End-to-End Quantum Algorithm for Topology Optimization in Structural Mechanics

Leonhard Hölscher[1,2,3,*] Oliver Ahrend[4], Lukas Karch[1], Carlotta L'Estocq[1], Marc Marfany Andreu[1], Tobias Stollenwerk[3], Frank K. Wilhelm[2,3], and Julia Kowalski[4]

[1] *BMW Group, 80809 Munich, Germany*

[2] *Theoretical Physics, Saarland University, 66123 Saarbrücken, Germany*

[3] *Institute for Quantum Computing Analytics (PGI 12), Forschungszentrum Jülich, 52425 Jülich, Germany and*

[4] *Chair of Methods for Model-based Development in Computational Engineering, RWTH Aachen University, 52062 Aachen, Germany*

(Dated: November 18, 2025)

Topology optimization is a key methodology in engineering design for finding efficient and robust structures. Due to the enormous size of the design space, evaluating all possible configurations is typically infeasible. In this work, we present an end-to-end, fault-tolerant quantum algorithm for topology optimization that operates on an exponentially large Hilbert space representing the design space. We demonstrate the algorithm on the two-dimensional Messerschmitt-Bölkow-Blohm (MBB) beam problem. By restricting design variables to binary values, we reformulate the compliance minimization task as a combinatorial satisfiability problem, solved using Grover's algorithm. Within Grover's oracle, the compliance is computed through the finite-element method (FEM) using established quantum algorithms, including block-encoding of the stiffness matrix, Quantum Singular Value Transformation (QSVT) for matrix inversion, Hadamard test, and Quantum Amplitude Estimation (QAE). The complete algorithm is implemented and validated using classical quantum-circuit simulations. A detailed complexity analysis shows that the method evaluates the compliance of exponentially many structures in quantum superposition in polynomial time. In the global search, our approach maintains Grover's quadratic speedup compared to classical unstructured search. Overall, the proposed quantum workflow demonstrates how quantum algorithms can advance the field of computational science and engineering.

I. INTRODUCTION

Topology optimization (TO) is a computational design methodology that determines the optimal material distribution within a design space to maximize structural performance of the system under given load and boundary conditions and possibly further constraints [1, 2]. Structural performance can be measured in various ways and depends on the exact task, for example structural stiffness against deformation, minimizing mass or material volume, or excited and avoided eigenfrequencies. By integrating simulation-based performance and optimization, TO enables the creation of lightweight and robust structures across engineering applications, including aerospace, automotive, and additive manufacturing.

In its standard formulation, TO discretizes the design space into a so called conforming mesh, i.e. the cells are non-overlapping and there are no gaps between adjacent cells. The objective is then to find the optimal configuration of cells that should contain material. This optimization problem is typically formalized in either of two ways. The first using a binary decision variable for each cell that denotes whether the cell contains material or not. This might also be extended to multiple materials, where the binary variable is replaced by an integer variable, denoting the material or void. The second approach uses a continuous variable to represent the extent to which the cell should be filled, sometimes referred to as density - not to be confused with the physical density of the actual material. Then, the Finite-Element-Method (FEM) is employed to evaluate the system performance of a candidate material distribution. Without loss of generality, TO can also be applied to other fields, for example fluid mechanics and electromagnetism, using other modeling frameworks, for example Computational-Fluid-Dynamics (CFD) or laboratory experiments. As our use-case originates from structural mechanics, we restrict our paper to FEM-based TO. Sigmund & Maute [2] give a review of the foundations of TO, and discuss different methods and implications.

Despite its widespread success, TO remains computationally demanding. Each candidate design's structural performance is evaluated by solving a large system of linear equations, and the number of possible candidates grows exponentially with the number of cells of the dicretized design space. To manage this complexity, most classical implementations make use of the continuous approach using cell densities between zero and one. This allows the use of gradient-based solvers but introduces intermediate, non-interpretable material states, where the cell densities must later be rounded to zero or one, or penalized right away during the optimization [3, 4]. This is because a cell being partially filled with material imposes another TO problem to find the respective material distribution within the

* leo.h@mail.de

cell. In contrast, the binary formulation preserves physical interpretability but turns the optimization into an exponentially hard combinatorial problem. Evaluating vast numbers of structural configurations, each involving a costly FEM simulation, represents a central computational bottleneck.

Quantum computing offers a fundamentally different route to overcome this challenge. By encoding each candidate structure as a quantum state, a quantum computer can exploit an exponentially large Hilbert space to process all possible structures in superposition. Grover's algorithm [5], for example, naturally exploits this exponential space for global search, enabling a quadratic speedup in identifying optimal or near-optimal configurations. At the same time, repeatedly solving large systems of linear equations arising from the FEM analysis is another major computational bottleneck in TO. Here, quantum linear system algorithms, such as the Harrow-Hassidim-Lloyd (HHL) algorithm [6], the Quantum Singular Value Transform (QSVT) [7, 8], and others [9–11] have the potential to achieve exponential speedups. Combining the combinatorial search over candidate structures and a simulation-based performance assessment of the underlying physical system could benefit from substantial quantum acceleration. However, the central challenge is to integrate these components into a coherent quantum workflow without introducing (classical) overheads that would eliminate any achievable speedup [12], for example, state preparation of large matrices or read-outs of displacement vectors.

Recent studies have started to investigate the integration of quantum algorithms with TO. Sato et al. [13] successfully demonstrated the application of Variational-Quantum-Algorithms (VQAs) to solve the binary TO setting. The variational approach shifts the complexity of solving a combinatorial problem to a continuous problem in the parameters of quantum gates, i.e. rotation angles. While this approach can be tuned to provably provide the best solution of the initial combinatorial problem [14], in practice, the continuous optimization problem is often still intractable due to its highly nonlinear nature. Specialized classical optimizers are being developed to tackle this problem [15], but an overall quantum advantage remains questionable. Ye and Pan [16] reformulated TO with multiple materials as a mixed-integer linear program solved via a Modified Dantzig-Wolfe decomposition [17], where local binary sub-problems are expressed as Quadratic-Unconstrained-Binary-Optimization problems (QUBOs) for quantum annealing. Sukulthanasorn et al. [18] propose to reduce the number of equation systems required to solve by leveraging an updater, which is tasked to find the optimal distribution of material with as few evaluations of the system performance as possible. The updater is formulated as a QUBO problem, which is then solved by quantum annealing. Beyond quantum approaches specifically targeting TO, Stein et al. [19] proposed Quantum Simulation-based Optimization (QuSO), embedding numerical simulations or the corresponding system of linear equations within the Quantum Approximate Optimization Algorithm (QAOA). The QuSO algorithm has been successfully applied to optimize power grids [20] and automotive cooling systems [21], highlighting the potential and limitations of this framework. However, the variational nature of QAOA does not allow us to give a general statement about any quantum advantage.

In this work, we present a fault-tolerant quantum algorithm for TO that combines FEM analysis and global optimization within a single coherent algorithm. By fault-tolerant, we refer to the class of quantum algorithms meant to be executed on error-corrected quantum computers. Using the well-known bending beam problem, also called the Messerschmitt-Bölkow-Blohm (MBB) beam problem [2–4], as a demonstrator, we reformulate TO as a quantum search over binary material configurations. Each candidate design is represented as a quantum state, and Grover's algorithm is employed to identify optimal configurations. Within Grover's oracle, the structural performance is quantified through the compliance, which is computed using a block-encoded stiffness matrix, QSVT for matrix inversion, the Hadamard test, and Quantum Amplitude Estimation (QAE). In contrast to previous quantum approaches to TO that rely on quantum annealing or VQAs, our method is composed entirely of fault-tolerant quantum subroutines. This allows us to derive a full algorithmic complexity analysis demonstrating that the overall workflow maintains Grover's quadratic speedup compared to classical unstructured search.

The remainder of this paper is organized as follows. Section II reviews the classical TO framework and its FEM formulation, including the discretization of the design space and the binary and density-based representation of the material distribution. This motivates the binary formulation adopted for our quantum implementation. Section III introduces the proposed quantum algorithm and explains each component of the workflow, from problem definition to circuit-level realization. Section IV presents numerical experiments that demonstrate the algorithm's functionality. Section V discusses the implications, limitations, and possible extensions of the approach. Finally, Section VI summarizes the main findings and outlines directions for future research.

II. TOPOLOGY OPTIMIZATION

The objective in this TO is to design structures that deform as little as possible under given supports and loads. A common performance measure is the compliance [1, 2]

$$c(\rho) = \int_\Omega \mathbf{u}(\rho, \mathbf{r}) \cdot \boldsymbol{f}(\mathbf{r})\, \mathrm{d}\mathbf{r}, \tag{1}$$

where $\rho(\mathbf{r}) \in [0,1]$ denotes the relative density of material, as a function of spatial coordinates $\mathbf{r}$, $\mathbf{u}(\rho, \mathbf{r})$ the displacement field, and $\boldsymbol{f}(\mathbf{r})$ the external load over the domain Ω. Eq. (1) shows, that the compliance measures the deformation in the same direction as the load. Deformation orthogonal to the load does not contribute to the compliance. As a sidenote, minimizing compliance is the same as maximizing stiffness. To avoid the trivial all-solid design, one typically constrains the total amount of material. We want to emphasize that relative density ρ is independent of the material density and is introduced only for the definition of the minimization problem. Therefore, without loss of generality ρ may be dimensionless, and the integral

$$V(\rho) = \int_\Omega \rho \, \mathrm{d}\mathbf{r} \Big/ \int_\Omega 1 \, \mathrm{d}\mathbf{r} \tag{2}$$

is interpreted as the fraction of volume containing material. The continuous optimization problem then reads

$$\begin{aligned} &\min_\rho \; c(\rho), \\ &\text{subject to } V(\rho) = V_0, \end{aligned} \tag{3}$$

where $V_0 \in (0,1)$ specifies the target volume fraction.

A. Finite Element Discretization

In this paper, we consider the 2D MBB beam problem, which is a common benchmark in TO [2–4]. We discretize a two-dimensional beam into $n_x \times n_y$ identical cells, where n_x and n_y is the number of cells in horizontal x- and vertical y-direction, respectively. For the sake of simplicity, each cell is then used as an element for the FEM. This is not a requirement of the TO, and one could remesh the cells for an arbitrary FEM discretization. However, maintaining a fully structured, equidistant, FEM mesh significantly simplifies the matrix encoding in the quantum algorithm. Therefore, we use bilinear quadrilateral elements that have four nodes each with two translational degrees of freedom (DoFs) per node. Fig. 1 illustrates the FEM discretization. Thus, the element force vector $\mathbf{f}^{\mathrm{el}}$ and the resulting element displacement vector $\mathbf{u}^{\mathrm{el}}$ have eight entries, ordered according to Fig. 1. In the linear elasticity regime, force and displacement are related linearly by

$$\mathbf{K}^{\mathrm{el}} \mathbf{u}^{\mathrm{el}} = \mathbf{f}^{\mathrm{el}}, \tag{4}$$

where $\mathbf{K}^{\mathrm{el}}$ is the 8×8 element stiffness matrix. Entry $\mathbf{K}^{\mathrm{el}}_{i,j}$ relates the displacement of DoF j ($\mathbf{u}^{\mathrm{el}}_j$) to the force acting in direction of DoF i ($\mathbf{f}^{\mathrm{el}}_i$), and physically has units of force per length. More details on $\mathbf{K}^{\mathrm{el}}$ are given in the Appendix A.

The domain is assembled by stitching the $n_{\mathrm{el}} = n_x n_y$ elements together. Since neighboring elements share two nodes, we need to introduce a global node ordering.

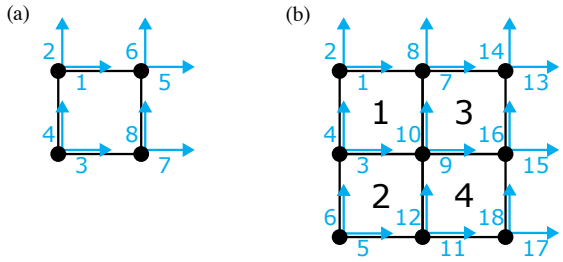

FIG. 1. FEM domain discretization using quadrilateral elements. (a) shows a single element with four nodes, each having a horizontal and a vertical degree of freedom (DoF), and (b) depicts a domain consisting of 2×2 elements. Either way, we order the DoFs by counting column-wise from top to bottom from left to right. Horizontal DoFs have precedence over vertical DoFs.

Therefore, we count column-wise from top to bottom, then left to right, as shown in Fig. 1(b). This gives us the global displacement and force vector $\mathbf{u}, \mathbf{f} \in \mathbb{R}^{n_{\mathrm{DoF}}}$ with the total number of DoFs

$$n_{\mathrm{DoF}} = 2(n_x + 1)(n_y + 1). \tag{5}$$

Similarly to Eq. (4), we have a global linear system of equations

$$\mathbf{K}\mathbf{u} = \mathbf{f}, \tag{6}$$

where the global stiffness matrix $\mathbf{K}$ is obtained by mapping element stiffness matrices into the global DoF space and summing all element contributions. This is also referred to as assembling the global stiffness matrix. In the context of TO, stiffness matrices depend on the material distribution, introduced as ρ above, as outlined in the following.

B. Density-Based Formulation

In common density-based approaches, such as the Solid Isotropic Material with Penalization (SIMP) method [4], the material distribution is represented by $\mathbf{x} = (x_1, \ldots, x_{n_{\mathrm{el}}})^\top$ with $x_e \in [0,1]$ denoting the relative density of element e. Here and in the remainder, $\mathbf{x}$ is the discretized version of ρ. The global stiffness matrix then reads

$$\mathbf{K}(\mathbf{x}) = \sum_{e=1}^{n_{\mathrm{el}}} x_e \, \tilde{\mathbf{K}}^{\mathrm{el}}(e), \tag{7}$$

where $\tilde{\mathbf{K}}^{\mathrm{el}}(e)$ denotes the element matrix embedded at the appropriate global rows and columns. Fig. 11 visualizes this global matrix assembly. Supports are enforced by removing the corresponding rows and columns from $\mathbf{K}$, thereby imposing zero displacement on the respective DoF.

