Towards Quantum Computational Mechanics

Burigede Liu[a,*], Michael Ortiz[b], Fehmi Cirak[a]

[a]*Department of Engineering, University of Cambridge, Cambridge, CB2 1PZ, UK*
[b]*Division of Engineering and Applied Science, California Institute of Technology, Pasadena, CA 91125, USA*

Abstract

The rapid advancements in quantum computing as ushered in a new era for computer simulations, presenting groundbreaking opportunities across diverse disciplines. Central to this revolution is the quantum processor's capacity to entangle qubits, unlocking unprecedented possibilities for addressing computational challenges on an extreme scale, far beyond the reach of classical computing. In this study, we explore how quantum computing can be employed to enhance computational mechanics. Our focus is on the analysis of Representative Volume Element (RVE) within the framework of multiscale solid mechanics. We introduce an innovative quantum algorithm designed to solve the RVE problem. This algorithm is capable of compute RVEs of discretization size N in $\mathcal{O}(\text{Poly}\log(N))$ time, thus achieving an exponential speed-up over traditional classical computing approaches that typically scales linearly with N. We validate our approach with case studies including the solution of one and two dimensional Poisson's equation, as well as an RVE of a composite bar with piece-wise constant phases. We provide quantum circuit designs that requires only $\mathcal{O}(\text{Poly}\log(N))$ universal quantum gates,underscoring the efficiency of our approach. Our work suggests a major way in which quantum computing can be combined with and brought to bear on computational mechanics.

Keywords: Quantum computing, multiscale analysis, quantum Fourier transform

Contents

[*] Corresponding author
Email addresses: bl377@cam.ac.uk (Burigede Liu), ortiz@aero.caltech.edu (Michael Ortiz), fc286@cam.ac.uk (Fehmi Cirak)

1. Introduction

The phenomenal advances in computational mechanics from its early beginnings in the late 1950s to today follow the exponential increase in computing power over the same period of time [23, 37]. This progress is most evident from the increase of finite element problem sizes from less than hundreds ($< 10^2$) to hundreds billions ($> 10^{11}$) of elements over the same time period [12, 26]. As widely known, Moore's law stipulates a two-fold increase in the density of transistors on a circuit every two years. The consensus is that future improvements in computing power will be limited because of the physical limitations in shrinking logic gate sizes further. A naive scaling up of current computing technologies without an increase is transistor densities is believed as highly problematic because of excessive cost and energy consumption. Using quantum systems to perform computations provides one possible solution to this impasse. Quantum computers were envisaged in early 1980s by Feynman [19], Benioff [5] and Manin [28]. Feynman was motivated by the impossibility of simulating a reasonably sized *quantum system* with classical computers. For instance, the state of a quantum system consisting of n_p particles contained in a box in $\mathbb{R}^3$ is described by a complex-valued *wave function* $\psi(\boldsymbol{x}) : \mathbb{R}^{3n_p} \mapsto \mathbb{C}$. The discretisation of this configuration space with only ten cells in each dimension leads to a discretised system with 10^{3n_p} degrees of freedom, rendering the problem unsolvable beyond a few particles.

Feynman argued to efficiently simulate quantum systems, one needs non-classical computers that must use phenomena unique to quantum mechanics. The same argument was put forward by Manin in his book in Russian; see [34]. In contrast to both, Benioff was concerned with energy dissipation in classical computers resulting from the irreversibility of the elementary operations. For instance, in a classical computer an AND gate has two binary inputs and one binary output and is irreversible as it is impossible to deduce from the single output the respective inputs. This irreversibility is associated with a loss of energy [6, 27]. Benioff proposed an alternative quantum-based classical computer design in which every operation is encoded as the solution of the reversible time-dependent Schrödinger equation governing quantum mechanics. The notion of a quantum computer, as envisaged by Feynman and Manin, was finally formalised in the seminal work by Deutsch [16] a few years later. In the same work, Deutsch discussed the advantages of a quantum computer in solving problems beyond the simulation of quantum systems, which led to an initial flurry of quantum algorithms, like Grover's algorithm of unstructured search [21] and Shor's algorithm for prime factorisation [39].

To develope an intuitive understanding of quantum computing, adopting a physical viewpoint of computing is expedient. Briefly, computing is always tied to a physical representation, and a computer is a machine. As Deutsch stated: *"a computing machine is any physical system whose dynamical evolution takes it from one of a set of 'input' states to one of a set of 'output' states"*. The input state of a quantum computer is chosen from the configuration space of a quantum system. According to the postulates of quantum mechanics, the configuration of the system with n quantum particles is described by a *state vector* $|q\rangle$, see [36].[1] The ket $|\cdot\rangle$ merely indicates that q is a quantum mechanical vector. Each particle is referred as a *qubit* and has two states. These two-states can be, for instance, the up and down spin states of a spin 1/2 particle, ground and excited states of an atom, or the horizontal and vertical polarisation of a photon [17]. Owing to *quantum entanglement*, the state vector of n qubits is the Hilbert space $\mathbb{C}^{2^n}$, i.e. $|q\rangle \in \mathbb{C}^{2^n}$. Notice that this space is exponentially larger than the configuration space of n classical particles which is only $\mathbb{C}^{2n}$, see [24]. The time evolution of the state vector $|q(t)\rangle$ is governed by the Schrödinger equation and the state vector $|q(T)\rangle$ at time $t = T$ is given by the unitary mapping $U_C : |q(0)\rangle \mapsto |q(T)\rangle$. The evolution operator $U_C \in \mathbb{C}^{2^n \times 2^n}$ is unitary, i.e. $U_C^{-1} = U_C^{\dagger}$, i.e. length preserving and bijective. A *universal quantum computer* can implement any such unitawhitry operator U_C. Hence, as a physical system a quantum computer performs computation by evolving a chosen $|q(0)\rangle$ to $|q(T)\rangle$ using a prudently designed unitary U_C. Therefore, quantum computers are sometimes referred as *analog machines* and their ability to simultaneously evolve the t state vector $|q(t)\rangle$ as *quantum parallelism*.

We provided so far a simplified and abstract description of quantum computing which requires further elaboration. In computational mechanics the initial state $|q(0)\rangle$ may represent the forcing, the final state $|q(T)\rangle$ the solution and U_C the discretised inverse solution operator. A classical forcing vector $\boldsymbol{q} \in \mathbb{C}^N$, with $N = 2^n$, is encoded as the components, usually called *amplitudes*, $q_i(0) \in \mathbb{C}$ of a state vector $|q(0)\rangle = \sum_i q_i(0)\,|i\rangle$ of a system with n qubits. The basis vectors $|i\rangle \in \{|0\rangle, |1\rangle, \dots |2^n - 1\rangle\}$ provide a labelling of the 2^n possible states of the quantum system. This encoding is referred to as *state preparation* and, for instance, can be accomplished with the unitary mapping $U_I(\boldsymbol{q})\colon\ |0\dots 0\rangle \mapsto |q(0)\rangle$. Here, $|0\dots 0\rangle$ is a known initial state and $U_I(\boldsymbol{q})$ a unitary matrix. Algorithms for state preparation can be found, e.g., in [3, 30, 38]. Subsequent application of the previously mentioned unitary U_C performs the desired computation. As widely studied, any unitary matrix can be constructed as the composition of a few elementary unitary matrices [4, 15]. These elementary unitaries are of size 2×2 and 4×4 and are applied to a single or two qubits as a time. In gate-based quantum computing the composition of the unitaries U_I and U_C from elementary unitaries, also referred to as gates, are visualised in circuit diagrams. On a more practical level, a quantum algorithm is implemented by designing the arrangement of the elementary gates in a quantum SDK, like Qiskit [13], Cirq [11], PyQuil [40] and Pennylane [7]. See also the textbooks [1, 25, 35, 46] on known quantum algorithms and their circuit implementations. The state vector $|q(T)\rangle$ after the completion of the computation is interrogated by *measurement*. According to the measurement postulate of quantum mechanics, the magnitude of the amplitude $|q_i|^2$ is equal to the probability to find the quantum system in the state labelled $|i\rangle$, see [36]. After the state is *collapsed* into the state $|i\rangle$ repeated measurements will yield the same result $|i\rangle$. More specifically, we can only infer the statistics of components of $|q(T)\rangle$ or, more generally, the statistics of the projection of $|q(T)\rangle$ to some subspace.

As detailed, quantum computing relies exclusively on unitary operations and measurement, so it is essential to reassess existing computational mechanics approaches for their suitability and develop new methods. In this paper, we consider the homogenisation of microstructured materials, like multi-phase composites consisting of materials with different properties or polycrystalline materials. The characteristic length scale of the microstructure is much smaller than the size of the macroscopic specimen to be analysed. Therefore, it is reasonable to derive the effective material properties, like diffusivity or Young's modulus, of the homogenised macroscopic boundary value problem from micro-level boundary value problems, fully considering the microstructure. In computational homogenisation, the micro- and macro-level boundary value problems are all suitably discretised and solved sequentially. The micro-level domain is usually a rectangular prism called the representative volume element (RVE). The boundary conditions for the RVE are chosen as periodic, and the prescribed average of the gradient of the solution, or strain gradient, is given by the macroscale boundary value problem. We solve the periodic microscale problem by discretising all fields using a band-limited Fourier interpolation, also known as Whittaker-Shannon interpolation, [10, 43]. In computational homogenisation, this approach was first pioneered by Moulinec and Suquet [31]; see also the recent review paper [20]. The resulting discrete system of equations is classically solved using a fixed-point iteration and the Fast Fourier

[1] The state vector $|q\rangle$ and wave function $\psi(\boldsymbol{x})$ are equivalent representations of a quantum system. In quantum computing usually only state vectors are used.

Transform (FFT). The FFT has the computational complexity $O(N \log N)$, where N is the number of degrees of freedom. The corresponding quantum algorithm, the Quantum Fourier Transform (QFT), has remarkably the polylog complexity $O((\log N)^2)$, i.e. a polynomial of $\log N$, see [14]. The solution of the RVE problem on a quantum computer requires, beyond QFT and the already mentioned state preparation, the implementation of simple algebraic operations, like the division by a scalar, in the Fourier space. To this end, we first approximate functions by piecewise polynomials using Chebyshev interpolation and subsequently encode the polynomials utilising a sequence of elementary rotation gates [41, 45]. Furthermore, we propose a quantum version of the fixed-point iteration to minimise the number of costly measurement operations.

2. Computational homogenisation

We review the computational homogenisation of periodic composites with a scalar solution field governed by the non-homogenous Poisson equation [29, 31, 32]. Such problems arise, for instance, in electrical and heat conduction, flow in porous media or antiplane elasticity. In computational homogenisation the solution field is determined by solving iteratively a sequence of macroscopic specimen level and microstructure level problems. The gradient of the solution field of the specimen-level problem serves as the input to the microstructure-level problems. In turn, the specimen level fluxes are obtained by averaging the microstructure level fluxes.

2.1. Problem formulation

The microstructure-level problem is given by the representative volume element (RVE). We refer to this problem as an RVE even in the one and two-dimensional case as typical in the literature. We assume an RVE domain in the shape of the square $\square = (0, L)^2 \in \mathbb{R}^2$ with an edge length of L, and the coordinates of the points $\boldsymbol{x} \in \square$ are denoted by $\boldsymbol{x} = (x_0 \quad x_1)^\mathsf{T}$. Note that the microstructure level problem is usually referred to as an RVE even in case of one and two-dimensional problems. We consider, for the sake of concreteness, an antiplane elasticity problem and name our variables accordingly. The shear modulus $\mu(\boldsymbol{x})$ in the RVE is periodic, i.e.,

$$\mu(x_0, x_1) = \mu(x_0 + L, x_1) = \mu(x_0, x_1 + L) . \tag{1}$$

Furthermore, the RVE is subject to the uniform average strain vector $\overline{\boldsymbol{\gamma}} \in \mathbb{R}^2$ determined by solving the macroscopic specimen level problem. The respective deformation of the RVE is characterised by a scalar transverse displacement field $u(\boldsymbol{x}) \in \mathbb{R}$ with the shear strain vector

$$\boldsymbol{\gamma}(\boldsymbol{x}) = \nabla u(\boldsymbol{x}) , \tag{2}$$

where ∇ denotes the gradient operator and $\gamma(\boldsymbol{x}) \in \mathbb{R}^2$. The displacement field $u(\boldsymbol{x}) \in \mathbb{R}$ consists of an affine component due to the applied $\overline{\boldsymbol{\gamma}}$ and a fluctuating component $v(\boldsymbol{x})$ so that

$$u(\boldsymbol{x}) = \overline{\boldsymbol{\gamma}} \cdot \boldsymbol{x} + v(\boldsymbol{x}) , \tag{3}$$

which implies the strain decomposition

$$\boldsymbol{\gamma}(\boldsymbol{x}) = \overline{\boldsymbol{\gamma}} + \nabla v(\boldsymbol{x}) . \tag{4}$$

The fluctuating displacement $v(\boldsymbol{x}) \in \mathbb{R}$ must satisfy the periodicity condition

$$v(x_0, x_1) = v(x_0 + L, x_1) = v(x_0, x_1 + L) . \tag{5}$$

The decomposition (4) and the periodicity condition (5) ensure that the applied $\overline{\boldsymbol{\gamma}}$ is indeed the average strain

$$\frac{1}{L^2} \int_0^L \int_0^L \gamma \, \mathrm{d}x_0 \, \mathrm{d}x_1 = \frac{1}{L^2} \int_0^L \int_0^L (\overline{\boldsymbol{\gamma}} + \nabla v(\boldsymbol{x})) \, \mathrm{d}x_0 \, \mathrm{d}x_1 = \overline{\boldsymbol{\gamma}} . \tag{6}$$

The stress field vector $\boldsymbol{\sigma}(\boldsymbol{x}) \in \mathbb{R}^2$ of the micro-structure level satisfies the equilibrium equation

$$\nabla \cdot \boldsymbol{\sigma}(\boldsymbol{x}) = 0\,, \tag{7}$$

and the constitutive equation

$$\boldsymbol{\sigma}(\boldsymbol{x}) = \mu(\boldsymbol{x})\boldsymbol{\gamma}(\boldsymbol{x}) = \mu(\boldsymbol{x})\left(\overline{\boldsymbol{\gamma}} + \nabla v(\boldsymbol{x})\right)\,. \tag{8}$$

Hence, the boundary value problem for micro-structure can be summarised as

$$\nabla \cdot (\mu(\boldsymbol{x})\nabla v(\boldsymbol{x})) + \overline{\boldsymbol{\gamma}} \cdot \nabla\mu(\boldsymbol{x}) = 0\,. \tag{9}$$

To ensure the periodicity of the displacement field $v(\boldsymbol{x})$ the Dirichlet boundary conditions must be chosen such tat $v(x_0,\, 0) = v(x_0,\, L)$ and $v(0,\, x_1) = v(L,\, x_1)$.

It is not possible to solve the boundary value problem (11) using a Fourier-based approach because of the non-constant $\mu(\boldsymbol{x})$. Therefore, we first introduce a constant reference shear modulus μ^0 and rewrite the constitutive equation as (8) as

$$\boldsymbol{\sigma}(\boldsymbol{x}) = (\mu(\boldsymbol{x}) - \mu^0)\gamma(\boldsymbol{x}) + \mu^0\gamma(\boldsymbol{x}) = \boldsymbol{\tau}(\boldsymbol{x}) + \mu^0\gamma(\boldsymbol{x})\,, \tag{10}$$

where $\boldsymbol{\tau}(\boldsymbol{x})$ is referred to as the polarisation stress. Subsequently, we use the equilibrium (7) to define the fixed point iteration

$$\mu^0\nabla \cdot \nabla v^{(s+1)}(\boldsymbol{x}) + \nabla \cdot \boldsymbol{\tau}^{(s)}(\boldsymbol{x}) = 0\,. \tag{11}$$

The polarisation stress $\boldsymbol{\tau}^{(s)}$ at iteration step s depends on the known displacement $v^{(s)}$ and the applied strain $\overline{\boldsymbol{\gamma}}$.

2.2. *Periodic solutions*

In preparation for the Fourier discretisation of the incremental RVE problem, we consider a polarisation stress vector consisting of a single wave with a period L, (spatial) frequency vector $\boldsymbol{\xi} = (\xi_0 \quad \xi_1)^{\mathsf{T}}$ with $\xi_0 = \xi_1 = 2\pi/L$ and a complex amplitude $\hat{\boldsymbol{\tau}} \in \mathbb{C}^2$, i.e.,

$$\boldsymbol{\tau}(\boldsymbol{x}) = \hat{\boldsymbol{\tau}}(\boldsymbol{\xi})e^{i\boldsymbol{\xi}\cdot\boldsymbol{x}}\,, \tag{12}$$

where $i^2 = -1$ and $\boldsymbol{\xi} \cdot \boldsymbol{x} = \xi_0 x_0 + \xi_1 x_1$. The corresponding solution will be a single wave with the same frequency vector $\boldsymbol{\xi}$ but a yet unknown amplitude $\hat{v}(\boldsymbol{\xi}) \in \mathbb{C}^2$:

$$v(\boldsymbol{x}) = \hat{v}(\boldsymbol{\xi})e^{i\boldsymbol{\xi}\cdot\boldsymbol{x}}\,. \tag{13}$$

Note that the components of $\boldsymbol{\tau}(\boldsymbol{x})$ and $v(\boldsymbol{x})$ can have different phases given that the amplitudes $\hat{\boldsymbol{\tau}}$ and $\hat{v}(\boldsymbol{\xi})$ are complex valued. Introducing the expressions (12) and (13) into (11), while ignoring the iteration counter s, yields the amplitude of the solution,

$$-\mu^0(\boldsymbol{\xi}\cdot\boldsymbol{\xi})\hat{v}(\boldsymbol{\xi}) + i\boldsymbol{\xi}\cdot\hat{\boldsymbol{\tau}}(\boldsymbol{\xi}) = 0 \quad \Rightarrow \quad \hat{v}(\boldsymbol{\xi}) = \frac{1}{\mu^0}\frac{i\boldsymbol{\xi}\cdot\hat{\boldsymbol{\tau}}(\boldsymbol{\xi})}{\boldsymbol{\xi}\cdot\boldsymbol{\xi}}\,. \tag{14}$$

Therefore, the solution of the RVE problem is a wave given by

$$v(\boldsymbol{x}) = \frac{1}{\mu_0}\frac{i\boldsymbol{\xi}\cdot\hat{\boldsymbol{\tau}}(\boldsymbol{\xi})}{\boldsymbol{\xi}\cdot\boldsymbol{\xi}}e^{i\boldsymbol{\xi}\cdot\boldsymbol{x}}\,. \tag{15}$$

Differentiating the solution vector yields according to (2) the strain

$$\boldsymbol{\gamma}(\boldsymbol{x}) = -\frac{1}{\mu_0}\boldsymbol{\xi}\frac{\boldsymbol{\xi}\cdot\hat{\boldsymbol{\tau}}(\boldsymbol{\xi})}{\boldsymbol{\xi}\cdot\boldsymbol{\xi}}e^{i\boldsymbol{\xi}\cdot\boldsymbol{x}} = \hat{\boldsymbol{\Gamma}}(\boldsymbol{\xi})\cdot\hat{\boldsymbol{\tau}}(\boldsymbol{\xi})e^{i\boldsymbol{\xi}\cdot\boldsymbol{x}}\,, \tag{16}$$

where $\mathbf{\Gamma}(\boldsymbol{\sigma}) \in \mathbb{R}^2$.

