Quantum Information Processing (2023) 22:258
https://doi.org/10.1007/s11128-023-04020-2

Near term algorithms for linear systems of equations

Aidan Pellow-Jarman[1,2] · Ilya Sinayskiy[1,3] · Anban Pillay[1,4] · Francesco Petruccione[1,3,5]

Received: 1 December 2022 / Accepted: 9 June 2023

Abstract
Finding solutions to systems of linear equations is a common problem in many areas of science and engineering, with much potential for a speed up on quantum devices. While the Harrow–Hassidim–Lloyd (HHL) quantum algorithm yields up to an exponential speed up over classical algorithms in some cases, it requires a fault-tolerant quantum computer, which is unlikely to be available in the near-term. Thus, attention has turned to the investigation of quantum algorithms for noisy intermediate-scale quantum (NISQ) devices where several near-term approaches to solving systems of linear equations have been proposed. This paper focuses on the Variational Quantum Linear Solvers (VQLS), and other closely related methods and adaptions. Several contributions are made in this paper, which include: the first application of the Evolutionary Ansatz to the VQLS (EAVQLS), the first implementation of the Logical Ansatz VQLS (LAVQLS), based on the Classical Combination of Quantum States (CQS) method, a proof of principle demonstration of the CQS method on real quantum hardware and

✉ Ilya Sinayskiy
sinayskiy@ukzn.ac.za

Aidan Pellow-Jarman
aidanpellow@gmail.com

Anban Pillay
pillayw4@ukzn.ac.za
http://www.cair.org.za

Francesco Petruccione
petruccione@sun.ac.za

1 University of KwaZulu-Natal, University Road, Chiltern Hill, Westville 3629, South Africa

2 Qunova Computing Inc., Creativity Hall, C2119, KAIST-Munji Campus, 193 Munji-ro, Yuseong-gu, Daejeon 34051, South Korea

3 National Institute for Theoretical and Computational Sciences (NITheCS), Stellenbosch, South Africa

4 Centre for Artificial Intelligence Research (CAIR), Stellenbosch, South Africa

5 School of Data Science and Computational Thinking, Stellenbosch University, Stellenbosch, South Africa

Published online: 24 June 2023

a method for the implementation of the Adiabatic Ansatz on the VQLS (AAVQLS). These approaches are implemented and contrasted. The CQS method is run with moderate success on a real quantum device. The EAVQLS and AAVQLS show promise as possible improvements to the standard VQLS algorithm once refined.

Keywords Variational · Linear equations · Near-term quantum · Quantum-classical · VQLS

1 Introduction

Systems of linear equations play an important role in many areas of science and engineering, making the potential quantum speed-up for solving them of great interest. Solving a system of N linear equations with N unknowns, expressible as $\mathbf{Ax} = \mathbf{b}$, involves finding the unknown solution vector $\mathbf{x}$. This is known as the Linear Systems Problem (LSP).

The Harrow–Hassidim–Lloyd (HHL) algorithm [1] is a quantum algorithm for the quantum linear systems problem (QLSP) [2], a quantum analogue of the LSP. The QLSP is stated as follows: Let $\mathbf{A}$ be an $N \times N$ Hermitian matrix (however, this algorithm is not limited to a Hermitian matrix) and let $\mathbf{x}$ and $\mathbf{b}$ be N dimensional vectors, satisfying $\mathbf{Ax} = \mathbf{b}$, having corresponding quantum states $|x\rangle$ and $|b\rangle$, such that

$$|x\rangle := \frac{\sum_i x_i |i\rangle}{||\sum_i x_i |i\rangle||_2}, \tag{1}$$

$$|b\rangle := \frac{\sum_i b_i |i\rangle}{||\sum_i b_i |i\rangle||_2}. \tag{2}$$

If $\mathbf{A}$ is not Hermitian, define $\tilde{\mathbf{A}} = \left(\begin{smallmatrix} 0 & \mathbf{A} \\ \mathbf{A}^\dagger & 0 \end{smallmatrix}\right)$, which is Hermitian, and instead solve the equation $\tilde{\mathbf{A}}\mathbf{y} = \left(\begin{smallmatrix} \mathbf{b} \\ 0 \end{smallmatrix}\right)$ and solve for $\mathbf{y} = \left(\begin{smallmatrix} 0 \\ \mathbf{x} \end{smallmatrix}\right)$. Given access to matrix $\mathbf{A}$ by means of an oracle, and a unitary gate U such that $U|0\rangle = |b\rangle$, output a quantum state $|x'\rangle$ such that $|||x\rangle - |x'\rangle||_2 \leq \epsilon$, where ϵ is the error-bound of the approximate solution.

The HHL algorithm is a quantum algorithm expected to give a substantial speed-up over classical approaches, providing up to an exponential speed-up over known classical algorithms in cases where the linear system is sparse, the condition number is low, and the actual solution vector is not required to be read out, but instead some scalar measure on the solution vector is of interest. As with many promising quantum algorithms, the HHL algorithm requires a fault-tolerant quantum computer to be successfully implemented, predicted to only be available in the long-term future.

Approaches at finding algorithms for noisy intermediate-scale quantum (NISQ) devices [3], available in the near-term future, have focused mainly on a class of algorithms known as Variational Hybrid Quantum Classical Algorithms (VHQCAs). The idea behind VHQCAs is to utilize a quantum-classical feedback loop. Here, a quantum device is used to compute a cost function for a parameterized quantum circuit (ansatz), much more efficiently than is possible on a classical device [6], while a classical device

is used to optimize the selection of the ansatz parameters. VHQCAs rely on the use of short depth quantum circuits to make them more resistant to noise and allowing them to be successfully run on NISQ quantum hardware. The main difficulties of this approach lie in overcoming the noise inherent in quantum devices and the difficulty of optimizing the ansatz parameters. An example of these difficulties is the barren plateau problem [4].

The Variational Quantum Eigensolver (VQE) [5] is one notable VHQCA that solves the optimization problem,

$$E = \min_{\theta} \langle V(\theta)|H|V(\theta)\rangle, \tag{3}$$

whereby the minimum eigenvalue E of some Hamiltonian H is approximated through the optimization of θ for some ansatz $V(\theta)$.

The Variational Quantum Linear Solver [6–8] is based on the VQE, recently proposed to solve the quantum linear systems problem. Since its proposal, many variations have also been presented in order to overcome various difficulties faced by the algorithm, and VHQCAs in general. Attempts to combat these difficulties include training approaches like the Adiabatic Assisted VQE [7] and ansatz variations such as the Logical Ansatz, being a Classical Combination of various Ansätze [7, 10] and the Evolutionary Ansatz [11], an evolutionary approach for Ansatz construction. The evolutionary ansatz was initially proposed for use in the VQE and has been applied here for the VQLS variation. Another non-variational approach to solving the quantum linear systems problem is the Classical Combination of Quantum States (CQS) method [7], of which the Logical Ansatz approach outlined in this paper is an adaption.

The following approaches were implemented and will be discussed in this paper:

Method	Abbreviation
Variational quantum linear solver	(VQLS)
Adiabatic ansatz VQLS	(AAVQLS)
Evolutionary ansatz VQLS	(EAVQLS)
Classical combination of quantum state	(CQS)
Logical ansatz VQLS	(LAVQLS)

This paper makes several contributions to the literature on the VQLS. Firstly, we present the first application of the Evolutionary Ansatz to the VQLS. The Evolutionary Ansatz has previously been applied to the Variational Quantum Eigensolver [11]. Secondly, the implementation proposed for solving systems of linear equations using the AAVQLS is new to this work. Also, a proof of principle demonstration of the CQS method on a real quantum device is conducted. Lastly, the first known implementation of the Logical Ansatz VQLS is given, with proposed training methods that are new to this work. All implementations of these approaches may be found in the Github repository. [1]

[1] https://github.com/aidanpellow/vqls.

This paper begins with a description of the above near-term approaches for the Quantum Linear Systems Problem. Some experiments designed for an evaluation of these approaches are outlined in the section following that. We then present and discuss the results of the experiments.

2 Near-term algorithms

The inputs to the all the near-term algorithms below are the matrix $\mathbf{A}$, and vector $\mathbf{b}$. $\mathbf{A}$ is given in a slightly different form here than in the QLSP. Here, $\mathbf{A}$ is given by m unitary matrices A_i, implemented as unitary gates, such that $\mathbf{A} = \sum_{i=1}^{m} c_i A_i$, $c_i \in \mathbb{C}$ (any Hermitian matrix in a finite-dimensional space can be written as a linear combination of unitary matrices). Also given is a unitary gate U, such that $U|0\rangle = |b\rangle$. VQLS cost functions often require either or both A_i and U to be given as controlled gates, which is assumed possible.

