scientific reports

Check for updates

OPEN

# A fast quantum algorithm for solving partial differential equations

Azim Farghadan[1], Mohammad Mahdi Masteri Farahani[1] & Mohsen Akbari[2]✉

**The numerical solution of partial differential equations (PDEs) is essential in computational physics. Over the past few decades, various quantum-based methods have been developed to formulate and solve PDEs. Solving PDEs incurs high-time complexity for high-dimensional real-world problems, and using traditional methods becomes practically inefficient. This paper presents a fast hybrid classical-quantum paradigm based on successive over-relaxation (SOR) to accelerate solving PDEs. Using the discretization method, this approach reduces the PDE solution to solving a system of linear equations, which is then addressed using the block SOR method. The block SOR method is employed to address qubit limitations, where the entire system of linear equations is decomposed into smaller subsystems. These subsystems are iteratively solved block-wise using Advantage quantum computers developed by D-Wave Systems, and the solutions are subsequently combined to obtain the overall solution. The performance of the proposed method is evaluated by solving the heat equation for a square plate with fixed boundary temperatures and comparing the results with the best existing method. Experimental results show that the proposed method can accelerate the solution of high-dimensional PDEs by using a limited number of qubits up to 2 times the existing method.**

**Keywords** Successive over-relaxation, Partial differential equations, D-Wave systems, Discretization method, Heat equation

Differential equations are indispensable tools for modeling a wide range of phenomena in physics, engineering, and other applied sciences as noted by[1]. Their practical applications extend to numerous industries, where accurate modeling and prediction are crucial for innovation and efficiency as noted by[2]. There are two primary approaches to solving differential equations: analytical methods and numerical methods. However, obtaining an analytical solution for complex systems often presents significant challenges as noted by[3], necessitating the development of numerical methods as noted by[4]. One of the most widely recognized numerical methods for solving linear differential equations is the finite difference discretization technique, which involves dividing the domain into cells or volumes as noted by[5]. This technique transforms linear differential equations into a system of linear equations, $A\boldsymbol{x} = \boldsymbol{b}$, where $A$ is typically a sparse matrix as noted by[6]. As a result of this approach, solutions can be approximated at specific points (center of cells or volumes) within the spatial domain. The most time-consuming part of this method is solving a system of linear equations. The fastest classical algorithm for solving these systems has a time complexity of $\mathcal{O}(N.\sqrt{\kappa})$, where $N$ is the dimension of the matrix $A$ as noted by[7] and $\kappa$ is its condition number. As the number of cells increases in real-world applications, the size of the linear equation system grows correspondingly, leading to increased computation time.

Quantum algorithms present a promising alternative to classical algorithms, enabling significant reductions in computational complexity as noted by[8–12]. Among these, the Harrow, Hassidim, and Lloyd (HHL) algorithm stands out as a well-known quantum algorithm specifically designed to solve systems of linear equations[8]. The HHL algorithm achieves a time complexity of $\mathcal{O}(\mathrm{Poly}(\log(N), \kappa))$ when the matrix $A$ is sparse, offering an exponential speedup compared to the fastest classical algorithms. This positions the HHL algorithm as a powerful tool for solving large linear systems. Following its development, researchers have employed the HHL algorithm as a subroutine in solving both ordinary and partial differential equations as noted by[13–17]. More specifically, the paper[18] applied the HHL algorithm to solve steady-state heat equations derived from Sturm-Liouville problems. Furthermore, the study presented in[19] explored the application of the HHL algorithm to accelerate the solution of PDEs, with a particular focus on aerospace applications. However, the output of the HHL-based methods is a quantum state $|x\rangle$ proportional to the solution, necessitating very resource-intensive quantum tomography to determine the complete solution $\boldsymbol{x}$[20]. Additionally, the HHL algorithm needs a state preparation step to prepare

[1]Iranian Quantum Technologies Research Center (IQTEC), Tehran, Iran. [2]Quantum Optics Lab, Department of Physics, Kharazmi University, Tehran, Iran. ✉email: mohsen.akbari@khu.ac.ir

the quantum state $|b\rangle$ proportional to $\boldsymbol{b}$ at the input of the algorithm, which is also challenging. Moreover, HHL-based methods require extensive quantum resources, such as qubits and gates, making practical application challenging in the era of NISQ (Noisy Intermediate-Scale Quantum) computers[21].

An alternative approach for solving a system of linear equations is quantum annealing, where the solution $|x\rangle$ corresponds to the ground state of the final Hamiltonian[22]. Several studies[23–26] proposed a method to encode $A\boldsymbol{x} = \boldsymbol{b}$ to a quadratic unconstrained binary optimization (QUBO) problem through a well-defined protocol and implemented it on a D-Wave quantum annealing system[27]. Moreover, Suresh et al. proposed a QUBO-based approach for the computation of the Sparse Approximate Inverse (SPAI) of a matrix[28]. Quantum annealing addresses certain challenges associated with HHL-based methods, such as state preparation and tomography. However, the limited number of qubits available on current D-Wave quantum systems (5000+ on the Advantage quantum computer[29]) restricts the dimensions of linear systems and, consequently, the practical application of PDEs. In response to this limitation, Pollachini et al.[6] have adopted a hybrid block Gauss–Seidel method, which decomposes large linear systems into smaller subsystems and solves each subsystem individually, facilitating the solution of large-scale problems with limited quantum resources. Despite its utility, the block Gauss–Seidel method suffers from a suboptimal convergence rate, making it an inefficient iterative approach[30]. This limitation becomes particularly pronounced for high-dimensional or poorly conditioned problems, where the increased number of iterations required significantly impacts computational performance.

The main purpose of this paper is to solve the steady-state 2D heat and Poisson equation by a fast hybrid classical-quantum algorithm to decrease computational complexity and increase the convergence rate. Through finite difference discretization, the resulting linear systems are formulated as QUBO problems and solved using the Advantage system of the D-Wave. Accordingly, this paper proposes a fast paradigm based on the block SOR to accelerate the solution of linear PDEs. This method has superior convergence speed with an optimal choice of over-relaxation parameter[31] to substantially reduce the number of iterations required for convergence compared to the block Gauss–Seidel method. The experimental results validate that this method not only half the convergence time compared to conventional techniques but also exhibits robustness against significant perturbations under different noise conditions. The analysis further shows that the method consistently outperforms Gauss–Seidel in iteration count for $1 < \omega < 1.8$, highlighting its practicality even when $\omega_{opt}$ is not precisely determined.

The rest of this paper is organized as follows. The following section provides a detailed explanation of the proposed approach for solving differential equations, including discussions on finite difference methods, the QUBO formulation of the problem, and the block SOR method. The third section presents two examples to illustrate the applications of the methodology. The fourth and fifth sections discuss the experimental results and offer concluding remarks.

## Proposed method

A description of the proposed algorithm for solving linear PDEs is presented in this section. Linear PDEs are used to model quantum mechanics as noted by[32] and various physical processes, such as heat transfer as noted by[33], and wave propagation as noted by[34]. However, solving PDEs for real-world problems with high dimensions remains challenging. Consequently, this paper proposes an efficient hybrid framework based on classical and quantum methods to facilitate this problem. The proposed method follows the sequence shown in FIG. 1. A linear PDE is specified as input to the system, and a heat map is generated as output. This algorithm has three main stages.