2.3. *Fourier discretisation*

We approximate on the bounded RVE domain $\square = (0, L)^2 \subset \mathbb{R}^2$ periodic functions with band-limited interpolation, also referred to as Whittaker-Shannon interpolation [43]. The band-limited interpolation of the solution field is defined as

$$v(\boldsymbol{x}) \approx v^h(\boldsymbol{x}) = \frac{1}{N} \sum_{k^0=0}^{N-1} \sum_{k^1=0}^{N-1} \hat{v}_{k^0 k^1} e^{i \boldsymbol{\xi}_{k^0 k^1} \cdot \boldsymbol{x}} = \frac{1}{N} \sum_{k^0=0}^{N-1} \sum_{k^1=0}^{N-1} \hat{v}_{\boldsymbol{k}} e^{i \boldsymbol{\xi}_{\boldsymbol{k}} \cdot \boldsymbol{x}} , \tag{17}$$

with the multi-index $\boldsymbol{k} = (k^0, k^1)$ and the frequency vector $\boldsymbol{\xi}_{\boldsymbol{k}} = (\xi_{0,k^0} \quad \xi_{1,k^1})^\mathsf{T}$; hence $\boldsymbol{\xi}_{\boldsymbol{k}} \cdot \boldsymbol{x} = \xi_{0,k^0} x_0 + \xi_{1,k^1} x_1$. As will be explained shortly, the multi-index $\boldsymbol{k}$ can be identified as grid point labels of the RVE domain discretisation. The components of the frequency vector for a specific $\boldsymbol{k}$ are given by

$$\xi_{0,k^0} = \frac{2\pi r(k^0)}{L} , \quad \xi_{1,k^1} = \frac{2\pi r(k^1)}{L} . \tag{18}$$

Here and elsewhere in the paper $N \geq 2$ is an even integer. N is an integer because $v(\boldsymbol{x})$ is required to be periodic. The function $r(l)$ is introduced to relabel the frequency components as

$$r(k^0) = \begin{cases} k^0 & 0 \leq k^0 < N/2 \\ k^0 - N & N/2 \leq k^0 < N ; \end{cases} \tag{19}$$

see also [43, Ch. 3] and [9, Ch. 2].

To determine the amplitudes $\hat{v}_{k^0 k^1}$ from the function values $v(\boldsymbol{x})$, we introduce a discretisation consisting of $N \times N$ cells of size $h \times h$, i.e. $h = L/N$, and the grid point coordinates

$$\boldsymbol{x}_{\boldsymbol{k}} \equiv \boldsymbol{x}_{k^0 k^1} = \frac{L}{N} \begin{pmatrix} k^0 \\ k^1 \end{pmatrix} = h \begin{pmatrix} k^0 \\ k^1 \end{pmatrix} , \quad k^0 = 0, \ldots, N-1 , \quad k^1 = 0, \ldots, N-1 . \tag{20}$$

We denote the function values at the grid points with $v_{\boldsymbol{k}} = v(\boldsymbol{x}_{\boldsymbol{k}})$. According to (17), the function values at the grid points are given by

$$v^h_{k^0 k^1} = \frac{1}{N} \sum_{j^0=0}^{N-1} \sum_{j^1=0}^{N-1} \hat{v}_{j^0 j^1} e^{i \left(\xi_{0,j^0} k^0 h + \xi_{1,j^1} k^1 h \right)} = \frac{1}{N} \sum_{j^0=0}^{N-1} \sum_{j^1=0}^{N-1} \hat{v}_{j^0 j^1} e^{i \frac{2\pi}{N} \left(r(j^0) k^0 + r(j^1) k^1 \right)} . \tag{21}$$

It is easy to show that this equation represents the inverse DFT of $\hat{v}_{k^0 k^1}$, i.e.,

$$v^h_{k^0 k^1} = \frac{1}{N} \sum_{j^0=0}^{N-1} \sum_{j^1=0}^{N-1} \hat{v}_{j^0 j^1} e^{i \frac{2\pi}{N} \left(j^0 \cdot k^0 + j^1 \cdot k^1 \right)} . \tag{22}$$

The inverse of this mapping, i.e. the DFT of v^h_{jk}, takes the form

$$\hat{v}_{j^0 j^1} - \frac{1}{N} \sum_{k^0=0}^{N-1} \sum_{k^1=0}^{N-1} v^h_{k^0 k^1} e^{-i \frac{2\pi}{N} \left(j^0 \cdot k^0 + j^1 \cdot k^1 \right)} . \tag{23}$$

As will be detailed in Section 4.5, the DFT and its inverse can be very efficiently computed on a quantum computer using the QFT algorithm.

Clearly, all periodic fields appearing in the RVE problem (11) can be approximated using band-limited interpolation. The band-limited approximation of a periodic field consists of a linear combination, or superposition, of waves with N^2 different frequencies. The RVE solution for each wave component is the same as the single-wave

Algorithm 1 Fourier-based iterative solution of the RVE problem on $\square = (0,\, L)^2 \subset \mathbb{R}^2$.

```
Initialization:
$\boldsymbol{\gamma}_{\boldsymbol{k}}^{(s=0)} \equiv \boldsymbol{\gamma}_{k^0k^1}^{(s=0)} = \bar{\boldsymbol{\gamma}}$
$\boldsymbol{\sigma}_{\boldsymbol{k}}^{(s=0)} \equiv \boldsymbol{\sigma}_{k^0k^1}^{(s=0)} = \mu(\boldsymbol{x}_{\boldsymbol{k}})\boldsymbol{\gamma}_{\boldsymbol{k}}^{(0)} \quad \forall\, \boldsymbol{x}_{\boldsymbol{k}} \equiv \boldsymbol{x}_{k^0k^1} \in \square$

Iteration: $\boldsymbol{\gamma}_{\boldsymbol{k}}^{(s)}$ and $\boldsymbol{\sigma}_{\boldsymbol{k}}^{(s)}$ at every grid point $\boldsymbol{k} = (k^0,\, k^1)$ are $\boldsymbol{x}_{\boldsymbol{k}}$ known
(a)    $\boldsymbol{\tau}_{\boldsymbol{k}}^{(s)} = \boldsymbol{\sigma}_{\boldsymbol{k}}^{(s)} - \mu^0\boldsymbol{\gamma}_{\boldsymbol{k}}^{(s)} = \left(\mu(\boldsymbol{x}_{\boldsymbol{k}}) - \mu^0\right)\boldsymbol{\gamma}_{\boldsymbol{k}}^{(s)}$
(b)    $\hat{\boldsymbol{\tau}}_{\boldsymbol{j}}^{(s)} = \mathrm{DFT}(\boldsymbol{\tau}_{\boldsymbol{k}}^{(s)})$
(c)    Convergence  test
(d)    $\hat{\boldsymbol{\gamma}}_{\boldsymbol{j}}^{(s+1)} = \begin{cases} -\hat{\boldsymbol{\Gamma}}_{\boldsymbol{j}} \cdot \hat{\boldsymbol{\tau}}_{\boldsymbol{j}}^{(s)} & \text{if } j^0 \neq N/2 \text{ and } j^1 \neq N/2 \\ \sqrt{N}\bar{\boldsymbol{\gamma}} & \text{if } j^0 = j^1 = N/2 \end{cases}$
(e)    $\boldsymbol{\gamma}_{\boldsymbol{k}}^{(s+1)} = \mathrm{DFT}^{-1}\left(\hat{\boldsymbol{\gamma}}_{\boldsymbol{j}}^{(s+1)}\right)$
(f)    $\boldsymbol{\sigma}_{\boldsymbol{k}}^{(s+1)} = \mu(\boldsymbol{x}_{\boldsymbol{k}})\boldsymbol{\gamma}_{\boldsymbol{k}}^{(s+1)}$
(g)    $s \leftarrow s + 1$
```

solution (24) derived in the previous section. The entire solution is obtained as the superpositon of the solutions of the N^2 frequencies, i.e.,

$$v^h(\boldsymbol{x}) = \frac{1}{\mu_0} \sum_{k^0=0}^{N-1} \sum_{k^1=0}^{N-1} \frac{i\boldsymbol{\xi}_{\boldsymbol{k}} \cdot \hat{\boldsymbol{\tau}}_{\boldsymbol{k}}}{\boldsymbol{\xi}_{\boldsymbol{k}} \cdot \boldsymbol{\xi}_{\boldsymbol{k}}} e^{i\boldsymbol{\xi}_{\boldsymbol{k}} \cdot \boldsymbol{x}} \,. \tag{24}$$

Finally, the sketched solution approach can be used for solving the homogenisation boundary value problem problem (11) with a non-homogenous shear modulus as detailed in Algorithm 1.

3. Essentials of quantum computing

In this Section we provide a summary of the foundations of the quantum information processing covering aspects of quantum mechanics and computing. For a more comprehensive review we refer to textbooks [1, 25, 35, 46] and tutorial paper [2].

3.1. Notation and definitions

We use in this paper both the Dirac notation from quantum mechanics, also known as bra-ket notation, and the matrix notation familiar from linear algebra. As can be glanced from Table 1 the two notations widely mirror each other.

3.2. Quantum systems

In quantum computing data is encoded via the state of a quantum system consisting of one or more qubits, i.e. quantum particles, with two different distinct states. In physical terms, the two states of a qubit can be realised, e.g., as the polarisation of a photon or two energy levels of an electron. In contrast, the two states of a classical particle, like a coin, are its head and tail states.

3.2.1. One-qubit systems

The state $|q\rangle \in \mathbb{C}^2$ of a single qubit is expressed as the *superposition*, or linear combination, of two states abstractly labelled as $\{|0\rangle\,, |1\rangle\}$ in Dirac notation or $\{\boldsymbol{e}_0 = (1\;\;0)^\mathsf{T},\, \boldsymbol{e}_1 = (0\;\;1)^\mathsf{T}\}$ in matrix notation, i.e.,

$$|q\rangle = q_0\,|0\rangle + q_1\,|1\rangle = q_0\begin{pmatrix}1\\0\end{pmatrix} + q_1\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}q_0\\q_1\end{pmatrix}, \tag{25}$$

	Matrix notation	Dirac notation	Examples / Notes
Scalar	$z \in \mathbb{C}$	$z \in \mathbb{C}$	
Complex conjugate of z	$z^* \in \mathbb{C}$	$z^* \in \mathbb{C}$	$(1+i)^* = (1-i)$
Modulus of z	$\lvert z\rvert \in \mathbb{R}$	$\lvert z\rvert \in \mathbb{R}$	$\lvert z\rvert = \sqrt{z^* z} = \sqrt{zz^*}$
Vector (or, a ket)	$\boldsymbol{q} \in \mathbb{C}^N$	$\lvert q\rangle \in \mathbb{C}^N$	Components: q_j $j = \{0, 1, N-1\}$
Dual vector (or, the bra) of $\boldsymbol{q}$	$\boldsymbol{q}^\dagger \in \mathbb{C}^N$	$\langle q\rvert \in \mathbb{C}^N$	$\boldsymbol{q}^\dagger = (\boldsymbol{q}^\mathsf{T})^*$
Inner product between $\boldsymbol{q}$ and $\boldsymbol{r}$	$\boldsymbol{q} \cdot \boldsymbol{r}$	$\langle q\vert r\rangle$	$\boldsymbol{q} \cdot \boldsymbol{r} = \boldsymbol{q}^\dagger \boldsymbol{r}$
Kronecker product	$\boldsymbol{q} \otimes \boldsymbol{r} \in \mathbb{C}^{N\times N}$	$\lvert q\rangle \otimes \lvert r\rangle \in \mathbb{C}^{N\times N}$	$\lvert q\rangle \otimes \lvert r\rangle \equiv \lvert q\rangle \lvert r\rangle \equiv \lvert qr\rangle$
Matrix	$\boldsymbol{A} \in \mathbb{C}^{N\times N}$	$A \in \mathbb{C}^{N\times N}$	Components: A_{jk} $j, k = \{0, 1, N-1\}$
Hermitian conjugate of A	$\boldsymbol{A}^\dagger \in \mathbb{C}^{N\times N}$	$A^\dagger \in \mathbb{C}^{N\times N}$	$\boldsymbol{A}^\dagger = (A^\mathsf{T})^* = (A^*)^\mathsf{T}$
Inner product between $\boldsymbol{q}$ and $\boldsymbol{Ar}$	$\boldsymbol{q}^\dagger \boldsymbol{A} \boldsymbol{r}$	$\langle q\vert A\vert r\rangle$	$\langle q\vert A\vert r\rangle^* = \langle r\vert A\vert q\rangle$
Standard basis vector (or, the computational basis)	$\boldsymbol{e}_0, \boldsymbol{e}_1, \dots, \boldsymbol{e}_{N-1}$	$\lvert 0\rangle, \lvert 1\rangle, \dots, \lvert N-1\rangle$	$\boldsymbol{e}_0 = (1 \;\; 0 \;\; \dots \;\; 0)^\mathsf{T}$ $\boldsymbol{e}_1 = (0 \;\; 1 \;\; \dots \;\; 0)^\mathsf{T}$

Table 1: Summary of the definitions and notation used in this paper.

where $q_0, q_1 \in \mathbb{C}$ are two coefficients called *amplitudes*. For comparison, the same state $|p\rangle$ expressed in matrix notation reads

$$\boldsymbol{q} = q_0\boldsymbol{e}_0 + q_1\boldsymbol{e}_1 = q_0\begin{pmatrix}1\\0\end{pmatrix} + q_1\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}q_0\\q_1\end{pmatrix} . \tag{26}$$

The kets $|0\rangle$ and $|1\rangle$, or $\boldsymbol{e}_0$ and $\boldsymbol{e}_1$, are two basis vectors which are referred to as the *computational basis*.

The state vector $|q\rangle$ expressed in a different orthonormal basis (spanning the same Hilbert space $\mathbb{C}^2$) will have different coefficients. Assuming that the measuring device can measure the two states $\{|0\rangle, |1\rangle\}$, the qubit will be found either in state $|0\rangle$ or $|1\rangle$ when observed. In quantum mechanics terminology, the state $|q\rangle$ will *collapse* either into state $|0\rangle$ or $|1\rangle$. According to the postulates of quantum mechanics, specifically the *Born rule*, the probability to observe $|0\rangle$ is

$$p(|0\rangle) = |\langle q|0\rangle|^2 = |q_0^* \langle 0|0\rangle + q_1^* \langle 1|0\rangle|^2 = |q_0^*|^2 = |q_0|^2 . \tag{27}$$

Note that $\langle q| = q_0^* \langle 0| + q_1^* \langle 1|$ according to Table 1. Similarly, the probability to observe $|1\rangle$ is $p(|1\rangle) = |q_1|^2$. The two probabilities must satisfy $|q_1|^2 + |q_2|^2 = 1$. Once the qubit has been observed, i.e. the state has collapsed, it will maintain its observed state and repeated measurements will yield the same result.

As will be detailed in Section 3.4, we note that the two states $|0\rangle$ and $|1\rangle$ are a property of the measuring device rather than the quantum system. For instance, a measuring device that can measure only the orthogonal states $\{|r\rangle, |r^\perp\rangle\}$ will find the qubit in one these two states. The respective probabilities are determined by replacing $|0\rangle$ with $|r\rangle$ or $|r^\perp\rangle$ in (27). Hence, the representation of the intrinsic qubit state $|q\rangle$ in a specific basis is a matter of choice by the observer.

3.2.2. *Multi-qubit systems*

To begin with, we consider a two-qubit system ($n = 2$) with the state vector $|q\rangle \in \mathbb{C}^{2n}$. The elements of the respective 2^n dimensional basis are given by the Kronecker product of the basis states $\{|0\rangle, |1\rangle\}$ for each of the qubits, i.e.,

$$\{|0\rangle \otimes |0\rangle\,,\ |0\rangle \otimes |1\rangle\,,\ |0\rangle \otimes |1\rangle\,,\ |1\rangle \otimes |1\rangle\}\,. \tag{28}$$

These four basis states are often abbreviated as

$$|0\rangle \otimes |0\rangle \equiv |0\rangle\,|0\rangle \equiv |00\rangle\ , \quad |0\rangle \otimes |1\rangle \equiv |0\rangle\,|1\rangle \equiv |01\rangle\ , \quad |1\rangle \otimes |0\rangle \equiv |1\rangle\,|0\rangle \equiv |10\rangle\ , \quad |1\rangle \otimes |1\rangle \equiv |1\rangle\,|1\rangle \equiv |11\rangle\ . \tag{29}$$

Converting the four binary numbers 00, 01, 10 and 11 to the decimals 0, 1, 2 and 3, respectively, yields an alternative labelling for the states

$$|0\rangle \equiv |0\rangle \otimes |0\rangle\ , \quad |1\rangle \equiv |0\rangle \otimes |1\rangle\ , \quad |2\rangle \equiv |1\rangle \otimes |0\rangle\ , \quad |3\rangle \equiv |1\rangle \otimes |1\rangle\ . \tag{30}$$

We obtain after evaluating the Kronecker products

$$|0\rangle = \begin{pmatrix} 1\begin{pmatrix}1\\0\end{pmatrix} \\ 0\begin{pmatrix}1\\0\end{pmatrix} \end{pmatrix} = \begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 1\begin{pmatrix}0\\1\end{pmatrix} \\ 0\begin{pmatrix}0\\1\end{pmatrix} \end{pmatrix} = \begin{pmatrix}0\\1\\0\\0\end{pmatrix}, \quad |2\rangle = \begin{pmatrix} 0\begin{pmatrix}1\\0\end{pmatrix} \\ 1\begin{pmatrix}1\\0\end{pmatrix} \end{pmatrix} = \begin{pmatrix}0\\0\\1\\0\end{pmatrix}, \quad |3\rangle = \begin{pmatrix} 1\begin{pmatrix}0\\1\end{pmatrix} \\ 0\begin{pmatrix}0\\1\end{pmatrix} \end{pmatrix} = \begin{pmatrix}0\\0\\0\\1\end{pmatrix}. \tag{31}$$

Using these basis vectors, the state $|q\rangle \in \mathbb{C}^4$ of a two-qubit system is given by

$$|q\rangle = \sum_{k=0}^{3} q_k\,|k\rangle = \begin{pmatrix}q_0\\q_1\\q_2\\q_3\end{pmatrix}, \tag{32}$$

with the coefficients $q_k \in \mathbb{C}$. The coefficients q_k must be normalised as the probability to observe the qubit in state $|k\rangle$ is given by $p(|k\rangle) = |q_k|^2$.