The discretized compliance corresponding to Eq. (1) is given by

$$c(\mathbf{x}) = \mathbf{u}^\top \mathbf{f} = \mathbf{u}^\top \mathbf{K}(\mathbf{x})\mathbf{u} \tag{8}$$
$$= \mathbf{f}^\top \mathbf{K}^{-1}(\mathbf{x})\mathbf{f} \quad . \tag{9}$$

While the rightmost form in Eq. (8) is often used in literature [3, 4], we use the equivalent form in Eq. (9), as it is easier to implement later on. Again without loss of generality, this assumes symmetric matrix $\mathbf{K}$. The material volume fraction becomes

$$V(\mathbf{x}) = \frac{1}{n_{\text{el}}} \sum_{e=1}^{n_{\text{el}}} x_e. \tag{10}$$

The discrete optimization problem is therefore

$$\begin{aligned} &\min_{\mathbf{x}} c(\mathbf{x}), \\ &\text{subject to } V(\mathbf{x}) = V_0. \end{aligned} \tag{11}$$

Fig. 2 shows a solution of the MBB problem obtained with the SIMP method, which resembles the expected truss structure. Here, the volume fraction was set to $V_0 = 1/2$, meaning half of the design domain is filled with material.

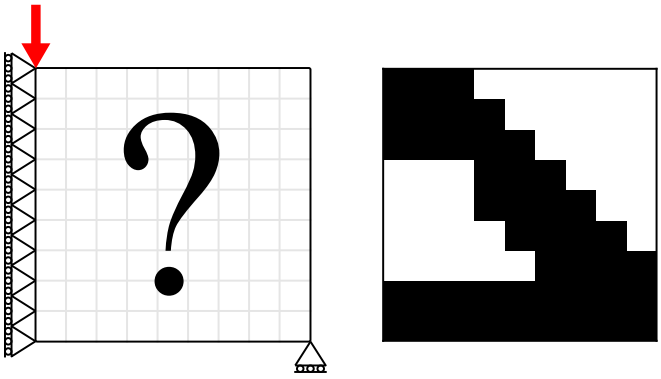

FIG. 2. Example solution for an MBB beam problem. The imposed boundary conditions and the square domain are displayed on the left. The horizontal DoFs of all nodes on the left boundary are fixed as well as the vertical DoF at the bottom-right corner. A single force pointing downwards is applied at the top-left corner. The right image shows a rounded optimized structure for 9×9 elements and a volume fraction of $V_0 = 1/2$ using the SIMP method outlined in Ref. [4].

III. QUANTUM ALGORITHM

In the following, we introduce our fault-tolerant quantum algorithm for TO of the MBB beam benchmark problem.

A. Quantum Topology Optimization Problem

To solve the MBB problem on a quantum computer, we adapt the density-based TO problem in Eq. (11). We restrict the design variables to binary values,

$$x_e \in \{0, 1\}, \qquad x_e = 1 \text{ solid}, \quad x_e = 0 \text{ void}. \tag{12}$$

This enables a natural representation of candidate solutions as computational basis states $|x_1 x_2 \dots x_{n_{\text{el}}}\rangle$ and is known as TO of binary structures [22]. In addition, we reformulate the problem as a satisfiability problem:

$$\text{Find } \mathbf{x} \in \{0,1\}^{n_{\text{el}}} \quad \text{such that} \quad \begin{cases} c(\mathbf{x}) < c_0, \\ V(\mathbf{x}) = V_0. \end{cases} \tag{13}$$

This makes it native to our quantum approach. By iteratively lowering the threshold value c_0, we can find the material distribution that minimizes compliance [23]. However, we want to mention that comparing multiple candidate solutions allows to consider additional latent constraints, for example from the perspective of manufacturing.

B. Grover's Algorithm

Grover's algorithm is a well-known quantum algorithm for unstructured search [5, 24]. With high probability, it returns one of M marked items in an unstructured search space of size N using $\mathcal{O}(\sqrt{N/M})$ oracle calls. Since the classical counterpart would require $\mathcal{O}(N/M)$ evaluations, Grover's algorithm gives a quadratic speedup. We use the search algorithm to solve the satisfiability problem in Eq. (13) and thus to find stiff structure designs $\mathbf{x}$.

The quantum circuit is shown in Fig. 3. We have

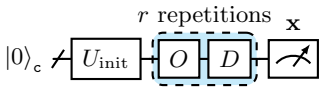

FIG. 3. Quantum circuit for Grover's algorithm. U_{init} is used to prepare the superposition of the search space. The following r iterations apply Oracle O and Diffusion D repeatedly, before finally the mesurement is conducted.

a register $\mathtt{c}$ containing $n_{\mathtt{c}} = n_{\text{el}}$ qubits to represent structure configurations. The initialization U_{init} prepares a superposition over the search space. If we search in the full space of size $N = 2^{n_{\text{el}}}$, this operation simply becomes $U_{\text{init}} = H^{\otimes n_{\text{el}}}$. However, to enforce the volume constraint $V(\mathbf{x}) = V_0$, we instead prepare the Dicke state [25]

$$U_{\text{init}} |0\rangle_{\mathtt{c}} = \left| D_k^{(n_{\text{el}})} \right\rangle_{\mathtt{c}} = \frac{1}{\sqrt{N}} \sum_{|\mathbf{x}|=k} |\mathbf{x}\rangle_{\mathtt{c}}\,, \tag{14}$$

which is a superposition over the $N = \binom{n_{\text{el}}}{k}$ states with Hamming weight $k = \lfloor n_{\text{el}} V_0 \rfloor$.

After the initialization, we have r repetitions of Grover's oracle O and Grover's diffusion operator D. O marks every solution state as

$$O|\mathbf{x}\rangle = \begin{cases} -|\mathbf{x}\rangle & \text{for a solution } \mathbf{x}, \\ |\mathbf{x}\rangle & \text{for any other } \mathbf{x}. \end{cases} \tag{15}$$

Grover's diffusion operator is given by

$$D = -U_{\text{init}} S_0 U_{\text{init}}^\dagger \quad \text{with} \quad S_0 = \mathbb{1} - 2|0\rangle\langle 0|. \tag{16}$$

Fig. 4 gives a geometric understanding of Grover's algorithm. We begin in the initial state $U_{\text{init}}|0\rangle = |\psi_{\text{init}}\rangle$.

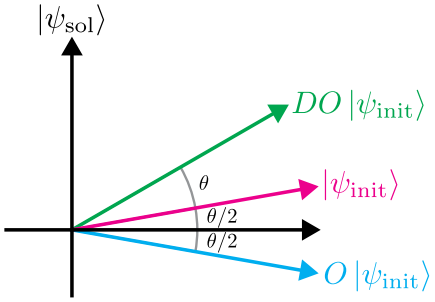

FIG. 4. Geometrical illustration of a single Grover iteration.

Grover's oracle reflects that state about the state orthogonal to our solution state $|\psi_{\text{sol}}\rangle$. The diffusion operator is another reflection about $|\psi_{\text{init}}\rangle$. By repeating these two operators, the state is rotated by θ toward the solution state. The angle θ is given by the expression

$$\sin\left(\frac{\theta}{2}\right) = \langle\psi_{\text{sol}}|\psi_{\text{init}}\rangle = \sqrt{\frac{M}{N}}, \tag{17}$$

and the number of repetitions r should be chosen to maximize the overlap

$$\langle\psi_{\text{sol}}|\,(DO)^r\,|\psi_{\text{init}}\rangle = \sin\left(\left(r+\frac{1}{2}\right)\theta\right). \tag{18}$$

Consequently,

$$r = \left\lfloor \frac{\pi}{4\arcsin\sqrt{M/N}} - \frac{1}{2} \right\rfloor \overset{M\ll N}{\approx} \left\lfloor \frac{\pi}{4}\sqrt{\frac{N}{M}} - \frac{1}{2} \right\rfloor. \tag{19}$$

C. Grover's Oracle

Previously, Grover's oracle O was introduced as marking solution states as shown in Eq. (15). In order to solve the satisfiability problem in Eq. (13), the oracle should mark states where $c(\mathbf{x}) < c_0$. When the configuration register is initialized as a Dicke state, the volume constraint $V(\mathbf{x}) = V_0$ is already enforced and need not be checked by the oracle.

We implement O as depicted in Fig. 5. Here, the

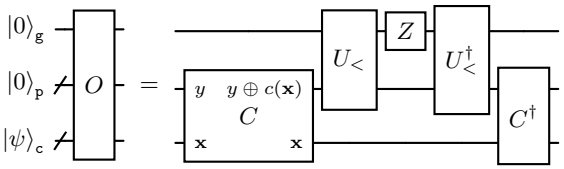

FIG. 5. Quantum circuit for Grover's Oracle operation O.

subroutine C writes the compliance into an ancilla register $\mathtt{p}$ as

$$C|0\rangle_{\mathtt{p}}|\mathbf{x}\rangle_{\mathtt{c}} = |c(\mathbf{x})\rangle_{\mathtt{p}}|\mathbf{x}\rangle_{\mathtt{c}}. \tag{20}$$

Thereafter, the operation $U_<$ flips the ancilla qubit $\mathtt{g}$ if $c(\mathbf{x}) < c_0$. This is followed by a Z gate on register $\mathtt{g}$ producing the phase flip of Eq. (15), and we uncompute previous operations.

D. Compliance Computation

Next, we dive into our implementation of C computing the compliance as defined in Eq. (20). By recalling Eq. (9), we can express $c(\mathbf{x})$ as

$$\tfrac{1}{\alpha}\,c(\mathbf{x}) = \langle\mathbf{x}|_{\mathtt{c}}\,\langle\mathbf{f}|_{\mathtt{d}}\,\langle 0|_{\mathtt{qlvzb}}\,U_{\mathbf{K}^{-1}}\,|0\rangle_{\mathtt{qlvzb}}\,|\mathbf{f}\rangle_{\mathtt{d}}\,|\mathbf{x}\rangle_{\mathtt{c}} \tag{21}$$

up to a known normalization constant α as described later. Here, $U_{\mathbf{K}^{-1}}$ is a unitary operator providing the inverse global stiffness matrix $\mathbf{K}^{-1}$ as a block-encoding for the configuration $|\mathbf{x}\rangle_{\mathtt{c}}$. The registers $\mathtt{qlvzb}$ serve as ancillas for the block-encoding, and the Hilbert space of the data register $\mathtt{d}$ carries $\mathbf{f}$ and $\mathbf{K}$. Further details on the block-encoding follow in the next two sections.

To compute the expectation value in Eq. (21), we use a Hadamard test [26] A with an ancilla qubit $\mathtt{h}$ as shown in Fig. 6. The state preparation $V_{\mathbf{f}}|0\rangle_{\mathtt{d}} = |\mathbf{f}\rangle$ loads the

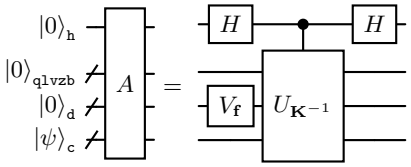

FIG. 6. Quantum circuit diagram of the Hadamard Test A.

normalized force vector. The action of A is

$$\begin{aligned} &A|0\rangle_{\mathtt{h}}|0\rangle_{\mathtt{qlvzb}}|0\rangle_{\mathtt{d}}|\mathbf{x}\rangle_{\mathtt{c}} \\ &= \frac{1}{2}\Big[|0\rangle_{\mathtt{h}}(\mathbb{1}+U_{\mathbf{K}^{-1}})|0\rangle_{\mathtt{qlvzb}}|\mathbf{f}\rangle_{\mathtt{d}}|\mathbf{x}\rangle_{\mathtt{c}} \\ &\quad + |1\rangle_{\mathtt{h}}(\mathbb{1}-U_{\mathbf{K}^{-1}})|0\rangle_{\mathtt{qlvzb}}|\mathbf{f}\rangle_{\mathtt{d}}|\mathbf{x}\rangle_{\mathtt{c}}\Big]. \end{aligned} \tag{22}$$

We embed A into a QAE [27] to estimate the absolute amplitude $|h_0|$ of $|0\rangle_{\mathtt{h}}$. The corresponding quantum circuit is given in Fig. 7. In general, QAE is a quantum phase estimation circuit [28] with the Grover

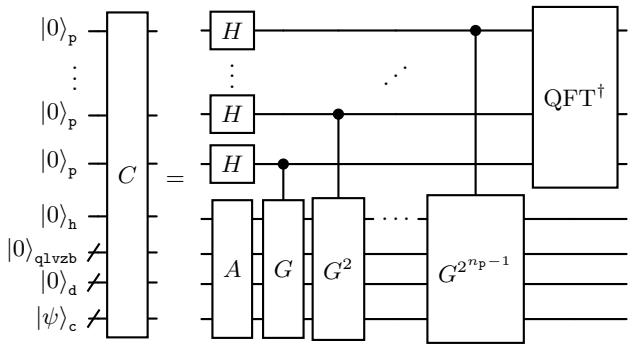

FIG. 7. Quantum circuit for Quantum Amplitude Estimation within the compliance computation operation C. The $\mathrm{QFT}^\dagger$ block represents an inverse Quantum Fourier Transform (QFT).

operator G as central unitary. The definition of G is shown in Fig. 8. Note that G is unrelated to Grover's algorithm discussed in Section III B. Consequently, the

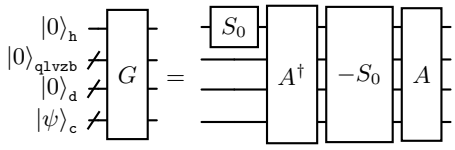

FIG. 8. Grover Operator G within the Quantum Amplitude Estimation circuit.