2.1 Variational quantum linear solver

The standard VHQCA approach for the quantum linear systems problem is the Variational Quantum Linear Solver, itself being a basic application of the VQE. The VQLS simply involves the selection of a suitable ansatz, cost function, and classical optimizer. The algorithm runs in a simple feedback-loop, whereby the classical optimizer finds the optimal parameters for the ansatz circuit, by iteratively evaluating the cost function on the quantum device, and updating the parameters until a minimum cost value is achieved. The quantum device is used to evaluate the cost function, because it is much more efficient than any known method on a classical device for this step [6].

Let the ansatz be denoted by $V(\alpha)$, and let the optimal ansatz parameters be denoted by α^*. Then, once the VQLS algorithm terminates, $V(\alpha^*)|0\rangle = |x'\rangle$, where $|||x\rangle - |x'\rangle||_2 \leq \epsilon$, where ϵ is the error-bound of the approximate solution, and $|x\rangle$ is the exact solution as described by Eq. (3) (Fig. 1).

2.1.1 VQLS ansatz

Broadly speaking, there are two types of ansätze; hardware-efficient (agnostic) ansätze and problem-specific ansätze.

Hardware-efficient ansätze are designed without taking into account the specific problem being solved, that is matrix $\mathbf{A}$ and $|b\rangle$, but rather only the topology (backend connectivity of the qubits) and available gates of a specific quantum computer. A hardware-efficient ansatz can be denoted by a sequence of n parameterized quantum gates as,

$$V_{\text{Agnostic}}(\alpha) = \gamma_1(\alpha_1)\gamma_2(\alpha_2)\cdots\gamma_n(\alpha_n), \tag{4}$$

where γ_i denotes a specific parameterized gate in the quantum circuit, and α_i denotes that parameters value.

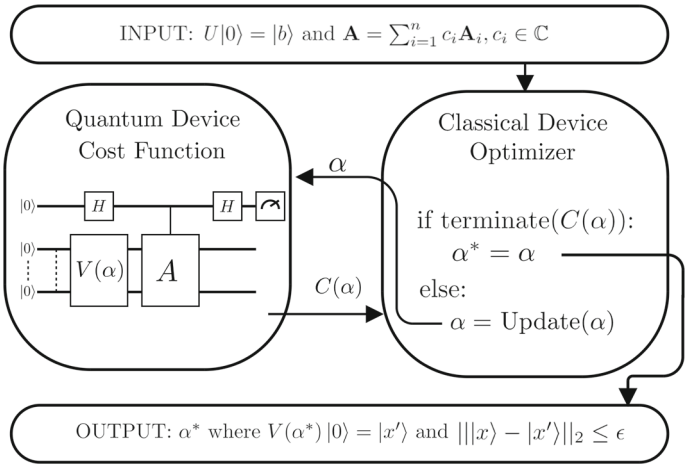

Fig. 1 VQLS Schematic: The algorithm runs in a simple feedback loop, whereby a classical optimizer finds the optimal parameters for the ansatz circuit, by iteratively evaluating the cost function on the quantum device and updating the parameters, until a minimum cost value is achieved. The matrix $\mathbf{A}$ is given as a linear sum of unitaries $\mathbf{A}_i$ with complex coefficients c_i, and the vector $\mathbf{b}$ is given as a unitary U such that $U|0\rangle = |b\rangle$. $V(\alpha)$ denotes the parameterized ansatz, α^* denotes the optimal parameters. At termination, $V(\alpha^*)|0\rangle = |x'\rangle$, where $|||x\rangle - |x'\rangle||_2 \leq \epsilon$, with $|x\rangle$ being the exact solution to the QLSP, and ϵ being the error-bound of the approximate solution

These ansätze can be constructed to be more resistant to noise on any specific available quantum device, but they may fall short finding a solution $|x'\rangle$, as any particular hardware-efficient ansätze is not guaranteed to span the region of the Hilbert space containing any good approximation of the solution $|x'\rangle$. Therefore, a hardware-efficient ansatz effectively trades potential relevance to the specific problem, for increased noise resistance.

Problem-specific ansätze on the other hand do not take into account the specific quantum device being used, and rather try to exploit the knowledge of the problem available. The Quantum Alternating Operator Ansatz (QAOA) [6] is one such proposed problem-specific ansatz, using two Hamiltonians, known as the *driver* and the *mixer*, denoted by H_D and H_M, respectively, constructed from specific knowledge of the problem, namely $\mathbf{A}$ and $\mathbf{b}$. This problem-specific ansatz can be denoted by a repeating sequence of *driver* and *mixer* Hamiltonian simulations, each being applied for a variable amount of time. These time parameters α are the trainable aspect of this problem specific ansatz, which are optimized by some classical device. The QAOA can be denoted as,

$$V_{QAOA}(\alpha) = e^{-iH_M\alpha_{2p}} e^{-iH_D\alpha_{2p-1}} \dots e^{-iH_M\alpha_2} e^{-iH_D\alpha_1}. \tag{5}$$

The requirement of Hamiltonian simulation from the QAOA makes these ansätze far less near-term; therefore, these ansätze are not considered further in this paper. More information on the specific construction of the QAOA, including operators H_D and H_M, is given in [6].

2.1.2 VQLS cost functions

The cost function Hamiltonian is where the application of the VQE algorithm to solving systems of linear equations is implemented. Various different cost functions have been proposed for the VQLS [6, 7]. For simplicity, denote the state $V(\alpha)|0\rangle$, as $|x\rangle$, and let $|\psi\rangle = A|x\rangle$. Ref. [6] proposes a cost function based on the overlap between the projector $|\psi\rangle\langle\psi|$ and the subspace orthogonal to $|b\rangle$. This cost function also normalizes the expectation value of the Hamiltonian to improve performance. The cost function is given by,

$$C_G = \frac{\langle x|H_G|x\rangle}{\langle\psi|\psi\rangle}, \tag{6}$$

where the Hamiltonian H_G is given by,

$$H_G = A^\dagger \left(\mathbb{1} - |b\rangle\langle b|\right) A. \tag{7}$$

This cost function can have gradients that vanish exponentially with the number of qubits. To improve on this shortfall, the cost function C_L is proposed by replacing H_G with a local version of the Hamiltonian, H_L, improving the trainability of the ansatz, given by,

$$H_L = A^\dagger U \left(\mathbb{1} - \frac{1}{n}\sum_{j=1}^{n} |0_j\rangle\langle 0_j| \otimes \mathbb{1}_{\bar{j}}\right) U^\dagger A, \tag{8}$$

where $\mathbb{1}_{\bar{j}}$ denotes identity on all qubits except qubit j. The cost function C_L can be computed using the Hadamard Test as shown in Fig. 2. C_L has been shown to be equivalent cost function to C_G, however, having improved performance [6], and is explicitly given by,

$$C_L = \frac{\langle x|H_L|x\rangle}{\langle\psi|\psi\rangle}. \tag{9}$$

2.1.3 Classical optimizers

The VQLS admits the use of either gradient-free or gradient-based optimizers. For gradient-based optimizers, gradient values can be found analytically [6, 12], or estimated through finite differences. The classical optimizer chosen has a large impact on how well the optimization process deals with the noise inherent in NISQ devices. Some classical optimizers handle noise better than others [13], making classical optimizer selection important.

2.2 Adiabatic-assisted VQLS

The Adiabatic-Assisted Variational Quantum Linear Solver (AAVQLS) [7] simply augments the standard VQLS approach, by proposing a variation in the Hamiltonian

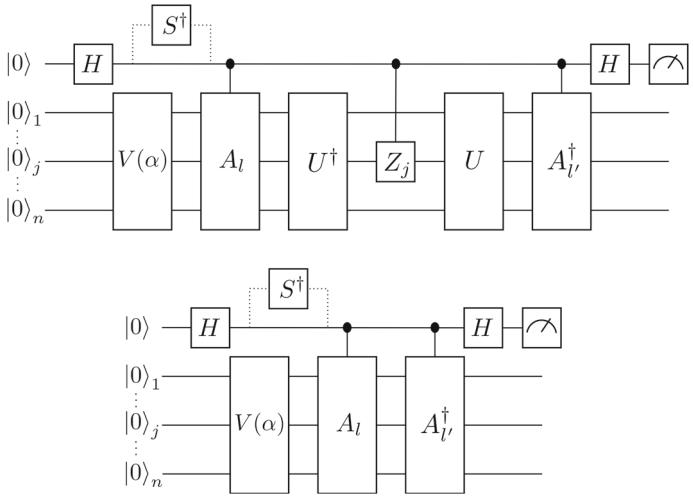

Fig. 2 Hadamard Test Circuits for cost function C_L Eq. (9). The top circuit is employed when calculating the value of the numerator $\langle x|H_L|x\rangle$, while the bottom circuit is employed when calculating the value of the denominator, $\langle\psi|\psi\rangle$. The $S^\dagger$ gate is included when calculating imaginary-valued parts of the cost function and excluded when calculating the real-valued parts, and therefore, drawn in as a dotted line. $V(\alpha)$ denotes the ansatz, A_l the l-th unitary from the linear sum $A = \sum_i c_i A_i$ and U the unitary such that $U|0\rangle = |b\rangle$. Z_j denotes a standard Z gate on the j-th qubit, and H being a standard Hadamard gate

over time, inspired by adiabatic quantum computing methods [14], in an attempt to allow the ansatz state to always be close to the ground state of the Hamiltonian. Let H_0 be a Hamiltonian with a known ground state, and let H_1 be the Hamiltonian whose ground state corresponds to the solution of the linear system in question. Let the Hamiltonian of the AAVQLS be given by $(1-s)H_0 + sH_1$, where s is a discrete parameter, varying from $s = 0$, to $s = 1$, in T discrete intervals, during the optimization process.