It bears emphasis that the Kronecker product of two one-qubit states is different from the two-qubit state (32). The Kronecker product of two one-qubit states $|r\rangle$ and $|s\rangle$ yields the state

$$|r\rangle \otimes |s\rangle = (r_0\,|0\rangle + r_1\,|1\rangle) \otimes (s_0\,|0\rangle + s_1\,|1\rangle) = \begin{pmatrix}r_0 s_0\\r_0 s_1\\r_1 s_0\\r_1 s_1\end{pmatrix}. \tag{33}$$

The Kronecker product of two one-qubit states cannot express every valid quantum state such as the state

$$|q\rangle = \frac{1}{\sqrt{2}}\,|0\rangle \otimes |0\rangle + \frac{1}{\sqrt{2}}\,|1\rangle \otimes |1\rangle = \begin{pmatrix}1/\sqrt{2}\\0\\0\\1/\sqrt{2}\end{pmatrix}, \tag{34}$$

which requires $r_0 s_0 = r_1 s_1 = 1/\sqrt{2}$, but at the same time $r_0 s_1 = 0$ and $r_1 s_0 = 0$. The states that cannot be represented as the Kronecker product of one-qubit states are referred to as *entangled states*. The non-entangled states are the *separable states* or *product states*.

Similar to the two-qubit case the state of n qubits is given by

$$|q\rangle = \sum_{k=0}^{2n-1} q_k\,|k\rangle = \begin{pmatrix}q_0\\q_1\\\vdots\\q_{2n-1}\end{pmatrix}, \tag{35}$$

where $q_k \in \mathbb{C}$. The index k is frequently expressed as a binary

$$k = k_0 2^{n-1} + k_1 2^{n-2} + \ldots + k_{n-2} 2^1 + k_{n-1} 2^0 \equiv k_0 k_1 \ldots k_{n-1} k_n \,, \tag{36}$$

so that (35) can be written as

$$|q\rangle = \sum_{k=0}^{2n-1} q_k \, |k\rangle = \sum_{k_0=0}^{1} \sum_{k_1=0}^{1} \cdots \sum_{k_{n-2}=0}^{1} \sum_{k_{n-1}=0}^{1} q_{k_0 k_1 \ldots k_{n-2} k_{n-1}} \, |k_0 k_1 \ldots k_{n-2} k_{n-1}\rangle \ . \tag{37}$$

Moreover, similar to two qubit states each of the states $|k\rangle$ are obtained as the Kronecker products of single qubit states which becomes evident when writing

$$|k_0 k_1 \ldots k_{n-2} k_{n-1}\rangle \equiv |k_0\rangle \, |k_1\rangle \ldots |k_{n-2}\rangle \, |k_{n-1}\rangle \equiv |k_0\rangle \otimes |k_1\rangle \otimes \ldots \otimes |k_{n-2}\rangle \otimes |k_{n-1}\rangle \ . \tag{38}$$

The inevitability of entangled states is even more apparent in case of $n > 2$. Each separate qubit has two coefficients so that a separable state with n qubits can have only up to $2n$ distinct coefficients, in contrast an entangled state can have up to 2^n distinct coefficients. Without going into details, we note that the entanglement property of quantum systems is in addition to their superposition property fundamentally different from classical systems such as a system with n binary coins with the states head and tail [24].

3.3. Quantum gates and circuits

At the most abstract level, a quantum computer maps a state $|q\rangle$ to a new state $U\,|q\rangle$ for a given unitary matrix U. The coefficients of the state $|q\rangle$ are the input and the coefficients of $U\,|q\rangle$ are the output. The unitarity requirement, $U^\dagger = U^{-1}$, is dictated by quantum mechanics and ensures that the mapped state remains normalised. In the *circuit model* of quantum computing, the unitary matrix U is composed of a sequence of smaller elementary unitary matrices acting only one or a few qubits at a time. This approach is possible because the product of unitary matrices is also unitary. The elementary unitary matrices correspond in the *quantum circuit* to *gates* so that we refer to elementary matrices interchangeably as gates. This section provides a summary of the mostly widely used gates and discusses how they are combined to quantum circuits implementing the unitary transformation U.

3.3.1. One-qubit gates

The three most common one-qubit gates are the Pauli gates

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \tag{39}$$

and the unitary gate

$$I = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{40}$$

Proposition 1. *Any unitary matrix $U \in \mathbb{C}^{2\times 2}$, or one-qubit gate, can be expressed as the linear combination of the four unitary matrices, or gates, $\{I, X, Y, Z\}$.*

The Pauli gates are Hermitian and involutory (self inverse). As an example, the X gate applied twice to a one-qubit state yields, owing to the involutory property,

$$X \begin{pmatrix} q_0 \\ q_1 \end{pmatrix} = \begin{pmatrix} q_1 \\ q_0 \end{pmatrix}; \quad X \begin{pmatrix} q_1 \\ q_0 \end{pmatrix} = \begin{pmatrix} q_0 \\ q_1 \end{pmatrix} . \tag{41}$$

When exponentiated the Pauli matrices yield the three rotation gates

$$R_X(\theta) = e^{-i\theta X/2} = , \quad R_Y(\theta) = e^{-i\theta Y/2} \,, \quad R_Z(\theta) = e^{-i\theta Z/2} \,, \tag{42}$$

which evaluate to

$$R_X(\theta) = \begin{pmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix}, \quad R_Y(\theta) = \begin{pmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix}, \quad R_Z(\theta) = \begin{pmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{pmatrix}. \tag{43}$$

Proposition 2. *The exponential $e^{i\theta A}$ of a Hermitian matrix $A = A^\dagger$ is a unitary matrix and is equal to*

$$e^{i\theta A} = I\cos(\theta) + iA\sin(\theta). \tag{44}$$

Other widely used one-qubit gates include the Hadamard and the phase gates,

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad P(\theta) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}, \tag{45}$$

respectively.

3.3.2. Multi-qubit gates

We first consider gates defined as the Kronecker products of one-qubit gates. To this end, note that the Kronecker product of two or more unitary matrices is a unitary as well. By way of example, consider the two-qubit gate

$$X \otimes Y = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix} = \begin{pmatrix} 0\begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix} & 1\begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix} \\ 1\begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix} & 0\begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & i \\ 0 & 0 & -i & 0 \\ 0 & i & 0 & 0 \\ -i & 0 & 0 & 0 \end{pmatrix}. \tag{46}$$

If this gate is applied to a non-entangled product state $|r\rangle \otimes |s\rangle$ it is equivalent to applying first X to $|r\rangle$ and Y to $|s\rangle$ and then taking their Kronecker product, i.e.,

$$(X \otimes Y)(|r\rangle \otimes |s\rangle) = (X|r\rangle) \otimes (Y|s\rangle). \tag{47}$$

Hence, starting with a product multi-qubit state applying only single qubit gates the state will remain a non-entangled product state.

To achieve entanglement a controlled-not or *CNOT* is required. The *CNOT* gate is defined as

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{48}$$

In studying the action of the *CNOT* gate, or any other gate for that matter, on a state

$$|q\rangle = \sum_{k=0}^{3} q_k |k\rangle = \sum_{k_0=0}^{1}\sum_{k_1=0}^{1} q_{k_0k_1} |k_0k_1\rangle, \tag{49}$$

it is sufficient to focus on its action on a single basis elements $|k\rangle = |k_0k_1\rangle$. This is possible because of the linearity of quantum transformations. According (48), the *CNOT* gate maps the four basis elements as follows

$$CNOT: \ |0\rangle\otimes|0\rangle \mapsto |0\rangle\otimes|0\rangle; \ |0\rangle\otimes|1\rangle \mapsto |0\rangle\otimes|1\rangle; \ |1\rangle\otimes|0\rangle \mapsto |1\rangle\otimes|1\rangle; \ |1\rangle\otimes|1\rangle \mapsto |1\rangle\otimes|0\rangle. \tag{50}$$

Here, the left qubit k_0 is the *control* and the right qubit k_1 is the *target.* Only when the control qubit k_1 is in state $|1\rangle$ the target qubit k_0 is flipped else its state is maintained.

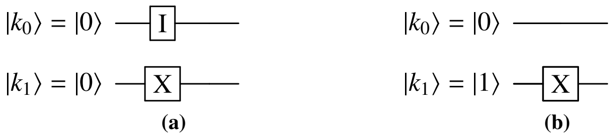

$|k_0\rangle = |0\rangle$
I
$|k_1\rangle = |0\rangle$
X
(a)
$|k_0\rangle = |0\rangle$
$|k_1\rangle = |1\rangle$
X
(b)

Figure 1: A basic quantum circuit with two qubits k_0 and k_1. As indicated in (b) the identity gate I is usually omitted in circuit diagrams so that circuits (a) and (b) are equivalent. The states of both qubits are initialised with $|k_0\rangle = |0\rangle$ and $|k_1\rangle = |0\rangle$ so that $|q\rangle = |k_0\rangle \otimes |k_1\rangle = |0\rangle \otimes |0\rangle$ which is abbreviated as $|q\rangle = |00\rangle$. The output of the circuit is $(I \otimes X)(|0\rangle \otimes |0\rangle) = I\,|0\rangle \otimes X\,|0\rangle = |0\rangle \otimes |1\rangle$.

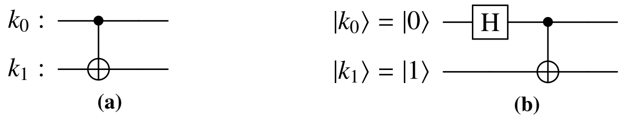


Figure 2: Quantum circuits with two qubits k_0 and k_1. The circuit (a) depicts the *CNOT* gate where k_0 is the control and k_1 the target wires. The respective unitary matrix is given in (48). The state of the target k_1 is only flipped only when the state of the control is $k_0 = 1$ else it is maintained. The output of the circuit (b) is the entangled state $\frac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + |3\rangle)$.

In passing, we note that *CNOT* is referred as a controlled Pauli X gate given that

$$CNOT = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} . \tag{51}$$

It goes without saying that *CNOT* is like any other quantum gate a unitary. To obtain the controlled two-qubit versions of other single-qubit gates it is sufficient to replace X above with the respective single qubit gate.

3.3.3. Quantum circuits

As stated earlier, quantum computation can be summarised as the mapping of a multi qubit state $|q\rangle \in \mathbb{C}^{2n}$ into the state $U\,|q\rangle \in \mathbb{C}^{2n}$ for the purpose of doing some desired computation. The construction of the unitary matrix $U \in \mathbb{C}^{2n \times 2n}$ out of one-qubit and multi-qubit gates is commonly visualised using *circuit diagrams* [18]. We should bear in that

Proposition 3. *The product $U_1 U_2$ of two unitary matrices U_1 and U_2 is again a unitary.*

Proposition 4. *The Kronecker product $U_1 \otimes U_2$ of two unitary matrices U_1 and U_2 is again a unitary.*

In the circuit diagrams depicted in Figures 1 and 2 each line is referred to as a *wire* and represents a qubit. The circuit is read from left to the right and shows the sequence of applied single and two-qubit gates. There is no joining or splitting of wires which would violate the unitarity requirement for U. As illustrated in Figure 1 the identity gates are usually omitted in the diagrams. In comparing and interpreting circuit diagrams note that there are two different common qubit labelings, see Section 3.2.2.

In the basic circuit depicted in Figure 1 the computation starts with two qubits k_0 and k_1 both in the state $|0\rangle$. The corresponding state vector is

$$|q\rangle = |k_0\rangle \otimes |k_1\rangle = |0\rangle \otimes |0\rangle = |0\rangle \otimes |0\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{52}$$

Next, we apply to the qubit k_1 the Pauli X gate and to the qubit k_0 the identity gate I. The resulting state is given by

$$|q\rangle = (I \otimes X)(|k_0\rangle \otimes |k_1\rangle) = I\,|0\rangle \otimes X\,|0\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}0 & 1\\1 & 0\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\1\\0\\0\end{pmatrix} . \tag{53}$$

Notice that $U = I \otimes X$ is the unitary for entire circuit.

The entangled state given in equation (34) can be obtained with the circuit depicted in Figure 2b. The sequence of the mappings is applied as follows:

$$\begin{aligned} &(H \otimes I)\colon\ |0\rangle \otimes |0\rangle \mapsto \frac{1}{\sqrt{2}}\,(|0\rangle + |1\rangle) \otimes |0\rangle\ , \\ &CNOT\colon\ \frac{1}{\sqrt{2}}\,(|0\rangle + |1\rangle) \otimes |0\rangle \mapsto \frac{1}{\sqrt{2}}\,(|0\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle)\ . \end{aligned} \tag{54}$$

For the $CNOT$ the left qubit (k_1) is the control qubit and the right (k_0) is the target qubit. The target qubit state is flipped when the control qubit is in state $|1\rangle$. This can be also more formally explained obtain the definition (51) of the $CNOT$ and noting the orthonormality of the states $|0\rangle$ and $|1\rangle$.

3.4. Measurement

As mentioned before, in quantum systems the outcome of a measurement is random and measurement has an irreversible effect on the state of a system. When a quantum system with the state vector $|q\rangle \in \mathbb{C}^{2^{n-1}}$, i.e. $|q\rangle = \sum_{k=0}^{2^{n-1}} q_k\,|k\rangle$, is measured we will find it in one of its 2^{n-1} possible states. According to the Born rule, the probability of observing the system in the specific state $|k^*\rangle$ is $p(|k^*\rangle) = |q_{k^*}|^2$. The measurement process collapses the quantum state so that after measurement the state vector becomes $|q\rangle = |k^*\rangle$. Consequently, we can infer the absolute values of the coefficients $\{|q_0|^2, |q_1|^2, \dots, |q_{2^{n-1}}|^2\}$ only by considering the statistics of several quantum computations with the same circuit.

Clearly, a state vector $|q\rangle$ can be expressed in different basis so that the possible outcomes of a measurement depend on the specific basis imposed by the measurement apparatus. This realisation leads to the notion of projective measurement.

Proposition 5. *A complete set of orthogonal projectors P_j has the properties $\sum_j P_j = I$ and $P_j P_k = \delta_{jk}$ and can be, for instance, obtained from an orthonormal basis $\{|r_0\rangle, \dots, |r_{2^{n-1}}\rangle\}$ according to the definition*

$$P_j = |r_j\rangle\langle r_j|\ . \tag{55}$$

In projective measurement the probability of observing the system in state j is given by

$$p(j) = \|P_j\,|q\rangle\| = \langle q|P_j|q\rangle\ . \tag{56}$$

Note that the probabilities for all 2^{n-1} states add up to one as desired. Furthermore, during the measurement the system collapses to the state

$$\frac{P_j\,|q\rangle}{\|P_j|\,|q\rangle\|} = \frac{P_j\,|q\rangle}{\sqrt{p(j)}}\ . \tag{57}$$

Projective measurement can be used to derive the rules for partial measurement. For instance, in a two qubit system the two projectors

$$P_0 = I \otimes |0\rangle\langle 0|\ , \quad P_1 = I \otimes |1\rangle\langle 1|\ , \tag{58}$$

form a complete set. They measure the probability of finding the right qubit in the state $|0\rangle$ or $|1\rangle$, respectively. Hence, applying P_1 to a quantum state $|q\rangle = \frac{1}{\sqrt{2}}(|1\rangle|0\rangle + |1\rangle|1\rangle)$ yields the probability

$$p(0) = \langle q|P_0|q\rangle = \frac{1}{2}, \tag{59}$$

and the state colapses to

$$\frac{P_0|q\rangle}{\sqrt{p(0)}} = \sqrt{2}P_0|q\rangle = |1\rangle|0\rangle\ . \tag{60}$$

3.5. Quantum algorithm complexity

The complexity of a quantum algorithm can be measured by three major ways: the number of universal quantum gates, the depth of the quantum circuit, and the number of queries of an oracle (black box) model. In this work, we aim to make all of our quantum algorithms explicit with no refence to oracle black box. We define the complexity of our quantum algorithm by the number of quantum gates present in our quantum circuit. For all cases studied in this work, we shall transpile our quantum circuits with a universal set of quantum gates made of the two-qubit Controlled-Not (or $CNOT$) gate, and the one-qubit rotation gate (or U_3 gate). We further assume the ideal scenario where we ignore the error of fundamental quantum gates, and subsequently the cost for error correction.

4. Quantum computing algorithms

In this Section we provide a summary of quantum algorithms and circuits needed in our quantum computational scheme. We also make comments on the computational complexity of these quantum algorithms.

4.1. Multi-controlled operation

One fundamental operation in scientific computing involves conditional commands for making decisions. In classical computing, the "if-then" structure is commonly used to handle such conditions. In the realm of quantum computing, the equivalent of this conditional structure can be realized through multi-controlled operations.

As previously discussed in Section 3.3.2, the controlled-Not ($CNOT$) gate serves as a prototype circuit for such operations. As elaborated in that section, it is possible to create a general controlled unitary circuit, denoted as the controlled-U (CU) gate, by replacing the (NOT) gate with an arbitrary single-qubit rotation gate (U). Mathematically, the CU gate is represented as:

$$CU = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes U. \tag{61}$$

It is evident that the CU gate is an operator that maps the state $|c\rangle|t\rangle$ to $|c\rangle U|t\rangle$. In simpler terms, this operation can be described as follows: "If the control qubit is $|c = 1\rangle$, then apply the operator $|U\rangle$ to the target qubit $|t\rangle$."

Generalizing the concept of the CU gate leads to the multi-controlled rotation gate, which performs the mapping:

$$|c_0c_1...c_n\rangle|t\rangle \mapsto |c_0c_1...c_n\rangle U|t\rangle, \tag{62}$$

where the rotation U is applied only if all the control qubits are set to $|c_0c_1...c_n\rangle = |11...1\rangle$. In our work, we adopt the "V-chain" approach proposed by Nelson and Chuang[1] for multi-controlled operations.

To illustrate, we provide an example quantum circuit with 3 control qubits in Figure 3, employing the V-chain architecture shown in Figure 3 (b). In the computational basis $|c_0c_1c_2\rangle$, the first Toffoli gate (i.e., the controlled $CNOT$ gate) transforms the state of the first ancilla qubit into $|a_0 = c_0 \cdot c_1\rangle$, and the second Toffoli gate updates the state of the second ancilla qubit to $|a_1 = c_0 \cdot c_1 \cdot c_2\rangle$. A controlled-$U$ gate is subsequently applied if $|a_1\rangle = 1$, indicating that this operation occurs only when all control qubits are in the state $|1\rangle$. The circuit is then un-computed by re-applying the Toffoli gates to ensure reversibility.

Complexity A general Multi-controlled operation with n control qubits requires $n - 1$ ancilla qubits and $2n - 2$ Toffoli gates. We note that Toffoli gates can be realized by the universal gate set defined in 3.5, with 10 U gate and

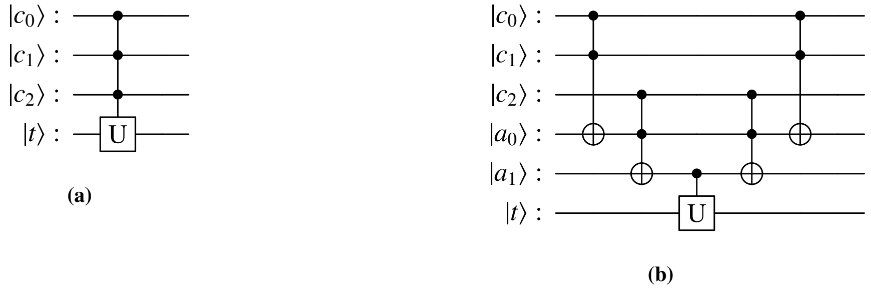


Figure 3: Quantum Circuit for Multi-Controlled Operations: (a) A multi-controlled rotation circuit featuring 3 control qubits $|c_0c_1c_2\rangle$ and one target qubit $|t\rangle$. (b) An implementation of the circuit from (a) utilizing the V-chain circuit as described in [1], augmented with 2 ancilla qubits $|a_0a_1\rangle$. We note that the operators represented by (a) and (b) are equivalent.

