order algorithm to solve $\min_{\pi\in\mathcal{S}} C(\pi)$ is to apply the projected subgradient method[14,39], which iteratively applies the iteration (16), which we rewrite below for convenience

$$\begin{aligned} &1)\ \text{Select an operator } g_i \text{ from } \partial C(\pi_i),\\ &2)\ \text{Update } \pi_{i+1} = \mathcal{P}_\mathcal{S}(\pi_i - a_i g_i), \end{aligned} \tag{55}$$

where i is the iteration index and a_i a learning rate.

The above algorithm differs from standard gradient methods in two aspects: (i) the update rule is based on the subgradient, which is defined even for non-smooth functions; (ii) the operator $\pi_i - a_i g_i$ is generally not a quantum state, so the algorithm fixes this issue by projecting that operator back to the closest quantum state, via Eq. (54). The algorithm converges to the optimal solution π_* (approximating the optimal program $\tilde{\pi}$) as[14]

$$C(\pi_i) - C(\pi_*) \le \frac{e_1 + G\sum_{k=1}^i a_k^2}{2\sum_{k=1}^i a_k} =: \epsilon, \tag{56}$$

where $e_1 = \|\pi_1 - \pi_*\|_2^2$ is the initial error (in Frobenius norm) and G is such that $\|g\|_2^2 \le G$ for any $g \in \partial C$. Popular choices for the learning rate that assure convergence are $a_k \propto 1/\sqrt{k}$ and $a_k = a/(b+k)$ for some $a, b > 0$.

In general, the projection step is the major drawback, which often limits the applicability of the projected subgradient method to practical problems. Indeed, projections like Eq. (54) require another full optimization at each iteration that might be computationally intensive. Nonetheless, we show in the following theorem that this issue does not occur in learning quantum states, because the resulting optimization can be solved analytically.

Theorem 3 *Let X be a Hermitian operator in a d-dimensional Hilbert space with spectral decomposition $X = UxU^\dagger$, where the eigenvalues x_j are ordered in decreasing order. Then $\mathcal{P}_\mathcal{S}(X)$ of Eq. (54) is given by*

$$\mathcal{P}_\mathcal{S}(X) = U\lambda U^\dagger,\ \lambda_i = \max\{x_i - \theta, 0\}, \tag{57}$$

where $\theta = \frac{1}{s}\sum_{j=1}^{s}(x_j - 1)$ *and*

$$s = \max\left\{k \in [1,\ldots,d] : x_k > \frac{1}{k}\sum_{j=1}^{k}(x_j - 1)\right\}. \tag{58}$$

Proof Any quantum (program) state can be written in the diagonal form $\pi = V\lambda V^\dagger$, where V is a unitary matrix, and λ is the vector of eigenvalues in decreasing order, with $\lambda_j \ge 0$ and $\sum_j \lambda_j = 1$. To find the optimal state, it is required to find both the optimal unitary V and the optimal eigenvalues λ with the above property, i.e.,

$$\mathcal{P}_\mathcal{S}(X) = \operatorname*{argmin}_{V,\lambda} \|X - V\lambda V^\dagger\|_2. \tag{59}$$

For any unitarily invariant norm, the following inequality holds (ref. [55], Eq. IV.64):

$$\|X - \pi\|_2 \ge \|x - \lambda\|_2, \tag{60}$$

with equality when $U = V$, where $X = UxU^\dagger$ is a spectral decomposition of X such that the x_j's are in decreasing order. This shows that the optimal unitary in Eq. (59) is the diagonalization matrix of the operator X. The eigenvalues of any density operator form a probability simplex. The optimal eigenvalues λ are then obtained thanks to Algorithm 1 from ref. [17]. ■

In the following section, we present an alternative algorithm with faster convergence rates, but stronger requirements on the function to be optimized.