First, a system of linear equations ($A\boldsymbol{x} = \boldsymbol{b}$) is generated by discretizing the domain and applying the finite difference method. Secondly, the system of linear equations is fed into the hybrid solver stage. Due to the large size of the system of linear equations in real-world problems, a hybrid solver based on block iterative methods is used. Block iterative methods can decompose large linear equation systems into smaller, more manageable subsystems that can be run on NISQ computers. First, the classical computer decomposes the system of linear equations into smaller subsystems (the blocking stage), and then the quantum system solves these subsystems using the quantum annealing algorithm. When the quantum computer provides solutions, the classical computer consolidates these solutions and calculates the error to determine whether to terminate or continue the iterative process. This process is repeated until convergence of all the subsystems is obtained. As a result of solving these subsystems, we can approximate the desired function at discretized points within the domain. Finally, a heat map of the solution is generated, and an error curve is plotted against the number of iterations to determine the convergence rate.

Each phase of the proposed algorithm, the finite difference discretization technique, a fast iterative approach using block SOR, and the quantum annealing algorithm for solving a system of linear equations using the QUBO formula are explained in the following subsections.

### Finite difference method

Solving a differential equation necessitates determining the value of an unknown function at every point within a specified spatial domain. However, in a continuous space, there are an infinite number of such points. To address this challenge, the finite difference discretization technique is employed. This technique discretizes the space by dividing it into a finite number of points and then approximating the derivatives of the unknown function values at these discrete points. The discretization of $\mathbb{R}^2$ can be written as:

$$x \to x_i, \quad y \to y_j, \ i \in \{1, \ldots, n\}, \quad j \in \{1, \ldots, m\}. \tag{1}$$

The steps in directions $x$ and $y$ can be defined as Eqs. (2) and (3).

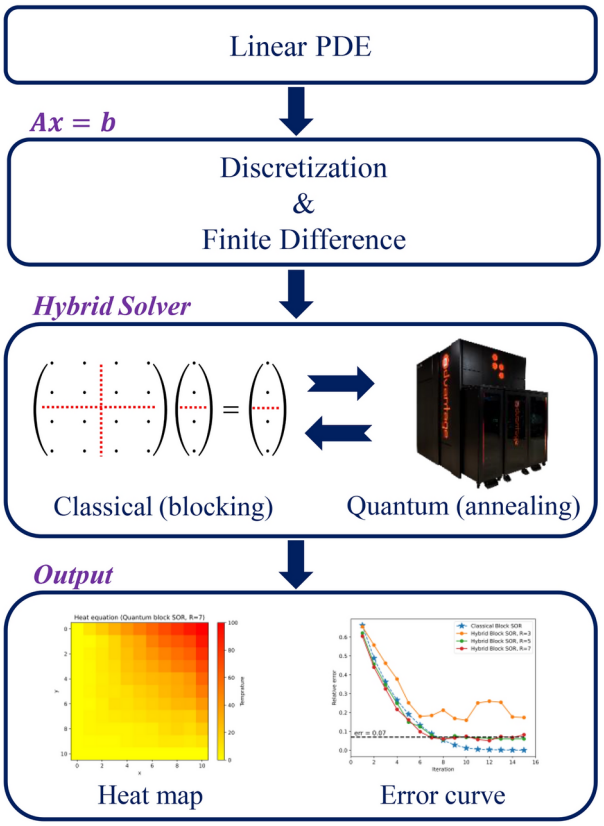



**Figure 1**. Schematic of the flowchart of proposed method.

$$\Delta x = \frac{x_n - x_1}{n-1}, \qquad \Delta y = \frac{y_m - y_1}{m-1} \tag{2}$$

$$x_{i+1} = x_i + \Delta x, \qquad y_{j+1} = y_j + \Delta y \tag{3}$$

For simplicity, it is assumed that $\Delta x = \Delta y = h$. Following the discretization of the spatial domain, we apply the finite difference method, which utilizes the Taylor series expansion to approximate the function values at adjacent points $x_{i-1} = x_i - h$ and $x_{i+1} = x_i + h$ as follows:

$$f(x_i - h) = f(x_i) - h\frac{df}{dx}\bigg|_{x_i} + \frac{h^2}{2}\frac{d^2f}{dx^2}\bigg|_{x_i} - \mathcal{O}(h^3), \tag{4}$$

$$f(x_i + h) = f(x_i) + h\frac{df}{dx}\bigg|_{x_i} + \frac{h^2}{2}\frac{d^2f}{dx^2}\bigg|_{x_i} + \mathcal{O}(h^3). \tag{5}$$

Accordingly, by subtracting and adding Eqs. (4) and (5), the first and second-order derivatives are obtained with an accuracy of $h^2$ by Eqs. (6) and (7), respectively.

$$f(x_i + h) - f(x_i - h) = 2h\frac{df}{dx}\bigg|_{x_i} + \mathcal{O}(h^3) \quad \Longrightarrow \quad \frac{df}{dx}\bigg|_{x_i} = \frac{f(x_i + h) - f(x_i - h)}{2h} + \mathcal{O}(h^2) \tag{6}$$

$$f(x_i + h) + f(x_i - h) = 2f(x_i) + h^2\frac{d^2f}{dx^2}\bigg|_{x_i} + \mathcal{O}(h^4) \quad \Longrightarrow \quad \frac{d^2f}{dx^2}\bigg|_{x_i} = \frac{f(x_i + h) - 2f(x_i) + f(x_i - h)}{h^2} + \mathcal{O}(h^2) \tag{7}$$

Given the small value of $h$, the terms involving $h^2$ can be neglected. As a result, to approximate the partial derivatives of $f(x, y)$, Eqs. (8) and (9) must be applied to all discretized points within the domain.

$$\frac{\partial f}{\partial x}(x_i, y_j) \approx \frac{f(x_i + h, y_j) - f(x_i - h, y_j)}{2h}, \quad \frac{\partial f}{\partial y}(x_i, y_j) \approx \frac{f(x_i, y_j + h) - f(x_i, y_j - h)}{2h} \tag{8}$$

$$\frac{\partial^2 f}{\partial x^2}(x_i, y_j) \approx \frac{f(x_i + h, y_j) - 2f(x_i, y_j) + f(x_i - h, y_j)}{h^2}, \quad \frac{\partial^2 f}{\partial y^2}(x_i, y_j) \approx \frac{f(x_i, y_j + h) - 2f(x_i, y_j) + f(x_i, y_j - h)}{h^2} \tag{9}$$

This approximation yields a system of linear equations that, when solved, provide approximate values of the desired function at discretized spatial locations within the domain.