6 *CNOT* gate. We conclude that for a system with computational basis $N = 2^n$, the Multi-controlled operation has a computational complexity of O (Poly (n) hence is logrithmatic in N.

4.2. Encoding of polynomials

The encoding of functions on a quantum computer is relevant for most applications involving differential equations. We focus in the following on the encoding of polynomials given that any function on a closed and bounded domain can be approximated by polynomials. Specifically, we consider the approach proposed in [41, 45] to encode a real-valued univariate polynomial $f(k) \in \mathbb{R}$ of the form

$$f(k) = \sum_{m=0}^{p} \alpha_m k^m \,, \tag{63}$$

where $k \in \{0,\ 1,\ 2,\ \dots, 2^{n-1}\}$ is the domain, n the number of qubits, p the polynomial degree and $\alpha_m \in \mathbb{R}$ the prescribed coefficients. As proposed in [41, 45], $f(k)$ can be encoded by constructing a unitary $U_P \in \mathbb{C}^{2^{n+1}\times 2^{n+1}}$ which is applied to the augmented input state $|k\rangle\,|0\rangle \equiv |k\rangle \otimes |0\rangle$. The augmented state consists of the tensor product of the argument $|k\rangle$ and an ancillary qubit $|0\rangle$. As will become clear, such ancillary, or helper, qubits are widely used in quantum algorithms to embed the problem in a higher dimensional space. Later, the sought solution is determined by projecting the output state to a lower dimensional space. The unitary U_P is applied to the augmented input state yielding the output state

$$U_P\colon\ |k\rangle\,|0\rangle \mapsto |k\rangle\,(\cos(\epsilon f(k))\,|0\rangle + \sin(\epsilon f(k))\,|1\rangle) = \cos(\epsilon f(k))\,|k\rangle\,|0\rangle + \sin(\epsilon f(k))\,|k\rangle\,|1\rangle\ . \tag{64}$$

It is easy to see that U_P makes use of the rotation gate $R_Y(2\epsilon f(k))$ in (43) and concerns only the ancillary qubit originally in state $|0\rangle$. Intuitively, U_P applies a rotation by $\epsilon f(k)$ in the plane spanned by the first (right-most) qubit. The parameter $\epsilon \in \mathbb{R}$ is a small scaling factor, which is chosen such that $\epsilon f(k) \ll 1$ so that the linearisation of the sine term in (65) yields the approximation

$$\cos(\epsilon f(k))\,|k\rangle\,|0\rangle + \sin(\epsilon f(k))\,|k\rangle\,|1\rangle \approx \sqrt{1-(\epsilon f(k))^2}\,|k\rangle\,|0\rangle + \epsilon f(k)\,|k\rangle\,|1\rangle\ . \tag{65}$$

That is, for sufficiently small ϵ, the amplitude is equal to the sought function value $\epsilon f(k)$ when the ancillary qubit is in state $|1\rangle$. In contrast, when the ancillary qubit is in state $|0\rangle$ the amplitude is of no interest and can be discarded. As mentioned the rotation gate $R_Y(2\epsilon f(k))$ is applied only to the ancillary qubit originally in state $|0\rangle$. Because $f(k)$ is additively composed of p scaled monomials, the gate $R_Y(2\epsilon f(k))$ can be multiplicatively composed of the p rotations $R_Y(2\epsilon a_m k^m)$. Furthermore, the order of the rotations $R_Y(2\epsilon\alpha_m k^m)$ is irrelevant because all the rotations pertain to the same ancillary qubit.

The implementation of the discussed algorithm as a quantum circuit can be best understood with an illustrative example. To this end, we consider, without loss of generality, the encoding of the quadratic polynomial

$$f(k) = \alpha_2 k^2 + \alpha_1 k + \alpha_0 \,. \tag{66}$$

The number of qubits n for representing k depends usually on the application. We choose here $n = 3$ so that k has the binary representation

$$k = k_0 k_1 k_2 = k_0 2^2 + k_1 2^1 + k_2 2^0 \,. \tag{67}$$

Before proceeding further, we note that $k_0^2 = k_0$, $k_1^2 = k_1$ and $k_2^2 = k_2$, given that each bit has either the value 0 or 1. The binary form introduced in the quadratic polynomial (66) yields

$$f(k) = 4(\alpha_1 + 4\alpha_2)k_0 + 2(\alpha_1 + 2\alpha_2)k_1 + (\alpha_1 + \alpha_2)k_2 + 8\alpha_2 k_0 k_2 + 4\alpha_2 k_1 k_2 + 16\alpha_2 k_0 k_1 + \alpha_0 \,. \tag{68}$$

To begin the encoding process, we start with a 4-qubit system $|k_0 k_1 k_2\rangle|0\rangle$. Recalling the definition of the R_Y gate, for arbitrary $\theta \in \mathbb{R}$, we have:

$$R_Y(2\theta)|0\rangle = e^{-i\theta Y}|0\rangle = cos(\theta)|0\rangle + sin(\theta)|1\rangle. \tag{69}$$

Let $\theta = f(k)$ (omitting the constant ϵ for simplicity), and our goal is to implement the operator $e^{-if(k)Y}$. Notably, the summation operator in the polynomial f_k is converted to the multiplication operator in the exponential form, as shown in Eqn 70.

$$\begin{aligned} e^{-if(k)Y} =& e^{-i4(\alpha_1+4\alpha_2)k_2 0} e^{-i2(\alpha_1+2\alpha_2)k_1 Y} e^{-i2(\alpha_1+\alpha_2)k_2 Y} e^{-i8\alpha_2 k_0 k_2 Y} e^{-i4\alpha_2 k_1 k_2 Y} e^{-i16\alpha_2 k_0 k_1 Y} e^{-i\alpha_0 Y} \\ =& R_Y(8(\alpha_1 + 4\alpha_2)k_0) R_Y(4(\alpha_1 + 2a_2)k_1) R_Y(2(\alpha_1 + \alpha_2)k_2) \cdot \\ & R_Y(16\alpha_2 k_0 k_2) R_Y(8\alpha_2 k_1 k_2) R_Y(32\alpha_2 k_0 k_1) R_Y(2\alpha_0) \end{aligned} \tag{70}$$

We can thus implement the desired operator using multi-controlled R_Y gates, as detailed in Section 4.1. For example, we shall apply $R_Y(16\alpha_2)$ to the last qubit if and only if $|k_2\rangle = 1$, which can be achieved with a controlled R_Y gate with rotation angle $\theta = 8(\alpha_1 + 4\alpha_2)$. We therefore achieve the encoding of the full polynomial by successively apply the multi-controled R_Y gates as depicted in Fig. 4

The above described technique can be easily generalised to multivariate polynomials. For instance, consider the bivariate polynomial

$$f(k,l) = \sum_{m=0}^{p} \sum_{j=0}^{q} a_{mj} k^m l^j \,, \tag{71}$$

where p and q are the polynomial degrees of the arguments k and l, respectively, and $a_{mj} \in \mathbb{R}$ are the coefficients. The argument pair take the values $k,\ l \in \{0,\ 1,\ 2,\ \ldots, 2^{n-1}\}$, so that there are in total n^2 qubits. The augmented unitary operator $U_P \in \mathbb{C}^{2^{n^2+1} \times 2^{n^2+1}}$ implements the mapping

$$U_P\colon\ |k\rangle\,|l\rangle\,|0\rangle \mapsto \cos(cf(k,l))\,|k\rangle\,|l\rangle\,|0\rangle + \sin(cf(k,l))\,|k\rangle\,|l\rangle\,|1\rangle \ . \tag{72}$$

To determine such a unitary U_P, we first express the arguments k and l of the polynomial as binaries $k = k_{n-1} \ldots k_1$ and $l = l_{n-1} \ldots l_1 l_0$. Substituting them into (71) and the application of the multinomial expansion theorem yields

$$\begin{aligned} f(k,l) &= \sum_{m=0}^{p} \sum_{j=0}^{q} \alpha_{mj} (2^n k_n + ... + 2^1 k_1 + 2^0 k_0)^m (2^n l_n + ... + 2^1 l_1 + 2^0 l_0)^j \\ &= \sum_{m=0}^{p} \sum_{j=0}^{q} \sum_{b_1+b_2+...b_n=m} \sum_{c_1+c_2+...c_n=j} \alpha_{mj} \frac{m!}{b_1!b_2!...b_n!} \frac{j!}{c_1!c_2!...c_n!} \Pi_{s=0}^{n-1} \Pi_{t=0}^{n-1} k_s^{b_s} l_t^{c_t} \,. \end{aligned} \tag{73}$$

The respective quantum circuit consists of multi-controlled R_Y gates conditioned on the n^2 qubits with the rotation

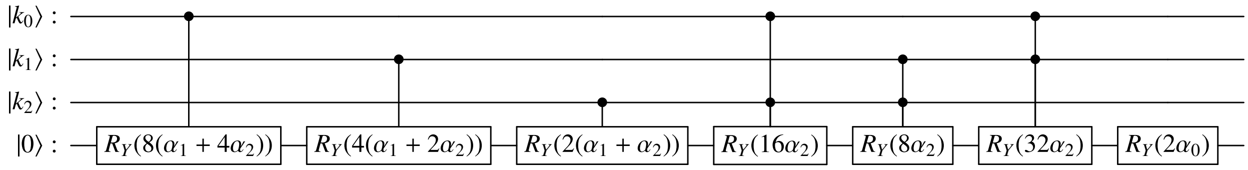


Figure 4: Quantum circuit for implementing the mapping $|k\rangle|0\rangle \mapsto \cos(f(k))|k\rangle|0\rangle + \sin(f(k))|k\rangle|1\rangle$. Here, $k = k_0k_1k_2$ is represented with $n = 3$ qubits and the ancillary qubit is initially in state $|0\rangle$. For instance, providing the circuit with the input $|k\rangle = |1\rangle|1\rangle|0\rangle$, i.e., $k = 2^2 + 2^1 = 6$, it will yield the output $\cos(f(6))|1\rangle|1\rangle|0\rangle|0\rangle + \sin(f(6))|1\rangle|1\rangle|0\rangle|1\rangle$.

angles equal to $\alpha_{mj}\frac{m!}{b_1!b_2!...b_n!}\frac{j!}{c_1!c_2!...c_n!}$.

Complexity Finally, for a polynomial function of degree p and domain size $N = 2^n$, the circuits can be evaluated in $\mathcal{O}$ (Poly (pn)) time, hence is logarithmic in N. A proof of this can be found in Appendix C

4.3. State preparation via function encoding

On a quantum computer, a classical vector $|q\rangle \in \mathbb{C}^N$ with an L_2-norm $\| |q\rangle \| = 1$ can be encoded into the amplitudes of $n = \log_2 N$ qubits. This means, for instance, that $n = 3$ qubits are sufficient to encode a classical vector with $N = 8$ components. Considering that each of the $n = 3$ qubits has only two states, the encoding of a vector with $N = 8$ components requires entanglement. The encoding of a classical vector into a quantum state is referred to as *state preparation*.

There are a number of approaches and algorithms for efficient state preparation [3, 30, 38]. In all algorithms a quantum state $|q\rangle$ is prepared by applying a suitably constructed unitary U_S to the zero-state, i.e.,

$$U_S: \ |0\rangle^{\otimes n} = |\underbrace{0\dots0}_{n}\rangle \mapsto |q\rangle = \sum_{k=0}^{2^n} q_k |k\rangle \ , \tag{74}$$

so that the amplitudes q_k are equal to the components of the classical vector we seek to encode. It is possible to use the polynomial encoding approach introduced in the previous section to devise a technique for constructing an approximate unitary. To this end, we first determine a polynomial $f(k) \approx q_k$, for all $k = 1, 2, \cdots, N$, approximating the components of a classical vector $|q\rangle \in \mathbb{R}^N$. We determine the polynomial $f(k)$ on a classical computer using standard polynomial approximation techniques, specially Chebyshev interpolation. After a $f(k)$ is determined, we can use the method described in the previous section to construct a unitary U_P and to evaluate it for different k on a quantum computer, cf. (64). We can make use of quantum superposition to evaluate $f(k)$ for all possible values of k simultaneously. To this end, we first generate a uniform quantum state composed of all basis vectors before applying the unitary U_P. As illustrated in Figure 5 for $n = 3$ a uniform quantum state is obtained by applying Hadamard gates. More specifically, the three Hadamard gates in Figure 5 accomplish the following transformation

$$\begin{aligned}(H \otimes H \otimes H \otimes I)\colon \ |0\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle &\mapsto \frac{1}{2\sqrt{2}}(|0\rangle + |1\rangle) \otimes (|0\rangle + |1\rangle) \otimes (|0\rangle + |1\rangle) \otimes |0\rangle \\ &= \frac{1}{2\sqrt{2}} \sum_{k_0=0}^{2} \sum_{k_1=0}^{2} \sum_{k_2=0}^{2} |k_0\rangle|k_1\rangle|k_2\rangle|0\rangle = \frac{1}{2\sqrt{2}} \sum_{k=0}^{7} |k\rangle|0\rangle \ .\end{aligned} \tag{75}$$

According to (64) and (65), the application of the unitary $U_P(cf(k))$ to this state yields

$$\begin{aligned}U_P(cf(k)): \ \frac{1}{2\sqrt{2}} \sum_{k=0}^{7} |k\rangle|0\rangle &\mapsto \frac{1}{2\sqrt{2}} \sum_{k=0}^{7} (\cos(cf(k))|k\rangle|0\rangle + \sin(cf(k))|k\rangle|1\rangle) \\ &\approx \frac{1}{2\sqrt{2}} \sum_{k=0}^{7} \left(\sqrt{1 - (cf(k))^2}|k\rangle|0\rangle + cf(k)|k\rangle|1\rangle \right) .\end{aligned} \tag{76}$$

An alternative state preparation approach for real-valued vectors $|q\rangle \in \mathbb{R}^N$ is introduced in Appendix B.

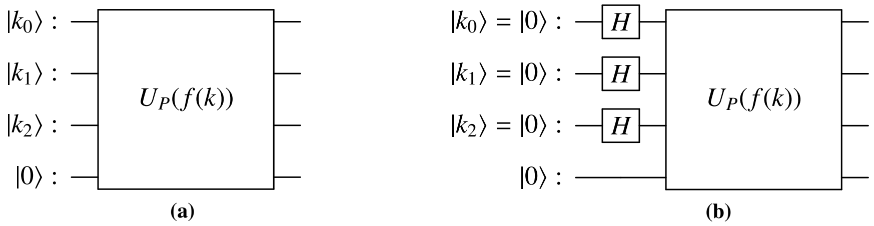


Figure 5: Quantum state preparation via function encoding. The four-qubit gate in (a) represents the unitary $U_P(f(k))$ for encoding $f(k)$ and is composed as the circuit depicted in Figure 4. Prepending the circuit with three Hadamard gates as shown in (b) yields a uniform superposition state, which becomes after multiplication by $U_P(f(k))$ an entangled state.

4.4. Encoding of arbitrary functions

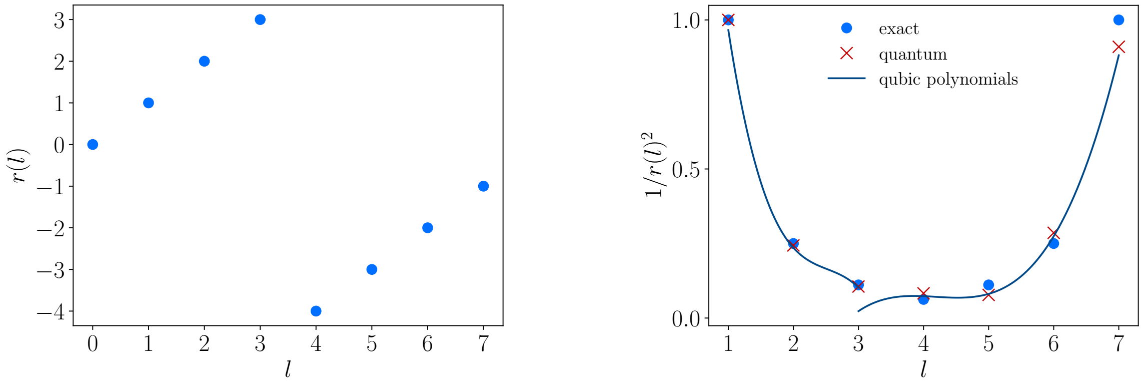
The accurate approximation of arbitrary functions by polynomials is an extensively studied topic in numerical analysis, so a wide range of efficient algorithms and implementations are available [8]. An arbitrary function $r(k) \in \mathbb{R}$ can be encoded on a quantum computer by first determining its piecewise polynomial approximation and then encoding the obtained polynomials. We perform the piecewise polynomial approximation on a classical computer using the Python Numpy library. Subsequently, we encode the obtained polynomials using the approach introduced in Section 4.2.

In piecewise approximation the polynomial degree p in the intervals and the number of the subintervals n_ω are both fixed and chosen empirically. The polynomials of degree p are determined so that they interpolate the given function $r(k)$ at $p + 1$ collocation points. As known, choosing the Chebyshev nodes as the collocation points yields a particularly efficient approximation with small interpolation errors. Furthermore, at the Chebyshev nodes of the first kind the Chebyshev polynomials satisfy a certain orthogonality relationship so that the interpolation can be performed without inverting a (Vandermonde) matrix. After determining the polynomial in the Chebyshev basis it is straightforward to convert it into a monomial basis as required for encoding. Furthermore, if $n_\omega > 1$, it is necessary to determine in which interval the argument k of $r(k)$ lies in order to evaluate the polynomial approximant corresponding to the interval. The interval number is determined by comparing the position of k with respect to the interval boundaries.

We consider an example to discuss the sketched approach in more detail. In homogenisation the inverse square of the relabelling function $r(k)$ introduced in (19) is required, which is as depicted in Figure 6a discontinuous at the centre of the interval $[0, 7]$. Note that the $r(k)$ corresponds to a physical domain is discretised with $N = 8$ grid cells and the respective nodal vectors have $N = 8$ components so that they require $n = \log_2 8 = 3$ qubits to represent on a quantum computer. The square inverse $1/(r(k))^2$ is plotted in Figure 6b and is C^0 continuous is at the centre of the interval. We approximate the square inverse with two polynomials $f^0(k)$ and $f^0(k)$ of degree $p = 3$ over the intervals $\omega^1 = [1, 3)$ and $\omega^2 = [3, 7]$, i.e. $n_\omega = 2$. The accuracy of the approximation can be improved by increasing the polynomial degree p and the number ob subintervals n_h.