This approach is the same as the VQLS with respect to the cost function and classical optimizer; however, the ansatz must be chosen such that it can be easily initialized in the ground state of H_0 at the start of the algorithm. The only added procedure to the AAVQLS from the VQLS occurs during the training phase, where the parameter s is varied in T discrete intervals from $s = 0$ to $s = 1$, thereby varying the Hamiltonian from H_0 to H_1.

Proposed here is one way in which the AAVQLS can be implemented as an adaption of the VQLS. Firstly, the linear system is reformulated as,

$$[(1-\bar{s})\mathbb{1} + \bar{s}\mathbf{A}]\mathbf{x} = \mathbf{b}, \tag{10}$$

where $\bar{s}$ can be varied from 0 to 1, with $\mathbf{x} = \mathbf{b}$, when $\bar{s} = 0$, and $[(1-\bar{s})\mathbb{1} + \bar{s}\mathbf{A}]\mathbf{x} = \mathbf{b}$, equivalent to $\mathbf{Ax} = \mathbf{b}$, when $\bar{s} = 1$.

Then, for a suitable ansatz $V(\alpha) = \gamma_1(\alpha_1)\gamma_2(\alpha_2)\cdots\gamma_n(\alpha_n)$ where α can be initialized such that $V(\alpha) = \mathbb{1}$, append to it the unitary U (for creating state $|b\rangle$) to create ansatz $V_{AAVQLS}(\alpha)$ as below,

$$V_{AAVQLS}(\alpha) = UV(\alpha). \tag{11}$$

Here, $V_{AAVQLS}|0\rangle$ is indeed equal to $|b\rangle$ when α is initialized appropriately (such that $V(\alpha) = \mathbb{1}$).

The AAVQLS algorithm then proceeds as follows:

1. Let $\overline{s} = 0$ and let ansatz parameters $\alpha = \alpha^*$, such that $V(\alpha^*)$ is equal to $\mathbb{1}$. Then, $V_{\text{AAVQLS}}(\alpha)$ gives the solution to (9). Let $\alpha_{\text{previous}} = \alpha$.
2. Increment $\overline{s}$ by $\frac{1}{T}$.
3. Using the VQLS approach, with initial ansatz parameters set to α_{previous}, find the new optimal parameters α_{current} with ansatz V_{AAVQLS}.
4. Let $\alpha_{\text{previous}} = \alpha_{\text{current}}$ and if $\overline{s} \neq 1$ return to step 2; else $\alpha_{\text{final}} = \alpha_{\text{current}}$.
5. $V_{\text{AAVQLS}}(\alpha_{\text{final}})|0\rangle = |x'\rangle$, where $|||x\rangle - |x'\rangle|| \leq \epsilon$, where $|x\rangle$ is the exact solution to the QLSP and ϵ is the error-bound of the approximate solution.

2.3 Evolutionary ansatz VQLS

The Evolutionary Ansatz [11] utilizes a genetic algorithm to construct the parameterized quantum circuit, while concurrently optimizing its parameters. This approach adapts the ansatz structure to both the specified problem and the backend configuration of the quantum device available, so it can be thought of as constructing an ansatz that is both hardware-efficient and problem-specific. This specialized ansatz would be highly noise resistant (being shallower and requiring fewer 2-qubit gates) while remaining applicable to the specific problem being solved. This approach still requires a VQLS cost function and classical minimizer, as standard with VQLS.

The main outline of the Evolutionary Ansatz algorithm is presented here. For a full explanation, please see the original paper[11]. Some specific details about the evolutionary ansatz implementation in this discussion differ from the original paper.

Evolutionary Algorithms are based on the principle of natural selection. The Evolutionary Ansatz VQLS (EAVQLS) mimics this principle to adapt the ansatz choice by evolving a set of candidate ansatze, known as the population, through random mutations (the EAVQLS only mimics asexual reproduction, there are no crossovers between candidate solutions in the population). The candidate ansätze being evolved are known as genomes. A genome consists of a list of genes, and for the EAVQLS is given as follows. If $V_i(\alpha)$ is any potential ansatz in the population, $V_i(\alpha)$ is expressed as a genome g_i that can be written as,

$$g_i = [\gamma_1(\alpha_1), \gamma_2(\alpha_2), \cdots, \gamma_m(\alpha_m)],$$

where each γ_i is a gene. Each gene γ_i is a layer of gates such that each qubit of the ansatz is assigned a gate. This set of gates is chosen from a gate set,

$$\mathbb{G} = \{\mathbb{I}_2, U_3, \wedge_1 U_3\},$$

where $\mathbb{I}_2$ is the single qubit identity gate, U_3 is the universal single qubit rotation, and $\wedge_1 U_3$ is the controlled version of U_3. Other gate sets may be used, for example, if the

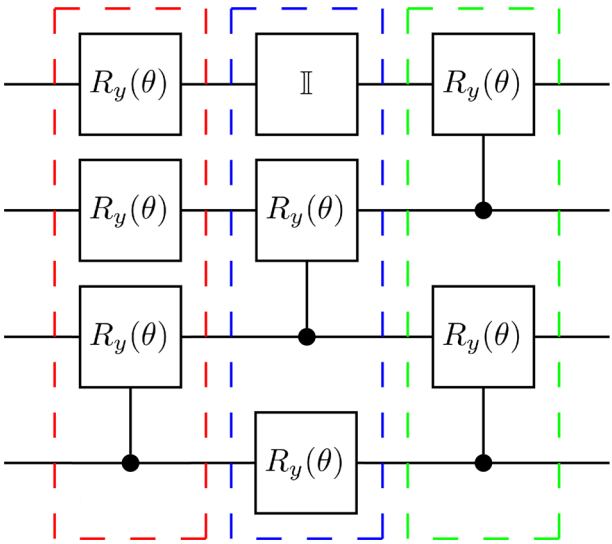

Fig. 3 4 Qubit evolutionary Ansatz genome schematic: three different genes (ansatz rows) are each outlined in red, blue and green in the genome above. The qubits are initialized as $H^{\otimes n}|0\rangle^{\otimes n}$, allowing the first gene in the genome to contain controlled gates that are not redundant. $R_y(\theta)$ denotes a the standard y-rotation gate (these could be replaced with U_3 in the case of a complex-valued solution), and $\mathbb{I}$ the standard identity gate

VQLS problem only deals with real valued **A** and **b**, an appropriate gate set is given by,

$$\mathbb{G}_{\mathbb{R}} = \{\mathbb{I}_2, R_y, \wedge_1 R_y\}.$$

An example of an evolutionary ansatz genome is shown in Fig. 3.

Initially, the evolutionary algorithm begins with a population consisting of random genomes, each only consisting of a single gene. The qubits in all ansatz circuits are initialized with the gate $H^{\otimes n}$, allowing the first gene in the genome to contain controlled gates that are not redundant when the ansatz corresponding to the genome is applied to the state $|0\rangle^{\otimes n}$. These genomes are evolved asexually, through the use of 3 genetic operators, *topological search*, *parameter search* and *removal*.

The *topological search* operator, τ, explores the space of possible ansätze, by adding a new random gene to the genome, which is equivalent to adding a new random layer of gates to the ansatz represented by the genome. The new gene added to the genome is initialized as identity, to ensure that the genome's fitness may only improve, or at worst, remain the same. This new gene also takes into account the gates of the previous gene in the genome, eliminating potential redundant gates from being added to the genome with the new gene (two of the same gate on the same qubit/s in a row results in a redundancy). The identity gate, $\mathbb{I}_2$, is only used whenever adding a different gate would cause some redundancy. The operation performed by τ is given by,

$$\tau : [\gamma_1(\alpha_1), \cdots, \gamma_m(\alpha_m)] \rightarrow [\gamma_1(\alpha_1), \cdots, \gamma_m(\alpha_m), \gamma_{m+1}(\alpha_{m+1})].$$

The *removal* operator, ρ, acting on a genome, removes some number of genes from the genome, starting from the end of the gene list. The operation performed by ρ is given by,

$$\rho : [\gamma_1(\alpha_1), \cdots, \gamma_p(\alpha_p), \cdots, \gamma_m(\alpha_m)] \rightarrow [\gamma_1(\alpha_1), \gamma_2(\alpha_2), \cdots, \gamma_p(\alpha_p)]$$

where $p \in \{1, 2, \cdots, m\}$.