## Hybrid solver (block SOR)

A well-established approach to solving a system of linear equations is the use of block iterative methods[35], which improve the solution through a series of iterative approximations. Block iterative methods are useful in numerical linear algebra and computational science when working with partitioned or structured matrices. A block iterative method divides a system of linear equations into smaller blocks and solves each separately. The results of all blocks are then combined to produce the overall solution. If the matrix $A = [A_{i,j}]$ in the linear system $A\boldsymbol{x} = \boldsymbol{b}$ has dimensions $(m \cdot N_b) \times (m \cdot N_b)$, it can be decomposed into blocks as Eq. (10). Where the blocks $A_{i,j}$ have size $m \times m$.

$$\begin{pmatrix} \boxed{A_{1,1}} & \boxed{A_{1,2}} & \cdots & \boxed{A_{1,N_b}} \\ \boxed{A_{2,1}} & \boxed{A_{2,2}} & \cdots & \boxed{A_{2,N_b}} \\ \vdots & \vdots & \ddots & \vdots \\ \boxed{A_{N_b,1}} & \boxed{A_{N_b,2}} & \cdots & \boxed{A_{N_b,N_b}} \end{pmatrix} \begin{pmatrix} \boxed{\boldsymbol{x}_1} \\ \boxed{\boldsymbol{x}_2} \\ \vdots \\ \boxed{\boldsymbol{x}_{N_b}} \end{pmatrix} = \begin{pmatrix} \boxed{\boldsymbol{b}_1} \\ \boxed{\boldsymbol{b}_2} \\ \vdots \\ \boxed{\boldsymbol{b}_{N_b}} \end{pmatrix} \tag{10}$$

Here, each vector $\boldsymbol{x}_i$ and $\boldsymbol{b}_i$ has dimensions $m \times 1$. An important aspect of block iterative methods is to convert Eq. (10) into a framework that can be implemented iteratively. As a result, the matrix $A$ is splitted as $A = E - F$ [31]. In this equation, $E$ and $F$ have block form similar to the matrix $A$ (i.e., matrices $E$=$[E_{i,j}]$ and $F$=$[F_{i,j}]$ have dimension $(m \cdot N_b) \times (m \cdot N_b)$) and also $E$ must be a non-singular matrix. When the system is divided into $N_b$ blocks, each smaller system has dimensions $m \times m$ compatible with NISQ computers. Consequently, the linear system is transformed into the block iterative form as Eq. (11):

$$\sum_j E_{i,j}\boldsymbol{x}_j^{(k+1)} = \sum_j F_{i,j}\boldsymbol{x}_j^{(k)} + \boldsymbol{b}_i. \tag{11}$$

where, $\boldsymbol{x_j}^{(k+1)}$ represents the approximate solution of block $j$ after $k+1$ iterations which was computed using the solution of the previous iteration ($\boldsymbol{x_j}^{(k)}$). The iterative procedure initiates with an initial approximation $\boldsymbol{x_j}^{(0)}$, which may be set to zero, i.e., $\boldsymbol{x_j}^{(0)} = 0$. The critical question then arises: what should the matrices $E$ and $F$ in Eq. (11) be?

Various selections of the matrices $E$ and $F$ lead to different block iterative methods. There are advantages and disadvantages to each of these block iterative methods. Since the block SOR iterative method has a high convergence rate, this paper employs it to solve each subsystem. In this way, the convergence rate of the hybrid algorithm for solving linear PDEs is improved. In the block SOR method, the matrices $E_{i,j}$ and $F_{i,j}$ are defined as Eqs. (12) and (13) respectively.

$$E_{i,j} = \frac{1}{\omega} D_{i,j} + L_{i,j} \tag{12}$$

$$F_{i,j} = \left(\frac{1}{\omega} - 1\right) D_{i,j} - U_{i,j} \tag{13}$$

where,

$$D_{i,j} = \begin{cases} A_{i,j}, & \text{if } i = j. \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

$$U_{i,j} = \begin{cases} A_{i,j}, & \text{if } i < j. \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

$$L_{i,j} = \begin{cases} A_{i,j}, & \text{if } i > j. \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

The parameter $\omega$ is the over-relaxation parameter, which must be selected optimally to achieve a faster convergence rate. In this method, the parameter $\omega$ typically lies between 1 and 2 to ensure stability and improved convergence[36]. The term "optimal" signifies that the block SOR method achieves its most rapid convergence rate when $\omega$ is set to $\omega_{SOR}^{opt}$, is given by[35]:

$$\omega_{SOR}^{opt} = \frac{2}{1 + \sqrt{1 - \rho(H_J)^2}}. \tag{17}$$

where $\rho(H_J)$ denotes the spectral radius (i.e., the maximum absolute value of the eigenvalues) of the Jacobi iteration matrix, which is defined as Eq. (18).

$$H_J = -D^{-1}(L+U) \tag{18}$$

The number of iterations required for convergence is not predetermined but is determined by a convergence criterion. Here, we monitor the relative error of the solution vector at the k-th iteration $\boldsymbol{x}^{(k)}$ as a convergence criterion. The relative error is defined as Eq.(19). In this equation, $\boldsymbol{x}^{(k)}$ is the approximate solution after $k$ iterations and $\boldsymbol{x}$ is the exact solution.

$$e^{(k)} = \frac{\left\|\boldsymbol{x}^{(k)} - \boldsymbol{x}\right\|}{\|\boldsymbol{x}\|} \tag{19}$$

This error is due to both classical and quantum sources. The classical error is due to the PDE's discretization, as seen from Eqs. (6) and (7). For example, $h = 0.1$, the discretization error is about 0.01, so results are meaningful up to two decimal places. There are two significant sources of quantum errors in D-Wave, which are linked. As a result, the total error cannot be split into partial contributions.

1. *Precision limitations* Since each element of the solution vector is described by a finite number of bits, *R*, the quantization error arises.
2. *Device noise* Intrinsic noise in an Advantage QPU -or any other present quantum device- adds inaccuracy to the computed solutions of the above problems.Recognizing that iterative methods do not universally guarantee convergence for every matrix $A$ is essential. The conditions under which convergence occurs are fundamental to the theoretical framework of iterative methods. Specifically, an iterative method for solving a system of linear equations $A\boldsymbol{x} = \boldsymbol{b}$ will converge from any initial guess $\boldsymbol{x}^{(0)}$ if and only if the *spectral radius* of the iteration matrix is less than one[37]. Consequently, for the block SOR method, we have:

$$\rho(H_{SOR}) < 1, \tag{20}$$

$$H_{SOR}(\omega) = (D - \omega L)^{-1}\left[(1-\omega)D + \omega U\right]. \tag{21}$$

## Hybrid solver (quantum annealing)

Obtaining the ground state of a Hamiltonian, particularly in quantum chemistry[38] and quantum biology[39], is an important aspect of quantum mechanics. It is generally not feasible to calculate the ground state of an arbitrary Hamiltonian analytically due to its algebraic complexity as noted by[40]. A popular method for obtaining the ground state is quantum annealing as noted by[41]. The concept of quantum annealing is derived from the Hamiltonian in Eq. (22). The total evolution time is represented by $T$. The Hamiltonian is $H_i$ at $t = 0$, and at $t = T$, it transitions to $H_f$.

$$H(t) = \left(1 - \frac{t}{T}\right) H_i + \frac{t}{T} H_f \tag{22}$$

The adiabatic theorem[42] states that if the evolution time $T$ is sufficient and the initial state of the system is the ground state of $H_i$, then the state of the system at time $T$ will, with a high probability, be $H_f$. In general, an evolution time $T$ is considered sufficient when it is on the order of $g_{min}^{-2}$, where $g_{min}$ represents the minimum energy gap between the ground state and the first excited state as noted by[42,43].