The quantum circuit for evaluating the obtained piecewise polynomial approximation is shown in Figure 7. The top three qubits $|k_0\rangle$, $|k_1\rangle$ and $|k_2\rangle$ represent the state vector and the remaining four qubits are ancillary qubits. After execution, the function value is $r(k)$ is equal to the amplitude of the second component of the ancillary qubit $|a_3\rangle$. The two gates U_{P_0} and U_{P_1} represent the circuits for evaluating the two polynomials $f^0(k)$ and $f^1(k)$, and are constructed as described in Section 4.2. Note that U_{P_0} and U_{P_1} are conditioned on the ancilla $|a_0\rangle$ and are executed only when $|a_1\rangle \equiv |1\rangle$. The two gates U_{C_0} and U_{C_1} determine whether k is in interval ω_1 or ω_2 and set $|a_1\rangle$ accordingly to state k_1. This is achieved by integer comparison of k and the interval boundaries. The inverse gates $U^\dagger_{C_0}$ and $U^\dagger_{C_1}$ reset, or unset, the $|a_1\rangle$ to its original state. Their inverses are obtained by simply executing the subcircuits backwards and taking the inverses of each of the gates in the subcircuits given that quantum computing is reversible. See also [22] for the implementation of gates U_{C_0} and U_{C_1}, and [44] for the overall circuit for encoding arbitrary functions. The sketched circuit is available in the qiskit PiecewiseChebyshev class.

ComplexityWe observe that the operators U_{C_0} and U_{C_1} can be effectively implemented using quantum bit string comparator algorithms, as elaborated in Oliveira et al. (2007) [33]. This reference demonstrates that a subtraction-



(a)

(b)

Figure 6: Mappings of the $N = 8$ components of the state vector. In (a) the mapping of the components to wave numbers $r(k)$ according to (19) is plotted. In (b) the function $1/r(k)^2$ needed for computing the second derivatives and its approximation are illustrated. The dots (•) denote the exact values which are approximated with two qubic polynomials in the two intervals [1, 3) and [3, 7]. The polynomials are obtained by interpolating $1/r(k)^2$ at the four Chebyshev points of the first kind. The four Chebyshev points in the two intervals are not shown. The crosses (×) denote the values obtained from the quantum circuit in Figure 7.

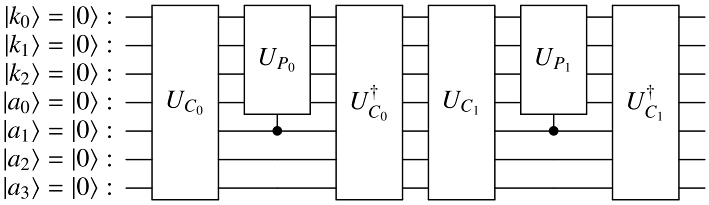

$|k_0\rangle = |0\rangle$:
$|k_1\rangle = |0\rangle$:
$|k_2\rangle = |0\rangle$:
$|a_0\rangle = |0\rangle$:
$|a_1\rangle = |0\rangle$:
$|a_2\rangle = |0\rangle$:
$|a_3\rangle = |0\rangle$:
U_{C_0}
U_{P_0}
$U^\dagger_{C_0}$
U_{C_1}
U_{P_1}
$U^\dagger_{C_1}$

Figure 7: Quantum circuit for implementing the mapping $|k\rangle |0\rangle^{\otimes 4} \mapsto \cos(r(k)) |k\rangle |0\rangle |0\rangle^{\otimes 3} + \sin(r(k)) |k\rangle |1\rangle |0\rangle^{\otimes 3}$. This circuit is constructed using the qiskit PiecewiseChebyshev class and encodes the function $r(k)$ plotted in Figure 6b. The function $r(k)$ is defined over two intervals. The two unitaries U_{C_0} and U_{C_1} set the the ancillary qubit $|a_1\rangle$ to state $|1\rangle$ when the evaluation point $|k\rangle$ lies in the their respective intervals. The corresponding unitaries $U^\dagger_{C_0}$ and $U^\dagger_{C_1}$ set the ancillary qubit $|a_1\rangle$ back to state $|0\rangle$. The polynomials for the two intervals are evaluated with the controlled unitaries U_{P_0} and U_{P_1} conditioned on the ancillary qubit $|a_1\rangle$. After the circuit is executed the function value $\sin(r(k)) |k\rangle$ is encoded in the top three qubits when the ancillary qubit $|a_0\rangle$ is found in state $|1\rangle$. The other three ancillary qubits a_1, a_2 and a_3 are only used temporarily by the circuit during execution and are of no relevance.

based algorithm facilitates the construction of these operators, with the complexity scaling according to $O(\text{Poly}(n))$. Furthermore, as presented in Section 4.2, we can implement the operators U_{P_0} and U_{P_1} using a polynomial encoding algorithm. The controlled versions of U_{P_0} and U_{P_1} are realized through the V-chain algorithm, detailed in Section 4.1. Notably, both of these algorithms fall within the complexity class of $O(\text{Poly}(n))$. Based on this analysis, we conclude that the overall algorithm maintains a complexity of $O(\text{Poly}(n))$.

4.5. Quantum Fourier Transform

We introduced the classical Discrete Fourier Transform (DFT) of a vector $v \in \mathbb{C}^N$ in Section 2.3. In preparation for the derivation of the equivalent Quantum Fourier Transform we denote DFT more compactly as

$$v = F_N \widehat{v} \tag{77}$$

where $F_N \in \mathbb{C}^{N\times N}$ is the Fourier matrix

$$F_N = \frac{1}{\sqrt{N}} \begin{pmatrix} & \vdots & \\ \dots & \omega_N^{jk} & \dots \\ & \vdots & \end{pmatrix} , \tag{78}$$

and ω_N is the N-th root of unity

$$\omega_N = e^{\frac{2\pi i}{N}} \ . \tag{79}$$

It is straightforward to show that the Fourier matrix F_N is unitary, i.e. $F_N^{-1} = F_N^{\dagger}$. Consequently, DFT can be readily implemented on a quantum computer. Note that in contrast to mathematical literature in quantum computing, the roles of $\hat{v}$ and v in (77) are usually switched. This is inconsequential as long as consistency is maintained.

In a quantum system with n qubits the two vectors $\hat{v} \in \mathbb{C}^N$ and $v \in \mathbb{C}^N$ are two state vectors with $N = 2^n$. To emphasise the switch from DFT to QFT, we rewrite (77) in bra-ket notation as

$$\widehat{|q\rangle} = F_N \, |q\rangle \quad \Rightarrow \quad \sum_{j=0}^{N} \widehat{q}_j \, |j\rangle = F_N \sum_{k=0}^{N} q_k \, |k\rangle \ . \tag{80}$$

As always, in order to implement QFT on a quantum computer the unitary matrix F_N must be decomposed into elementary gates acting on one or two qubits at a time. In coming up with such a decomposition, it is, as usual, convenient to consider only the mapping of individual basis vectors, i.e.,

$$F_N|k\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega_N^{kj} \, |j\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{\frac{2\pi i}{N} kj} \, |j\rangle \ . \tag{81}$$

Subsequently, the obtained decomposition and quantum circuit can be applied to the entire state vector $|q\rangle$ with no further changes.

We choose, without loss of generality, $n = 3$ to illustrate the decomposition of F_N into elementary one and two qubit gates. We make use of the binary representation of indices

$$k = k_0 k_1 k_2 = k_0 2^2 + k_1 2^1 + k_2 2^0, \quad j = j_0 j_1 j_2 = j_0 2^2 + j_1 2^1 + j_2 2^0 , \tag{82}$$

cf. (36), and introduce the binary fraction notation

$$\frac{k}{2^{n=3}} = 0.k_0 k_1 k_2 = \frac{k_0}{2^1} + \frac{k_1}{2^2} + \frac{k_2}{2^3} \, . \tag{83}$$

Furthermore, noting that $|k\rangle = |k_0\rangle \, |k_1\rangle \, |k_2\rangle$ and $|j\rangle = |j_0\rangle \, |j_1\rangle \, |j_2\rangle$, equation (81) can be rewritten as

$$\begin{aligned} F_N|k_0\rangle \, |k_1\rangle \, |k_2\rangle &= \frac{1}{\sqrt{2^3}} \sum_{j_0=0}^{1} \sum_{j_1=0}^{1} \sum_{j_2=0}^{1} \left(e^{2\pi i (0.k_0k_1k_2) j_0 2^2} \, |j_0\rangle\right) \left(e^{2\pi i (0.k_0k_1k_2) j_1 2^1} \, |j_1\rangle\right) \left(e^{2\pi i (0.k_0k_1k_2) j_2 2^0} \, |j_2\rangle\right) , \\ &= \frac{1}{\sqrt{2^3}} \left(|0\rangle + e^{2\pi i (0.k_0k_1k_2) 2^2} \, |1\rangle\right) \left(|0\rangle + e^{2\pi i (0.k_0k_1k_2) 2^1} \, |1\rangle\right) \left(|0\rangle + e^{2\pi i (0.k_0k_1k_2) 2^0} \, |1\rangle\right) . \end{aligned} \tag{84}$$

The exponentials can be further simplified because $e^{2i\pi m} = \cos(2\pi m) + i \sin(2i\pi m) \equiv 1 \ \forall m \in \mathbb{Z}^+$, i.e.,

$$e^{2\pi i (0.k_0k_1k_2) j_0 2^2} = e^{2\pi i (0.k_2) j_0}, \quad e^{2\pi i (0.k_0k_1k_2) j_1 2^1} = e^{2\pi i (0.k_1k_2) j_1}, \quad e^{2\pi i (0.k_0k_1k_2) j_0 2^0} = e^{2\pi i (0.k_0k_1k_2) j_2} \, , \tag{85}$$

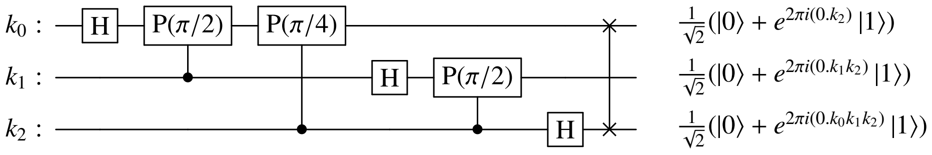

k_0 :
k_1 :
k_2 :
H
P($\pi/2$)
P($\pi/4$)
H
P($\pi/2$)
H
$\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.k_2)}\,|1\rangle)$
$\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.k_1k_2)}\,|1\rangle)$
$\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.k_0k_1k_2)}\,|1\rangle)$

Figure 8: Quantum Fourier Transform. One Hadamard gate H and a variable number of phase gates $P(\cdot)$ are applied to each qubit. For instance, providing the circuit with the intput $|k\rangle = |1\rangle|1\rangle|0\rangle$ so that $0.k_2 = 0$, $0.k_1k_2 = 1/2$ and $0.k_0k_1k_2 = 3/4$, cf. (83) , it will yield the output state $1/(2\sqrt{2})(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)(|0\rangle - i\,|1\rangle)$. After evaluating the Kronecker products the result expressed in matrix form reads $(1, -i, -1, +i, 1, -i, -1, i)$.

Finally, a basis vector $|k\rangle$ is mapped using QFT as

$$F_N|k_0\rangle\,|k_1\rangle\,|k_2\rangle = \frac{1}{\sqrt{2^3}}\left(|0\rangle + e^{2\pi i(0.k_2)}\,|1\rangle\right)\left(|0\rangle + e^{2\pi i(0.k_1k_2)}\,|1\rangle\right)\left(|0\rangle + e^{2\pi i(0.k_0k_1k_2)}\,|1\rangle\right). \tag{86}$$

Each of the three (round) brackets correspond to a single qubit. We obtain each of the bracketed expressions by successively mapping a basis $|k\rangle = |k_0\rangle\,|k_1\rangle\,|k_2\rangle$ using the Hadamard and phase gates, H and $P(\theta)$, introduced in (45). For instance focusing on the qubit $|k_0\rangle$, the application of the Hadamard gate yields

$$H\colon\ |k_0\rangle \mapsto \frac{1}{\sqrt{2}}\left(|0\rangle + (-1)^{k_0}\,|1\rangle\right) = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{\frac{2\pi i}{2}k_0}\,|1\rangle\right) = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i(0.k_0)}\,|1\rangle\right). \tag{87}$$

Subsequent application of the controlled phase gate $P(\pi/2)$ with $|k_1\rangle$ as the control qubit gives

$$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{\frac{\pi}{2}ik_1}e^{2\pi i(0.k_0)}\,|1\rangle\right) = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i(0.k_0k_1)}\,|1\rangle\right). \tag{88}$$

Note that $k_1 \in \{0,\ 1\}$ and $P(\pi/2)$ is only applied when $k_1 = 1$. One last application of the controlled phase gate $P(\pi/4)$ yields the expression within the last bracket in (86), i.e.,

$$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{\frac{\pi}{4}ik_2}e^{2\pi i(0.k_0k_1)}\,|1\rangle\right) = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i(0.k_0k_1k_2)}\,|1\rangle\right). \tag{89}$$

The expressions within the second and first bracket can be obtained in a similar way. The final circuit diagram is depicted in Figure 8. Although the output of the circuit is correct the values are reversed in comparison to standard DFT so that it is necessary to apply a swap gate to the qubits $|k_2\rangle$ and $|k_0\rangle$.

At last, it is simple to obtain from the introduced circuit for the inverse Fourier matrix F_N to obtain a circuit for the inverse Fourier matrix F_N^{-1}. The quantum circuit must be run in reverse and the each gate replaced with its inverse, i.e. its conjugate transpose given that all gates are unitary.

Complexity It is well known from [1] that the QFT circuit requires n Hardarmard gate and $\frac{n(n+1)}{2}$ controled Phase gate. Therefore the over all complexity of the QFT circuit is $O(n^2)$.

4.6. Base swap

We introduce the base swap circuit for the implementation of the boundary condition in Alg. 1 (c). Given an entangled state $\sum_{j=0}^{N-1} y_j|j\rangle$ and we wish to swap the two basis $|m\rangle, |l\rangle$ where $m, l \in \{0, 1, ..., N-1\}$. We add an ancilla qubits to this system and consider the agumented system $\sum_{j=0}^{N-1} y_j|j\rangle|0\rangle$. We flag the desired states $|m\rangle$ and $|l\rangle$ using the multi-controlled C-NOT gate (c.f., Fig. 9). We end up with states

$$\sum_{j=0, j\neq m, j\neq l}^{N-1} y_j|j\rangle|0\rangle + y_m|m\rangle|1\rangle + y_l|l\rangle|1\rangle \tag{90}$$

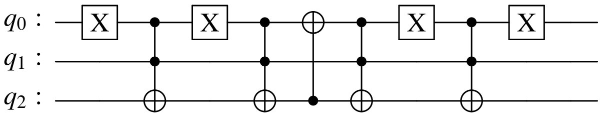


Figure 9: An example base-swap circuit used for imposing the boundary conditions. This circuits swap the amplitudes corresponding to basis $|0\rangle|1\rangle$ and $|1\rangle|1\rangle$.

We then conduct a controlled-swap of the qubits in $|m\rangle$ and $|l\rangle$, conditioned on the ancilla qubits taking the value $|1\rangle$. Afterwards, we uncompute the multi-controled C-NOT gate and shall end up with our desired state.

$$\sum_{j=0,j\neq m,j\neq l}^{N-1} y_j|j\rangle|0\rangle + y_m|l\rangle|0\rangle + y_l|m\rangle|0\rangle \tag{91}$$

An example circuit that swap the basis $|0\rangle|1\rangle$ and $|1\rangle|1\rangle$ is shown in Fig. 9.

Complexity Key to the implementation of the general base-swap circuit is the multi-controlled C-NOT gate. The efficient implementation of a general multi-controlled unitary operator is a research topic of current interest in the quantum computation community. In this work, we took the ”V-chain” approach proposed by Nelson and Chuang [1]. Given $N = 2^n$ computational basis embedid with n qubits, the V-chain algorithm is known to require an additional $n-1$ ancilla qubits, with a total of $2n-1$ CNOT gate. We therefore conclude that one can realize the general base swap algorithm with $4n-7$ CNOT gate and at most $4n$ NOT gate. This is clearly a Poly $\log(N)$ scaling as the number of universal quantum gate in our circuit grows only logrithmatically with respect to the number of degree of freedom N.

5. Quantum computational homogenisation

We proceed to the implementation of the periodic homogenisation approach introduced in Section 2 on a quantum computer with the help of the quantum algorithms described in Section 4. As detailed in Section 2.3, we approximate all periodic fields using band-limited Fourier interpolation. Consequently, we solve linear differential equations by solving a set of decoupled algebraic equations in the Fourier space. The transformations between the physical and Fourier spaces are accomplished with QFT. To ease the discussion of the proposed approach, we first consider the solution of periodic Poisson problems in 1D and 2D and, subsequently, the homogenisation problem on an RVE.

All the proposed quantum algorithms are implemented using the Python-based Qiskit SDK [13] and executed on a noiseless simulator. We verify the correctness of the implementations by comparing the quantum solution encoded in the output state vector with analytical and classical numerical solutions.

5.1. Periodic Poisson problems

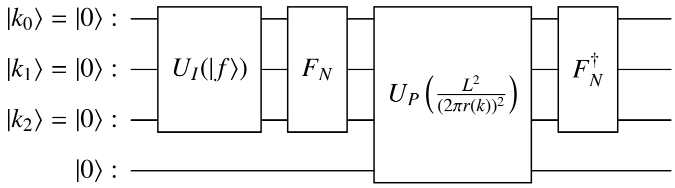
5.1.1. One-dimensional case

We seek the solution of the one-dimensional periodic Poisson problem

$$-\frac{\mathrm{d}^2 v(x)}{\mathrm{d}x^2} = f(x)\,, \tag{92}$$

The solution $v(x)$ and the source $f(x)$ have the periodicity $L = 1$, i.e. $v(x) = v(x+L)$ and $f(x) = f(x+L)$. For solving this problem, it is sufficient to consider the domain $\Omega = (0, L) \in \mathbb{R}$, which is discretised with a uniform grid of N cells of size h. A grid point with the index $k \in \{0, 1, \dots, N-1\}$ has the coordinate $x_k = kh$ and the source value $f_k = f(x_k)$. The source values at the N grid points are collected in the ket

$$|f\rangle = \sum_{k=0}^{N-1} f_k\,|k\rangle\ . \tag{93}$$



$|k_0\rangle = |0\rangle$:
$|k_1\rangle = |0\rangle$:
$|k_2\rangle = |0\rangle$:
$|0\rangle$:
$U_I(|f\rangle)$
F_N
$U_P\left(\frac{L^2}{(2\pi r(k))^2}\right)$
$F_N^\dagger$

Figure 10: Quantum circuit for solving the one-dimensional periodic Poisson problem discretised with a uniform grid of $N = 8$ cells. Spatial and Fourier coefficients at the $N = 8$ grid points are encoded as the amplitudes of a quantum state of the $n = 3$ qubits $|k_0\rangle\,|k_1\rangle\,|k_2\rangle$. The left most gate U_I encodes the source $|f\rangle$ as a quantum state. The second gate applies QFT to determine $\widehat{|f\rangle}$. In the Fourier space the Poisson problem is solved by scaling the Fourier components of $\widehat{|f\rangle}$ with the the third gate U_P. The last gate applies the inverse QFT to obtain the solution $|v\rangle$. The unnamed last qubit is an ancillary needed internally by U_P as discussed in Section **??**.