QAE routine estimates the absolute amplitude

$$\begin{aligned}|h_0| &= \sqrt{\tfrac{1}{2} + \tfrac{1}{2}\,\mathrm{Re}\Big\{\langle \mathbf{x}|_{\mathtt{c}}\,\langle \mathbf{f}|_{\mathtt{d}}\,\langle 0|_{\mathtt{qlvzb}}\; U_{\mathbf{K}^{-1}}\; |0\rangle_{\mathtt{qlvzb}}\,|\mathbf{f}\rangle_{\mathtt{d}}\,|\mathbf{x}\rangle_{\mathtt{c}}\Big\}} \\ &= \sqrt{\tfrac{1}{2} + \tfrac{1}{2\alpha}\,c(\mathbf{x})}\end{aligned} \tag{23}$$

and writes the phase

$$\theta(\mathbf{x}) = \pm\tfrac{1}{\pi}\arcsin\left(\sqrt{\tfrac{1}{2} + \tfrac{1}{2\alpha}c(\mathbf{x})}\right) \tag{24}$$

into the phase register $\mathtt{p}$. Thus, our implementation of C differs from the ideal definition in Eq. (20) and computes

$$C\,|0\rangle_{\mathtt{p}}\,|\mathbf{x}\rangle_{\mathtt{c}} = \frac{1}{\sqrt{2}}\Big(|\theta(\mathbf{x})\rangle_{\mathtt{p}} + |-\theta(\mathbf{x})\rangle_{\mathtt{p}}\Big)\,|\mathbf{x}\rangle_{\mathtt{c}}\,. \tag{25}$$

We consider this as a technical detail, since the $\pm$ branches of $\theta(c(\mathbf{x})) = \theta(\mathbf{x})$ are strictly increasing bijections of $c(\mathbf{x})$, and we equivalently compare $|\theta(\mathbf{x})| < \theta_0 = \arcsin\left(\sqrt{\frac{1}{2} + \frac{1}{2\alpha}c_0}\right)/\pi$ inside $U_<$.

E. Matrix Inversion

In this section, we explain how to implement the unitary $U_{\mathbf{K}^{-1}}$ block-encoding the inverse global stiffness matrix $\mathbf{K}^{-1}(\mathbf{x})$ for a configuration $\mathbf{x}$. We employ the QSVT algorithm [7, 8] to invert the global stiffness matrix $\mathbf{K}(\mathbf{x})$. Therefore, we assume access to the corresponding block-encoding $U_{\mathbf{K}}$ which is detailed in the next section. QSVT applies a polynomial transformation to the singular values of the block-encoded matrix without explicitly accessing the singular value decomposition. In our case, this matrix is $\mathbf{K}$ up to a scaling constant β. Hence,

$$\frac{1}{\beta}\mathbf{K} = \mathbf{V}\mathbf{\Sigma}\mathbf{W}^\dagger \overset{\mathrm{QSVT}}{\mapsto} \mathbf{V}P(\mathbf{\Sigma})\mathbf{W}^\dagger, \tag{26}$$

where $P(\cdot)$ is the QSVT polynomial, $\mathbf{\Sigma}$ contains the singular values $\sigma_i \in [0,1]$ in its diagonal, and $\mathbf{V}, \mathbf{W}$ are the unitaries of the singular value decomposition (SVD). Since $\mathbf{K} = \mathbf{K}^\top$, the SVD is equivalent to the eigenvalue decomposition and $\mathbf{V} = \mathbf{W}$, however we are not restricted to symmetric matrices in general. For matrix inversion, the QSVT polynomial approximates the reciprocal function $P_{\mathrm{inv}}(x) \approx \gamma/x$ on the interval $x \in [\mu, 1]$, where $\mathcal{O}(\gamma) = \mathcal{O}(\mu)$ ensures that the function values lie within $[-1, 1]$, and μ is the smallest relevant singular value. Consequently,

$$\frac{1}{\beta}\mathbf{K} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^\dagger \overset{\mathrm{QSVT}}{\mapsto} \mathbf{V}P_{\mathrm{inv}}(\mathbf{\Sigma})\mathbf{V}^\dagger \approx \gamma\beta\mathbf{K}^{-1}. \tag{27}$$

The QSVT algorithm is defined as an alternating sequence of the block-encoding $U_{\mathbf{K}}$ (or its conjugate transpose) and projector-controlled phase-shifts $\Pi_{\phi_k} = e^{i\phi_k(2\Pi - \mathbb{1})}$. Here, Π is the projector to the block within $U_{\mathbf{K}}$ corresponding to $\mathbf{K}$, and $\{\phi_k\}$ are phase angles. The QSVT theorem states that it is possible to find d angles $\{\phi_1, \phi_2, \ldots, \phi_d\}$ that implement any polynomial $P(\cdot)$ with $\deg(P) \le d$, parity $d \bmod 2$, and specific norm to preserve normalization. We discuss the choice of the polynomial approximation and the angle finding algorithms in Section III G. Consequently, we have

$$\begin{aligned}U_{\mathbf{K}^{-1}} &= \begin{cases} \Pi_{\phi_1}U_{\mathbf{K}}\prod_{k=1}^{(d-1)/2}\Pi_{\phi_{2k}}U_{\mathbf{K}}^\dagger\Pi_{\phi_{2k+1}}U_{\mathbf{K}}, & \text{for odd } d \\ \prod_{k=1}^{d/2}\Pi_{\phi_{2k-1}}U_{\mathbf{K}}^\dagger\Pi_{\phi_{2k}}U_{\mathbf{K}}, & \text{for even } d \end{cases} \\ &= \begin{pmatrix} \mathbf{V}P_{\mathrm{inv}}(\mathbf{\Sigma})\mathbf{V}^\dagger & * \\ * & * \end{pmatrix}.\end{aligned} \tag{28}$$

The corresponding quantum circuit looks like Fig. 9, where we use the ancilla qubit $\mathtt{q}$ to implement Π_{ϕ_k}, and $U_{\mathbf{K}}$ depends on the block-encoding ancillas $\mathtt{lvzb}$, the data register $\mathtt{d}$, and the configuration register $\mathtt{c}$.

To implement Π_{ϕ_k}, one typically uses the projector $\Pi = |0\rangle\langle 0|_{\mathtt{lvzb}} \otimes \mathbb{1}_{\mathtt{d}}$ pointing to the top-left block of the unitary. We modify the projector to remove the rows and columns corresponding to the fixed DoFs

$$\Pi = |0\rangle\langle 0|_{\mathtt{lvzb}} \otimes \left(\mathbb{1}_{\mathtt{d}} - \sum_{\text{fixed DoFs } i} |i\rangle\langle i|\right). \tag{29}$$

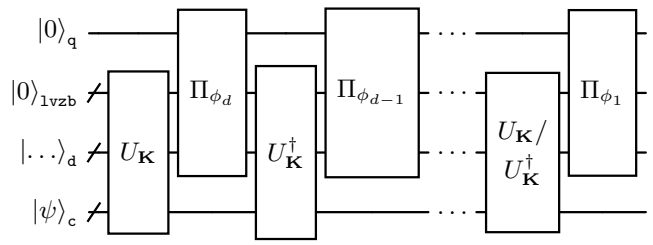

FIG. 9. Quantum circuit realizing a block-encoding of the inverse stiffness matrix $\mathbf{K}^{-1}$ via Quantum Singular Value Transform.

Fig. 10 shows the circuit implementation of Π_{ϕ_k} (up to a global phase) for the example 2×2 elements example, where we remove the DoFs with indices 1, 3, 4, and 18 to encode horizontal support on the left boundary and vertical support at the bottom-right corner corresponding to the example depicted in fig. 2.

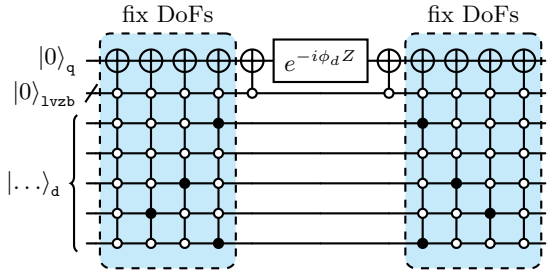

FIG. 10. Exemplary circuit implementation of the projector-controlled phase-shift Π_{ϕ_d}. The blue shaded operations remove the fixed DoFs with indices 1, 3, 4, and 18 (least significant bit on the bottom).

F. Matrix Block-Encoding

The QSVT circuit for matrix inversion is based on $U_{\mathbf{K}}$ block-encoding the global stiffness matrix $\mathbf{K}$ in the top-left block

$$U_{\mathbf{K}} = \frac{1}{\beta}\begin{pmatrix} \mathbf{K} & * \\ * & * \end{pmatrix}, \tag{30}$$

where β is a scaling constant ensuring $\|\mathbf{K}/\beta\| \leq 1$. Furthermore, $U_{\mathbf{K}}$ depends on the structure design $\mathbf{x}$ stored in the configuration register $|\mathbf{x}\rangle_{\mathtt{c}}$. From Eq. (7), we know that $\mathbf{K}$ is the sum of the element stiffness matrices $\mathbf{K}^{\mathrm{el}}$ embedded into the global DoF space. Fig. 11 illustrates this assembly for a $n_x = 3$, $n_y = 4$ domain. The matrix $\mathbf{K}$ is sparse, banded, and displays a repeating block pattern due to the uniform element topology and global DoF ordering. Each element e contributes a transformed element stiffness matrix $\tilde{\mathbf{K}}^{\mathrm{el}}(e)$ to $\mathbf{K}$. $\tilde{\mathbf{K}}^{\mathrm{el}}(e)$ is $\mathbf{K}^{\mathrm{el}}$ with an offset $\Delta(e)$ in both rows and columns and a gap of size $2(n_y - 1)$ splitting the 8×8 matrix into four

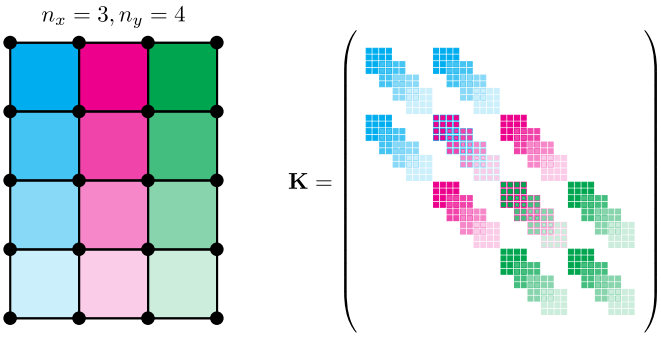

FIG. 11. Element-wise contributions to the global stiffness matrix for a 3×4 domain. The element's colors match their contribution.

4×4 blocks. This transformation is visualized by

$$\mathbf{K}^{\mathrm{el}} = \left(\begin{array}{c}\blacksquare\end{array}\right) \mapsto \tilde{\mathbf{K}}^{\mathrm{el}}(e) = \left(\begin{array}{c}\Delta(e)\\ \blacksquare \quad \blacksquare \\ 2(n_y-1) \\ \blacksquare \quad \blacksquare\end{array}\right), \tag{31}$$

where $\tilde{\mathbf{K}}^{\mathrm{el}}(e)$ has size $n_{\mathrm{DoF}} \times n_{\mathrm{DoF}}$. This transformation embeds each local matrix into the global domain, aligning the DoFs with the column-wise global ordering. The diagonal offset is

$$\Delta(x, y) = 2\left[(x-1)(n_y+1) + (y-1)\right] \tag{32}$$

with the element coordinates $1 \leq x \leq n_x$ and $1 \leq y \leq n_y$. If we count the elements column-wise, we have

$$e = (x-1)n_y + y \quad \text{with} \quad e = 1, \ldots, n_{\mathrm{el}} \tag{33}$$

and conversely

$$x = \left\lfloor \frac{e-1}{n_y} \right\rfloor + 1 \quad \text{and} \quad y = (e-1) \bmod n_y + 1. \tag{34}$$

Consequently, the offset for an element e is

$$\Delta(e) = 2\left(e - 1 + \left\lfloor \frac{e-1}{n_y} \right\rfloor\right). \tag{35}$$

Based on the matrix structure, we implement the block-encoding of $\mathbf{K}$ using the following six-step strategy.

1. *Block-encoding of the element stiffness matrix*

We begin by block-encoding the element stiffness matrix $\mathbf{K}^{\mathrm{el}}/\delta$, where δ is a scaling constant. Since $\mathbf{K}^{\mathrm{el}}$ has constant size, this block-encoding can be implemented

in $\mathcal{O}(1)$ steps using for example the Quantum Shannon Decomposition [29] to decompose

$$U_{\mathbf{K}^{\mathrm{el}}} = \frac{1}{\delta}\begin{pmatrix} \mathbf{K}^{\mathrm{el}} & \sqrt{\delta^2\mathbb{1} - \mathbf{K}^{\mathrm{el}^2}} \\ \sqrt{\delta^2\mathbb{1} - \mathbf{K}^{\mathrm{el}^2}} & \mathbf{K}^{\mathrm{el}} \end{pmatrix}, \tag{36}$$

into basis gates with block-encoding ancilla qubit `b`.

2. Zero-padding to global size

To embed $\mathbf{K}^{\mathrm{el}}$ into the global DoF space, we pad it with $n_{\mathrm{DoF}} - 8$ zeros. Therefore, we expand the data register `d` by

$$d_{\mathrm{z}} = \left\lceil \log_2 \frac{n_{\mathrm{DoF}}}{8} \right\rceil \tag{37}$$

qubits. To remove non-zero entries originating from implicit identity gates on these d_{z} qubits, an operation F flips an additional block-encoding ancilla qubit `z` whenever the added d_{z} qubits differ from $|0\cdots0\rangle$.

3. Gap insertion

Due to the column-wise ordering of global DoFs, a gap of size $2(n_y-1)$ must separate the left and right nodes of each element. We implement this by permuting matrix indices with a permutation unitary P_{gap} acting on the data register,

$$P_{\mathrm{gap}} = P_{+4}\left(P_{+2(n_y-1)}\ (\text{controlled on MSB})\right)P_{-4}, \tag{38}$$

where $P_{\pm k}$ adds/subtracts k to/from the binary index (modulo the register range) and are realized with standard reversible adders. Here, P_{-4} shifts the beginning of the gap to the zero-index. The gap is inserted via $P_{+2(n_y-1)}$, which is controlled on the most significant bit and acts on the remaining bits, thereby shifting only the first half of the matrix entries. P_{+4} is reversing the initial shift. Fig. 12 shows the quantum circuit for P_{gap}.

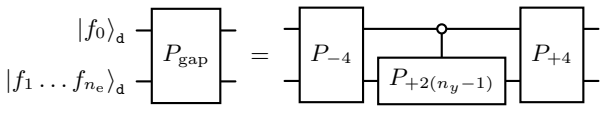

FIG. 12. Quantum circuit for the permutation operation P_{gap} introducing the gap of size $2(n_y-1)$ as shown in Eq. (31).