D-Wave has successfully implemented quantum annealing based on the Ising Hamiltonian model[44]. In this model, the initial and final Hamiltonians are expressed as Eqs. (23) and (24), respectively. In these equations, $\sigma_x^{(i)}$ and $\sigma_z^{(i)}$ are spin operators in the $x$ and $z$ directions on the $i$-th particle, respectively. The coefficients $\alpha_i$ and $\beta_{ij}$ are arbitrary real numbers. Since the objective is to find the ground state of the final Hamiltonian, the initial state must be the ground state of the initial Hamiltonian, which is given by $|g\rangle = |+\rangle^{\otimes n}$. The final Hamiltonian represents the Ising model, whose eigenstates can be easily calculated, but determining its ground state remains challenging. The primary objective of the D-Wave quantum system is to find the ground state of the Ising model using quantum annealing[44].

$$H_i = -\sum_{i=1}^{n} \sigma_x^{(i)} \tag{23}$$

$$H_f = \sum_{i=1}^{n} \alpha_i \sigma_z^{(i)} + \sum_{i,j=1}^{n} \beta_{ij} \sigma_z^{(i)} \sigma_z^{(j)} \tag{24}$$

Moreover, the Ising model's algebraic structure corresponds to the objective function of the QUBO problems. The QUBO problems constitute a significant class of optimization problems, characterized by binary decision variables, a quadratic objective function, and the absence of constraints on the variables[45]. Consequently, a QUBO formulation can be employed to solve a system of linear equations on a D-Wave quantum system. A linear system $A\boldsymbol{x} = \boldsymbol{b}$ can be formulated as an optimization problem by Eq. (25).

$$\boldsymbol{x}_{sol} = \arg\min_{\boldsymbol{x}} \ (A\boldsymbol{x} - \boldsymbol{b})^{\dagger}(A\boldsymbol{x} - \boldsymbol{b}) \tag{25}$$

It is clear that the objective function of Eq. (25) is quadratic, and by expressing the elements of $\boldsymbol{x}$ in binary form, it conforms to the QUBO framework, making it solvable using a D-Wave quantum system. However, the elements $x_i$ of $\boldsymbol{x}$ are arbitrary real numbers since it is impossible to represent them in binary form. So, the algorithm starts by converting the real-valued number $x_i$ into an $R$ bits binary format as Eq. (26). Where $x_i$ is scaled and shifted such that $x_i \in [-d_i, 2c_i - d_i)$, where $c_i$ and $d_i$ can be selected depending on the specific problem. In this equation $q_r^i$ is the $r$-th bit of $x_i$ with an $R$-bit approximation. When $d_i > 0$ and $c_i > d_i/2$, the value of $x_i$ may be positive or negative. This reduces the qubits required for a given floating-point accuracy of $x_i$. There is great benefit in reducing the number of required qubits in the NISQ era systems, where only a few thousand qubits are available.

$$x_i = c_i \sum_{r=1}^{R} q_r^i 2^{-r} - d_i \tag{26}$$

After substituting $x_i$ from Eq. (26) into the objective function in Eq. (25), the objective function can be rewritten approximately as Eq. (27).

$$H = \sum_{r=1}^{R}\sum_{i=1}^{N} \alpha_r^i q_r^i + \sum_{r,s=1}^{R}\sum_{i,j=1}^{N} \beta_{rs}^{ij} q_r^i q_s^j \tag{27}$$

In this equation, the coefficients are calculated as Eqs. (28) and (29). Here, $A_{ij}$ denotes the $ij$-th element of matrix $A$ and $b_i$ are components of the vector $b$.

$$\alpha_r^i = -2^{-r+1}\left(\sum_{j,k=1}^{N} A_{ki}A_{kj}c_i d_j + \sum_{j=1}^{N} A_{ji}c_i b_j\right) \tag{28}$$

$$\beta_{rs}^{ij} = 2^{-(r+s)}\left(\sum_{k=1}^{N} A_{ki}A_{kj}c_i c_j\right) \tag{29}$$

D-Wave has developed a software development interface known as Ocean[27] to facilitate communication with quantum hardware. Accordingly, the coefficients $\alpha_i$ and $\beta_{ij}$ must be provided to the D-Wave Ocean program to solve a system of linear equations. The solution of a system of linear equations on a D-Wave quantum system requires a bit string of $N \cdot R$ logical qubits. As a result, a substantial number of qubits is necessary when dealing with large matrices (large $N$) or when high accuracy is required (large $R$). This emphasizes the importance of hybrid block iterative methods.

Table 1 shows the comparative resources achieved by the iterative and non-iterative QUBO. The columns of Table 1 contain the names of the iterative methods, the number of qubits for representing the solution, and run-time respectively. The variable $T_{N,R}$ in Table 1 represents the time required to run a single QUBO algorithm for solving a linear system of size $N$, with $R$ qubits for representing the solution. In contrast, $T_{N/N_{b,R}}$ denotes the time needed to solve a system of size $N/N_b$. The variable $n_{\text{iter}}$ in the table specifies the number of iterations in the iterative QUBO method, which is decreased in our proposed method. Both $T_{N,R}$ and $T_{N/N_{b,R}}$ are dependent on the annealing time and the number of runs necessary to achieve the solutions.

## Example

The performance of the proposed method is evaluated using one of the most important PDEs in physics, namely the heat equation. For example, the heat transfer problem within a given region requires the solution of the heat equation. The heat equation is expressed as Eq. (30).

$$\frac{\partial u}{\partial t} = \alpha^2 \nabla^2 u \tag{30}$$

| Method | Number of qubits | Time |
|---|---|---|
| Original QUBO | $N \cdot R$ | $T_{N,R}$ |
| Iterative QUBO | $\frac{N}{N_b} \cdot R$ | $T_{N/N_b,R} \cdot N_b \cdot n_{\text{iter}}$ |

**Table 1**. Comparison between original QUBO and iterative QUBO methods.

The proposed method is detailed for a square region of length $L$, with fixed temperatures at the edges. For the stationary case, where the temperature $u$ is independent of time, the heat equation simplifies to Eq. (31). Where $u = u(x, y, t)$ denotes the temperature at the point $(x, y)$ at time $t$.

$$\nabla^2 u = 0 \tag{31}$$

While this study primarily focuses on the heat equation, the methodology is inherently general and applicable to a wide range of linear PDEs. For instance, the approach has been extended to the Poisson equation by analyzing the heat equation (Eq. 30) with a source term, which, under steady-state conditions, modifies to the Poisson equation (Eq. 32). As a result of this extension, the versatility of the proposed method can be demonstrated. Crucially, the convergence of the proposed method depends on the properties of the coefficient matrix formed after discretization-a condition preserved across this transformation-thus demonstrating the broader applicability of the technique.

$$\frac{\partial u}{\partial t} - \alpha^2 \nabla^2 u = f, \quad \text{if} \quad \frac{\partial u}{\partial t} = 0 \quad \Longrightarrow \quad \nabla^2 u = -\frac{f}{\alpha^2} \tag{32}$$

The initial phase of the proposed method involves spatial domain discretization. The region is discretized into a grid consisting of $11 \times 11$ cells. The discretized spatial points are depicted in Fig. 2. In this illustration, green points represent the domain's boundaries, while yellow points represent its interior. Discrete points are the centers of each cell, where each point corresponds to a number. It is possible to use different numbering schemes for discrete points. As illustrated, we offer snake numbering[46]. This special numbering yields advantages, such as facilitating access to points during programming. For example, each pair of vertically aligned points has an index difference of 9. Furthermore, this numbering results in a sparse tridiagonal matrix when applied to the finite difference method[30].