The QFT of this source is defined as

$$\widehat{|f\rangle} = F_N\,|f\rangle\,, \tag{94}$$

or more explicitly,

$$\sum_{j=0}^{N-1} \hat{f}_j\,|j\rangle = F_N \sum_{k=0}^{N-1} f_k\,|k\rangle\ ; \tag{95}$$

see Section 4.5. Following the derivation in Sections 2.2 and 2.3, it is easy to see that the solution of the periodic Poisson problem in the Fourier space is given by

$$\hat{v}_j = \left(\frac{L}{2\pi r(j)}\right)^2 \hat{f}_j\,, \tag{96}$$

where the relabelling function $r(j)$ is defined as in (19). The inverse DFT of the Fourier coefficients $\{\hat{v}_j\}_{j=0}^{N-1}$ is the solution at the grid points, i.e.,

$$|v\rangle = F_N^\dagger \widehat{|v\rangle} = F_N^\dagger \sum_{j=0}^{N-1} \hat{v}_j\,|j\rangle\ . \tag{97}$$

Furthermore, according the band-limited interpolation equation (17) the approximate solution field is given by

$$v^h(x) = \frac{1}{N}\sum_{l=0}^{N-1} \hat{v}_l e^{i\frac{2\pi r(l)}{L}x}\,. \tag{98}$$

The proposed quantum circuit for solving the one-dimensional Poisson problem is depicted in Figure 10 for $N = 8$ grid cells. The circuit has been implemented using the Qiskit SDK. It has one ancillary qubit and three qubits $|k\rangle \equiv |k_0 k_1 k_2\rangle$ for representing the source $|f\rangle$ and solution $|v\rangle$. All qubits are initialised in the state $|0\rangle$. The first unitary U_I prepares a quantum state representing the components of the source $|f\rangle$. U_I can be chosen as the state preparation technique introduced in Section 4.3 or one of the many other known approaches. The unitaries F_N and $F_N^\dagger$ apply the QFT and its inverse. Finally the unitary U_P divides the Fourier coefficients by $L^2/(2\pi r(l))^2$.

More succinctly, the sequence of the mappings implemented by the circuit with $n = 3$ qubits can be summarised as follows:

$$U_I(|f\rangle)\otimes I\colon \quad |0\rangle^{\otimes 3}|0\rangle \mapsto \sum_{k=0}^{7} f_k |k\rangle |0\rangle \ , \tag{99a}$$

$$F_N\otimes I\colon \quad \sum_{k=0}^{7} f_k |k\rangle |0\rangle \mapsto \sum_{j=0}^{7} \hat{f}_j |j\rangle |0\rangle \ , \tag{99b}$$

$$U_P\left(\frac{L^2}{(2\pi r(j))^2}\right)\colon \quad \sum_{j=0}^{7} \hat{f}_j |j\rangle |0\rangle \mapsto Junk + \sum_{j=0}^{7} \left(\frac{L}{2\pi r(j)}\right)^2 \hat{f}_j |j\rangle |1\rangle \ , \tag{99c}$$

$$F_N^\dagger\otimes I\colon \quad Junk + \sum_{j=0}^{7} \left(\frac{L}{2\pi r(j)}\right)^2 \hat{f}_j |j\rangle |1\rangle \mapsto Junk + \sum_{k=0}^{7} \left(\frac{L}{2\pi r(k)}\right)^2 f_k |l\rangle |1\rangle \ . \tag{99d}$$

In this derivation, all the irrelevant terms for the final solution are denoted as *Junk*.

Next, we study the convergence properties and the computational complexity of solving a Poisson problem with the source

$$f(x) = \exp\left(-\frac{(x-c_0)^2}{c_1^2}\right) + c_2 \, , \tag{100}$$

where the three parameters are chosen as $c_0 = 0.3$, $c_1 = 0.1$, $c_2 = -0.1772$. This function is depicted in Figure 11(a) and is approximated as periodic. The respective analytical solution is given by

$$v(x) = \frac{c_1\sqrt{\pi}}{2}\left((x-c_0)\operatorname{erf}\left(\frac{x-c_0}{c_1}\right) + \frac{c_1}{\sqrt{\pi}}\exp\left(-\frac{(x-c_0)^2}{c_1^2}\right)\right) + \frac{c_2x^2}{2} + c_3x + c_4 \, . \tag{101}$$

The periodic boundary conditions $u(0) = u(1) = 0$ yield for the two integration constants $c_3 = 0.053$ and $c_4 = -0.0266$. The

In Figure 11(b) the analytic solution $v(x)$ and the approximate solution $v^h(x)$ obtained with quantum computing are compared, indicating an excellent agreement between the two. The convergence of the L_2-norm error for an increasing number of grid cells is shown in Figure 11(c). The error is defined as

$$\eta = \| \, |v\rangle - |v^{ex}\rangle \, \| \, , \tag{102}$$

where $|v^{ex}\rangle = \sum_{k=0}^{N-1} v(x_k)\,|k\rangle$ is the exact solution at the grid points. The relative error has a convergence order of $1/2$ which is mainly governed by the piecewise polynomial approximation in the unitary $U_P(L^2/(2\pi r(l))^2)$.

Computational Complexity Finally, we assess the computational complexity of the proposed algorithm by expressing the quantum circuit using only the single-qubit rotation gate $U3$ and the two-qubit controlled gate $CNOT$. This two gates are universal in the sense that any arbitrary multi-qubit unitary can be composed using these two gates. In quantum computing the process of expressing a given circuit in terms of a different gate set is referred to as *transpiling* and can be automatically performed in Qiskit. The total number of $U3$ and $CNOT$ gates in dependence of the number of grid cells N is plotted in Figure 11(c), showing a logarithmic scaling for both the $U3$ and $CNOT$ gates. This confirms that the proposed algorithm's computational complexity is $\mathcal{O}\,(\text{Poly}(\log(N)))$.

5.1.2. Two-dimensional case

We introduce next the discretisation of a two-dimensional periodic Poisson problem

$$-\frac{\partial^2 v(\boldsymbol{x})}{\partial x_0^2} - \frac{\partial^2 v(\boldsymbol{x})}{\partial x_1^2} = f(\boldsymbol{x}) \, . \tag{103}$$

The solution and source fields $v(\boldsymbol{x})$ and $f(\boldsymbol{x})$ have the periodicity $L = 1$ and must satisfy $v(\boldsymbol{x}) = v(\boldsymbol{x} + \boldsymbol{e}_0 L) = v(\boldsymbol{x} + \boldsymbol{e}_1 L)$ and $f(\boldsymbol{x}) = f(\boldsymbol{x} + \boldsymbol{e}_0 L) = f(\boldsymbol{x} + \boldsymbol{e}_1 L)$, where $\boldsymbol{e}_0 = (1 \quad 0)^\mathsf{T}$ and $\boldsymbol{e}_1 = (0 \quad 1)^\mathsf{T}$ are the standard basis vectors.

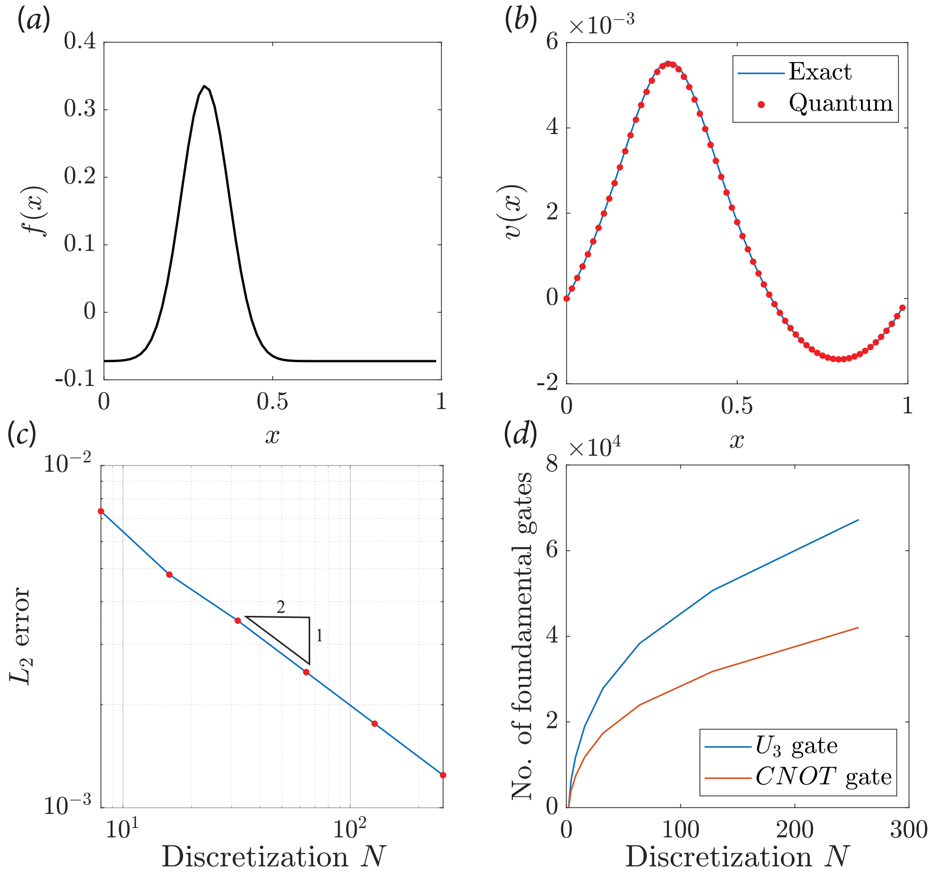


Figure 11: One-dimensional periodic Poisson problem. (a) Source function $f(x)$. (b) Comparison of the quantum solution on a grid with 64 cells and the exact (analytical solution). (c) Convergence of the relative L_2-norm. (d) Scaling of the total number of $U3$ and $CNOT$ gates.

We discretise the domain $\square = (0,\, L)^2 \subset \mathbb{R}^2$ by uniformly partitioning with $N \times N$ cells of size $h \times h$. Consequently, a grid point with the multi-index $\boldsymbol{k} = (k^0,\, k^1)$, where $k^0, k^1 \in \{0,\, 1,\, \dots,\, N-1\}$, has the coordinates

$$\boldsymbol{x}_{\boldsymbol{k}} = \begin{pmatrix} x_{0,\boldsymbol{k}} & x_{1,\boldsymbol{k}} \end{pmatrix}^{\mathsf{T}} = \begin{pmatrix} x_{0,k^0k^1} & x_{1,k^0k^1} \end{pmatrix}^{\mathsf{T}} , \tag{104}$$

and the source value $f_{\boldsymbol{k}} = f(\boldsymbol{x}_{\boldsymbol{k}})$. We collect the source values in the ket

$$|f\rangle = \sum_{\boldsymbol{k}=0}^{N-1} f_{\boldsymbol{k}} \, |k\rangle = \sum_{k^0=0,k^1=0}^{N-1} f_{k^0k^1} \, |k^0k^1\rangle \; . \tag{105}$$

As in standard DFT, the two-dimensional QFT is the Kronecker product of two one-dimensional QFT's $F_{0,N}$ and $F_{1,N}$. Hence, we can compactly write

$$\widehat{|f\rangle} = (F_{0,N} \otimes F_{1,N}) \, |f\rangle \; ; \tag{106}$$

or in more in detail,

$$\sum_{j^0=0,j^1=0}^{N-1} \hat{f}_{j^0j^1} \, |j^0j^1\rangle = (F_{0,N} \otimes F_{1,N}) \sum_{k^0=0,k^1=0}^{N-1} f_{k^0k^1} \, |k^0k^1\rangle \; . \tag{107}$$

Here, $F_{0,N}$ and $F_{1,N}$ denote the QFT with respect to the first and second indices of the multi-index. Building on the periodic one-dimensional Poisson problem introduced before, the solution of the two-dimensional problem in the Fourier space is given by

$$\hat{v}_{j^0j^1} = \left(\left(\frac{L}{2\pi r(j^0)} \right)^2 + \left(\frac{L}{2\pi r(j^1)} \right)^2 \right) \hat{f}_{j^0j^1} \; ; \tag{108}$$

where $r(j^0)$ and $r(j^1)$ are defined as in (19). Finally, the inverse DFT of the computed Fourier coefficients yields the solution at the grid points, i.e.,

$$|v\rangle = (F^{\dagger}_{0,N} \otimes F^{\dagger}_{0,N}) \widehat{|v\rangle} = (F^{\dagger}_{0,N} \otimes F^{\dagger}_{0,N}) \sum_{j^0=0j^1=0}^{N-1} \hat{v}_{j^0j^1} \, |j^0j^1\rangle \; . \tag{109}$$

The quantum circuit for solving the two-dimensional Poisson problem can be designed by closely following the one-dimensional construction introduced in the previous section. As an example, in Figure 12 the circuit for a grid with 4×4 cells is shown. In mapping the source $|f\rangle$ and other kets to qubits we make use of the following expansion of the multi-indices

$$|\boldsymbol{k}\rangle = |k^0\rangle \, |k^1\rangle = |k^0_0 k^0_1\rangle \, |k^1_0 k^1_1\rangle \equiv |k^0_0\rangle \, |k^0_1\rangle \, |k^1_0\rangle \, |k^1_1\rangle \; . \tag{110}$$

After the second equality sign we express the two integers $k^0, k^1 \in \{0,\, 1,\, 2,\, 3\}$ as binaries. In Figure 12 the unitary U_I for state preparation and the unitaries $F_{0,N}$ and $F_{1,N}$ and their inverses for the QFT are equivalent to the unitaries in the circuit for the one-dimensional problem in Figure 10. The main difference concerns the unitary U_P which encodes a bivariate polynomial for implementing (108). Importantly, owing to the quantum parallelism, the QFT's with respect to the two multi-indices can be applied simultaneously as evident from the quantum circuit. The efficiency implications of this parallelism to high-dimensional problems are evident.

As a concrete example, we consider on a grid of size $N \times N = 64 \times 64$ the solution of a Poisson problem with the source

$$f(x_0,\, x_1) = \sum_{l=0}^{M} c_{0,l} \sin(c_{1,l}\pi x_0 + c_{2,l}) \sin(c_{3,l}\pi x_1 + c_{4,l}) \, , \tag{111}$$

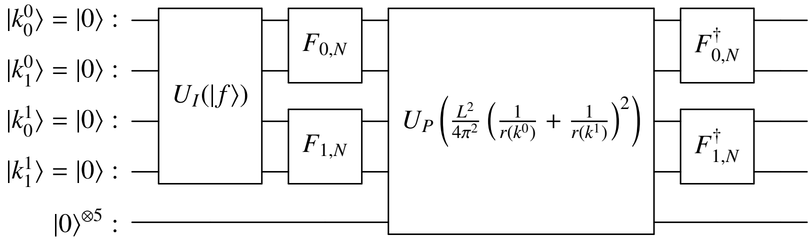

$|k_0^0\rangle = |0\rangle$:
$|k_1^0\rangle = |0\rangle$:
$|k_0^1\rangle = |0\rangle$:
$|k_1^1\rangle = |0\rangle$:
$|0\rangle^{\otimes 5}$:
$U_I(|f\rangle)$
$F_{0,N}$
$F_{1,N}$
$U_P\left(\frac{L^2}{4\pi^2}\left(\frac{1}{r(k^0)} + \frac{1}{r(k^1)}\right)^2\right)$
$F_{0,N}^\dagger$
$F_{1,N}^\dagger$

Figure 12: Quantum circuit for solving the two-dimensional periodic Poisson problem discretised with a uniform grid of $N \times N = 4 \times 4$ cells. Spatial and Fourier coefficients at the $N^2 = 16$ grid points are encoded as the amplitudes of a quantum state of the $n = 4$ qubits $|k_0^0\rangle\, |k_1^0\rangle\, |k_0^1\rangle\, |k_1^1\rangle$. The left most gate U_I encodes the source $|f\rangle$ as a quantum state. The second layer of gates $F_{0,N}$ and $F_{1,N}$ apply simultaneously the QFT with respect to the multi-indices k^0 and k^1. In the Fourier space the Poisson problem is solved by scaling the Fourier components of $\widehat{|f\rangle}$ with the gate U_P. The last two gates apply the inverse QFT to obtain the solution $|v\rangle$. The unnamed last five qubits are ancillaries needed by U_P, see Section 4.

where $c_{0,l}$, $c_{2,l}$, $c_{4,l} \in \mathbb{R}$ and $c_{1,l}$, $c_{3,l} \in \mathbb{Z}$ are five random parameters. The real parameters are sampled from a uniform probability measure $\mathcal{D}_R(-1, 1)$ and the integer parameters from a uniform probability measure $\mathcal{D}_Z(-20, 20)$. An example $f(x_0, x_1)$ with $M = 3$ is shown in Fig 13(a). The approximate solution $|v\rangle$ is depicted in Figure 13(b), and the corresponding relative absolute error distribution is shown in 13 (c). The true solution is obtained by solving the same problem using a classical solver on a fine grid of size 1024 × 1024. We observe the high accuracy of the proposed quantum algorithm with the largest relative error around 1%. Finally, we assess our algorithm's complexity by counting total number of the U_3 and CNOT gates for an increasing number of grid cells N^2, see Figure 13(d). We again observe a $O(\text{poly} \log(N^2))$ scaling as evident from the plot.

5.2. Solution of the unit cell problem

We consider the solution of the periodic homogenisation problem on a quantum computer. The RVE is a one-dimensional domain $\square = (0, L = 1) \subset \mathbb{R}$ with piecewise constant shear modulus $\mu(x)$ as depicted in Figure 16(a). This problem represents the homogenisation of an elastic composite rod with a periodic Young's modulus $\mu(x)$. As discussed in Section 2, homogenisation aims to determine the stress field $\sigma(x)$ for a prescribed scalar applied strain $\overline{\gamma}$. We achieve this by solving a sequence of Poisson problems as outlined in Algorithm 1. In the following, we first introduce our quantum circuit for computing one iteration step and subsequently explain how to execute several steps.

5.2.1. Incremental problem

We closely follow Algorithm 1 in devising a quantum circuit for solving the incremental homogenisation problem. Assuming that the RVE is discretised with $N = 2^n$ cells, the quantum state vector $|q\rangle \in \mathbb{C}^{N+3}$ for implementing our circuit has the following structure

$$|q\rangle = \underbrace{|0\rangle^{\otimes n}}_{\text{field}} \underbrace{|a_0 = 0\rangle\, |a_1 = 0\rangle}_{\text{controls}} \underbrace{|b = 0\rangle}_{\text{AS}} \underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} , \tag{112}$$

where the first n qubits encode field vectors, like the shear modulus $|\mu\rangle \in \mathbb{R}^N$ or strain $|\gamma\rangle \in \mathbb{R}^N$, the qubits $|a_0\rangle$ and $|a_1\rangle$ control the application of unitaries and the qubit $|b\rangle$ is used to encode applied strain $\overline{\gamma} \in \mathbb{R}$. The last n qubits serve as helpers throughout the circuit.