The *parameter search* runs an optimization subroutine, $\mathbb{O}$, on each individual gene in the genome, in a random order. The optimization subroutine $\mathbb{O}$ is implemented for a genome g_i and a gene $y_i(\alpha_i)$, by using the ansatz $V(\alpha)$ represented by genome g_i in the standard VQLS optimization routine, while only optimizing the specific parameters α_i of the gene $y_i(\alpha_i)$. This optimization is done per gene so that *removal* operator does not affect the training of the rest of the ansatz.

The fitness f_i of genome g_i is calculated using the value of the cost achieved by the ansatz represented by the genome, as well as the depth of the ansatz (number of genes) and the number of 2-qubit gates. The fitness value is given by,

$$f_i = \mathbb{C}(g_i) + \alpha \cdot |g_i| + \beta \cdot |\wedge(g_i)|,$$

where α and β are weighted variables that can be assigned, $\mathbb{C}(g_i)$ is the cost value of the ansatz of represented by g_i, $|g_i|$ is the number of genes in g_i, and $|\wedge(g_i)|$ is the number of 2-qubit gates in g_i. This genome fitness value is then averaged among genomes of the same species and is called the species-adjusted fitness. Species are defined by a genetic distance measure, given by the average distance of a common ancestor between two separate genomes. Two genomes with an average distance of a common ancestor less than some given value may be assigned to the same species.

The EAVQLS Algorithm runs as follows:

1. Generate a population P of n empty genomes g_i, and apply $\tau(g_i)$ to each genome.
2. Apply the optimization subroutine $\mathbb{O}$ to the last gene in each genome for all genomes in P.
3. Group the genomes in P by species, then calculate the fitness and then species-adjusted fitness of each genome g_i in P.
4. Randomly select n parent genomes with replacement from P, inversely proportional to their fitness values, forming the next generation P'.
5. For all of the genomes g_i in P', apply the three genetic operators to g_i with some predefined probabilities.
6. If a termination condition is met, return the fittest genome in P', else let $P = P'$ and return to step 2.

2.4 Classical combination of quantum states (CQS)

The Classical Combination of Quantum States (CQS) approach detailed in [7] is the most unique near-term approach presented in this paper. The CQS approach is not a variational algorithm, meaning that there is no classical optimization of ansatz parameters. The CQS approach solves the linear system by finding the solution as a

linear combination of quantum states. A similar method is detailed in [9], where the first proof of principle demonstration of this method is given.

Given a set of n states $S = \{|s_1\rangle, \cdots, |s_n\rangle\}$, the CQS method aims to find a linear combination x' approximating the solution x to the linear systems problem $A\mathbf{x} = \mathbf{b}$ where,

$$x' = \sum_{i=1}^{n} \alpha_i |s_i\rangle, \alpha \in \mathbb{C}. \tag{12}$$

Note that, differing from the above-mentioned approaches, the solution x' is never actually created on the quantum device. It is not necessarily normalized and is proportional to the solution to the same problem solved using the other variational methods.

Starting with $m = 1$, and the set $S = \{|s_1\rangle\}$ where $|s_1\rangle$ is a state that can be prepared by some efficient quantum circuit. The CQS Algorithm then runs as follows:

1. Solve the for the optimal values of α^* such that $x' = \Sigma_{i=1}^{m} \alpha_i^* |s_i\rangle$, where x' is the an approximation of x, given the set S.
2. Using the value of α^*, find the next circuit generating the state $|s_{m+1}\rangle$ and add $|s_{m+1}\rangle$ to S.
3. Set $m = m + 1$ and repeat from step 1 until x' is a sufficiently good solution.

Given a set of n efficiently prepared states, $S = \{|s_1\rangle, \cdots, |s_n\rangle\}$, containing a close approximation of the solution x as a linear combination, the linear coefficients, c_i, can be found efficiently by a hybrid quantum-classical procedure outlined below.

The standard regression function used to solve a linear system is given by,

$$L_R(x) := ||Ax - |b\rangle||_2^2 = x^\dagger A^\dagger A x - 2\mathrm{Re}[\langle b|Ax] + 1. \tag{13}$$

Given $x = \sum_{i=1}^{m} \alpha_i |s_i\rangle$, let $V = (v_1, \ldots, v_m)$ where $v_i = A|s_i\rangle$. Now, $Ax = \sum_{i=1}^{m} \alpha_i A|s_i\rangle = V\alpha$. Equation (13) can be rewritten as,

$$||V\alpha - |b\rangle||_2^2 = \alpha^\dagger V^\dagger V \alpha - 2\mathrm{Re}\{q^\dagger \alpha\} + 1. \tag{14}$$

where $q_i = \langle i|V^\dagger|b\rangle = \langle s_i|A^\dagger|b\rangle$. A simple regression problem for α can be obtained with the kernel matrix $V^\dagger V$, where $(V^\dagger V)_{ij} = \langle s_i|A^\dagger A|s_j\rangle$. This quadratic optimization problem on complex α can be rewritten as a real optimization problem,

$$\min_z (z^\dagger Q z - 2r^T z + 1), \tag{15}$$

where $z = (\mathrm{Re}[\alpha], \mathrm{Im}[\alpha])$, and Q and r are given by,

$$Q = \begin{pmatrix} \mathrm{Re}[V^\dagger V] & \mathrm{Im}[V^\dagger V] \\ \mathrm{Im}[V^\dagger V] & \mathrm{Re}[V^\dagger V] \end{pmatrix}, \tag{16}$$

$$r = \begin{pmatrix} \mathrm{Re}[q] & \mathrm{Im}[q] \end{pmatrix}. \tag{17}$$

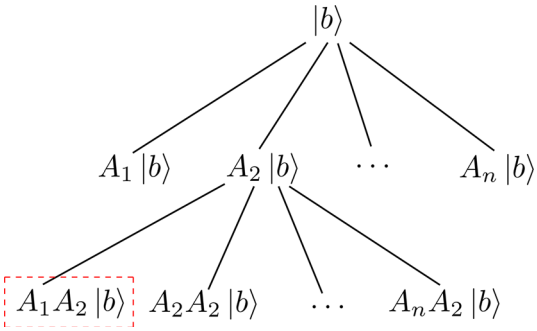

Fig. 4 Ansatz tree diagram: an example expansion of the Ansatz Tree. The highlighted node in the tree represents the unitary created by applying the unitaries A_2 and then A_1 to the state $|b\rangle$

Here, Eq. (15) can be solved using standard convex quadratic programming methods (Fig. 4).

The ansatz tree (Fig. 4) is a proposed structure used to obtain a good set of unitaries. As previously specified, the matrix A is given as a linear combination, $A = \sum_{i=1}^{n} c_i A_i, c_i \in \mathbb{C}$. The unitaries A_i are used in the construction of the ansatz tree as follows.

Each node in the ansatz tree represents a single state, $|s_i\rangle$, composed from the unitaries $\mathbf{A}_i$ and $|b\rangle$, which can be added to the ansatz set used to find the linear combination on a classical device. The ansatz tree can be expanded breadth-first, or searched via some heuristic.

A heuristic approach to searching the ansatz tree also proposed in [7] aims to expand the ansatz tree node by node. Let the current set of expanded nodes in the tree be given by the set S, and the current set of all potential child nodes of the nodes in S be the set $C(S)$. Let the current approximate solution for the set of expanded nodes be given by α^*. The ansatz tree is further explored by expanding the child node, $|\psi\rangle \in C(S)$, that has the greatest overlap with the current gradient, maximizing the function given by,

$$\langle\psi|\nabla L_R(x^s) = 2\sum_{|\psi_i\rangle\in S} \alpha_i^* \langle\psi|A^2|\psi_i\rangle - 2\langle\psi|A|b\rangle. \tag{18}$$

2.5 Logical ansatz VQLS

The logical ansatz approach is simply an adaption of the CQS method above, allowing for parameterized unitaries to be employed. This approach is then similar to the Standard VQLS approach, except it proposes that instead of a single ansatz circuit, a linear combination of multiple ansatz circuits be used. This greatly lowers the depth of any one of the multiple ansatz circuits. The Logical Ansatz is implemented here as suggested in [7], by implementing the CQS method with a selected set of n parameterized ansatze making the states $S = \{|s_1(\theta_1)\rangle, \cdots, |s_n(\theta_n)\rangle\}$, and repeating the optimization process outlined for the CQS with a classical minimizer optimizing the parameters θ_i of the ansatze creating the states $|s_i(\theta_i)\rangle$. This method avoids the ansatz tree expansion process for selecting unitaries, by instead optimizing a set of pre-selected parameter-

ized unitaries. The solution found is expressed by,

$$x' = \sum_{i=1}^{n} \alpha_i |s_i(\theta_i)\rangle, \alpha_i \in \mathbb{C}. \tag{19}$$

This logical ansatz implementation and training differs from that in [10].