Conjugate gradient method

The conjugate gradient method[15,39], sometimes called the Frank–Wolfe algorithm, has been developed to provide a better convergence speed and to avoid the projection step at each iteration. Although the latter can be explicitly computed for quantum states (thanks to our Theorem 3), having a faster convergence rate is important, especially with higher dimensional Hilbert spaces. The downside of this method is that it necessarily requires a differentiable cost function C, with gradient ∇C.

In its standard form, the conjugate gradient method to approximate the solution of $\mathrm{argmin}_{\pi\in\mathcal{S}} C(\pi)$ is defined by the following iterative rule

$$\begin{aligned} &1)\ \text{Find } \mathrm{argmin}_{\sigma\in\mathcal{S}}\ \mathrm{Tr}\left[\sigma\nabla C(\pi_i)\right],\\ &2)\ \pi_{i+1} = \pi_i + \tfrac{2}{i+2}(\sigma - \pi_i) = \tfrac{i}{i+2}\pi_i + \tfrac{2}{i+2}\sigma. \end{aligned} \tag{61}$$

The first step in the above iteration rule is solved by finding the smallest eigenvector $|\sigma\rangle$ of $\nabla C(\pi_i)$. Indeed, since π is an operator and $C(\pi)$ a scalar, the gradient ∇C is an operator with the same dimension as π. Therefore, for learning quantum programs, we find the iteration (17), that we rewrite below for convenience

$$\begin{aligned} &1)\ \text{Find the smallest eigenvalue } |\sigma_i\rangle \text{ of } \nabla C(\pi_i),\\ &2)\ \pi_{i+1} = \frac{i}{i+2}\pi_i + \frac{2}{i+2}|\sigma_i\rangle\langle\sigma_i|. \end{aligned} \tag{62}$$

When the gradient of C is Lipschitz continuous with constant L, the conjugate gradient method converges after $\mathcal{O}(L/\epsilon)$ steps[16,41]. The following iteration with adaptive learning rate a_i has even faster convergence rates, provided that C is strongly convex[54]:

$$\begin{aligned} &1)\ \text{Find the smallest eigenvalue } |\sigma_i\rangle \text{ of } \nabla C(\pi_i),\\ &2)\ \text{Find } a_i = \mathrm{argmin}_{a\in[0,1]} a\langle\tau_i, \nabla C(\pi_i)\rangle\\ &\quad + a^2\tfrac{\beta_C}{2}\|\tau_i\|_C^2,\ \text{for } \tau_i = |\sigma_i\rangle\langle\sigma_i| - \pi_i,\\ &3)\ \pi_{i+1} = (1 - a_i)\pi_i + a_i|\sigma_i\rangle\langle\sigma_i|. \end{aligned} \tag{63}$$

where the constant β_C and norm $\|\cdot\|_C$ depend on C (ref. [54]).

In spite of the faster convergence rate, conjugate gradient methods require smooth cost functions (so that the gradient ∇C is well defined at every point). However, cost functions based on trace distance (7) are not smooth. For instance, the trace distance in one-dimensional spaces reduces to the absolute value function $|x|$ that is non-analytic at $x = 0$. When some eigenvalues are close to zero, conjugate gradient methods may display unexpected behaviors, though we have numerically observed that convergence is always obtained with a careful choice of the learning rate. In the next section, we show how to formally justify the applicability of the conjugate gradient method, following Nesterov's smoothing prescription[41].

Smoothing: smooth trace distance

The conjugate gradient method converges to the global optimum after $\mathcal{O}(\frac{L}{\epsilon})$ steps, provided that the gradient of C is L-Lipschitz continuous[41]. However, the constant L can diverge for non-smooth functions like the trace distance (7) so the convergence of the algorithm cannot be formally stated, although it may still be observed in numerical simulations. To solidify the convergence proof (see also Supplementary Note 5.2), we introduce a smooth approximation to the trace distance. This is defined by the following cost function that is differentiable at every point

$$C_\mu(\pi) = \mathrm{Tr}\left[h_\mu(\chi_\pi - \chi_\mathcal{E})\right] = \sum_j h_\mu(\lambda_j), \tag{64}$$

where λ_j are the eigenvalues of $\chi_\pi - \chi_\mathcal{E}$ and h_μ is the so-called Huber

penalty function

$$h_\mu(x) := \begin{cases} \frac{x^2}{2\mu} & \text{if } |x| < \mu\,, \\ |x| - \frac{\mu}{2} & \text{if } |x| \le \mu\,. \end{cases} \tag{65}$$