In the initialisation step of Algorithm 1, we encode the prescribed applied strain $\overline{\gamma}$ and the predictor strain $|\gamma\rangle^{(s=0)} \in \mathbb{R}^N$ as amplitudes of the state vector $|q\rangle^{(s=0)}$. The applied strain $\overline{\gamma}$ is encoded as the amplitude of $|b\rangle$. This is easily accomplished by applying the rotation $R_Y(\theta)$ with $\theta = 2 \arcsin(\sqrt{N}\overline{\gamma})$ to qubit $|b\rangle$ so that the quantum state becomes

$$|q\rangle_1^{(0)} = \sqrt{N}\overline{\gamma} \underbrace{|0\rangle^{\otimes n}}_{\text{field}} \underbrace{|0\rangle\, |0\rangle}_{\text{controls}} \underbrace{|1\rangle}_{\text{AS}} \underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} + \sqrt{1 - (\sqrt{N}\overline{\gamma})^2} \underbrace{|0\rangle^{\otimes n}}_{\text{field}} \underbrace{|0\rangle\, |0\rangle}_{\text{controls}} \underbrace{|0\rangle}_{\text{AS}} \underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} . \tag{113}$$

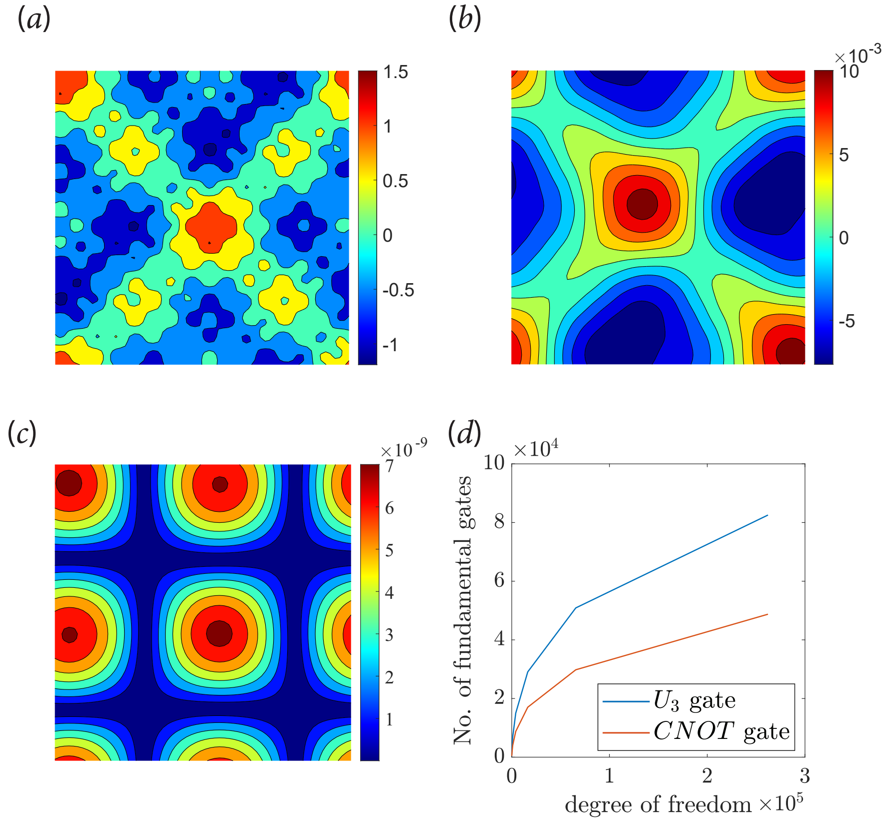


Figure 13: Two-dimensional periodic Poisson problem. (a) Source function $f(x)$ (b) Comparison of the quantum solution on a grid with $N \times N = 8 \times 8$ cells and a solution (analytical solution) (c) Convergence of the relative L_2-norm of the quantum solution. (d) Scaling of the total number of $U3$ and $CNOT$ gates.

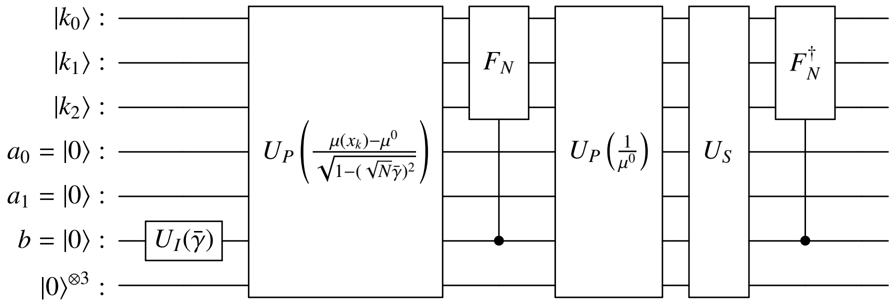


Figure 14: Quantum circuit for solving the incremental homogenisation problem discretised with a uniform grid of $N = 2^3$ cells. The left most gate U_I encodes the prescribed average strain $\bar{\gamma}$ as one of the two amplitudes of the $|b\rangle$ qubit. The two gates U_P approximately implement the functions in their arguments. The incremental problem is solved by scaling the Fourier coefficients using the right U_P gate. The gate U_S is to impose in the Fourier space the average prescribed strain.

The predictor strain $|\gamma\rangle^{(0)}$ can be encoded by applying a standard unitary $U_I(|\gamma\rangle^{(0)})$ for state preparation conditioned on $|b = 0\rangle$. The resulting new state reads

$$|q\rangle_2^{(0)} = \sqrt{N}\bar{\gamma}\,\underbrace{|0\rangle^{\otimes n}}_{\text{field}}\,\underbrace{|0\rangle|0\rangle}_{\text{controls}}\,\underbrace{|1\rangle}_{\text{AS}}\,\underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} + \sqrt{1-(\sqrt{N}\bar{\gamma})^2}\,\underbrace{\sum_{k=0}^{2^n-1}\gamma_k^{(s)}|k\rangle}_{\text{field}}\,\underbrace{|0\rangle|0\rangle}_{\text{controls}}\,\underbrace{|0\rangle}_{\text{AS}}\,\underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} \tag{114}$$

We are now ready to introduce the quantum circuit for solving the incremental problem in iteration step s. It is instructive to read the following discussion in light of Algorithm 1 and circuit diagram in Figure 14. We implement step (a) of the algorithm using the unitary mapping

$$U_P\left(\frac{\mu(x_k)-\mu^0}{\sqrt{1-(\sqrt{N}\bar{\gamma})^2}}\right) : |k\rangle|0\rangle \mapsto \sqrt{1-\frac{(\mu(x_j)-\mu^0)^2}{1-(\sqrt{N}\bar{\gamma})^2}}\,|k\rangle|0\rangle + \frac{(\mu(x_j)-\mu^0)}{\sqrt{1-(\sqrt{N}\bar{\gamma})^2}}\,|k\rangle|1\rangle\,. \tag{115}$$

The unitary U_P encodes the function in its argument by first approximating the function as a polynomial using Chebyshev interpolation. The unitary U_P is applied to the field qubits of $|q\rangle_2^{(0)}$ when the AS qubit is in state $|b = 0\rangle$. The resulting state reads

$$|q\rangle_3^{(s)} = \sqrt{N}\bar{\gamma}\,\underbrace{|0\rangle^{\otimes n}}_{\text{field}}\,\underbrace{|0\rangle|0\rangle}_{\text{controls}}\,\underbrace{|1\rangle}_{\text{AS}}\,\underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} + \underbrace{\sum_{k=0}^{2^n-1}\tau_k^{(s)}|k\rangle}_{\text{field}}\,\underbrace{|1\rangle|0\rangle}_{\text{controls}}\,\underbrace{|0\rangle}_{\text{AS}}\,\underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} + JUNK\,. \tag{116}$$

Here, $JUNK$ denotes the terms not further needed. The subsequent application of QFT by applying the unitary F_N to the field qubits gives Fourier space representation of the polarisation stress $|\tau\rangle^{(s)}$, i.e.,

$$|q\rangle_4^{(s)} = \sqrt{N}\bar{\gamma}\,\underbrace{|0\rangle^{\otimes n}}_{\text{field}}\,\underbrace{|0\rangle|0\rangle}_{\text{controls}}\,\underbrace{|1\rangle}_{\text{AS}}\,\underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} + \underbrace{\sum_{k=0}^{2^n-1}\hat{\tau}_k^{(s)}|k\rangle}_{\text{field}}\,\underbrace{|1\rangle|0\rangle}_{\text{controls}}\,\underbrace{|0\rangle}_{\text{AS}}\,\underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} + JUNK\,. \tag{117}$$

We can now compute the strain $\hat{\gamma}^{(s+1)}$. Note that for a one-dimensional problem the strain in the Fourier space is given by

$$|\hat{\gamma}\rangle = -\frac{1}{\mu^0}\,|\hat{\tau}\rangle \; ; \tag{118}$$

cf. (24). This operation is implemented using one more time the unitary mapping

$$U_P\left(-\frac{1}{\mu^0}\right): \; |k\rangle\,|0\rangle \mapsto \sqrt{1-\frac{1}{(\mu^0)^2}}\,|k\rangle\,|0\rangle - \frac{1}{\mu^0}\,|k\rangle\,|1\rangle \; . \tag{119}$$

Although we used the same symbol U_P above and in (115) the two unitaries approximate different functions and their implementation details will be different. The unitary U_P is applied to the field qubits when the first controls qubit is in state $|a_0 = 1\rangle$, yielding the state

$$|q\rangle_5^{(s)} = \sqrt{N}\bar{\gamma}\,\underbrace{|0\rangle^{\otimes n}}_{\text{field}}\,\underbrace{|0\rangle\,|0\rangle}_{\text{controls}}\,\underbrace{|1\rangle}_{\text{AS}}\,\underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} - \underbrace{\sum_{k=0}^{2^n-1}\frac{\hat{\tau}_k^{(s)}}{\mu^0}\,|k\rangle}_{\text{field}}\,\underbrace{|1\rangle\,|1\rangle}_{\text{controls}}\,\underbrace{|0\rangle}_{\text{AS}}\,\underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} + JUNK\,. \tag{120}$$

To impose the average strain $\overline{\gamma}$ we use the unitary U_S to swap as follows two of the amplitudes

$$\begin{aligned} U_S:\; \sqrt{N}\bar{\gamma}\,|0\rangle^{\otimes n}\,|0\rangle\,|0\rangle\,|1\rangle\,|0\rangle^{\otimes n} - \frac{\hat{\tau}_0^{(s)}}{\mu^0}\,|k=0\rangle\,|1\rangle\,|0\rangle\,|0\rangle\,|0\rangle^{\otimes n} + \ldots \\ \mapsto -\frac{\hat{\tau}_0^{(s)}}{\mu^0}\,|0\rangle^{\otimes n}\,|0\rangle\,|0\rangle\,|1\rangle\,|0\rangle^{\otimes n} + \bar{\gamma}\,|k=0\rangle\,|1\rangle\,|0\rangle\,|0\rangle\,|0\rangle^{\otimes n} + \ldots \,. \end{aligned} \tag{121}$$

After the amplitude swap the quantum state becomes

$$|q\rangle_6^{(s)} = -\frac{\hat{\tau}_0^{(s)}}{\mu^0}\,\underbrace{|0\rangle^{\otimes n}}_{\text{field}}\,\underbrace{|0\rangle\,|0\rangle}_{\text{controls}}\,\underbrace{|1\rangle}_{\text{AS}}\,\underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} + \bar{\gamma}\,\underbrace{|k\rangle}_{\text{field}}\,\underbrace{|1\rangle\,|1\rangle}_{\text{controls}}\,\underbrace{|0\rangle}_{\text{AS}}\,\underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} - \underbrace{\sum_{k=1}^{2^n-1}\frac{\hat{\tau}_k^{(s)}}{\mu^0}\,|k\rangle}_{\text{field}}\,\underbrace{|1\rangle\,|1\rangle}_{\text{controls}}\,\underbrace{|0\rangle}_{\text{AS}}\,\underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} + JUNK\,. \tag{122}$$

Finally, we compute the inverse QFT by applying $F_N^\dagger$ to the field qubits giving the desired state

$$|q\rangle_7^{(s)} = \underbrace{\sum_{k=0}^{2^n-1}\gamma_k^{(s+1)}\,|k\rangle}_{\text{field}}\,\underbrace{|1\rangle\,|1\rangle}_{\text{controls}}\,\underbrace{|0\rangle}_{\text{AS}}\,\underbrace{|0\rangle^{\otimes n}}_{\text{ancillas}} + JUNK\,. \tag{123}$$

The solution $|\gamma\rangle^{(s+1)}$ and its components can be obtained by measuring as outlined in Section 3.4.

5.2.2. Fixed-point iteration

We proceed with an explanation on how we achieved the fixed-point iteration for algorithm 1. A schematic of the over-all circuit is shown in Fig. 15. Let $S \in \mathbb{Z}^+$ denote the total number of iteration, we seek to store the solution in an expended system below,

$$\psi^S = \underbrace{|0\rangle}_{\text{n state qubits}}\;\underbrace{|a_{c0}^{s=0}a_{c1}^{s=0}\rangle|a_{c0}^{s=1}a_{c1}^{s=1}\rangle, ..., |a_{c0}^{s=2}a_{c1}^{s=2}\rangle}_{2S\text{ control qubits}}\;\underbrace{|b\rangle|0\rangle}_{\log(S)+1\text{ BC qubits}}\;\underbrace{|0\rangle}_{\text{n ancilla qubits}}\;. \tag{124}$$

The key idea is that we shall implement the incremental circuit (c.f. Fig. **??**) conditioned/controlled on the $2S$ control qubits, so that we store the solution of each iteration in different basis denoted by the a_c^s qubits.

We start with replacing the *Initialize BC* circuits in the incremental circuit (c.f. Fig.14), with a general initialization

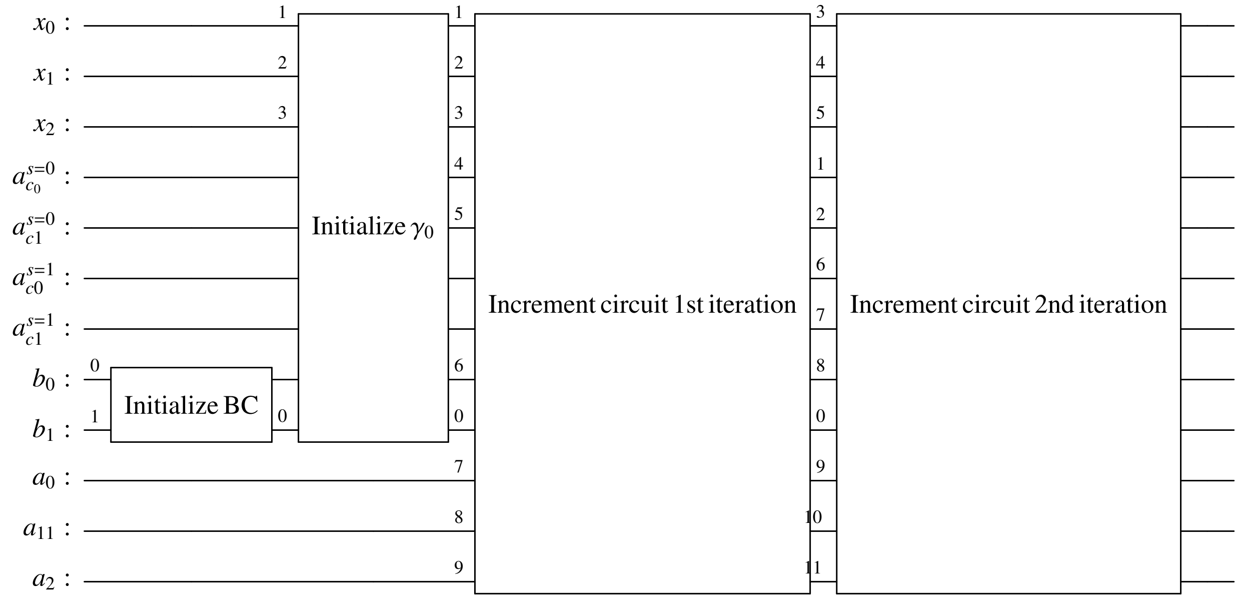


Figure 15: Schematic of the Fixed-point circuit for the QRVE algorithm with two iterations and 8 degree of freedom.

circuit where we employ a polynomial encoding algorithm to prepare the $|b_0\rangle$ nodes. Specifically, we construct a polynomial rotation circuit and prepare the initial state

$$\psi_0^S = \sqrt{1 - S\bar{\gamma}^2}|0\rangle|a_c = 0\rangle|b = 0\rangle|0\rangle|0\rangle + \sum_{s=0}^{S-1} \bar{\gamma}|0\rangle|a_c\rangle|b = s\rangle|1\rangle\rangle|0\rangle \tag{125}$$

.

We now initialize γ_0. This initialization can be arbitrary but we choose to set $\gamma_0(x) = \frac{1}{\sqrt{N}}, \forall x \in \Omega$ throughout this study. We now repeatedly implement the incremental circuit. After the first iteration the system's states becomes:

$$\psi_0^S = \sum_{j=0}^{N-1} \gamma_j^{s=0}|j\rangle|a_{c0}^{s=0}a_{c1}^{s=0} = 11\rangle|a_c^{s>0} = 0\rangle|b = 0\rangle|0\rangle|0\rangle + \sum_{s=1}^{S-1} \bar{\gamma}|0\rangle|a_c\rangle|b = s\rangle|1\rangle\rangle|0\rangle + ... \tag{126}$$

We now implement the second incremental circuit, and make it conditioned on the a_c qubits from the first iteration (i.e., $|a_{c0}^{s=0}a_{c1}^{s=0} = 11\rangle$)

The system's state then becomes

$$\psi_1^S = \sum_{j=0}^{N-1} \gamma_j^{s=0}|j\rangle|a_{c0}^{s=0}a_{c1}^{s=0} = 11\rangle|a_c^{s>0} = 0\rangle|b = 0\rangle|0\rangle|0\rangle + \sum_{s=2}^{S-1} \bar{\gamma}|0\rangle|a_c = 0\rangle|b = s\rangle|1\rangle\rangle|0\rangle + ... \tag{127}$$

After S iteration, the final sysmem's state becomes

$$\psi_S^S = \sum_{j=0}^{N-1} \gamma_j^{s=S-1}|j\rangle|a_{c0}^{s=1}a_{c1}^{s=1} = 11\rangle...|a_{c0}^{s=S-1}a_{c1}^{s=S-1} = 11\rangle|b = 0\rangle|0\rangle|0\rangle + ... \tag{128}$$

The solution is then built in the state in the basis with all control quibits $|a_c = 1\rangle$.