4. Shift by global offset $\Delta(e)$

Each element's contribution is shifted by $\Delta(e)$ into its global position using permutations $P_{\pm\Delta(e)}$, yielding

$$U_{\tilde{\mathbf{K}}^{\mathrm{el}}(e)} = P_{+\Delta(e)}P_{\mathrm{gap}}FU_{\mathbf{K}^{\mathrm{el}}}P_{\mathrm{gap}}^{\dagger}P_{-\Delta(e)}. \tag{39}$$

5. Sum of all $U_{\tilde{\mathbf{K}}^{el}(e)}$

We sum over all element contributions using the Linear Combination of Unitaries (LCU) method [30, 31]. Therefore, we prepare a block-encoding ancilla register `l` of $\lceil \log_2 n_{\mathrm{el}} \rceil$ qubits in a uniform superposition using Hadamards. This is sufficient, because using the fully structured mesh, all elements contribute equally to the global stiffness matrix. This is followed by the unitaries $U_{\tilde{\mathbf{K}}^{\mathrm{el}}(e)}$ controlled on the corresponding binary index $e-1$ in `l`. Finally, Hadamards on `l` uncompute the initial superposition. This circuit block-encodes the sum

$$\frac{1}{n_{\mathrm{el}}}\sum_{e=1}^{n_{\mathrm{el}}} U_{\tilde{\mathbf{K}}^{\mathrm{el}}(e)}. \tag{40}$$

Because only the outer permutations in Eq. (39) depend on e, the inner operations (gap insertion, zero-padding, and $U_{\mathbf{K}^{\mathrm{el}}}$) can be shared across all elements.

6. Conditioning on material distribution $\mathbf{x}$

To reflect the actual material distribution $\mathbf{x}$, we introduce a void ancilla `v` flipped when $x_e = 0$ and when the `l` register from step 5 is in state $|e-1\rangle_{\mathtt{l}}$. Thereby, we set the contribution of an empty element to zero and $\mathbf{K}$ is constructed only from filled elements and depends coherently on $\mathbf{x}$.

By implementing these six steps, we successfully construct the global block-encoding $U_{\mathbf{K}}$ of Eq. (30) with the scaling constant $\beta = n_{\mathrm{el}}\delta$. The complete circuit for a four-element example is shown in Fig. 13.

G. Choice of QSVT Polynomial

In Section III E, we explained how to use the QSVT algorithm for matrix inversion. Therefore, the QSVT polynomial approximates the reciprocal function $P_{\mathrm{inv}}(x) \approx \gamma/x$ on the interval $x \in [\mu, 1]$, and μ is the smallest singular value that is accurately inverted. Following Refs. [7, 8, 32], an odd target function is

$$f_{\mathrm{odd}}(x) = \begin{cases} \dfrac{\mu}{2x}, & |x| \geq \mu, \\ \psi_{\mathrm{odd}}(|x|), & \frac{\mu}{2} < |x| < \mu, \\ 0, & |x| \leq \frac{\mu}{2}, \end{cases} \tag{41}$$

with a smooth interpolation $\psi_{\mathrm{odd}} : [\mu/2, \mu] \to [0, 1/2]$ satisfying $\psi_{\mathrm{odd}}(\mu/2) = 0$ and $\psi_{\mathrm{odd}}(\mu) = 1/2$. Chebyshev techniques yield an $\epsilon\mu/2$-close polynomial approximation to f_{odd} of degree [8]

$$\deg(P_{\mathrm{odd}}) = \mathcal{O}\left(\frac{1}{\mu}\log\frac{1}{\mu\epsilon}\right). \tag{42}$$

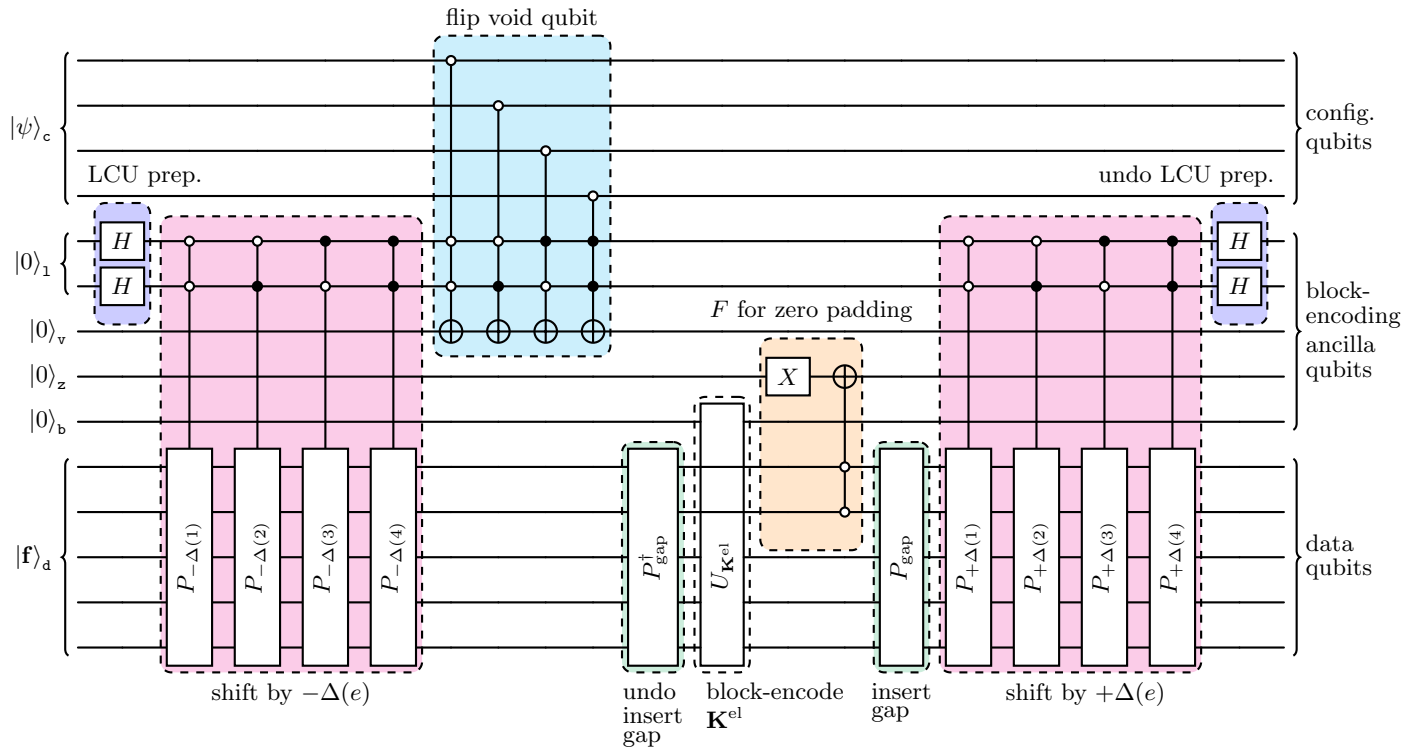

FIG. 13. Quantum circuit representation of the operator $U_{\mathbf{K}}$ block-encoding the global stiffness matrix $\mathbf{K}$ up to a scaling factor for a domain with four elements.

On $|x| \geq \mu$ this behaves like

$$P_{\text{odd}}(x) \approx \frac{\mu}{2}\frac{1}{x}, \tag{43}$$

so the effective scale factor is $\gamma_{\text{odd}} = \mu/2$.

However, in TO the global stiffness matrix $\mathbf{K}$ can become singular by design if elements are void or supports are insufficient. Then a singular value with $\sigma_i = 0$ is possible. Because $f_{\text{odd}}(0) = 0$, loads aligned with such singular vectors are mapped to zero rather than very large displacements, which underestimates compliance for infeasible configurations. To robustly penalize near-singular directions while keeping $|P_{\text{inv}}| \leq 1$, we use an even target function that acts as a singular value filter

$$f_{\text{even}}(x) = \begin{cases} \psi_{\text{even}}(x), & |x| < \mu, \\ \dfrac{y_0\,\mu}{|x|}, & |x| \geq \mu, \end{cases} \tag{44}$$

where $\psi_{\text{even}}(x)$ is a smooth and symmetric function satisfying $\psi_{\text{even}}(0) = 1$ and $\psi_{\text{even}}(x_1) > \psi_{\text{even}}(x_2)$ for $0 \leq |x_1| \leq |x_2| \leq \mu$, and $y_0 = f_{\text{even}}(\mu) \in (0, 1]$. On $|x| \geq \mu$ the polynomial approximation matches the scaled reciprocal,

$$P_{\text{even}}(x) \approx \frac{y_0\,\mu}{|x|}, \tag{45}$$

As with the odd case, one can construct a Chebyshev approximation on $[\mu, 1]$ with

$$\deg(P_{\text{even}}) = \mathcal{O}\left(\frac{1}{\mu}\log\frac{1}{\mu\epsilon}\right). \tag{46}$$

Fig. 14(a) shows $f_{\text{odd}}(x)$ and $f_{\text{even}}(x)$ scaled so that the function values lie in $[-1, 1]$ and match the reciprocal target function in $[\mu, 1]$.

In the end, our quantum approach computes the phase $\theta(\mathbf{x})$ from Eq. (24) associated with

$$\tilde{c}(\mathbf{x}) = \mathbf{f}^\top \mathbf{V} P_{\text{inv}}(\mathbf{\Sigma}) \mathbf{V}^\dagger \mathbf{f}, \tag{47}$$

where $\frac{1}{\beta}\mathbf{K}(\mathbf{x}) = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^\dagger$. For feasible structures, $P_{\text{inv}}(x) \approx \gamma/x$ gives the exact compliance up to a known scaling factor $\tilde{c}(\mathbf{x}) \approx \gamma\beta\, c(\mathbf{x})$. For infeasible designs with very small or zero singular values, the even polynomial saturates to 1, yielding large but finite displacements and thus a reasonable objective that mirrors the classical blow-up without numerical instabilities. In the SIMP method, one typically avoids the singular values with $\sigma_i = 0$ by setting $0 < x_e \ll 1$ for void elements. Fig. 14(b) reports compliance for all configurations of a 2×2 MBB beam, where we fixed all horizontal DoFs on the left boundary and the vertical DoF at the bottom-right corner. All three methods (SIMP with $x_e \in \{0.001, 1\}$, odd QSVT with $\mu = 10^{-3}$, and even QSVT with $\mu = 10^{-3}$ and $y_0 = 0.01$) agree on the three feasible designs, while the odd QSVT underestimates compliance elsewhere. This would mark all material

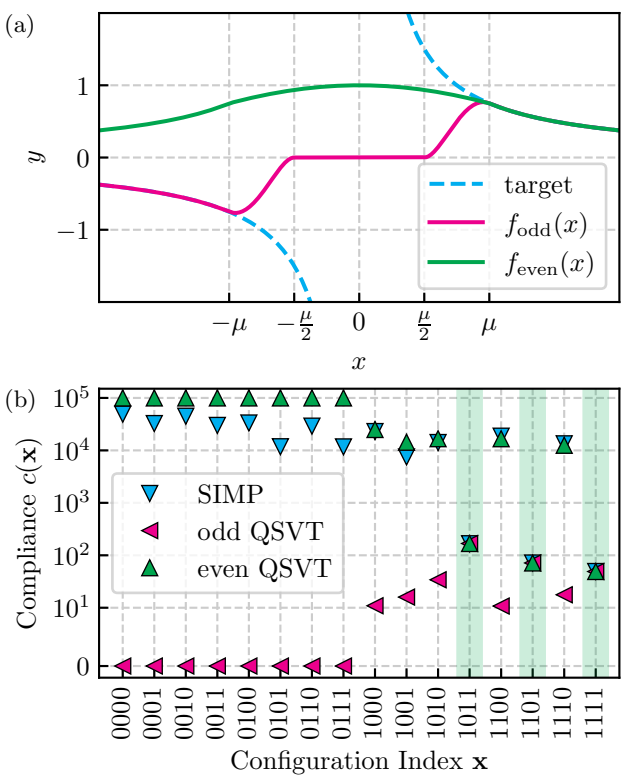

FIG. 14. Impact of the QSVT polynomial on compliance computation. (a) Target reciprocal function $\propto 1/x$ compared with its odd and even (polynomial) approximations over the interval $[\mu, 1]$. (b) Compliance values for a domain with $n_x = 2$, $n_y = 2$ obtained using the SIMP approach with $0 < x_e \ll 1$, and QSVT matrix inversion based on the odd or even approximation function. The green shaded region marks the three feasible configurations. One can observe that the odd QSVT leads to wrongfully small compliances for invalid structures. Consequently, Grover's algorithm would mark all configurations as solutions.

distributions $\mathbf{x}$ as valid solutions. The even QSVT leads to similarly large compliances on infeasible configurations as the SIMP method, making it a suitable choice for search and filtering.

H. Complexity Analysis

We summarize the algorithmic complexities $\mathcal{G}$ of the main subroutines and then combine them into end-to-end expressions. Therefore, we count logical resources only and do not account for error-correction overhead. In the following, we will use equal signs in combination with big O notation for convenience.