The previous definition of the trace distance, C_1 in Eq. (7), is recovered for $\mu \to 0$ and, for any non-zero μ, the C_μ bounds C_1 as follows

$$C_\mu(\pi) \le C_1(\pi) \le C_\mu(\pi) + \frac{\mu d}{2}, \tag{66}$$

where d is the dimension of the program state π. In Supplementary Note 5.2, we then prove the following result

Theorem 4 *The smooth cost function $C_\mu(\pi)$ is a convex function over program states and its gradient is given by*

$$\nabla C_\mu(\pi) = \Lambda^*[h'_\mu(\chi_\pi - \chi_\mathcal{E})], \tag{67}$$

where h'_μ is the derivative of h_μ. Moreover, the gradient is L-Lipschitz continuous with

$$L = \frac{d}{\mu}\,, \tag{68}$$

where d is the dimension of the program state.

Being Lipschitz continuous, the conjugate gradient algorithm and its variants[41,54] converge up to an accuracy ϵ after $\mathcal{O}(L/\epsilon)$ steps. In some applications, it is desirable to analyze the convergence in trace distance in the limit of large program states, namely for $d \to \infty$. The parameter μ can be chosen such that the smooth trace distance converges to the trace distance, namely $C_\mu \to C_1$ for $d \to \infty$. Indeed, given the inequality (66), a possibility is to set $\mu = \mathcal{O}(d^{-(1+\eta)})$ for some $\eta > 0$ so that, from Eq. (68), the convergence to the trace norm is achieved after $\mathcal{O}(d^{2+\eta})$ steps.

DATA AVAILABILITY

The datasets generated and analyzed during the current study are available from the corresponding author on reasonable request.

CODE AVAILABILITY

The codes used for this study are available from the corresponding author on reasonable request.

Received: 21 November 2019; Accepted: 25 March 2020;
Published online: 19 May 2020