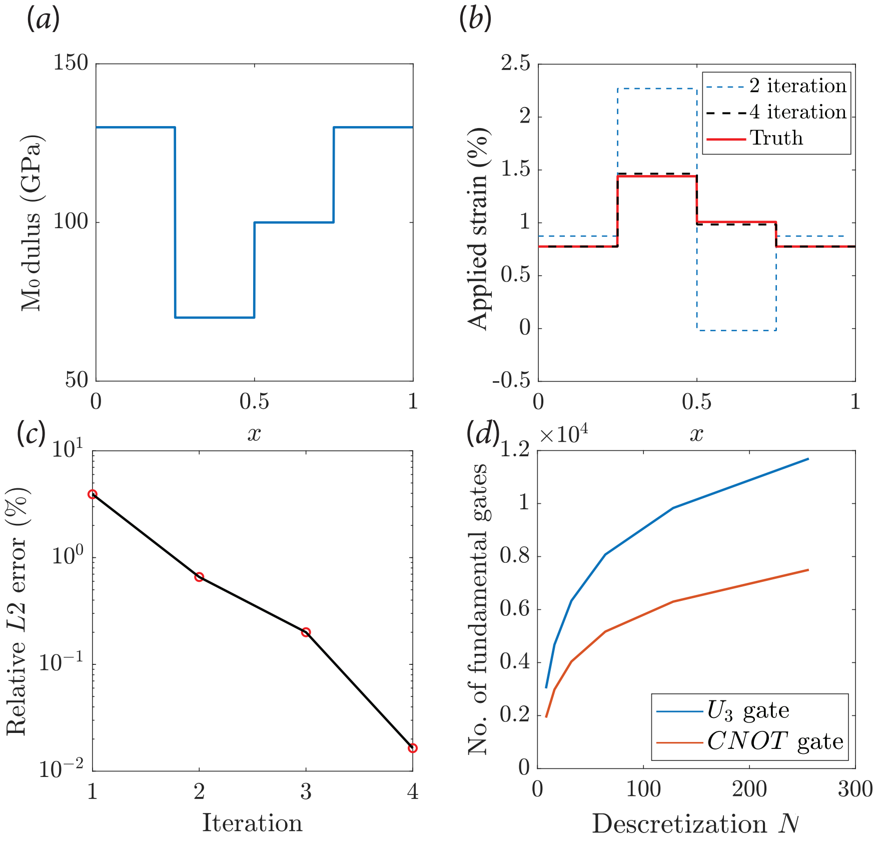


Figure 16: Result for the two dimension Poisson's equation. (a) source function $f(x, y)$ (b) Comprsion of QFT solution of 64 degree of freedom and truth (obtained with 2D FFT numerical solver on a classical computer) (c) Relative L_2 error versus the decretization of the domain Ω. (d) Number of universal quantum gates versus degree of freedom for incremental circuit shown in Fig. 14.

The QRVE result conducted using the abovementioned circuits is shown in Fig. 16 (b). We observe that the accuracy of the QRVE solution can be greatly improved by increasing number of iterations. This is further studied in Fig. 16 (c), with an exponential decrease in relative L_2 error observed with increasing number of iteration. Finally, we estimate the computational complexity of our algorithm by counting the number of universal gates in the incremental circuit, as shwon in Fig. 16 (d). The curve show that the scaling is logrithmatic meaning one only need Poly (log) (N) universal gates for the QRVE algorithm, hence proving the efficacy of our algorithm and circuit design.

6. Conclusions

We have presented a framework that enables exponential speed up in the computation homogenisation. In this framework, we encode the physical variables in the unit cell to the amplitudes of an entangled quantum system. The solution is then find by evoling this quantum system with a combination of unitary operators including the Quantum Fourier Transform and Polynomial Encoding. The effectiveness of the framework was validated through its application to one and two-dimensional Poisson's equations and the RVE problem for composite beams. In each instance, our framework exhibited exponential acceleration compared to traditional methods, a testament to the power of quantum computing in handling problems with a high degree of freedom.

The efficiency of our approach was particularly highlighted by the reduced requirement of only Poly log(N) universal quantum gates within our quantum computational circuits. This efficiency is not just a theoretical improvement but a practical step forward in computational mechanics, offering new possibilities for addressing previously intractable problems.

Looking to the future, this work opens up numerous research avenues. Among these are the extension of the quantum algorithm to two and three dimensions, as well as the development of algorithms for nonlinear materials and large deformations. This research not only paves the way for new computational strategies but also sets a precedent for the integration of quantum computing into broader scientific and engineering applications.

In conclusion, our quantum computational framework stands as a pioneering contribution to computational mechanics, offering a new paradigm that leverages the untapped potential of quantum computing. This work is a significant step towards addressing complex multi-scale problems, heralding a new era of computational capabilities and advancing our understanding and application of quantum mechanics in practical and theoretical scenarios.

Appendix A. Kronecker Product

The Kronecker product satisfies the following properties.

1. $(A \otimes C)(B \otimes D) = AB \otimes CD$
2. $(A \otimes C)(|u\rangle \otimes |w\rangle) = A\,|u\rangle \otimes C\,|w\rangle$
3. $(|u\rangle + |v\rangle) \otimes |w\rangle = |u\rangle\,|w\rangle \otimes |v\rangle \otimes |w\rangle$
4. $|u\rangle \otimes (|w\rangle + |x\rangle) = |u\rangle \otimes |w\rangle + |u\rangle \otimes |x\rangle$
5. $a\,|u\rangle \otimes b\,|w\rangle = ab\,|u\rangle \otimes |w\rangle$
6. $(A \otimes B)^\dagger = A^\dagger \otimes B^\dagger$

Appendix B. State preparation

In this section we provide an alternative state preparation approach as motivated by [3] and is closer to algorithms available in quantum libraries, like Qiskit. This approach is in contrast to the one introduced in Section 4.3 exact. To illustrate the basic idea, we consider the encoding of a classical vector $|q\rangle \in \mathbb{R}^{N=8}$ using the amplitudes of $n = 3$ qubits. The quantum state is prepared using the unitary mapping

$$|q\rangle = U_S\,|000\rangle\;. \tag{B.1}$$

For determining the unknown components of the matrix $U_S\mathbb{R}^{8\times 8}$ it is convenient to consider first the inverse mapping

$$|000\rangle = U_S^\dagger\,|q\rangle\;. \tag{B.2}$$

After determining $U_S^\dagger$ it is trivial to determine its conjugate transpose U_S. The components of $U_S^\dagger$ can be determined by applying either successive Givens rotations or Householder reflections well known from linear algebra, see e.g. [42]. In the following, we briefly sketch how the components of $U_S^\dagger$ can be obtained using Givens rotations. For the sake of clarity, the mapping (B.2) expressed in matrix notation reads

$$\begin{pmatrix}1\\0\\0\\0\\0\\0\\0\\0\end{pmatrix} = \begin{pmatrix}\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot\end{pmatrix}\begin{pmatrix}q_0\\q_1\\q_2\\q_3\\q_4\\q_5\\q_6\\q_7\end{pmatrix}, \tag{B.3}$$

where the yet unknown components are represented by a ”·”. Each Givens rotation represents a rotation in the plane spanned by two coordinate axis. In the first step, the rotation axis are chosen such that

$$\begin{pmatrix}c&s&0&0&0&0&0&0\\-s&c&0&0&0&0&0&0\\0&0&\cdot&\cdot&0&0&0&0\\0&0&\cdot&\cdot&0&0&0&0\\0&0&0&0&\cdot&\cdot&0&0\\0&0&0&0&\cdot&\cdot&0&0\\0&0&0&0&0&0&\cdot&\cdot\\0&0&0&0&0&0&\cdot&\cdot\end{pmatrix}\begin{pmatrix}q_0\\q_1\\q_2\\q_3\\q_4\\q_5\\q_6\\q_7\end{pmatrix} = \begin{pmatrix}r\\0\\\cdot\\0\\\cdot\\0\\\cdot\\0\end{pmatrix}. \tag{B.4}$$

Some of the non-zero components have been omitted for clarity. Notice that on the right-hand side that every other component is zero. The non-zero components of $U_s^\dagger$ in the upper left 2×2 block are given by

$$r = \sqrt{q_0^2 + q_1^2}\,, \quad c = \cos\beta = \frac{q_0}{r}\,, \quad s = \sin\beta = \frac{q_1}{r}\,. \tag{B.5}$$

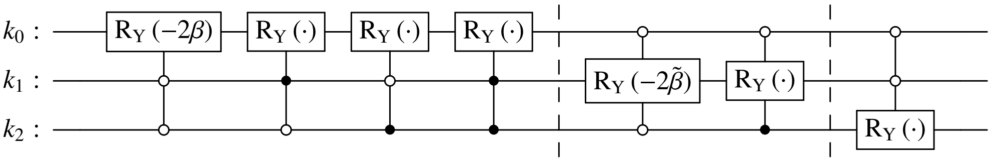


Figure B.17: Quantum circuit implementing the unitary $U_S^\dagger$ for mapping an arbitrary (real) state $|q\rangle = (q_0\ q_1\ q_2\ q_3\ q_4\ q_5\ q_6\ q_7)^\mathsf{T}$ to a state $|000\rangle = (1\ 0\ 0\ 0\ 0\ 0\ 0\ 0)^\mathsf{T}$. The two dashed helper lines correspond to the three distinct steps in constructing $U_S^\dagger$ mentioned in the main text. Furthermore, each R_Y is applied only to two selected components of the state vector. For instance, the most left R_Y gate is applied to the components q_0 and q_1, the second most left R_Y gate two the components q_2 and q_3, etc. The inverse circuit U_S for state preparation maps the state $|000\rangle$ to $|q\rangle$. The circuit for U_S is obtained by taking the conjugate transpose $R_Y^\dagger$ of each gate R_Y and applying the gates from right to the left.

The components of the other three 2 × 2 blocks are determined similarly. In the second step, the Givens rotations applied are chosen such that

$$\begin{pmatrix} \tilde{c} & 0 & \tilde{s} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\tilde{s} & 0 & \tilde{c} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdot & 0 & \cdot & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdot & 0 & \cdot & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} r \\ 0 \\ \cdot \\ 0 \\ \cdot \\ 0 \\ \cdot \\ 0 \end{pmatrix} = \begin{pmatrix} \tilde{r} \\ 0 \\ 0 \\ 0 \\ \cdot \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{B.6}$$

Notice that the right-hand side has now only two non-zero components. In the third and last step, the fifth component of the right-hand side can be set zero by applying one more Givens rotation yielding a vector with a 1 on its first component, i.e. $|000\rangle$.

Hence, the obtained unitary matrix $U^\dagger$ is multiplicatively composed of three unitary matrices and each matrix contains one or more of Givens rotations. Specifically, the fist matrix (B.3) contains three, the second matrix (B.6) two and the third matrix one Givens rotations. On a quantum computer each Givens rotation is realised using the R_Y (43). In order to apply each rotation to selected components of a vector we use the controlled versions of the R_Y gates. That is the R_Y gate is only applied when the control gate is in state $|1\rangle$. The circuit implementation of the introduced unitary $U^\dagger$ is depicted in Figure B.17. The circuit diagram for the inverse mapping $U^\dagger$ is obtained by taking the conjugate transpose of each R_Y and applying the gates from right to the left.

Appendix C. Complexity of the polynomial encoding algorithm

We provide an estimate of the computational complexity of the polynomial encoding circuit. Note in 71, the number of terms in $(2^{n-1}k_{n-1} + ... + 2^1 k_1 + 2^0 k_0)^m$ for a fixed m equals to $\binom{n+m-1}{n-1}$. The total number of terms with varying $m \in \{0, 1, ..., p\}$ can be computed as

$$\binom{n-1}{n-1} + \binom{n}{n-1} + \binom{n+1}{n-1} + ... + \binom{n+p-1}{n-1}. \tag{C.1}$$

Recall, for any $r \in \{0, ..., n\}$

$$\binom{n}{r} + \binom{n}{r+1} = \binom{n+1}{r+1}. \tag{C.2}$$

We conclude that the sum in C.1 equals to

$$\binom{n-1}{n-1} + \binom{n}{n-1} + \binom{n+1}{n-1} + ... + \binom{n+m-1}{n-1} \tag{C.3}$$

$$= \binom{n}{n} + \binom{n}{n-1} + \binom{n+1}{n-1} + ... + \binom{n+p-1}{n-1} \tag{C.4}$$

$$= \binom{n+1}{n} + \binom{n+1}{n-1} + ... + \binom{n+p-1}{n-1} \tag{C.5}$$

$$= \binom{n+p}{n} \tag{C.6}$$

Therefore, for one-dimensional and two dimensional polynomial encoding we conclude that we need a total of $\binom{n+p}{n}$ and $\binom{n+p}{n}\binom{n+q}{n}$ multi-controlled RY gate, respectively. Overall, we conclude that the polynomial encoding can be implemented with $O\left(\text{Poly}\left(pn\right)\right)$ in single variate polynomials of order and $O\left(\text{Poly}\left(pqn^2\right)\right)$ for bi-variate polynomials.

References

[1] Nielsen M A and Chuang I L. *Quantum Computation and Quantum Information*. Cambridge University Press, 10th anniversary edition, 2010.

[2] J Abhijith, A Adedoyin, J Ambrosiano, P Anisimov, A Bärtschi, Wi Casper, G Chennupati, C Coffrin, H Djidjev, D Gunter, S Karra, N Lemons, S Lin, A Malyzhenkov, D Mascarenas, S Mniszewski, B Nadiga, D O'Malley, D Oyen, S S Pakin, L Prasad, R Roberts, P Romero, N NSanthi, N Sinitsyn, P J Swart, J G Wendelberger, B Yoon, R Zamora, W Zhu, S Eidenbenz, A Bärtschi, P J Coles, M Vuffray, and A J Lokhov. Quantum algorithm implementations for beginners. *arXiv preprint arXiv:1804.03719*, 2018.

[3] I F Araujo, D K Park, F Petruccione, and A J da Silva. A divide-and-conquer algorithm for quantum state preparation. *Scientific Reports*, 11:6329, 2021.

[4] 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*, 52:3457–3467, 1995.

[5] P Benioff. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *Journal of Statistical Physics*, 22:563–591, 1980.

[6] C H Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 17:525–532, 1973.

[7] V Bergholm, J Izaac, M Schuld, C Gogolin, S Ahmed, V Ajith, M S Alam, G Alonso-Linaje, B AkashNarayanan, A Asadi, et al. Pennylane: Automatic differentiation of hybrid quantum-classical computations. *arXiv preprint arXiv:1811.04968*, 2018.

[8] J-P Berrut and L N Trefethen. Barycentric Lagrange interpolation. *SIAM Review*, 46:501–517, 2004.

[9] Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2019.

[10] C Canuto, M Y Hussaini, A Quarteroni, and T A Zang. *Spectral methods: fundamentals in single domains*. Springer, 2007.

[11] Cirq Developers. Cirq, 2023.

[12] Ray W Clough. Original formulation of the finite element method. *Finite elements in analysis and design*, 7(2):89–101, 1990.

[13] Qiskit contributors. *Qiskit: An Open-source Framework for Quantum Computing*, 2023.

[14] D Coppersmith. An approximate Fourier transform useful in quantum factoring. Technical Report RC 19642, IBM Research Division, 1994.

[15] D E Deutsch. Quantum computational networks. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 425:73–90, 1989.

[16] Ivan H Deutsch. Harnessing the power of the second quantum revolution. *PRX Quantum*, 1(2):020101, 2020.

[17] D P DiVincenzo. The physical implementation of quantum computation. *Progress of Physics*, 48:771–783, 2000.

[18] R P Feynman. Quantum mechanical computers. *Foundations of Physics*, 16:507–532, 1986.

[19] Richard P Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21:467–488, 1982.

[20] C Gierden, J Kochmann, J Waimann, B Svendsen, and S Reese. A review of FE-FFT-based two-scale methods for computational modeling of microstructure evolution and macroscopic material behavior. *Archives of Computational Methods in Engineering*, 29(6):4115–4135, 2022.

[21] L K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, pages 212–219, 1996.

[22] Thomas Häner. *Code Optimization for Quantum Computing*. PhD thesis, ETH Zurich, 2018.

[23] Mark Horowitz and Emily Grumbling. *Quantum computing: progress and prospects*. The National Academies Press, 2019.

[24] R Jozsa and N Linden. On the role of entanglement in quantum-computational speed-up. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 459:2011–2032, 2003.

[25] P Kaye, R Laflamme, and M Mosca. *An Introduction to Quantum Computing*. OUP Oxford, 2006.

[26] N Kohl, D Bauer, F Böhm, and U Rüde. Fundamental data structures for matrix-free finite elements on hybrid tetrahedral grids. *International Journal of Parallel, Emergent and Distributed Systems*, 0(0):1–24, 2023.

[27] R Landauer. Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 5(3):183–191, 1961.

[28] Yuri Manin. Computable and Uncomputable. *Sovetskoye Radio, Moscow*, 128, 1980.

[29] J-C Michel, H Moulinec, and P Suquet. Effective properties of composite materials with periodic microstructure: a computational approach. *Computer Methods in Applied Mechanics and Engineering*, 172:109–143, 1999.

[30] M Mottonen, J J Vartiainen, V Bergholm, and M M Salomaa. Transformation of quantum states using uniformly controlled rotations. *Quantum Information & Computation*, 5:467–473, 2005.

[31] H Moulinec and P Suquet. A numerical method for computing the overall response of nonlinear composites with complex microstructure. *Computer Methods in Applied Mechanics and Engineering*, 157:69–94, 1998.

[32] Toshio Mura. *Micromechanics of defects in solids*. Martinus Nijhoff Publishers, 1987.

[33] David Sena Oliveira and Rubens Viana Ramos. Quantum bit string comparator: circuits and applications. *Quantum Comput. Comput*, 7(1):17–26, 2007.

[34] John Preskill. Quantum computing 40 years later. In *Feynman Lectures on Computation*, pages 193–244. CRC Press, 2023.

[35] E G Rieffel and W H Polak. *Quantum Computing: A Gentle Introduction*. MIT Press, 2011.

[36] J J Sakurai. *Modern Quantum Mechanics*. Addison-Wesley, 1994.

[37] John Shalf. The future of computing beyond Moore's Law. *Philosophical Transactions of the Royal Society A*, 378:20190061, 2020.

[38] V V Shende, S S Bullock, and I L Markov. Synthesis of quantum logic circuits. In *Proceedings of the 2005 Asia and South Pacific Design Automation Conference*, 2005.

[39] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41, 1999.

[40] R S Smith, M J Curtis, and W J Zeng. A practical quantum instruction set architecture, 2016.

[41] N Stamatopoulos, D J Egger, Y Sun, C Zoufal, R Iten, N Shen, and S Woerner. Option pricing using quantum computers. *Quantum*, 4:1–20, 2020.

[42] G Strang. *Introduction to Linear Algebra*. Wellesley-Cambridge Press, 2016.

[43] L N Trefethen. *Spectral methods in MATLAB*. SIAM, 2000.

[44] A C Vazquez, R Hiptmair, and S Woerner. Enhancing the quantum linear systems algorithm using Richardson extrapolation. *ACM Transactions on Quantum Computing*, 3:1–37, 2022.

[45] S Woerner and D J Egger. Quantum risk analysis. *npj Quantum Information*, 5:1–15, 2019.

[46] T G Wong. *Introduction to Classical and Quantum Computing*. Rooted Grove, 2022.