The Logical Ansatz Linear Solver Algorithm runs much like the CQS method:

1. Select n parameterized ansatze each corresponding to some state $|s_i(\theta_i)\rangle$, forming set S.
2. Solve the for the optimal value of α^* such that $x' = \Sigma_{i=1}^{m} \alpha_i^* |s_i(\theta_i)\rangle$, where x' is the closest approximation of x, given the set S. Proceed either to 3 or 4 for method 1 or 2, respectively.
3. Method 1: For r training rounds, select each of the states $|s_i(\theta_i)\rangle$ at random and optimize their parameters θ_i with some classical optimizer, only solving the new α^* value after the parameter optimization of each ansatz.
4. Method 2: Treating the entire state $x' = \Sigma_{i=1}^{m} \alpha_i^* |s_i(\theta_i)\rangle$ as a single logical ansatz, optimize all parameters at once with a classical optimizer, solving for the new α^* value with each change of parameter during the optimization process.

3 Experimentation and evaluation

Tests of all above described methods follow below. The linear systems to be solved are given as a matrix $\mathbf{A}$, where $\mathbf{A} = \Sigma_l c_l A_l$, for l unitary gates, and a state $|b\rangle$, given as a unitary U, prepared by some efficient quantum circuit, such that $U|0\rangle = |b\rangle$, corresponding to some $\mathbf{b}$, as described in the formulation of the near-term Quantum Linear Systems Problem.

In all problem instances detailed below, the state $|b\rangle = H^{\otimes n}|0\rangle^n$, where n is the relevant number of qubits for the problem. These problems are also only real-valued linear systems; however, these approaches are not limited to real-values. The number of shots for the Qiskit's Qasm simulator is set to 10,000 (except for the CQS approach on the real device).

3.1 Variational quantum linear solver

Three problem instances of differing sizes were selected to investigate the performance of the standard VQLS. Two different classical optimizers (gradient-based vs gradient-free), and three levels of noise were tested in order to further investigate their role in the performance of the VQLS. The two chosen optimizers were the gradient-based Broyden–Fletcher–Goldfarb–Shanno algorithm (BFGS) [15] (using an analytic gradient function computed on the quantum hardware) and the gradient-free Simultaneous Perturbation Stochastic Approximation algorithm (SPSA) [16], based on performance in [13]. The three noise levels were chosen such to demonstrate the difference between zero noise, shot-noise only and realistic NISQ device noise, given by Qiskit's Stat-

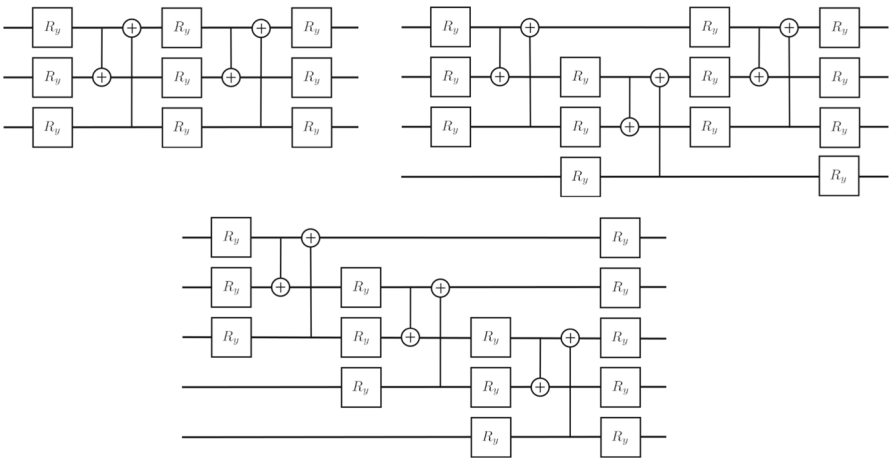

Fig. 5 The ansätze selected for the 3, 4 and 5 qubit problem instances. These consist of R_y and controlled-X gates, as the problems tested were real-valued

evector simulator, Qasm simulator and Qasm simulator with a realistic noise sample, respectively. The noise sample is taken from the IBM-Vigo quantum device.

The three problem instances, using 3, 4 and 5 qubits, respectively, are given by,

$$\begin{aligned}\mathbf{A}_1 &= H_1 + 0.25 \cdot Z_2 + 0.15 \cdot H_3,\\ \mathbf{A}_2 &= Z_1 + 0.25 \cdot Z_2 + 0.5 \cdot Z_4,\\ \mathbf{A}_3 &= H_1 + 0.25 \cdot Z_3 + 0.5 \cdot H_5,\end{aligned}$$

where Z_i, $(i = 1, 2, 3)$ indicates the matrix formed by the tensor product, with Pauli gate Z applied to qubit i and the identity gate applied to the remaining qubits. Notation is similarly defined with Hadamard gate H and Pauli gate X. $\mathbb{1}$ indicates an $N \times N$ identity matrix.

The local cost function (detailed above) was selected for all the VQLS runs. 100 runs of the VQLS algorithm were conducted for each problem instance, noise-level and classical optimizer. Furthermore, the same 100 random initial ansatz parameter values were used for all runs in the same problem instance across all noise levels and classical optimizers. This was done to make the results obtained for each problem instance comparable.

In order to ensure even resource distribution between the gradient-based BFGS and the gradient-free SPSA classical optimizers, the optimizers were only allowed a limited number of cost function evaluations. The resource cost of a gradient call can be given in terms of cost function calls, being exactly 2 cost function calls per ansatz parameter, and so this comparison can be done. The 3, 4 and 5 qubit problem instances were limited to 1000, 1500 and 2000 cost function evaluations, respectively.

The ansätze selected for each of the problem instances are shown in Fig. 5. Ansatz selection was not done with any specific backend in mind, and as such, the ansatz used

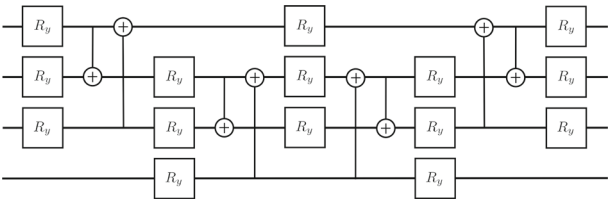

Fig. 6 The ansatz employed in the AAVQLS test. This ansatz can be initialized to identity with some random parameters. The circuit U for creating $|b\rangle$ is appending to the end of this ansatz in accordance with the method discussed in the AAVQLS description in order to create the adiabatic ansatz

have been assumed to be hardware efficient. No transpilation to any specific backend connectivity was done, even for the realistic noise simulation.

3.2 Adiabatic ansatz variational quantum linear solver

The following test of the AAVQLS algorithm compares the Adiabatic Ansatz method to a standard VQLS approach, for the three same noise levels as the test above. The same local cost function was used, and Powell's method was used as the classical optimizer [17], due to its noise resistance [13]. Both the AAVQLS and the VQLS used the same ansatz circuit given in Fig. 6. This ansatz can be initialized to identity with some nonzero random parameters, hence its use in the AAVQLS here. The circuit U for creating state $|b\rangle$ is appending to the end of this ansatz in accordance with the method discussed in the AAVQLS description, to create the full adiabatic ansatz.

The example five qubit problem tested was given by,

$$\mathbf{A}_4 = Z_1 + 0.15 \cdot Z_3 + 0.5 \cdot Z_4.$$

The AAVQLS approach was split into two trials. One using 10 steps and the other using 20 steps are denoted in the results with the suffix '1' and '2,' respectively. These were compared to a standard VQLS implementation. The same number of overall function evaluations were allowed for all approaches. Twenty runs for each approach and noise-level were conducted.