REFERENCES

1. Nielsen, M. A. & Chuang, I. L. Programmable quantum gate arrays. *Phys. Rev. Lett.* **79**, 321 (1997).
2. Nielsen, M. A. & Chuang, I. L. *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2000).
3. Watrous, J. *The Theory of Quantum Information* (Cambridge Univ. Press, 2018).
4. Knill, E., Laflamme, R. & Milburn, G. J. A scheme for efficient quantum computation with linear optics. *Nature* **409**, 46 (2001).
5. Bishop, C. M. *Pattern Recognition and Machine Learning* (Springer, 2006).
6. Wittek, P. *Quantum Machine Learning: What Quantum Computing Means to Data Mining* (Academic Press, Elsevier, 2014).
7. Biamonte, J. et al. Quantum machine learning. *Nature* **549**, 195 (2017).
8. Dunjko, V. & Briegel, H. J. Machine learning & artificial intelligence in the quantum domain: a review of recent progress. *Rep. Prog. Phys.* **81**, 074001 (2018).
9. Schuld, M., Sinayskiy, I. & Petruccione, F. An introduction to quantum machine learning. *Contemp. Phys.* **56**, 172–185 (2015).
10. Ciliberto, C. et al. Quantum machine learning: a classical perspective. *Proc. R. Soc. A* **474**, 20170551 (2018).
11. Tang, E. A quantum-inspired classical algorithm for recommendation systems. In Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC 2019). Association for Computing Machinery, New York, NY, USA, 217–228. https://doi.org/10.1145/3313276.3316310 (2019).
12. Tang, E. Quantum-inspired classical algorithms for principal component analysis and supervised clustering. Preprint at https://arxiv.org/abs/1811.00414 (2018).
13. Kitaev, A. Y., Shen, A. & Vyalyi, M. N. *Classical and Quantum Computation*, 47 (American Mathematical Society, Providence, Rhode Island, 2002).
14. Boyd, S., Xiao, L. & Mutapcic, A. "Subgradient methods." lecture notes of EE392o, Stanford University, Autumn Quarter 2004 (2003):2004–2005.
15. Jaggi, M. Convex optimization without projection steps. Preprint at https://arxiv.org/abs/1108.1170 (2011).
16. Jaggi, M. Revisiting frank-wolfe: projection-free sparse convex optimization. In *Proceedings of the 30th International Conference on International Conference on Machine Learning*, Vol 28 I–427(2013).
17. Duchi, J., Shalev-Shwartz, S., Singer Y. & Chandra, T. Efficient projections onto the l 1-ball for learning in high dimensions. In *Proceedings of the 25th international conference on Machine learning*, 272–279 (ACM, 2008).
18. Liu, J., Musialski, P., Wonka, P. & Ye, J. Tensor completion for estimating missing values in visual data. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**, 208–220 (2013).
19. Ishizaka, S. & Hiroshima, T. Asymptotic teleportation scheme as a universal programmable quantum processor. *Phys. Rev. Lett.* **101**, 240501 (2008).
20. Ishizaka, S. & Hiroshima, T. Quantum teleportation scheme by selecting one of multiple output ports. *Phys. Rev. A* **79**, 042306 (2009).
21. Ishizaka, S. Some remarks on port-based teleportation. Preprint at https://arxiv.org/abs/1506.01555 (2015).
22. Lloyd, S. Universal quantum simulators. *Science* **273**, 1073–1078 (1996).
23. Pirandola, S., Laurenza, R., Ottaviani, C. & Banchi, L. Fundamental limits of repeaterless quantum communications. *Nat. Commun.* **8**, 15043 (2017).
24. Pirandola, S. et al. Theory of channel simulation and bounds for private communication. *Quant. Sci. Tech* **3**, 035009 (2018).
25. Nechita, I., Puchała, Z., Pawela, Ł. & Życzkowski, K. Almost all quantum channels are equidistant. *J. Math. Phys.* **59**, 052201 (2018).
26. Fuchs, C. A. & van de Graaf, J. Cryptographic distinguishability measures for quantum-mechanical states. *IEEE Trans. Info. Theory* **45**, 1216–1227 (1999).
27. Pinsker, M. S. *Information and information stability of random variables and processes* (Holden-Day, San Francisco, 1964).
28. Carlen, E. A. & Lieb, E. H. Bounds for entanglement via an extension of strong subadditivity of entropy. *Lett. Math. Phys.* **101**, 1–11 (2012).
29. Watrous, J. Semidefinite programs for completely bounded norms. *Theory Comput.* **5**, 217–238 (2009).
30. Watrous, J. Simpler semidefinite programs for completely bounded norms. *Chicago J. Theor. Comput. Sci.* **8**, 1–19 (2013).
31. Vandenberghe, L. & Boyd, S. Semidefinite programming. *SIAM Rev.* **38**, 49–95 (1996).
32. Chao, H.-H. *First-Order Methods for Trace Norm Minimization* (University of California, Los Angeles, 2013).
33. Monteiro, R. D. C. First-and second-order methods for semidefinite programming. *Math. Program.* **97**, 209–244 (2003).
34. Spall, J. C. Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE Trans. Automat. Contr.* **45**, 1839–1853 (2000).
35. Zhuang, Q. & Zhang, Z. Physical-layer supervised learning assisted by an entangled sensor network. *Phys. Rev. X* **9**, 041023 (2019).
36. Harrow, A. & Napp, J. Low-depth gradient measurements can improve convergence in variational hybrid quantum-classical algorithms. Preprint at https://arxiv.org/abs/1901.05374 (2019).
37. Cai, J.-F., Candès, E. J. & Shen, Z. A singular value thresholding algorithm for matrix completion. *SIAM J. Optimiz.* **20**, 1956–1982 (2010).
38. Recht, B., Fazel, M. & Parrilo, P. A. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Rev.* **52**, 471–501 (2010).
39. Nesterov, Y. *Introductory Lectures on Convex Optimization: A Basic Course*, Vol 87 (Springer Science & Business Media, New York, 2013).
40. Coutts, B., Girard, M. & Watrous, J. Certifying optimality for convex quantum channel optimization problems. Preprint at https://arxiv.org/abs/1810.13295 (2018).
41. Nesterov, Y. Smooth minimization of non-smooth functions. *Math. Program.* **103**, 127–152 (2005).
42. Khaneja, N., Reiss, T., Kehlet, C., Schulte-Herbrüggen, T. & Glaser, S. J. Optimal control of coupled spin dynamics: design of nmr pulse sequences by gradient ascent algorithms. *J. Magn. Reson.* **172**, 296–305 (2005).
43. Banchi, L., Pancotti, N. & Bose, S. Quantum gate learning in qubit networks: toffoli gate without time-dependent control. *npj Quantum Info.* **2**, 16019 (2016).
44. Innocenti, L. Banchi, L. Ferraro, A. Bose S. & M. Paternostro, M. Supervised learning of time-independent hamiltonians for gate design. *New J. Phys*. (in press) https://doi.org/10.1088/1367-2630/ab8aaf (2020).
45. Mitarai, K., Negoro, M., Kitagawa, M. & Fujii, K. Quantum circuit learning. *Phys. Rev. A* **98**, 032309 (2018).
46. Bennett, C. H. et al. Teleporting an unknown quantum state via dual classical and einstein-podolsky-rosen channels. *Phys. Rev. Lett.* **70**, 1895 (1993).
47. Pirandola, S., Eisert, J., Weedbrook, C., Furusawa, A. & Braunstein, S. L. Advances in quantum teleportation. *Nat. Photon.* **9**, 641–652 (2015).
48. Pirandola, S., Laurenza, R., Lupo, C. & Pereira, J. L. Fundamental limits to quantum channel discrimination. *npj Quantum Info* **5**, 50 (2019).
49. Lloyd, S. Almost any quantum logic gate is universal. *Phys. Rev. Lett.* **75**, 346 (1995).