Grover's algorithm consists of the initialization U_{init} and $r = \mathcal{O}(\sqrt{N/M})$ (cf. Eq. (19)) Grover iterations of DO, where $N = \mathcal{O}(2^{n_{\text{el}}})$ is the size of the search space, and M is the number of marked solutions. Without volume constraints, $U_{\text{init}} = H^{\otimes n_{\text{el}}}$ and $\mathcal{G}[U_{\text{init}}] = n_{\text{el}}$. With volume constraints, U_{init} prepares the Dicke state and $\mathcal{G}[U_{\text{init}}] = \mathcal{O}(n_{\text{el}}k)$ [25]. The number of solid elements k depends on how fine our FEM discretization is and thus, we assume $k = \mathcal{O}(n_{\text{el}})$, which gives $\mathcal{G}[U_{\text{init}}] = \mathcal{O}(n_{\text{el}}^2)$. Grover's Diffuser D from Eq. (16) has similar gate complexity $\mathcal{G}[D] = \mathcal{G}[U_{\text{init}}] + \mathcal{G}[S_0] = \mathcal{O}(n_{\text{el}}^2)$ since the reflection S_0 corresponds to a multi-controlled Z operation with n_{el} controls, which can be implemented in $\mathcal{O}(n_{\text{el}})$ steps [33]. In general, we assume that any unitary U controlled on n qubits can be implemented in $\mathcal{O}(n + \mathcal{G}[U])$ steps by introducing an ancilla qubit [33]. Grover's oracle O (cf. Fig. 5) has complexity $\mathcal{G}[O] = \mathcal{G}[C] + \mathcal{G}[U_<]$, where $U_<$ is the bitwise comparator to a classical number. If $U_<$ is implemented via a ripple-carry adder, it can be implemented with $\mathcal{O}(n_{\mathtt{p}})$ operations [34]. Here, $n_i, i \in \{\mathtt{c}, \mathtt{p}, \mathtt{l}, \mathtt{d}, \mathtt{g}, \mathtt{h}, \mathtt{q}, \mathtt{v}, \mathtt{z}\}$ is the number of qubits of the respective register. Consequently, the gate complexity of Grover's algorithm is given by

$$\begin{aligned}\mathcal{G}[\text{Grover}] &= \mathcal{G}\left[U_{\text{init}}\right] + \mathcal{O}(r)(\mathcal{G}[D] + \mathcal{G}[O]) \\ &= \mathcal{O}\left(\sqrt{N/M}\left(n_{\text{el}}^2 + n_p + \mathcal{G}[C]\right)\right).\end{aligned} \tag{48}$$

The operation C computes the compliance $c(\mathbf{x})$ or equivalently the phase $\theta(\mathbf{x})$ from Eq. (24). It is a QAE circuit consisting of $n_{\mathtt{p}}$ Hadamards, a single A, $\mathcal{O}(2^{n_{\mathtt{p}}})$ controlled Grover operators G, and an inverse QFT with $\mathcal{O}(n_{\mathtt{p}})$ operations. The gate complexity of G is dominated by A and is thus given by $\mathcal{G}[G] = \mathcal{G}[A]$. As can be seen in Fig. 6, $\mathcal{G}[A] = \mathcal{O}(n_{\text{el}} + \mathcal{G}[U_{\mathbf{K}^{-1}}])$. Here, we assumed that the state preparation V_f requires $\mathcal{O}(n_{\text{el}})$ operations [35]. Accordingly, we write the complexity of C as

$$\begin{aligned}\mathcal{G}[C] &= \mathcal{O}(\mathcal{G}[H^{\otimes n_{\mathtt{p}}}] + \mathcal{G}[A] + 2^{n_{\mathtt{p}}}\mathcal{G}[G] + \mathcal{G}[\text{QFT}^\dagger]) \\ &= \mathcal{O}(2^{n_{\mathtt{p}}}(n_{\text{el}} + \mathcal{G}[U_{\mathbf{K}^{-1}}])).\end{aligned} \tag{49}$$

The operator $U_{\mathbf{K}^{-1}}$ block-encoding the inverse global stiffness matrix $\mathbf{K}^{-1}$ is the QSVT of $U_{\mathbf{K}}$ with an even polynomial $P_{\text{even}}(x)$ approximating $f_{\text{even}}(x)$ from Eq. (44). Thus, we have $\deg(P_{\text{even}})$ calls to $U_{\mathbf{K}}$ and Π_ϕ. For n_{fixed} DoFs, the projector-controlled phase-shift contains $\mathcal{O}(n_{\text{fixed}})$ multi-controlled X gates (MCXs) with up to $n_{\mathtt{d}} + n_{\mathtt{l}} + 3$ controls. Both, the data register $\mathtt{d}$ and the $\mathtt{l}$ register have $n_{\mathtt{d}} = n_{\mathtt{l}} = \mathcal{O}(\log n_{\text{el}})$ qubits. Hence, we have

$$\begin{aligned}\mathcal{G}[U_{\mathbf{K}^{-1}}] &= \deg(P_{\text{even}})(\mathcal{G}[U_{\mathbf{K}}] + \mathcal{G}[\Pi_\phi]) \\ &= \mathcal{O}\left(\tfrac{1}{\mu}\log\tfrac{1}{\mu\epsilon}(\mathcal{G}[U_{\mathbf{K}}] + n_{\text{fixed}}\log n_{\text{el}})\right).\end{aligned} \tag{50}$$

The block-encoding operator of the global stiffness matrix $U_{\mathbf{K}}$ requires the following gate operations. The LCU preparation step consists of $\mathcal{O}(n_{\mathtt{l}})$ Hadamards. Moreover, we have $\mathcal{O}(n_{\text{el}})$ adders $P_{\pm\Delta(e)}$ acting on $n_{\mathtt{d}}$ qubits controlled on $n_{\mathtt{l}}$ qubits. As previously mentioned, we assume ripple-carry adder implementations for estimating algorithmic complexity. This leads to linear complexity in the number of qubits [34]. The conditioning step on $|\mathbf{x}\rangle_{\mathtt{c}}$ introduces n_{el} MCXs (Multi-Controlled NOT) with $n_{\mathtt{l}} + 1$ controls. In addition, we

have the permutation operators P_{gap} and $P_{\text{gap}}^\dagger$ consisting of adders on $n_{\mathtt{d}}$ qubits. The block-encoding operator for the element stiffness matrix $U_{\mathbf{K}^{\text{el}}}$ is implemented in $\mathcal{O}(1)$ steps, F (see Section III F 2) corresponds to a MCX with $n_{\mathtt{d}} - 3$ controls. Under these considerations, we obtain the complexity

$$\begin{aligned}\mathcal{G}[U_{\mathbf{K}}] =& \mathcal{G}[H^{\otimes n_{\mathtt{l}}}] + \mathcal{O}(n_{\text{el}})\,\mathcal{G}[\text{MC}P_{\pm\Delta(e)}] \\ &+ \mathcal{O}(n_{\text{el}})\,\mathcal{G}[\text{MC}X] + \mathcal{G}[P_{\text{gap}}] + \mathcal{G}[U_{\mathbf{K}^{\text{el}}}] + \mathcal{G}[F] \\ =& \mathcal{O}(n_{\text{el}} \log n_{\text{el}}).\end{aligned} \tag{51}$$

This shows that $U_{\mathbf{K}}$ can efficiently implement $2^{n_{\text{el}}}$ possible global stiffness matrices $\mathbf{K}(\mathbf{x})$ in only $\mathcal{O}(n_{\text{el}} \log n_{\text{el}})$ steps.

Plugging everything together, we get the end-to-end complexity of our fault-tolerant quantum approach to TO

$$\begin{aligned}\mathcal{G}[\text{Grover}] = \mathcal{O}\Bigg(\sqrt{\tfrac{N}{M}}\Big(n_{\text{el}}^2 + 2^{n_{\mathtt{p}}}\tfrac{1}{\mu}\log\tfrac{1}{\mu\epsilon}(n_{\text{el}}\log n_{\text{el}} \\ + n_{\text{fixed}}\log n_{\text{el}})\Big)\Bigg).\end{aligned} \tag{52}$$

We can further simplify this expression by expressing the variables n_{fixed}, μ, and $n_{\mathtt{p}}$ in terms of n_{el}. The number of fixed DoFs scales like $n_{\text{fixed}} = \mathcal{O}(n_{\text{el}})$. The smallest singular value μ that we need to resolve within QSVT depends on the smallest non-zero eigenvalue within $\mathbf{K}$. In general, we can assume that the smallest non-zero eigenvalue scales like $\mathcal{O}(1/n_{\text{el}})$ [36, 37]. In addition, the sum in Eq. (40) gives another scaling with $1/n_{\text{el}}$. Thus, $\mu = \mathcal{O}(1/n_{\text{el}}^2)$. Finally, we need to estimate the number of phase qubits $n_{\mathtt{p}}$ storing the phase $\theta(\mathbf{x})$ associated with $c(\mathbf{x})$ as shown in Eq. (24). The compliance $c(\mathbf{x})$ converges for fine discretizations with large n_{el} and is thus independent of n_{el}. If we assume a normalized force vector $\mathbf{f}$, the scaling coefficients are $\alpha = 1/(\gamma\beta) = \mathcal{O}(n_{\text{el}})$. This leads to

$$\theta(\mathbf{x}) = 1/4 + \mathcal{O}(c(\mathbf{x})/n_{\text{el}}). \tag{53}$$

Hence, $2^{n_{\mathtt{p}}} = \mathcal{O}(n_{\text{el}})$, and we have the simplified complexities

$$\mathcal{G}[U_{\mathbf{K}}] = \mathcal{O}(n_{\text{el}} \log n_{\text{el}}), \tag{54}$$

$$\mathcal{G}[U_{\mathbf{K}^{-1}}] = \mathcal{O}(n_{\text{el}}^3 \log n_{\text{el}} \log(n_{\text{el}}/\epsilon)), \tag{55}$$

$$\mathcal{G}[C] = \mathcal{O}(n_{\text{el}}^4 \log n_{\text{el}} \log(n_{\text{el}}/\epsilon)), \tag{56}$$

$$\mathcal{G}[\text{Grover}] = \mathcal{O}\left(\sqrt{N/M}\, n_{\text{el}}^4 \log n_{\text{el}} \log(n_{\text{el}}/\epsilon)\right). \tag{57}$$

For completeness, we also give the space complexities for each qubit register

$$\begin{aligned} n_{\mathtt{c}} &= n_{\text{el}}, \\ n_{\mathtt{p}} &= \mathcal{O}(n_{\text{el}}), \\ n_{\mathtt{l}} &= \mathcal{O}(\log n_{\text{el}}), \\ n_{\mathtt{d}} &= \mathcal{O}(\log n_{\text{el}}), \\ n_{\mathtt{g}} &= n_{\mathtt{h}} = n_{\mathtt{q}} = n_{\mathtt{v}} = n_{\mathtt{z}} = 1. \end{aligned} \tag{58}$$

Thus, the algorithm requires $\mathcal{O}(n_{\text{el}})$ logical qubits overall, dominated by the phase register.

IV. NUMERICAL EXPERIMENTS

We evaluate our algorithm through numerical simulations with the PennyLane quantum computing framework [38]. All experiments are performed by classically simulating the quantum circuits, since the fault-tolerant nature of our approach makes it unsuitable for current NISQ devices. We use material parameters $E = 1.0$ and $\nu = 0.3$, with horizontal supports along the left edge, a vertical support at the bottom-right corner, and a unit force pointing downwards at the top-left corner, matching the setup of Fig. 2. We proceed step by step: first verifying compliance computation on a 2×2 domain, then demonstrating Grover's Search over the full binary space, and finally incorporating volume constraints with Dicke initialization on a 3×3 domain.

A. Compliance Computation

In the first experiment, we compute the compliance $c(\mathbf{x})$ or, more precisely, the phase $\theta(\mathbf{x})$ defined in Section III D. The corresponding operation C is a QAE circuit in which the Grover operator G contains the QSVT-based matrix inversion $U_{\mathbf{K}^{-1}}$ of the global stiffness matrix $\mathbf{K}$. Because C requires $\mathcal{O}(2^{n_{\mathtt{p}}})$ calls to $U_{\mathbf{K}^{-1}}$, direct simulation quickly becomes intractable. We therefore split the computation into two steps. First, we simulate the QSVT circuit to obtain the (pseudo-)inverse of $\mathbf{K}(\mathbf{x})$ for all material configurations. In a second step, we embed these (pseudo-)inverses as a block-encoding unitary into the Hadamard test (cf. Fig. 6) to construct A and G. See Appendix C 1 for further implementation details.

For demonstration, we consider a 2×2 domain without a volume constraint, yielding 16 possible configurations. The even QSVT polynomial is obtained by fitting a Chebyshev approximation to Eq. (44) with $\mu = 10^{-3}$ and $y_0 = 0.3$, subject to an absolute error tolerance of 10^{-3}. The resulting polynomial has degree $\deg(P_{\text{even}}) = 6610$, which is acceptable for proof-of-concept simulation. Further technical details on the polynomial construction are provided in Appendix B.

Figure 15(a) shows the probability distribution of the phase register with $n_{\mathtt{p}} = 5$ qubits after applying $C\,|0\rangle_{\mathtt{p}}\,|1111\rangle_{\mathtt{c}}$. Two sharp peaks appear at bitstrings 01000 and 11000, corresponding to phases ± 0.25 in big-endian notation. The exact value is $\theta(\mathbf{x} = 1111) = 0.25000826$, validating the simulation. Figure 15(b) reports $\theta(\mathbf{x})$ for all 16 configurations. For the three feasible cases $\mathbf{x} \in \{1011, 1101, 1111\}$, our quantum circuit results agree with classical FEM values up to the expected polynomial approximation error. Infeasible configurations would exhibit infinite compliance, but our quantum routine maps them to phases corresponding to large, finite values. Setting a threshold $\theta_0 = 0.263$ cleanly separates feasible from infeasible designs.

For comparison, Fig. 14(b) presented earlier shows

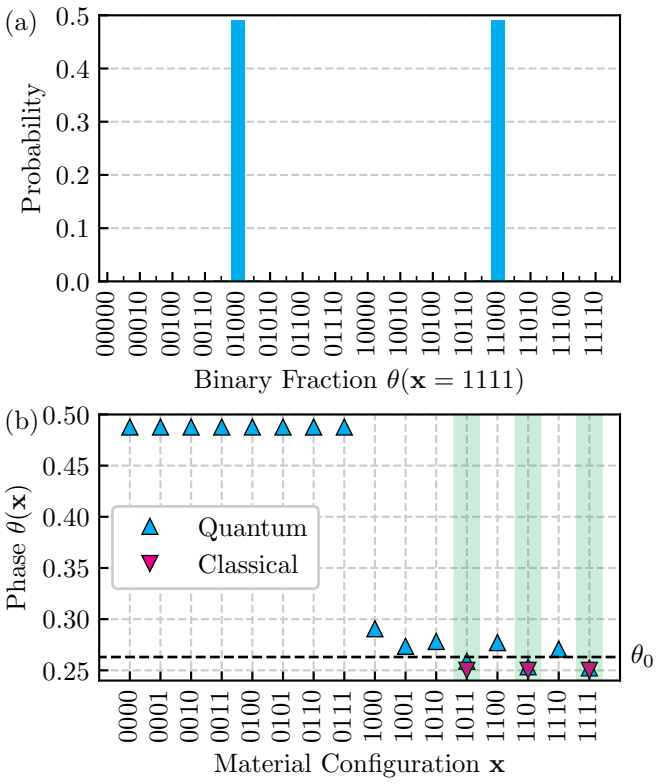

FIG. 15. Quantum compliance computation for the 2×2 MBB beam. (a) shows the probability distribution of measuring the phase register $\mathtt{p}$ after applying $C\,|0\rangle_{\mathtt{p}}\,|1111\rangle_{\mathtt{c}}$. (b) shows $\theta(\mathbf{x})$ for all 16 configurations. Green shading highlights feasible configurations.

the compliance for the same 2×2 domain computed with SIMP regularization, odd QSVT, and even QSVT. There, using smaller values $y_0 = 0.1$ produces a sharper separation between feasible and infeasible cases at the expense of an even higher polynomial degree as shown in Fig. 22. Additionally, the phase encoding $\theta(\mathbf{x}) \in [0.25, 0.5)$ compresses compliance values, bringing good and bad designs closer together on the phase scale.