3.3 Evolutionary ansatz variational quantum linear solver

The EAVQLS method was compared to the standard VQLS method. Again the local cost function was used for both the EAVQLS and the VQLS, and the 4 Qubit ansatz in Fig. 5 was used for the VQLS comparison. Three attempts of the Standard VQLS approach were performed, each with a different classical optimizer: COBYLA, BFGS and SPSA. The EAVQLS algorithm in two different hyper-parameters, with $\alpha = 0.0001, \beta = 0.001$, and $\alpha = 0.001, \beta = 0.01$. The α and β parameters punish ansätze for having too many genes (circuit layers), and having too many 2-qubit gates. These EAVQLS settings will be referred to as EAVQLS Lower and EAVQLS Higher, respectively. The EAVQLS used the COBYLA [18] optimizer, as the simulation was

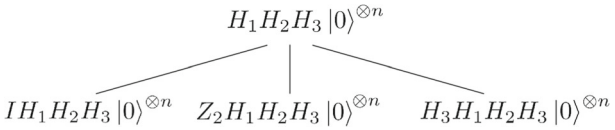

Fig. 7 CQS algorithm ansatz tree expansion used to solve the CQS test problem on the real quantum machine

done on a noise-free statevector simulator, and COBYLA has a very quick convergence rate on a noise-free simulation [13]. The test problem consisted of four qubits with A given by,

$$\mathbf{A}_5 = Z_1 + 0.15 \cdot X_2 Z_3 + 0.5 \cdot H_4.$$

Twenty runs of each EAVQLS genetic algorithm were performed, with a population of size 20 and 20 generations. The genetic hyper-parameters, topological search, parameter search and removal, were set to the values of 0.7, 0.2 and 0.4, respectively. Each VQLS instance was run 100 times, and the top 20 best runs were selected in the comparison. This is done to give a fairer comparison with the resource intense EAVQLS method. Once the statevector simulation of both the EAVQLS and VQLS runs was completed, 50 noisy shots, with a noise-model sampled from the IBM Belem device, were run on each final optimized ansatz solution, in order to compare the noise resistance of the solution EAVQLS ansatz, to the VQLS standard ansatz used. This comparison of noise resistance also compares the EAVQLS Higher and Lower settings to each other, with the expected outcome that the EAVQLS Lower will outperform the EAVQLS Higher in the statevector simulation, as it has greater freedom to employ 2-qubit gates and more genes (greater circuit depth); however, the EAVQLS Higher may outperform the EAVQLS Lower in the noisy shots, because the EAVQLS Higher circuits have fewer 2-qubit gates, and fewer number of genes (circuit depth).

3.4 Classical combination of quantum states

The CQS method is the only non-variational approach tested. The aim of this test was to see how accurately the real quantum machine could approximate the solution given the nodes. This test follows the standard implementation of the CQS algorithm; however, no ansatz tree expansion was conducted on the real device. Instead the ansatz tree was pre-expanded with the nodes as seen in Fig. 14, and then, the solution was approximated on the real device.

The three qubit test example used is given as follows,

$$\mathbf{A}_6 = \mathbb{1} + 0.25 \cdot Z_2 + 0.175 \cdot H_3.$$

This example was selected as a non-trivial problem that suited the topology of the real backend selected (the IBMQX2 quantum device). The number of shots per Hadamard test was set to 245760 (being 30 repetitions of the max 8192 shots per run).

3.5 Logical ansatz variational quantum linear solver

The logical ansatz used the same cost function as the CQS method as detailed above. Both COBYLA and Powell's method were tested as classical optimizers, and both a noise-free Statevector simulation, and a shot-noise Qasm simulation were tested. The logical ansatz was tested on the five qubit problem given by,

$$\mathbf{A}_7 = H_1 + 0.25 \cdot Z_3 + 0.5 \cdot H_4 + 0.5 \cdot Z_5.$$

The individual ansätze making up the logical ansatz were generated randomly. Each logical ansatz was made up of five shallow physical ansatze. Each ansatz consisting of 3 layers of gates was taken from the gate set $\{\mathbb{I}_2, R_y, \wedge_1 X\}$. Two different approaches to training were tested, each denoted in the results by the suffix '1' or '2', for the first and second approach, respectively. These two approaches are detailed as Method 1 and Method 2 in the Logical Ansatz section (Sect. 2.5). Twenty runs for each approach were performed in order to obtain the results.

4 Results and discussion

Please note that the cost values achieved by the VQLS, AAVQLS and EAVQLS are not directly comparable to the cost values achieved by the CQS and LAVQLS due to a differing cost function.

4.1 Variational quantum linear solver

The box plot Fig. 8 and line graph Fig. 9 show the range of the termination values and the average rate of convergence, respectively, for each problem instance, noise-level and classical optimizer used.

Figure 8 gives an idea of the difficulty of each problem instance and also, gives an idea as to the overall affect of noise on the optimization process. The 4 qubit problem instance appears to have been the most difficult, next being the 5 qubit instance, with the 3 qubit instance being the simplest to solve.

The VQLS algorithm performs very well in a noise-free state vector simulation, with the gradient-based BFGS undoubtedly performing the best outright. The inclusion of shot-noise alone does not appear to greatly disturb the optimization process much; however, BFGS is much more affected by the shot-noise than the gradient-free SPSA. SPSA very much outperforms BFGS in the presence of noise. The realistic noise levels appear to greatly affect the optimization process, and while again, SPSA is less affected than BFGS, both are heavily set back. These trends seen between the different noise levels and classical optimizers appear to hold regardless of the problem instances apparent difficulty.

Figure 9 shows the average convergence rate of the top 50 attempts, for each noise-level, classical optimizer and problem instance. In all, SPSA converges faster than

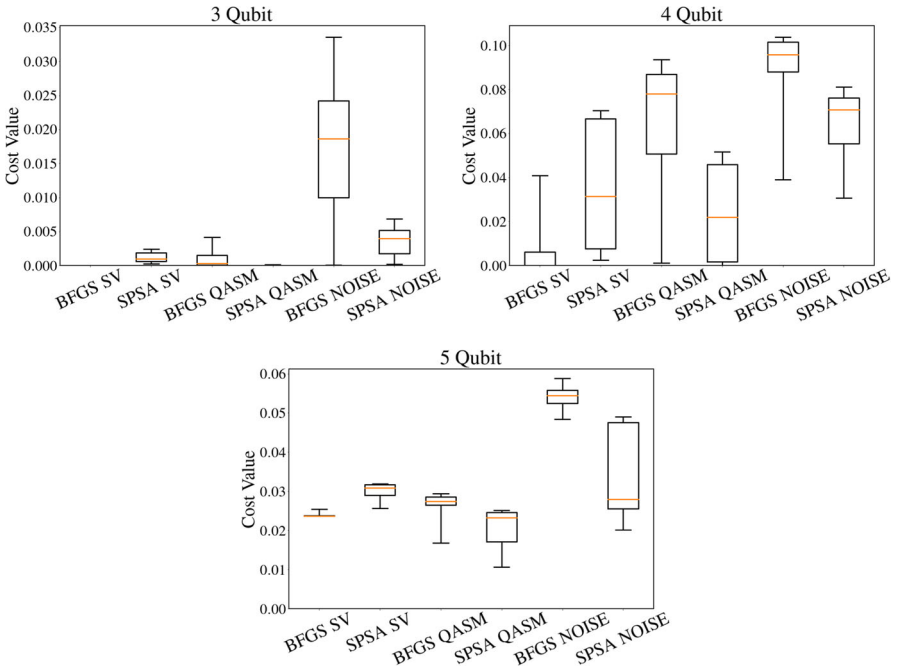

Fig. 8 Termination Value Box plots: These box plots capture the final value at the termination of top 50 attempts of the VQLS algorithm for both the SPSA and BFGS optimizers, with three levels of noise in simulation, for 3 problems. The standard VQLS performance is greatly affected by the presence of noise in the quantum simulation

BFGS regardless of noise-level, and the more difficult the problem, the slower the rate of convergence.

4.2 Adiabatic ansatz variational quantum linear solver

The box plot Fig. 10 shows the range of the final termination values achieved by the respective methods for the respective levels of noise. The results captured in Fig. 10 appear to indicate there is not much of a significant difference between the VQLS and both AAVQLS approaches. However, the state vector simulation clearly favors the standard VQLS approach, while the both AAVQLS approaches slightly outperform the VQLS in the realistic noise situation. Given that the state vector simulation is merely theoretical, there may be some merit to the AAVQLS approach. AAVQLS 2 also ever so slightly outperforms AAVQLS 1 in the noisy simulations, meaning shorter, more frequent steps may be the better approach between the two. The AAVQLS approach was split into two trials. One using 10 steps and the other using 20 steps, denoted in the results with the suffix '1' and '2,' respectively.