Dirichlet boundary conditions determine the values of a desired function in the boundary. These boundary conditions are defined in Eq. (33). Based on the known temperature values at the boundaries, excluding these cells yields a $9 \times 9$ internal cells grid. Due to this, 81 grid points need to be solved for the heat equation.

$$u(0, y) = u(x, 0) = 0\ ^\circ C, \quad u(L, y) = \frac{y}{L} \times 100\ ^\circ C, \quad u(x, L) = \frac{x}{L} \times 100\ ^\circ C \tag{33}$$

To approximate the heat equation, it is necessary to approximate the Laplacian operator using the finite difference method, as shown in Eq. (9), leading to the approximation given in Eq. (34).

$$\nabla^2 u(x_i, y_j) = \frac{\partial^2 u}{\partial x^2}(x_i, y_j) + \frac{\partial^2 u}{\partial y^2}(x_i, y_j) \approx \frac{u(x_i + h, y_j) + u(x_i - h, y_j) + u(x_i, y_j + h) + u(x_i, y_j - h) - 4u(x_i, y_j)}{h^2} \tag{34}$$

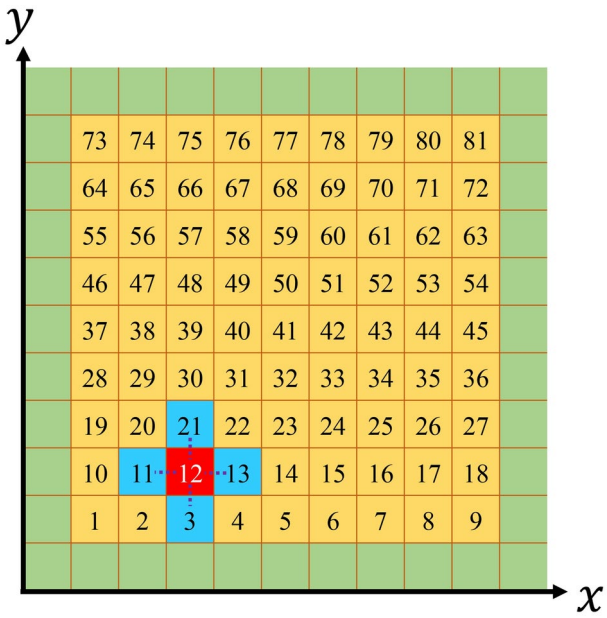



**Figure 2**. The discretization of the spatial domain ($\mathbb{R}^2$) to a grid of cells is depicted. It shows that the approximation of the Laplacian at point 12 requires the function values at points 3, 11, 13, and 21.

It is evident that to approximate the Laplacian operator at a specific point $(x_i, y_j)$, the values of the function at four neighboring points and the point itself are required. For example, in Fig. 2, the Laplacian operator approximation at point 12 is obtained from Eq. (35).

$$\nabla^2 u\Big|_{12} \approx \frac{u_3 + u_{11} + u_{13} + u_{21} - 4u_{12}}{h^2} \tag{35}$$

Applying the approximation of the Laplacian operator (Eq. 34) to all interior points within the domain results in a sparse system of linear equations.

$$A\boldsymbol{x} + \boldsymbol{b_c} = 0 \quad \Longrightarrow \quad A\boldsymbol{x} = -\boldsymbol{b_c} \tag{36}$$

In this system, the vector $\boldsymbol{x}$ represents the solution vector $\boldsymbol{x} = (u_1, u_2, \cdots, u_{81})^T$, and the vector $\boldsymbol{b_c}$ contains information regarding the boundary function values. The matrix $A$ has structure as shown in Eq. (37).

$$A = \begin{pmatrix} C & I & 0 & \cdots & 0 \\ I & C & I & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & I & C & I \\ 0 & \cdots & 0 & I & C \end{pmatrix} \tag{37}$$

This matrix has a sparse tridiagonal block form, where $I$ is the identity matrix, and the matrix $C$ is obtained as in Eq. (38).

$$C = \begin{pmatrix} -4 & 1 & 0 & \cdots & 0 \\ 1 & -4 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 1 & -4 & 1 \\ 0 & \cdots & 0 & 1 & -4 \end{pmatrix} \tag{38}$$

Consequently, solving the heat differential equation is reduced to solving a system of linear equations, which is then addressed using the block SOR method.

## Discussion and experimental results

The performance of the proposed method has been evaluated by solving the heat and the Poisson equation for a square plate with fixed temperatures using the Advantage D-Wave system with over 5000 qubits. The computational implementation of the proposed hybrid classical-quantum algorithm utilized both classical and quantum resources. The classical components were developed in Python. These experiments were conducted on a computer with an Intel Core i7 processor and 8 GB RAM. To assess efficiency and accuracy, the problem is numerically addressed through the finite difference method, partitioning the domain into $N_b$ segments of equal length. This approach allows for solving larger linear systems of equations, thus overcoming the constraints posed by the limited number of qubits in D-Wave quantum processing units.

The efficiency of this extension highlights the versatility of the proposed method. Crucially, the convergence of the method depends on the properties of the coefficient matrix formed after discretization-a condition preserved across these transformations-thus demonstrating the broader applicability of the technique.

The validation of the proposed method is integrated with the initial implementation of the block Gauss–Seidel method, as suggested by[6], due to its close resemblance to our proposed algorithm and its proven effectiveness in addressing challenges associated with high-dimensional problems. To address the limitations of the block Gauss–Seidel method, particularly its slow convergence, our proposed method was introduced, demonstrating faster convergence with fewer iterations and reducing overall execution time. The iterative process in the block SOR method is conceptually analogous to that of the block Gauss–Seidel method; for $\omega = 1$, the block SOR method transforms into the block Gauss–Seidel method.

The heat equation example demonstrates the effectiveness of the proposed methodology with three studies. These studies examine the effect of the number of qubits for representing the solution. The convergence criterion was selected by the method of[6] relative error = 0.08. That is done to keep the evaluation framework consistent and to provide a fair comparison with similar hybrid quantum-classical methods. Moreover, the selected criteria reflect the current capabilities of quantum hardware, particularly the D-Wave system, whose precision is influenced by hardware constraints such as the finite number of qubits and inherent noise. Using more stringent convergence criteria may be possible as quantum technology advances, potentially enhancing the accuracy and applicability of the proposed method. Figure 3a and b show the temperature distribution in the square plate after convergence.

The sensitivity of the proposed method concerning an over-relaxation parameter $\omega$ was investigated in simulations using a D-Wave quantum computer. Figure 4 shows the experiment results, where the vertical axis indicates the number of iterations required to achieve an error of 0.08. In contrast, the horizontal axis displays the different values of $\omega$. This figure illustrates that the number of iterations is the least for $\omega_{opt} \approx 1.53$ and gives the best performance at its neighborhood value. For $\omega = 1$, the scheme reduces to Gauss–Seidel. It is clear from

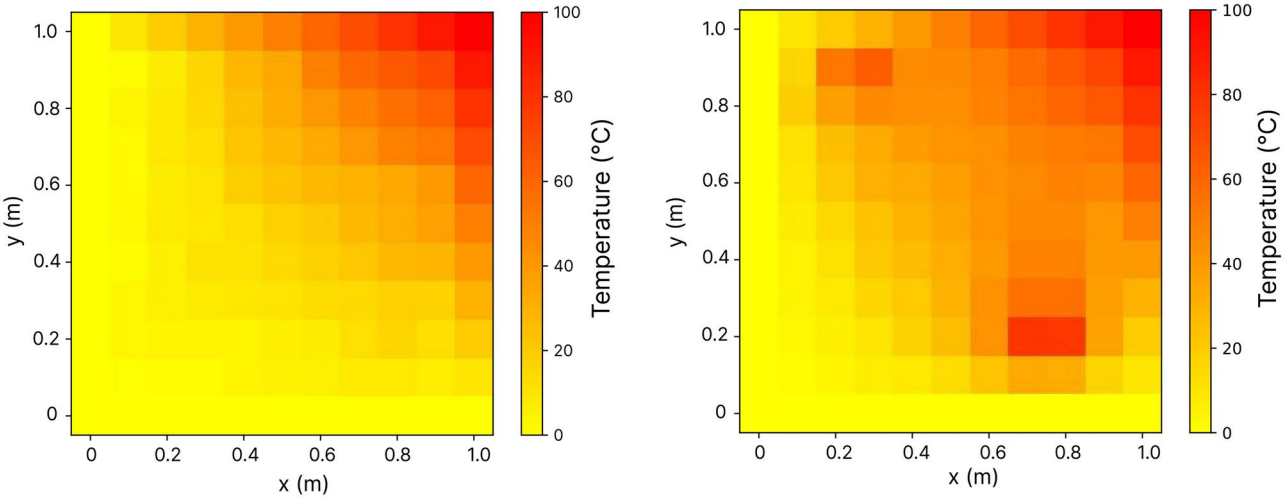



**Figure 3**. The solution of heat and Poisson equations with boundary condition in Eq. (33) with hybrid block SOR algorithm in the square plate.