B. Grover's Search

We next demonstrate Grover's algorithm introduced in Sections III B and III C on the same 2×2 domain. To keep the simulation tractable, we replace the full compliance computation C by a state-preparation routine that directly encodes the QAE outcome for each configuration $\mathbf{x}$ on $n_{\mathtt{p}} = 9$ qubits. See Appendix C 2 for further implementation details. We precompute the phases $\theta(\mathbf{x})$ with QSVT using $\mu = 10^{-3}$ and $y_0 = 0.3$. With a threshold of $\theta_0 = 0.263$, the oracle marks $M = 3$ feasible configurations out of the $N = 16$ candidates. After one Grover iteration ($r = 1$), their measurement probabilities are significantly amplified, as shown in Fig. 16. Feasible configurations dominate the distribution, while infeasible ones are strongly suppressed, demonstrating successful amplitude amplification.

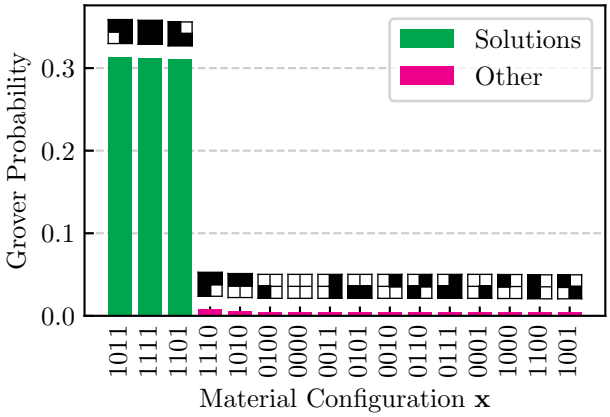

FIG. 16. Sorted probability distribution over all 16 configurations after running Grover's algorithm for a 2×2 beam. The corresponding structure is shown above each bar, where black elements indicate solid material.

C. Volume-Constrained Search

Finally, we extend the FEM discretization to a 3×3 domain while enforcing a fixed material volume of $V_0 = 5/9$ elements via Dicke state initialization $U_{\text{init}}\,|0\rangle_{\mathtt{c}} = |D_5^{(9)}\rangle_{\mathtt{c}}$. This reduces the search space from $2^9 = 512$ to $N = \binom{9}{5} = 126$ configurations. As before, we replace C by a state-preparation routine encoding the QAE outcome on $n_{\mathtt{p}} = 9$ qubits. Since the domain is larger with $n_{\text{el}} = 9$ elements, we choose a smaller $\mu = 10^{-5}$ (with $y_0 = 0.3$) for QSVT precomputation. After $r = 2$ Grover iterations with threshold $\theta_0 = 0.251$, the algorithm identifies $M = 8$ feasible configurations. Fig. 17 shows the resulting probability distribution, which correctly highlights all eight solutions.

The number of phase qubits $n_{\mathtt{p}} = 9$ is sufficient to distinguish between feasible and infeasible configurations, with phases up to $\theta = 0.2506$ and above $\theta = 0.2553$, respectively. However, resolving the optimal configuration among the eight feasible designs would require distinguishing $\Delta\theta \approx 5 \times 10^{-6}$, which would in turn demands at least $n_{\mathtt{p}} = 18$ qubits. From Eq. (53), we know that $\theta(\mathbf{x})$ approaches 0.25 as the number of elements n_{el} increases, explaining the high resolution required.

This limitation can be reframed as a feature. If our goal is only to identify feasible solutions under the volume constraint (rather than the single optimal solution), we can intentionally use fewer phase qubits. This effectively rounds all feasible phases to $\theta = 0.25$, as in our experiment above where $\theta = 0.2506 \approx 0.25$

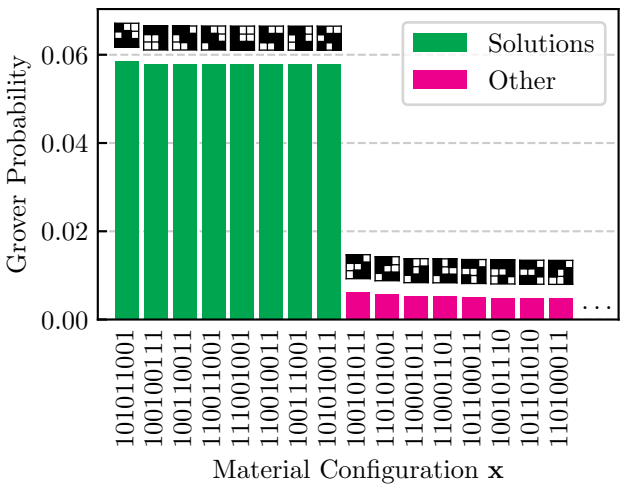

FIG. 17. Sorted probability distribution of the 16 most probable configurations for volume-constrained Grover's search on the 3×3 beam with $V_0 = 5/9$ filled elements enforced by Dicke initialization. The corresponding structure is shown above each bar, where black elements indicate solid material.

is rounded. A side effect is that the QAE probability distribution exhibits smoother amplitude accumulation instead of sharp peaks, which introduces noise into the Grover oracle. This explains why the separation between solution and non-solution probabilities is less pronounced here than in the unconstrained experiment.

V. DISCUSSION

While our numerical experiments showcase the functionality of the proposed algorithm, our complexity analysis illustrates where it has an advantage compared to classical approaches. The compliance computation C consists of the QSVT-based matrix inversion within the QAE circuit. It can in principle evaluate $N = \mathcal{O}(2^{n_{\text{el}}})$ candidate designs in polynomial time in n_{el}, see Eq. (56). For a single linear system, this offers no advantage over state-of-the-art classical linear equation system solvers, such as Gaussian elimination with $\mathcal{O}(n_{\text{el}}^3)$ or the conjugate gradient method with $\mathcal{O}(n_{\text{el}}\sqrt{\kappa})$ [39] in optimal cases, where $\kappa = \mathcal{O}(n_{\text{el}})$ [36, 37] is the condition number. However, since we never want to read out all displacement fields $\mathbf{u}(\mathbf{x}) = \mathbf{K}^{-1}(\mathbf{x})\mathbf{f}$, we can avoid the exponential output overhead [12]. Instead we evaluate exponentially many structures in coherent superposition using only $\mathcal{O}(n_{\text{el}}^4 \log n_{\text{el}} \log(n_{\text{el}}/\epsilon))$ quantum gates. On top of C, Grover's algorithm finds an optimal structure $\mathbf{x}$ satisfying Eq. (13) in quadratically less steps (cf. Eq. (57)) compared to a classical unstructured search algorithm. This poses a significant advantage over classical methods when one seeks not only the global optimum but all feasible solutions to a problem. More than one feasible solution might be desired because of latent requirements that cannot be considered in the optimization formulation. For example constraints from the chosen manufacturing procedure or human-product interaction. In the quantum setting, additional solutions can be found simply by repeated execution and sampling from the final quantum state. In contrast, classical gradient-based solvers typically require repeated experimentation with learning rates, penalty terms, and initialization strategies to find other solutions.

While the theoretical quadratic advantage seems promising, it might not be enough to be translated to an actual practical speedup [40]. Moreover, relaxation methods such as SIMP [3, 4] avoid exponential runtime by using gradient-based optimization. Although these methods may not find the global optimum, in practice a near-optimal result is often sufficient. Another practical limitation arises from the scaling $\mu = \mathcal{O}(1/n_{\text{el}}^2)$, which originates from the condition number κ and an additional factor of $1/n_{\text{el}}$ from the LCU in Eq. (40). The determination of the QSVT phases ϕ_d for small μ, and thus for high-degree QSVT polynomials, quickly becomes a non-negligible computational task itself [41–43]. Moreover, the runtime of QSVT, and consequently of our quantum topology algorithm, scales superlinearly with $1/\mu$ (cf. Eq. (52)).

One way to increase μ for larger n_{el} would be to introduce quantum-compatible preconditioners [44–46]. In the best case, this would have insignificant computational overhead and make κ independent of n_{el}, thereby reducing the scaling of μ by a factor of $1/n_{\text{el}}$. Another potential improvement could come from optimizing our block-encoding algorithm for $U_{\mathbf{K}}$ to mitigate the additional $1/n_{\text{el}}$ factor in the LCU in Eq. (40). However, it remains unclear how such an optimization could be achieved, as the current block-encoding is already highly efficient. In comparison to a conventional LCU block-encoding of a single stiffness matrix $\mathbf{K}$ with $\mathcal{O}(n_{\text{el}}^2)$ Pauli strings, our approach requires only $\mathcal{O}(n_{\text{el}} \log n_{\text{el}})$ gates. This does not even change when preparing all $2^{n_{\text{el}}}$ stiffness matrices $\mathbf{K}(\mathbf{x})$ in superposition.

Further improvements of our algorithm could be to reduce the QSVT circuit depth by implementing generalized QSVT [47], which extends projector-controlled phase-shifts beyond the Z axis. While we are using the nature of QSVT to handle singular matrices, one might also use other quantum linear system solvers with reduced runtime complexity [9–11]. Another possible extension would be to evaluate the structural performance of the system through frequency-domain response functions, as recently demonstrated by Danz et al. [48]. Their method computes the dynamic response of coupled oscillator systems directly from the FEM stiffness and mass matrices and could complement the present workflow by enabling dynamic or frequency-dependent topology optimization. Also their block-encoding with matrix access oracles [49] could

represent an attractive alternative to our block-encoding techniques. On the optimization side, Grover's algorithm could be replaced by alternative quantum optimization methods, such as QAOA [19–21, 50] or decoded quantum interferometry [51, 52]. In general, Bennett et al. [53] have shown that Grover's algorithm is asymptotically optimal for unstructured search. Nevertheless, since we can prepare the entire search space in polynomial time, Grover's algorithm becomes the bottleneck of our TO approach. Structured search algorithms that exploit the problem's inherent structure could potentially overcome this limitation.

VI. CONCLUSION

We have presented a fault-tolerant quantum algorithm for TO and demonstrated its feasibility on the 2D MBB beam problem. Our approach combines block-encoding of FEM stiffness matrices, QSVT-based matrix inversion, compliance computation through a Hadamard test and QAE, and Grover's search with Dicke state initialization to enforce volume constraints. By integrating these components, we establish an end-to-end demonstration of how TO can be reformulated and addressed within a coherent quantum workflow. In our detailed complexity analysis, we show that our algorithm maintains the quadratic Grover speedup for unstructured search.

Our numerical experiments validate each subroutine. Compliance computation via QSVT and QAE reproduces classical FEM results within polynomial approximation error, while Grover's search amplifies feasible, low-compliance designs in both unconstrained and volume-constrained settings. These proof-of-concept simulations confirm that the algorithm is logically consistent and illustrate the potential of quantum superposition to evaluate many candidate designs at once. At the same time, they highlight technical challenges, notably the growth of polynomial degree with system size and the high phase resolution required to distinguish near-optimal solutions.

Although our present study is limited to small-scale examples simulated on classical hardware, it establishes a blueprint for fault-tolerant quantum TO. As quantum hardware progresses toward large-scale fault-tolerance, the principles demonstrated here may enable practical advantages for engineering design tasks where the combinatorial explosion of candidate structures is the primary computational bottleneck.

DATA AND CODE AVAILABILITY

All code and Jupyter notebooks to reproduce the results and figures presented in this work are available at github.com/Emerging-Tech-MUC/QTO.

ACKNOWLEDGMENTS

We acknowledge the support of the BMW Group. L. Karch is partly funded by the German Ministry for Economic Affairs and Energy (BMWE) in the project QCHALLenge under Grant 01MQ22008D. O. Ahrend acknowledges support by the Interdisciplinary Doctoral Program in Quantum Systems Integration funded by the BMW Group.

Appendix A: Derivation of Element Stiffness Matrix

In the following, we provide the exact form of the element stiffness matrix $\mathbf{K}^{\text{el}}$ for a square, bilinear, four-noded quadrilateral element. Due to the element's simple geometry, $\mathbf{K}^{\text{el}}$ is determined by eight unique parameters

$$\begin{aligned} k_1 &= \tfrac{1}{2} - \tfrac{\nu}{6}, & k_2 &= -\tfrac{1}{8} - \tfrac{\nu}{8}, & k_3 &= \tfrac{\nu}{6}, & k_4 &= -\tfrac{1}{8} + \tfrac{3\nu}{8}, \\ k_5 &= -\tfrac{1}{4} - \tfrac{\nu}{12}, & k_6 &= \tfrac{1}{8} - \tfrac{3\nu}{8}, & k_7 &= -\tfrac{1}{4} + \tfrac{\nu}{12}, & k_8 &= \tfrac{1}{8} + \tfrac{\nu}{8} \end{aligned} \tag{A1}$$

yielding $\mathbf{K}^{\text{el}} = \frac{E}{1-\nu^2}\mathbf{K}_0$ with

$$\mathbf{K}_0 = \begin{pmatrix} k_1 & k_2 & k_3 & k_4 & k_5 & k_6 & k_7 & k_8 \\ k_2 & k_1 & k_6 & k_5 & k_4 & k_3 & k_8 & k_7 \\ k_3 & k_6 & k_1 & k_8 & k_7 & k_2 & k_5 & k_4 \\ k_4 & k_5 & k_8 & k_1 & k_2 & k_7 & k_6 & k_3 \\ k_5 & k_4 & k_7 & k_2 & k_1 & k_8 & k_3 & k_6 \\ k_6 & k_3 & k_2 & k_7 & k_8 & k_1 & k_4 & k_5 \\ k_7 & k_8 & k_5 & k_6 & k_3 & k_4 & k_1 & k_2 \\ k_8 & k_7 & k_4 & k_3 & k_6 & k_5 & k_2 & k_1 \end{pmatrix}. \tag{A2}$$

Here, Young's modulus E and Poisson's ratio ν are material properties.