The line graph Fig. 11 shows the value of the cost function along the adiabatic optimization path for the best performing run of both AAVQLS 1 and 2 for each noise level. Figure 11 shows that all methods kept fairly close to the ground state of the

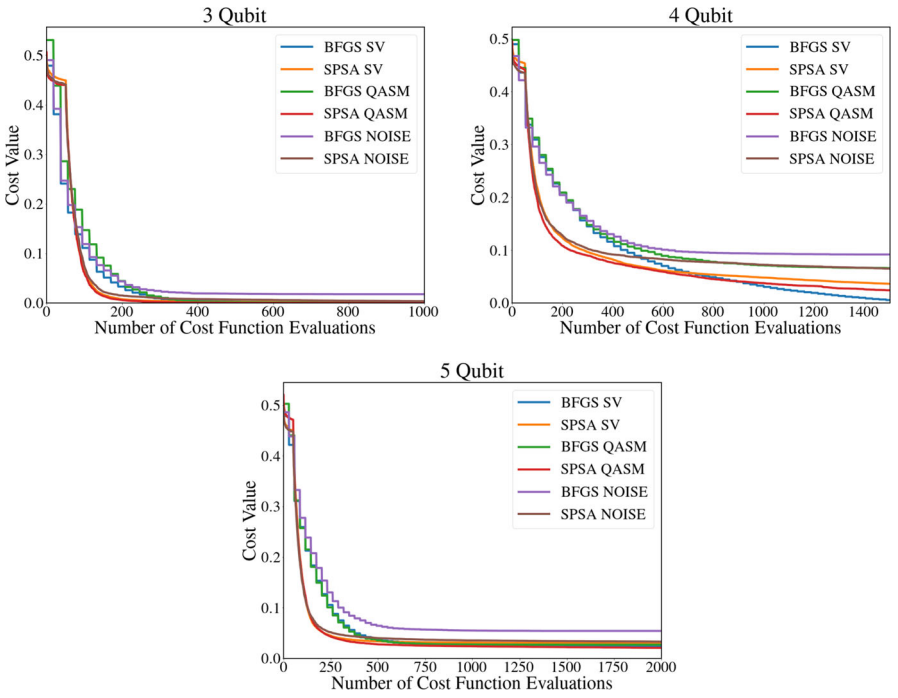

Fig. 9 Average optimization convergence: these line graphs capture the convergence of the top 50 attempts of the VQLS algorithm for both the SPSA and BFGS optimizers, with three levels of noise in simulation, for 3 problems. (BFGS convergence appears stepped as gradient function calls require multiple cost function evaluations). Both optimizers converge relatively quickly, close their final values, irrespective of the noise present

Hamiltonian during the initial phase of optimization, only to move further away from the ground state during the middle of the optimization process. The Statevector and Qasm simulations of both methods managed to move close to the ground state near the end of the optimization process; however, the noisy simulations did not. Ideally, all methods should have kept fairly close to the ground state throughout the optimization process.

4.3 Evolutionary ansatz variational quantum linear solver

Figure 12 shows the final cost values achieved by the EAVQLS algorithm (EAVQLS Higher and EAVQLS Lower) in 20 runs, and the cost value achieved by the top 20 out of 100 statevector simulation of the standard VQLS approach using SPSA, BFGS and COBYLA minimizers. Figure 13 shows the cost value achieved by the noise shots on the statevector optimized ansatz parameters of Fig. 12. Noise model was sampled from the IBM Belem device.

The EAVQLS Higher (set with $\alpha = 0.0001, \beta = 0.001$) outperforms the EAVQLS Lower (with $\alpha = 0.001, \beta = 0.01$) on the statevector simulation; however, the EAVQLS Lower outperforms the EAVQLS Higher on the noise shots. This is expected

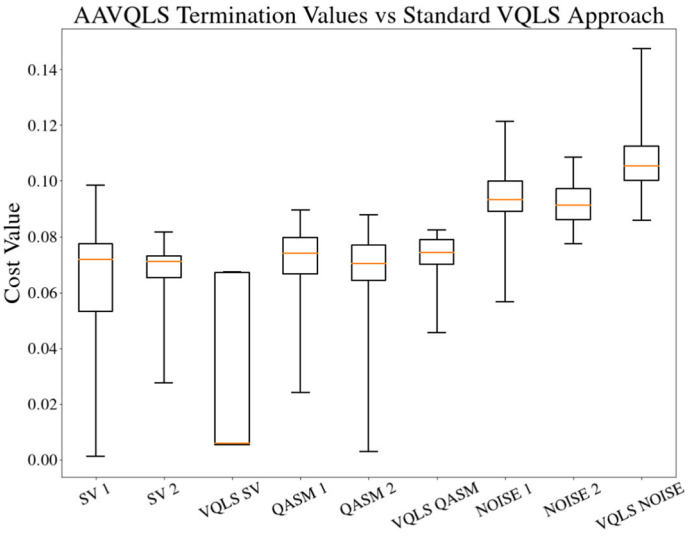

Fig. 10 AAVQLS termination values: AAVQLS termination values compared to a standard VQLS approach, for 2 different AAVQLS methods for 3 noise levels. These results appear to indicate there is not much of a significant difference between the VQLS and both AAVQLS approaches

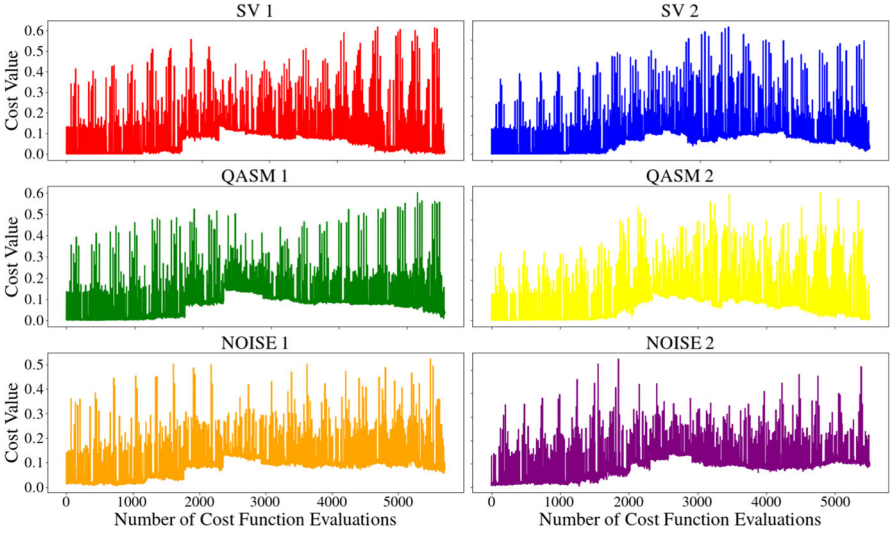

Fig. 11 Optimization convergence: these line graphs capture the cost value measured by the AAVQLS algorithm during the optimization process, for 2 different methods, '1' using 10 steps and '2' using 20 steps, and for 3 noise levels, a statevector simulation, a simulation including only shot-noise, and a currently realistic noise-level simulation. This corresponds to how close the ansatz kept to the ground state of the Hamiltonian during the optimization

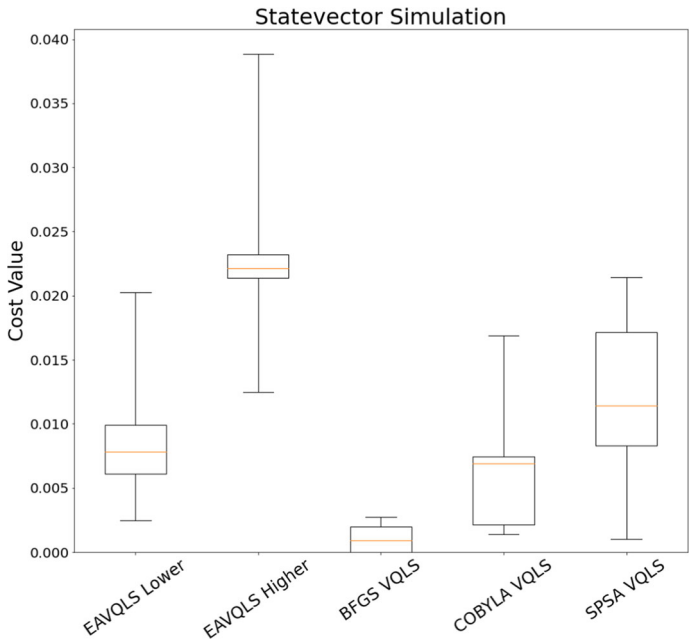

Fig. 12 Cost value achieved by the statevector simulation of the EAVQLS algorithm (EAVQLS Higher and EAVQLS Lower) in 20 runs, compared to the cost value achieved by the top 20 out of 100 statevector simulation of the standard VQLS approach using SPSA, BFGS and COBYLA minimizers

because the ansätze created by the EAVQLS Lower parameter settings are biased towards having fewer circuit layers and fewer 2-qubit gates, making them less susceptible to noise.

The EAVQLS Higher performance on the statevector simulation is worse than both the BFGS and COBYLA optimizers, indicating that for a statevector simulation the EAVQLS approach shows no advantage. On the noise shots, however, the EAVQLS Lower performs the best overall, as expected. This performance comes down to it having the simplest and most noise resistant ansätz circuits. The VQLS use the ansatz is shown in Fig. 5.