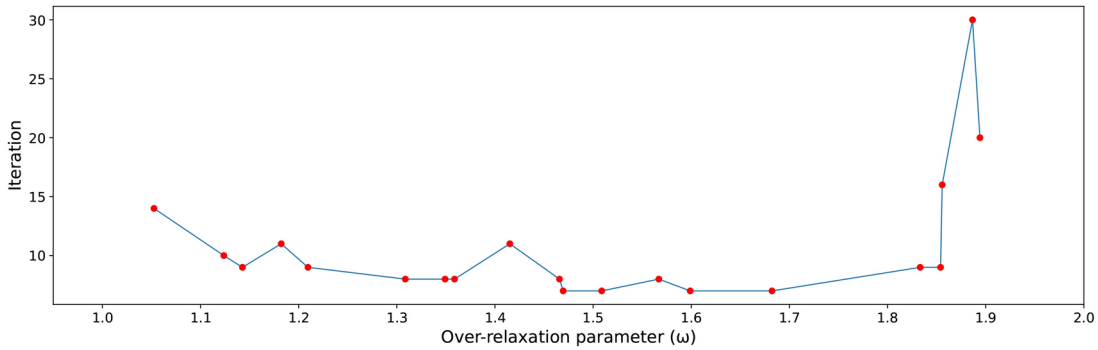



**Figure 4**. Independence of the over-relaxation parameter $(\omega)$ to the number of iterations needed to reach the relative error (see Eq. 19) equal to 0.08.

this graph that for any $1 < \omega < 1.8$, the number of iterations is less than that for Gauss–Seidel. This analysis underlines that the newly proposed method is not sensitive to small perturbations in the choice of $\omega$. Even without an optimum choice of $\omega_{opt}$, the procedure converges faster than Gauss–Seidel.

Stability is a fundamental property of numerical methods. It reflects the ability of the methods to provide reliable solutions when perturbations in input data are present. To investigate the stability of the proposed method, we added Gaussian noise with zero mean and different standard deviations to the right-hand side vector $\boldsymbol{b}$ in the system $A\boldsymbol{x} = \boldsymbol{b}$. Noise variance was given as a percentage of the magnitude of $\boldsymbol{b}$ to simulate real-world data imperfections. It follows from Table 2 that the method is highly stable; for noise variances as significant as $40\%$, the relative error in the solution does not exceed $13\%$, which is considerably smaller than the noise level itself. Moreover, it can be noticed that the algorithm's convergence rate is affected only slightly with an increase in noise. These results confirm that the proposed method is robust and capable of handling noisy input data effectively, which is appropriate for practical applications where imperfections in the data are unavoidable.

Figure 5 illustrates the relationship between the relative error, as defined in Eq. (19), and the number of iterations of the Block Gauss–Seidel and Block SOR methods, comparing their performance across varying quantities of qubits used in the numerical representation of the solution. We discussed the influence of the number of qubits for representing the solution (set to 3, 5, 7) on the convergence rate. This example shows the minimum error when seven qubits are utilized within the Advantage system.

The computation time of QUBO-based algorithms is primarily influenced by factors such as annealing time, readout time, embedding time, and others, which remain challenging to optimize[47]. Due to this, demonstrating computational time on small-scale examples may not effectively convey the advantages of the proposed approach. This quantum advantage in computational time is expected to become more pronounced with the development of quantum hardware incorporating a significantly more significant number of qubits. In this respect, its real application is anticipated to be substantially enhanced with the advent of next-generation quantum computers.

| Noise variance (%) | Relative error (%) | Number of iteration |
|---|---|---|
| 5 | 7.2 | 8 |
| 10 | 6.9 | 8 |
| 15 | 9.8 | 7 |
| 20 | 10.2 | 8 |
| 25 | 10.4 | 7 |
| 30 | 11.1 | 8 |
| 35 | 13.1 | 8 |
| 40 | 12.1 | 8 |

**Table 2**. Error relative to the noise-free solution for various noise levels.

As a result, the efficiency of the proposed method was illustrated by comparing it with the best existing method based on the number of iterations required to achieve convergence.

The experimental results indicate that the proposed method outperforms the block Gauss–Seidel method in all studied scenarios. The results show that our hybrid quantum-classical method accelerates the solution of PDEs by up to two times compared to the block Gauss–Seidel method. While the Gauss–Seidel method achieves an accuracy of 0.09 after 15 iterations, the proposed method reaches the desired accuracy by the 7th iteration. Unlike in classical algorithms, we observed that the error does not consistently decrease, partly due to the limited floating-point accuracy of the variables $x_i$, which prevents further improvement after a certain number of iterations. Moreover, the relative error reduces as the number of qubits for representing the solution increases since $R$ determines the significant figures of the elements in $\boldsymbol{x}$.

## Conclusion

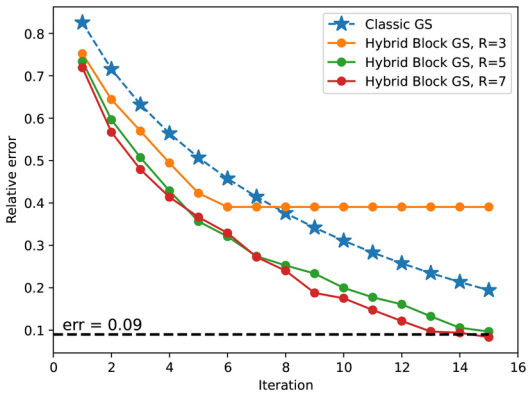
This paper presents a fast hybrid classical-quantum approach to solving PDEs using a combination of quantum annealing and the block SOR method, significantly reducing the computational complexity of solving PDEs. The proposed method addresses the challenge of high-dimensional PDEs, which are computationally intensive to solve using traditional methods, by leveraging the capabilities of quantum computing within a hybrid framework. Through finite difference discretization, the solution of PDEs is reduced to the solution of a system of linear equations, which is then solved using the block SOR method. This block-wise decomposition allows for efficiently handling larger systems by partitioning them into smaller subsystems that can be processed within the constraints of current quantum hardware. The quantum component of the approach, implemented on D-Wave's Advantage quantum computers, solves these systems at a subsystem level. In contrast, the classical component integrates the solutions to achieve the overall result.

The performance of the proposed hybrid method was evaluated by solving the heat equation for a square plate with fixed boundary temperatures. The experimental results indicate that the proposed hybrid quantum-classical method is a promising approach to solving PDEs more efficiently, particularly in high-dimensional scenarios. The method not only accelerates the computation but also achieves comparable accuracy to classical methods, highlighting its potential for practical applications in computational physics and beyond.