Next, we outline the derivation of $\mathbf{K}^{\text{el}}$ following standard FEM analysis. Fig. 18 shows the element in the isoparametric coordinate system $(\xi, \eta) \in [-1, 1]^2$.

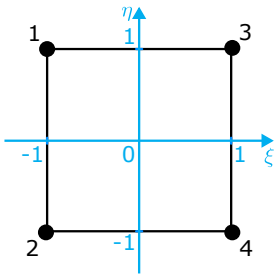

FIG. 18. Square, four-noded element defined in the isoparametric coordinate system.

The four corner nodes are

$$\begin{aligned} &(1)\ \text{top-left:} && \xi = -1, && \eta = 1, \\ &(2)\ \text{bottom-left:} && \xi = -1, && \eta = -1, \\ &(3)\ \text{top-right:} && \xi = 1, && \eta = 1, \\ &(4)\ \text{bottom-right:} && \xi = 1, && \eta = -1. \end{aligned} \tag{A3}$$

For a physical element of side length l, we have the mapping

$$x = \frac{l}{2}(\xi + 1) \quad \text{and} \quad y = \frac{l}{2}(\eta + 1). \tag{A4}$$

The Jacobi determinant of this transformation is $|\mathbf{J}| = l^2/4$. Using (ξ, η) lets one write a single element formulation that applies to any square or rectangular element. In the end, the factor $|\mathbf{J}|$ rescales the stiffness matrix to its physical dimensions.

Following typical finite element analysis, we define so-called bilinear shape functions for each node as

$$\begin{array}{ll} N_1 = \frac{1}{4}(1-\xi)(1+\eta), & N_2 = \frac{1}{4}(1-\xi)(1-\eta), \\ N_3 = \frac{1}{4}(1+\xi)(1+\eta), & N_4 = \frac{1}{4}(1+\xi)(1-\eta). \end{array} \tag{A5}$$

Each N_i equals one at its own node and zero at the others, so the combination $\sum_i N_i(\, u_{2i-1}, u_{2i})^\top$ provides a bilinear interpolation of displacement inside the element.

Next, we calculate the 3×8 strain-displacement matrix $\mathbf{B}$ that converts nodal displacements $\mathbf{u}^{\text{el}}$ to in-plane strains $\varepsilon = (\varepsilon_{xx}, \varepsilon_{yy}, \gamma_{xy})^\top$. Using the spatial derivatives

$$\frac{\partial N_i}{\partial x} = \frac{2}{l}\frac{\partial N_i}{\partial \xi} \quad \text{and} \quad \frac{\partial N_i}{\partial y} = \frac{2}{l}\frac{\partial N_i}{\partial \eta} \tag{A6}$$

the strain-displacement matrix is given by

$$\mathbf{B}(\xi,\eta) = \begin{pmatrix} \frac{\partial N_1}{\partial x} & 0 & \frac{\partial N_2}{\partial x} & 0 & \frac{\partial N_3}{\partial x} & 0 & \frac{\partial N_4}{\partial x} & 0 \\ 0 & \frac{\partial N_1}{\partial y} & 0 & \frac{\partial N_2}{\partial y} & 0 & \frac{\partial N_3}{\partial y} & 0 & \frac{\partial N_4}{\partial y} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial y} & \frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial y} & \frac{\partial N_3}{\partial x} & \frac{\partial N_4}{\partial y} & \frac{\partial N_4}{\partial x} \end{pmatrix}. \tag{A7}$$

For an isotropic, linear-elastic material under plane-stress conditions, Hooke's law reads $\sigma = \mathbf{D}\,\varepsilon$ with the constitutive matrix

$$\mathbf{D} = \frac{E}{1-\nu^2}\begin{pmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{pmatrix}, \tag{A8}$$

where Young's modulus E and Poisson's ratio ν are material parameters.

Finally, the element stiffness matrix is given by

$$\mathbf{K}^{\text{el}} = \int_{-1}^{1}\int_{-1}^{1} \mathbf{B}^\top \mathbf{D}\mathbf{B}|\mathbf{J}|\, d\xi\, d\eta \tag{A9}$$

leading to the definition in Eq. (A2). Notably, $\mathbf{K}^{\text{el}}$ is independent of l since the terms in $\mathbf{B}$ and $|\mathbf{J}|$ cancel out.

Appendix B: Practical Construction of the QSVT Polynomial

Constructing the QSVT polynomial is an essential step in our algorithm. Ref. [8] describes how one can analytically build an approximation of $f_{\text{odd}}(x)$ in Eq. (41) by combining known polynomial approximations of elementary functions such as rectangular step functions. This analytical approach has the important advantage that it allows a derivation of the asymptotic behavior of the resulting polynomial. However, in practice such constructions typically lead to very high polynomial degrees. Equivalent approximations of much lower degree can often be obtained numerically. For this reason, we demonstrate the numerical polynomial approximation of $f_{\text{even}}(x)$ in Eq. (44) in the following, rather than relying on an explicit analytical construction.

While the theoretical form in Eq. (44) specifies the general structure, the practical implementation requires a concrete choice for the smooth interpolation $\psi_{\text{even}}(x)$ in the region $|x| < \mu$. We use the cosine interpolation

$$\psi_{\text{even}}(x) = \cos\left(\frac{\arccos(y_0)}{\mu}\, x\right), \quad |x| < \mu, \tag{B1}$$

which ensures a smooth connection at $x = \pm\mu$ with $f_{\text{even}}(\pm\mu) = y_0$. The complete target function is then

$$f_{\text{even}}(x) = \begin{cases} \cos\left(\frac{\arccos(y_0)}{\mu}\, x\right), & |x| < \mu, \\ \frac{y_0\mu}{|x|}, & |x| \geq \mu, \end{cases} \tag{B2}$$

We approximate $f_{\text{even}}(x)$ using Chebyshev polynomials. Exploiting the even symmetry of the target function reduces the effective degree of the fitted polynomial by half, improving numerical stability and reducing computation time. Instead of approximating $f_{\text{even}}(x)$ directly on $[-1, 1]$, we approximate the auxiliary function $g(t) = f_{\text{even}}(\sqrt{t})$ on $t \in [0, 1]$. If

$$P(t) = \sum_{k=0}^{n} c_k T_k(t) \tag{B3}$$

is a Chebyshev polynomial approximation of $g(t)$, then

$$Q(x) = P(x^2) = \sum_{k=0}^{n} c_k T_k(x^2) \tag{B4}$$

automatically satisfies the symmetry $Q(-x) = Q(x)$ and approximates $f_{\text{even}}(x)$ on $[-1, 1]$. In practice, this corresponds to placing the coefficients c_k at even positions in the Chebyshev expansion,

$$Q(x) = \sum_{k=0}^{n} c_k T_{2k}(x) = c_0 T_0(x) + c_1 T_2(x) + c_2 T_4(x) + \cdots. \tag{B5}$$

We implemented this Chebyshev approximation procedure using the Python library `chebpy` [54]. Fig. 19 shows the even target function for $\mu = 0.01$, $y_0 = 0.5$, and a polynomial approximation with an absolute error tolerance of $\varepsilon = 10^{-3}$. The resulting polynomial has degree 382 and overlaps well with the target function.

Using this numerical construction, we study how the required polynomial degree scales with μ, ε, and y_0. The parameter μ, representing the smallest singular value inverted accurately, has a significant impact on

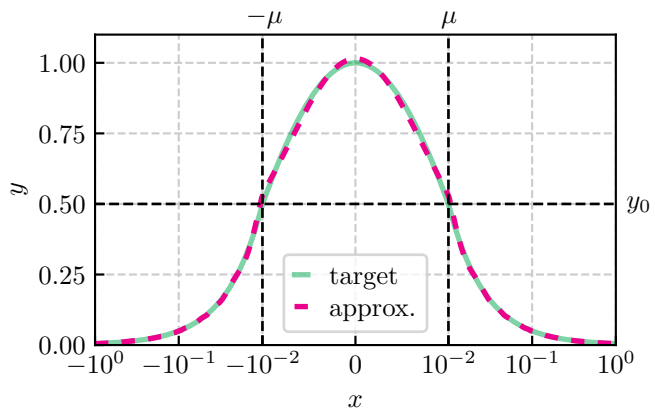

FIG. 19. Chebyshev polynomial approximation (degree 382) of $f_{\text{even}}(x)$ for $\mu = 0.01$, $y_0 = 0.5$, and $\varepsilon = 10^{-3}$. The horizontal axis is shown on a linear scale in the region $x \in [-\mu, \mu]$.

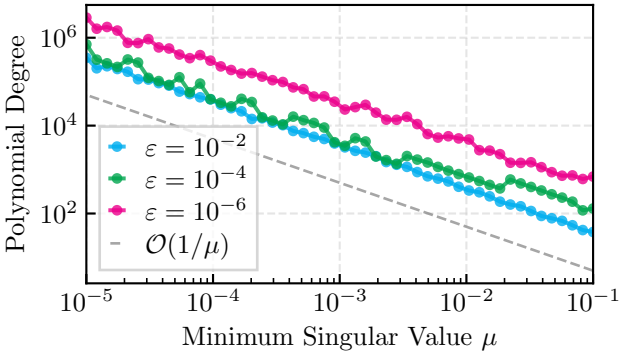

FIG. 20. Scaling of the polynomial degree with μ for different error tolerances ε. The observed behavior confirms the $\mathcal{O}(1/\mu)$ stated in Eq. (46).

runtime complexity. Fig. 20 shows the degree of the polynomial against μ for different ε, confirming the expected reciprocal scaling $\mathcal{O}(1/\mu)$ assumed in Eq. (46).

Similarly, Fig. 21 shows the polynomial degree as a function of ε, agreeing with the expected $\mathcal{O}(\log(1/\varepsilon))$ scaling. Note that Eq. (46) characterizes the case of relative error $\epsilon = \mathcal{O}(\varepsilon/\mu)$, whereas ε is the absolute error.

Finally, we analyze the influence of y_0 in Fig. 22. No clear scaling law emerges, but since y_0 can be chosen independently of μ and ε, we typically fix it to a moderate constant, e.g. $y_0 = 0.3$. The parameter y_0 controls the distance in $\theta(\mathbf{x})$ between feasible and infeasible solutions. Thus, in practice we aim to set y_0 as small as possible while still keeping the polynomial degree manageable.

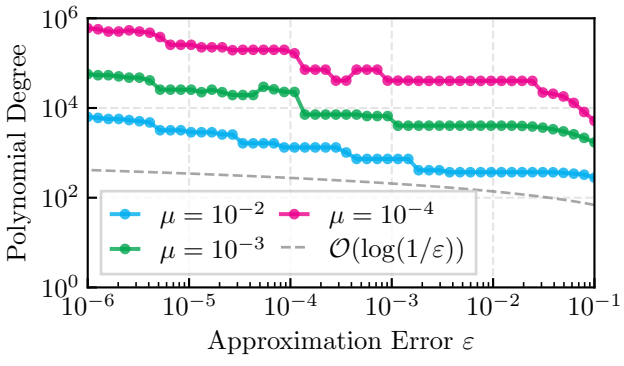

FIG. 21. Scaling of the polynomial degree with approximation error ε. The results agree with the expected $\mathcal{O}(\log(1/\varepsilon))$ scaling.

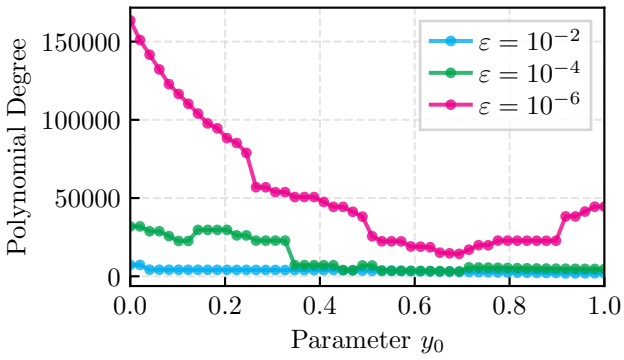

FIG. 22. Influence of y_0 on the required polynomial degree. Although no clear scaling law is visible, the polynomial degree typically increases with decreasing y_0.

Consequently, our numerical construction of $f_{\text{even}}(x)$ using Chebyshev polynomials provides a practical and efficient way to obtain QSVT polynomials of moderate degree while retaining the expected asymptotic scaling with μ and ε. This validates the theoretical complexity estimates given in Sec. III E, and confirms that the polynomial construction step does not introduce unexpected overhead beyond the known $\mathcal{O}(1/\mu)$ and $\mathcal{O}(\log(1/\varepsilon))$ behavior.

Appendix C: Implementation Details of Our Numerical Experiments

The end-to-end algorithm presented here yields a deep quantum circuit acting on many qubits. Its full execution would require fault-tolerant quantum computers, and its simulation quickly becomes intractable. Nevertheless, we presented simulation results of our algorithm in Section IV. To make these simulations feasible, we replaced certain costly quantum operations by hardcoded unitaries. While this makes simulations of small

systems (with few elements in the domain discretization) tractable, this approach of course does not work for executions on real machines and is not scalable for large simulations, as replacing these unitaries can lead to exponential overhead. In the following, we provide the exact details on how we simplified the simulations for our numerical experiments.

1. Details on Compliance Computation

In Section IV A, we split the compliance computation C in two steps. First, we use QSVT to calculate the (pseudo-)inverses $\mathbf{K}^{-1}(\mathbf{x})$ for each material configuration $\mathbf{x}$. Therefore, we use `chebpy` [54] to fit a Chebyshev polynomial to our even target function in Eq. (B2) with $\mu = 10^{-3}$ and an absolute error $\varepsilon = 10^{-3}$. This leads to a polynomial with degree 6610. We then convert the Chebyshev coefficients to the set of phase angles $\{\phi_k\}$ required for the projector-controlled phase-shifts Π_{ϕ_k} using the Python library `pyqsp` [8]. This allows us to calculate $U_{\mathbf{K}^{-1}}$ from Eq. (28) and then to extract $\mathbf{K}^{-1}$.

In a second step, we manually block-encode $\mathbf{K}^{-1}$ in a unitary operator as shown by Eq. (36). Thereby, we get a unitary $U_{\mathbf{K}^{-1}}$ which replaces the deep QSVT circuit within A (cf. Fig. 6) to compute the entire operation C.