50. D'Ariano, G. M. & Perinotti, P. Efficient universal programmable quantum measurements. *Phys. Rev. Lett.* **94**, 090401 (2005).
51. Pirandola, S., Bardhan, B. R., Gehring, T., Weedbrook, C. & Lloyd, S. Advances in photonic quantum sensing. *Nat. Photon* **12**, 724–733 (2018).
52. Uhlmann, A. The transition probability. *Rep. Math. Phys.* **9**, 273–279 (1976).
53. Duchi, J., Hazan, E. & Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.* **12**, 2121–2159 (2011).
54. Garber, D. & Hazan, E. Faster rates for the frank-wolfe method over strongly-convex sets. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, Vol 37, 541–549 (2015).
55. Bhatia, R. *Matrix Analysis*, Vol 169 (Springer Science & Business Media, New York, 2013).

ACKNOWLEDGEMENTS

L.B. acknowledges support by the program "Rita Levi Montalcini" for young researchers. S.P. and J.P. acknowledge support by the EPSRC via the 'UK Quantum Communications Hub' (Grants EP/M013472/1 and EP/T001011/1), and S.P. acknowledges support by the European Union via the project 'Continuous Variable Quantum Communications' (CiViQ, no 820466).

AUTHOR CONTRIBUTIONS

All authors contributed to prove the main theoretical results and to the writing of the manuscript.

COMPETING INTERESTS

The authors declare no competing interests

ADDITIONAL INFORMATION

Supplementary information is available for this paper at https://doi.org/10.1038/s41534-020-0268-2.

Correspondence and requests for materials should be addressed to L.B.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.