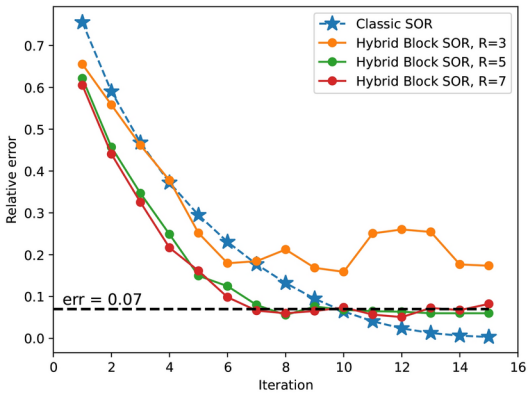
The proposed method enables a substantial reduction in the number of logical qubits, from $N \cdot R$ to $N \cdot R/N_b$, though this reduction comes at the cost of an increased number of iterations required to achieve the desired accuracy. Our results demonstrate that the block SOR method significantly outperforms the block Gauss–Seidel method, achieving convergence up to two times faster. It was observed that increasing the number of qubits used in the numerical representation of the unknown variables enhances the accuracy of the solution.

The proposed method employs a hybrid quantum-classical approach that is inherently sequential due to its block SOR structure, which restricts its potential for parallelization. As a result, implementing this algorithm on high-performance classical systems, such as parallelized SOR solvers running on GPUs or CPUs, is not feasible for the current approach. In contrast, implementing the proposed algorithm on the D-Wave system harnesses quantum computational power and demonstrates notable energy efficiency. While quantum annealers require cryogenic cooling, their energy consumption per computation is significantly lower than that of classical solvers, with GPUs using approximately 100 times more power as noted by[48]. In the hybrid approach, classical systems handle pre-and post-processing, reducing energy costs and optimizing scalability. D-Wave's power consumption, primarily for cryogenic refrigeration, has remained constant since 2011, and advancements in quantum hardware are expected to enhance computational power per watt, solidifying their energy-efficient potential for large-scale problems[48].

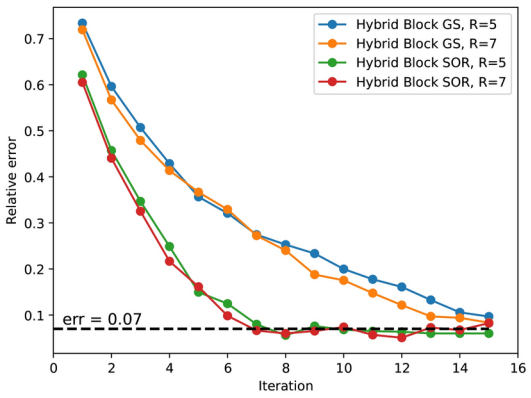
The proposed method shows promise for solving linear systems, but future research may explore its application to more complex, nonlinear systems. It is possible to extend the hybrid method by integrating it with Newton's method to solve nonlinear problems[49]. Newton's method, widely used for its rapid convergence in solving nonlinear equations, could benefit from quantum annealing by enhancing the performance of each algorithm iteration. Combining Newton's method's robustness and speed with quantum annealing's optimization capabilities provides a powerful tool for solving large-scale nonlinear systems. This method is widely used in fluid dynamics, material science, and machine learning. Moreover, incorporating preconditioning techniques is proposed to further enhance the method's stability and convergence under noisy conditions. These techniques are designed to mitigate error propagation and improve the robustness of the solution, thereby increasing the algorithm's resilience to imperfections commonly encountered in real-world data. Additionally, future research



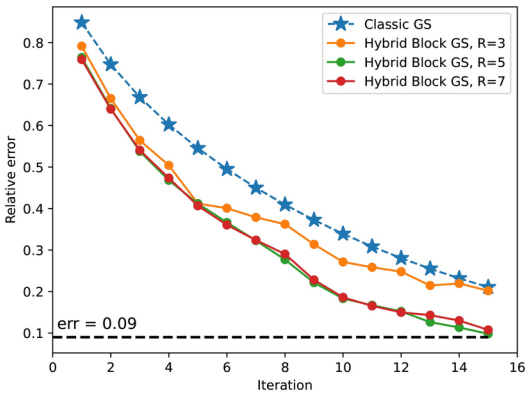

**(a)** Classic and hybrid block Gauss-Seidel method of the heat equation[6].



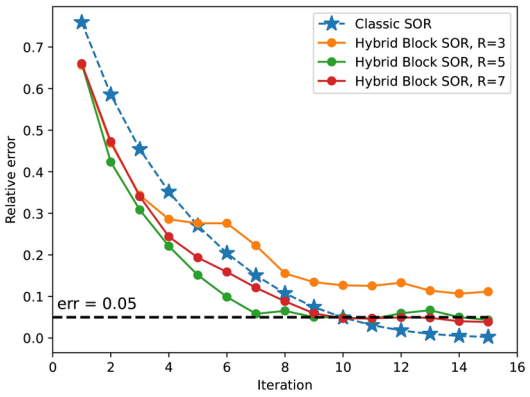

**(b)** Classic block SOR and hybrid block SOR method of the heat equation.



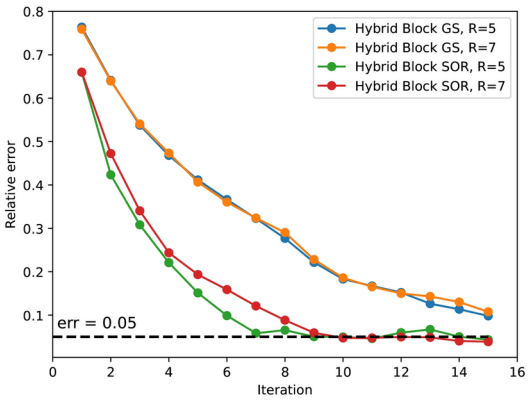

**(c)** Hybrid block Gauss-Seidel and block SOR method of the heat equation.




**(d)** Hybrid block Gauss-Seidel and block SOR method of the Poisson equation.




**(e)** Hybrid block Gauss-Seidel and block SOR method of the Poisson equation.




**(f)** Hybrid block Gauss-Seidel and block SOR method of the Poisson equation.

**Figure 5**. The relative error is analyzed as a function of the number of iterations for the Block Gauss–Seidel and Block SOR methods and a comparison of them, considering varying qubit counts in the numerical representation of the solution vector $x$. The dashed horizontal line represents the error value at which the method has approximately stabilized.

will examine the method's applicability to more complex PDEs, such as the wave equation, focusing on its convergence properties and assessing its computational efficiency. As a result of these efforts, the method is robust and versatile in solving various challenging mathematical problems.

## Data availibility

Data cannot be shared openly but are available on request from corresponding authors.