2. Details on Grover's Search

In Section IV B, we demonstrate Grover's Search for a 2×2 domain. There, we replace the compliance computation C by several multi-controlled state preparation routines. As shown in Fig. 15, C is a QAE circuit which estimates the absolute value of an amplitude a of some state

$$|\phi\rangle = a\,|\alpha\rangle_{\mathtt{d}} + \sqrt{1-a^2}\,|\alpha^\perp\rangle_{\mathtt{d}}\,. \tag{C1}$$

If we know $|a|$, we also know the analytical expression of state after QAE

$$\begin{aligned}\mathrm{QAE}\,|0\rangle_{\mathtt{p}}\,|\phi\rangle_{\mathtt{d}} = -\frac{i}{\sqrt{2}}\sum_{j=0}^{2^{n_{\mathtt{p}}}-1} e^{-i\pi\theta}\alpha_+(j)\,|j\rangle_{\mathtt{p}}\,|\psi_+\rangle_{\mathtt{d}} \\ - e^{i\pi\theta}\alpha_-(j)\,|j\rangle_{\mathtt{p}}\,|\psi_-\rangle_{\mathtt{d}}\,,\end{aligned} \tag{C2}$$

where the phase θ is given by

$$\theta = \pm\tfrac{1}{\pi}\arcsin(|a|), \tag{C3}$$

the amplitudes $\alpha_\pm(j)$ are

$$\alpha_\pm(j) = \frac{1}{2^{n_{\mathtt{p}}}}\sum_{l=0}^{2^{n_{\mathtt{p}}}-1} e^{-i2\pi l(2^{-n_{\mathtt{p}}}j \mp \theta)}, \tag{C4}$$

and

$$|\psi_\pm\rangle = \frac{1}{\sqrt{2}}(|\alpha\rangle \pm i\,|\alpha^\perp\rangle) \tag{C5}$$

are the eigenstates of the Grover operator G shown in Fig. 8 in the subspace spanned by $|\alpha\rangle$ and $|\alpha^\perp\rangle$. In our TO case, $|h_0(\mathbf{x})|$ in Eq. (23) is the absolute amplitude $|a|$ that we would like to estimate. We classically calculate

$$|h_0(\mathbf{x})| = \sqrt{\tfrac{1}{2} + \tfrac{1}{2\alpha}\mathbf{f}^\top \underbrace{\tfrac{\gamma}{\beta}\mathbf{K}^{-1}(\mathbf{x})}_{=\mathbf{V}f_{\mathrm{even}}(\mathbf{\Sigma})\mathbf{V}^\dagger}\mathbf{f}} \tag{C6}$$

for every configuration $\mathbf{x}$. Thereby, we also replace the deep QSVT circuit by its expected outcome $\mathbf{V}f_{\mathrm{even}}(\mathbf{\Sigma})\mathbf{V}^\dagger$. Note that we directly use the target function $f_{\mathrm{even}}(\cdot)$ from Eq. (B2) instead of fitting a Chebyshev polynomial to it. We use all values $|h_0(\mathbf{x})|$ to calculate the QAE amplitudes according to Eq. (C2) and prepare the quantum state accordingly controlled on the respective configuration $|\mathbf{x}\rangle_{\mathtt{c}}$. This allows us to embed these multi-controlled state preparations in Grover's oracle (cf. Fig. 5) replacing C.

3. Details on Volume-Constrained Search

In Section IV C, we demonstrate Grover's search for a 3×3 domain with an additional volume constraint. Grover's oracle is simulated as in the previous unconstrained 2×2 case. However, the operation U_{init} in Grover's algorithm needs to prepare the right Dicke state, whereas it consisted of simple Hadamards before. Instead of the efficient routine of Ref. [25], we employ a standard state preparation operator provided by PennyLane. This suffices for our purpose of demonstrating the algorithm.

[1] M. P. Bendsøe and O. Sigmund, *Topology Optimization* (Springer Berlin Heidelberg, Berlin, Heidelberg, 2004).

[2] O. Sigmund and K. Maute, Topology optimization approaches: A comparative review, Structural and Multidisciplinary Optimization **48**, 1031 (2013).

[3] O. Sigmund, A 99 line topology optimization code written in Matlab, Structural and Multidisciplinary Optimization **21**, 120 (2001).

[4] E. Andreassen, A. Clausen, M. Schevenels, B. S. Lazarov, and O. Sigmund, Efficient topology optimization in MATLAB using 88 lines of code, Structural and Multidisciplinary Optimization **43**, 1 (2011).

[5] L. K. Grover, A fast quantum mechanical algorithm for database search, in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96* (ACM Press, Philadelphia, Pennsylvania, United

States, 1996) pp. 212–219.
[6] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum Algorithm for Linear Systems of Equations, Physical Review Letters **103**, 150502 (2009).
[7] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe, Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics, in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing* (ACM, Phoenix AZ USA, 2019) pp. 193–204.
[8] J. M. Martyn, Z. M. Rossi, A. K. Tan, and I. L. Chuang, Grand Unification of Quantum Algorithms, PRX Quantum **2**, 040203 (2021).
[9] P. C. Costa, D. An, Y. R. Sanders, Y. Su, R. Babbush, and D. W. Berry, Optimal Scaling Quantum Linear-Systems Solver via Discrete Adiabatic Theorem, PRX Quantum **3**, 040303 (2022).
[10] Z. Liu, G. Li, and L. Li, Improved quantum linear system solver via quantum phase discrimination, The European Physical Journal Special Topics 10.1140/epjs/s11734-025-01627-7 (2025).
[11] A. M. Dalzell, A shortcut to an optimal quantum linear system solver (2024), version Number: 1.
[12] S. Aaronson, Read the fine print, Nature Physics **11**, 291 (2015).
[13] Y. Sato, R. Kondo, S. Koide, and S. Kajita, Quantum topology optimization of ground structures for near-term devices, in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE, Bellevue, WA, USA, 2023) pp. 168–176.
[14] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, and P. J. Coles, Variational Quantum Linear Solver, Quantum **7**, 1188 (2023).
[15] G. Iannelli, *Noisy Bayesian Optimization of Variational Quantum Eigensolvers*, Ph.D. thesis (2024), publisher: Humboldt-Universität zu Berlin.
[16] Z. Ye and W. Pan, Towards Quantum Accelerated Large-scale Topology Optimization (2025), arXiv:2507.14478 [physics].
[17] G. B. Dantzig and P. Wolfe, Decomposition Principle for Linear Programs, Operations Research **8**, 101 (1960).
[18] N. Sukulthanasorn, J. Xiao, K. Wagatsuma, R. Nomura, S. Moriguchi, and K. Terada, A novel design update framework for topology optimization with quantum annealing: Application to truss and continuum structures, Computer Methods in Applied Mechanics and Engineering **437**, 117746 (2025).
[19] J. Stein, L. Müller, L. Hölscher, G. Chnitidis, J. Jojo, A. Farea, M. S. Çelebi, D. Bucher, J. Wulf, D. Fischer, P. Altmann, C. Linnhoff-Popien, and S. Feld, Exponential Quantum Speedup for Simulation-Based Optimization Applications (2023), version Number: 3.
[20] M. Adler, J. Stein, and M. Lachner, Scaling Quantum Simulation-Based Optimization: Demonstrating Efficient Power Grid Management with Deep QAOA Circuits (2025), arXiv:2505.16444 [quant-ph].
[21] L. Hölscher, L. Müller, O. Samimi, and T. Danzig, Quantum Simulation-Based Optimization of a Cooling System (2025), version Number: 2.
[22] R. Picelli, R. Sivapuram, and Y. M. Xie, A 101-line MATLAB code for topology optimization using binary variables and integer programming, Structural and Multidisciplinary Optimization **63**, 935 (2021).
[23] C. Durr and P. Hoyer, A Quantum Algorithm for Finding the Minimum (1996), version Number: 2.
[24] M. Boyer, G. Brassard, P. Høyer, and A. Tapp, Tight Bounds on Quantum Searching, Fortschritte der Physik **46**, 493 (1998).
[25] A. Bärtschi and S. Eidenbenz, Deterministic Preparation of Dicke States, in *Fundamentals of Computation Theory*, Vol. 11651, edited by L. A. Gasieniec, J. Jansson, and C. Levcopoulos (Springer International Publishing, Cham, 2019) pp. 126–139, series Title: Lecture Notes in Computer Science.
[26] L. Lin, Lecture Notes on Quantum Algorithms for Scientific Computation (2022), version Number: 1.
[27] G. Brassard, P. Høyer, M. Mosca, and A. Tapp, Quantum amplitude amplification and estimation, in *Contemporary Mathematics*, Vol. 305, edited by S. J. Lomonaco and H. E. Brandt (American Mathematical Society, Providence, Rhode Island, 2002) pp. 53–74.
[28] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*, 1st ed. (Cambridge University Press, 2012).
[29] V. Shende, S. Bullock, and I. Markov, Synthesis of quantum-logic circuits, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **25**, 1000 (2006).
[30] A. M. Childs and N. Wiebe, Hamiltonian Simulation Using Linear Combinations of Unitary Operations, Quantum Information and Computation **12**, 10.26421/QIC12.11-12 (2012).
[31] C. Sünderhauf, E. Campbell, and J. Camps, Block-encoding structured matrices for data input in quantum computing, Quantum **8**, 1226 (2024).
[32] A. M. Childs, R. Kothari, and R. D. Somma, Quantum Algorithm for Systems of Linear Equations with Exponentially Improved Dependence on Precision, SIAM Journal on Computing **46**, 1920 (2017).
[33] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Elementary gates for quantum computation, Physical Review A **52**, 3457 (1995).
[34] S. A. Cuccaro, T. G. Draper, S. A. Kutin, and D. P. Moulton, A new quantum ripple-carry addition circuit (2004), version Number: 1.
[35] M. Mottonen, J. J. Vartiainen, V. Bergholm, and M. M. Salomaa, Transformation of quantum states using uniformly controlled rotations (2004), arXiv:quant-ph/0407010.
[36] I. Fried, Bounds on the spectral and maximum norms of the finite element stiffness, flexibility and mass matrices, International Journal of Solids and Structures **9**, 1013 (1973).
[37] L. K. Lennard Kamenski and W. H. Weizhang Huang, A Study on the Conditioning of Finite Element Equations with Arbitrary Anisotropic Meshes via a Density Function Approach, Journal of Mathematical Study **47**, 151 (2014).
[38] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, S. Ahmed, V. Ajith, M. S. Alam, G. Alonso-Linaje, B. AkashNarayanan, A. Asadi, J. M. Arrazola, U. Azad, S. Banning, C. Blank, T. R. Bromley, B. A. Cordier, J. Ceroni, A. Delgado, O. D. Matteo, A. Dusko, T. Garg, D. Guala, A. Hayes, R. Hill, A. Ijaz, T. Isacsson, D. Ittah, S. Jahangiri, P. Jain, E. Jiang, A. Khandelwal, K. Kottmann, R. A. Lang, C. Lee, T. Loke, A. Lowe,

K. McKiernan, J. J. Meyer, J. A. Montañez-Barrera, R. Moyard, Z. Niu, L. J. O'Riordan, S. Oud, A. Panigrahi, C.-Y. Park, D. Polatajko, N. Quesada, C. Roberts, N. Sá, I. Schoch, B. Shi, S. Shu, S. Sim, A. Singh, I. Strandberg, J. Soni, A. Száva, S. Thabet, R. A. Vargas-Hernández, T. Vincent, N. Vitucci, M. Weber, D. Wierichs, R. Wiersema, M. Willmann, V. Wong, S. Zhang, and N. Killoran, PennyLane: Automatic differentiation of hybrid quantum-classical computations (2022), arXiv:1811.04968 [quant-ph].

[39] G. H. Golub and C. F. Van Loan, *Matrix computations*, fourth edition ed., Johns Hopkins studies in the mathematical sciences (The Johns Hopkins University Press, Baltimore, 2013).

[40] R. Babbush, J. R. McClean, M. Newman, C. Gidney, S. Boixo, and H. Neven, Focus beyond Quadratic Speedups for Error-Corrected Quantum Advantage, PRX Quantum **2**, 010103 (2021).

[41] Y. Dong, X. Meng, K. B. Whaley, and L. Lin, Efficient phase-factor evaluation in quantum signal processing, Physical Review A **103**, 042419 (2021).

[42] L. Lin, Mathematical and numerical analysis of quantum signal processing (2025), arXiv:2510.00443 [quant-ph].

[43] H. Ni and L. Ying, Fast Phase Factor Finding for Quantum Signal Processing (2024), version Number: 2.

[44] B. D. Clader, B. C. Jacobs, and C. R. Sprouse, Preconditioned Quantum Linear System Algorithm, Physical Review Letters **110**, 250504 (2013).

[45] M. Deiml and D. Peterseim, Quantum Realization of the Finite Element Method (2024), version Number: 4.

[46] L. Lapworth and C. Sünderhauf, Preconditioned block encodings for quantum linear systems, Quantum Science and Technology **10**, 045064 (2025).

[47] D. Motlagh and N. Wiebe, Generalized Quantum Signal Processing, PRX Quantum **5**, 020368 (2024).

[48] S. Danz, M. Berta, S. Schröder, P. Kienast, F. K. Wilhelm, and A. Ciani, Calculating response functions of coupled oscillators using quantum phase estimation, Physical Review Research **7**, 023264 (2025).

[49] D. Camps, L. Lin, R. Van Beeumen, and C. Yang, Explicit Quantum Circuits for Block Encodings of Certain Sparse Matrices, SIAM Journal on Matrix Analysis and Applications **45**, 801 (2024).

[50] E. Farhi, J. Goldstone, and S. Gutmann, A Quantum Approximate Optimization Algorithm (2014), version Number: 1.

[51] S. P. Jordan, N. Shutty, M. Wootters, A. Zalcman, A. Schmidhuber, R. King, S. V. Isakov, T. Khattar, and R. Babbush, Optimization by Decoded Quantum Interferometry (2024), version Number: 4.

[52] F. Sabater, O. E. Harzli, G.-J. Besjes, M. Erdmann, J. Klepsch, J. Hiltrop, J.-F. Bobier, Y. Cao, and C. A. Riofrio, Towards solving industrial integer linear programs with Decoded Quantum Interferometry (2025), version Number: 1.

[53] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, Strengths and Weaknesses of Quantum Computing, SIAM Journal on Computing **26**, 1510 (1997).

[54] M. Richardson and contributors, chebpy: A Python library for Chebyshev spectral methods (2016).