Received: 30 October 2024; Accepted: 4 February 2025
Published online: 13 February 2025

## References

1. Brearley, P. & Laizet, S. Quantum algorithm for solving the advection equation using Hamiltonian simulation. *Phys. Rev. A* **110**, 012430 (2024).
2. Oz, F., San, O. & Kara, K. An efficient quantum partial differential equation solver with Chebyshev points. *Sci. Rep.* **13**, 7767 (2023).
3. Cveticanin, L. Analytical methods for solving strongly non-linear differential equations. *J. Sound Vib.* **214**, 325–338 (1998).
4. Fadugba, S., Nathaniel Ogunyebi, S., Adebayo, K. & Okunlola, J. Review of some numerical methods for solving initial value problems for ordinary differential equations. *Int. J. Appl. Math. Theor. Phys.* **6**, 7–13 (2020).
5. LeVeque, R. *Finite Difference Methods for Ordinary and Partial Differential Equations* (SIAM, 2007).
6. Pollachini, G., Salazar, J., Góes, C., Maciel, T. & Duzzioni, E. Hybrid classical-quantum approach to solve the heat equation using quantum annealers. *Phys. Rev. A* **104**, 032426 (2021).
7. Strang, G. Linear algebra and its applications (4th Edition).(Thomson/Brooks/Cole, 2006).
8. Harrow, A., Hassidim, A. & Lloyd, S. Quantum Algorithm for linear systems of equations. *Phys. Rev. Lett.* **103**, 150502 (2009).
9. Shor, P. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science* 124–134 (1994).
10. Grover, L. A fast quantum mechanical algorithm for database search. In *Proceedings of The Twenty-Eighth Annual ACM Symposium on Theory of Computing* 212–219 (1996)
11. Feynman, R. Simulating physics with computers. *Int. J. Theor. Phys.* **21**, 467. (1994).
12. Farhi, E., Goldstone, J. & Gutmann, S. A Quantum Approximate Optimization Algorithm. Preprint at https://arxiv.org/abs/1411.4028 (2014).
13. Clader, B., Jacobs, B. & Sprouse, C. Preconditioned quantum linear system algorithm. *Phys. Rev. Lett.* **110**, 250504 (2013).
14. Montanaro, A. & Pallister, S. Quantum algorithms and the finite element method. *Phys. Rev. A* **93**, 032324 (2016).
15. Wang, S. et al. Quantum fast Poisson solver: The algorithm and complete and modular circuit design. *Quantum Inf. Process.* **19**, 1–25 (2020).
16. Childs, A., Liu, J. & Ostrander, A. High-precision quantum algorithms for partial differential equations. *Quantum* **5**, 574 (2021).
17. Bagherimehrab, M., Nakaji, K., Wiebe, N. & Aspuru-Guzik, A. Fast quantum algorithm for differential equations. Preprint at https://arxiv.org/abs/2306.11802 (2023).
18. Das, . D. D., A. Quantum computing solution to Sturm–Liouville differential equation. *Int. J. Innov. Res. Phys.* **5**, 27–37 (2024).
19. Grilli, A. *Quantum algorithms for the solution of partial differential equations with applications in the aerospace sector*. Master's thesis, Department of Physics and Astronomy "Galileo Galilei"—DFA (2024).
20. Nielsen, M. & Chuang, I. *Quantum Computation and Quantum Information: 10th Anniversary Edition* (2011).
21. Suau, A., Staffelbach, G. & Calandra, H. Practical quantum computing: Solving the wave equation using a quantum approach. *ACM Trans. Quantum Comput.* **2**, 1–35 (2021).
22. Rajak, A., Suzuki, S., Dutta, A. & Chakrabarti, B. Quantum annealing: An overview. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **381**, 20210417 (2022).
23. O'Malley, D. & Vesselinov, V. ToQ.jl: A high-level programming language for D-Wave machines based on Julia. In *2016 IEEE High Performance Extreme Computing Conference (HPEC)* 1–7 (2016).
24. Rogers, M. & Singleton, R. Floating-point calculations on a quantum annealer: Division and matrix inversion. *Front. Phys.* **8**, 265 (2020).
25. Borle, A. & Lomonaco, S. Analyzing the quantum annealing approach for solving linear least squares problems. In *WALCOM: Algorithms And Computation: 13th International Conference, WALCOM 2019, Guwahati, India, February 27—March 2, 2019, Proceedings* 289–301 (2019).
26. Glover, F., Kochenberger, G. & Du, Y. A Tutorial on Formulating and Using QUBO Models. Preprint at https://arxiv.org/abs/1811.11538 (2019).
27. Inc, D. *Ocean Software Documentation*. https://docs.ocean.dwavesys.com/en/stable/
28. Suresh, S. & Suresh, K. Computing a sparse approximate inverse on quantum annealing machines. Preprint at https://arxiv.org/abs/2310.02388 (2023).
29. The D-Wave Advantage System: An overview. https://www.dwavesys.com/resources/white-paper/the-d-wave-advantage-system-an-overview/.
30. Saad, Y. *Iterative Methods for Sparse Linear Systems* (SIAM, 2003).
31. Zhang, C. Y., Xu, C. & Luo, S. Convergence of block iterative methods for linear systems with generalized H-matrices. *J. Comput. Appl. Math.* **229**, 70–84 (2009).
32. Rajchel, K. A new constructive method for solving the Schrödinger equation. *Symmetry* **13**, 1879 (2021).
33. Firmansah, E. & Rosyid, M. undefined (IOP Publishing, 2019).
34. Démoulin, B. Propagation of radiofrequency waves in space. Territory Movement J. Geography Planning, [Online], 51 | 2021, Online since 27 January 2022, connection on 11 February 2025. URL: http://journals.openedition.org/tem/8323; DOI: https://doi.org/10.4000/tem.8323 (2021).
35. Ciaramella, G. & Gander, M. *Iterative Methods and Preconditioners for Systems of Linear Equations* (SIAM, 2022).
36. Onabid, M. A. Solving three-dimensional (3D) Laplace equations by successive over-relaxation method. *Afr. J. Math. Comput. Sci. Res.* **5**, 204–208 (2012).
37. Robert, Y. Regular incomplete factorizations of real positive definite matrices. *Linear Algebra Appl.* **48**, 105–117 (1982).
38. McArdle, S., Endo, S., Aspuru-Guzik, A., Benjamin, S. & Yuan, X. Quantum computational chemistry. *Rev. Mod. Phys.* **92**, 015003 (2020).
39. Huelga, S. & Plenio, M. Vibrations, quanta and biology. *Contemp. Phys.* **54**, 181–207 (2013).
40. Lieb, E. & Yngvason, J. Ground state energy of the low density Bose gas. *Phys. Rev. Lett.* **80**, 2504–2507 (1998).
41. Farhi, E. et al. A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem. *Science* **292**, 472–475 (2001).
42. Jansen, S., Ruskai, M. & Seiler, R. Bounds for the adiabatic approximation with applications to quantum computation. *J. Math. Phys.* **48**, 102111 (2007).
43. Albash, T. & Lidar, D. Adiabatic quantum computation. *Rev. Mod. Phys.* **90**, 015002 (2018).

44. Johnson, M. et al. Quantum annealing with manufactured spins. *Nature* **473**, 194–8 (2011).
45. Kochenberger, G. et al. The unconstrained binary quadratic programming problem: A survey. *J. Comb. Optim.* **28**, 58–81 (2014).
46. Parhami, B. Introduction to Parallel Processing: Algorithms and Architectures. (Kluwer Academic Publishers, 1999).
47. Inc, D.-W. S. *Operation and Timing*. https://docs.dwavesys.com/docs/latest/c_qpu_timing.html.
48. Developer Group, D.-W. *Computational Power Consumption and Speedup*. https://www.dwavesys.com/resources/ white-paper/computational-power-consumption-and-speedup/.
49. Remani, C. *Numerical Methods for Solving Systems of Nonlinear Equations* Vol. 77 (Lakehead University Thunder Bay, 2013).

## Author contributions
All authors reviewed the manuscript

## Declarations

## Competing interests
The authors declare no competing interests.

## Additional information
**Correspondence** and requests for materials should be addressed to M.A.

**Reprints and permissions information** is available at www.nature.com/reprints.

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