4.3.1 Classical combination of quantum states

Relatively good results achieved on the real machine Fig. 15, given the noise levels of current NISQ devices, however, the problem was specifically tailored to suite the connectivity of the selected backend. The optimal cost value achievable using the nodes in the ansatz tree given is equal to approximately 0.00324.

Figure 16 shows the spread of the final cost values of the logical ansatz approach. It is clear that the noise introduced by the Qasm simulator greatly affects the performance of the Logical Ansatz VQLS. The effect of noise may be increased when using a logical ansatz as multiple different Hadamard test runs are needed, one for each pairing of the different physical ansatze, each adding some potential for error.

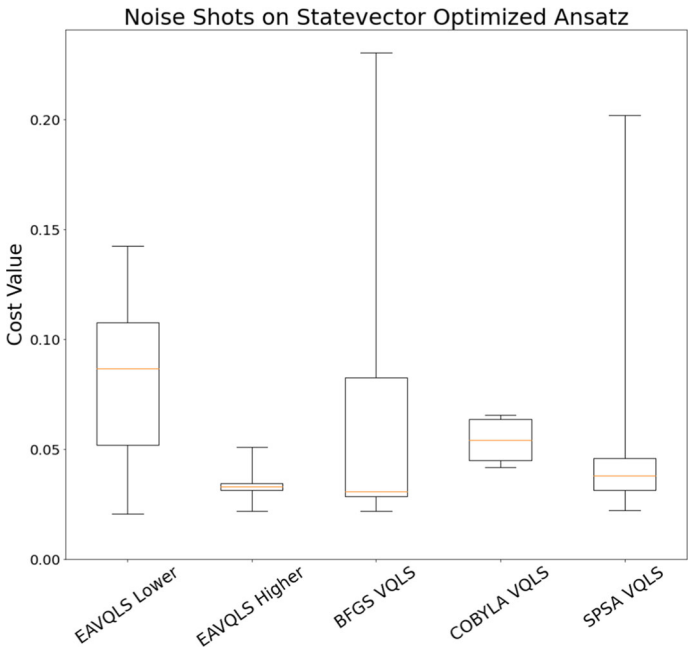

Fig. 13 Cost value achieved by the noise shots on the statevector optimized ansatz parameters of the above Fig. 12. Noise model was sampled from the IBM Belem device

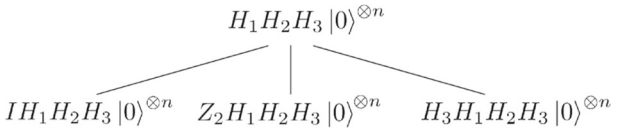

Fig. 14 CQS algorithm ansatz tree expansion used to solve the specified linear systems problem on the real machine. H_i denotes a Hadamard gate on qubit i, I denotes an identity gate, and Z_i denotes a Z gate on qubit i

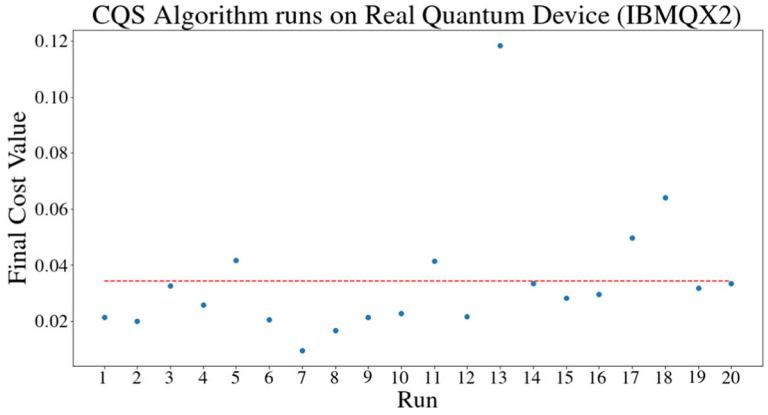

Fig. 15 20 runs of the CQS Algorithm on IBMQX2 machine, with the average cost indicated in red. The CQS algorithm runs with moderate success on a real quantum device

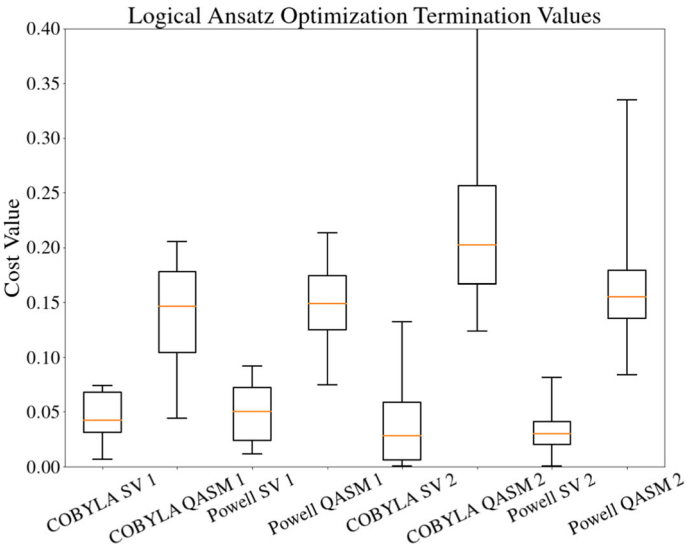

Fig. 16 Termination Values achieved by the logical ansatz, for 2 methods, 2 noise levels and 2 different optimizers. Both training methods 1 and 2 performed similarly well

5 Conclusions

The success of the standard VQLS approach appears to be greatly dependent on the noise levels on the quantum device. The cost values achieved remain fairly consistent after the addition of shot-noise and decline substantially when realistic levels of noise are added to the simulated quantum device.

The AAVQLS approach presents an ansatz optimization strategy that in theory could keep the ansatz near the ground state of the system's Hamiltonian, allowing a low cost value to be easily found. Considering the best runs recorded in Fig. 11, this trend is observed during the first part of the optimization where at all levels of noise, the system remained close the ground state of the Hamiltonian. This trend faded just before midway through the optimization process, where all simulations, regardless of the level of noise, moved away from the ground state. This presents a particular flaw in this approach, whereby the system can leave the ground state. One possible explanation for this is that the optimization process had a step size that was too large (the evolution of the Hamiltonian was too fast), or the ground state was not contained in the Hilbert space spanned by the particular ansatz used. The latter issue may be avoided by evolving the initial Hamiltonian to the final Hamiltonian along a different path. In the later half of the optimization process, the Statevector and QASM simulations recovered the ground state while the realistic noise simulation did not. That the AAVQLS performs similarly to the VQLS in the QASM and Noisy simulations, as seen in Fig. 10, suggests that there may be some merit to this approach, especially because the best cost values achieved by the AAVQLS for those two simulations were quite substantially lower, even if, on average, they performed similarly.

The EAVQLS performed worse than the VQLS for the specific problem instance simulated using statevectors in these results in Fig. 12. The EAVQLS (EAVQLS Lower) outperforms all the VQLS approaches in the noise shots in Fig. 13, however, as the ansätze are biased towards being more noise resistant as they are constructed in the evolutionary algorithm. For a more realistic test of the EAVQLS algorithm, the full parameter optimization needs to be done under noisy conditions. It has only been indicated here in these results that the EAVQLS ansätze show more noise resistance, when the algorithm parameters are set appropriately.

The CQS approach was the only approach tested on the real quantum device, in order to gauge its effectiveness on near-term quantum hardware. With 20 runs on the IBMQX2 device, the CQS approach managed to achieve some fairly low cost values and a decent average cost value. This is positive for this approach; however, it is noted that the specific problem that was solved may have been quite simple, yet still non-trivial.

The LAVQLS, being an adaptation of the CQS method, appears to work well in a noise-free simulation; however, the shot-noise alone heavily reduced the final cost value achieved by the method. This may be because the many Hadamard tests required to evaluate the cost function amplify the noise. This is not good because a proposed feature of the LAVQLS was noise resistance due to the use of shorter individual ansäze making up the logical ansatz.

In this paper, a few approaches to solving systems of linear equations on near-term quantum hardware have been presented. Each approach that differs from the standard VQLS approach tries to offer some advantage over the standard approach; however, whether the proposed advantages of these algorithms that actually apply in implementation are yet to be conclusively seen. Some potential problems with these approaches have been highlighted, and it is left to a future work to investigate the realistic advantages of these approaches. It may be the case that some of these approaches offer significant advantages over the standard VQLS approach; however, this is still unclear. It is left to a future work to develop better training procedures for the AAVQLS algorithm, to avoid it losing the ground state, and to fully investigate the noise-resistance properties of the EAVQLS algorithm.

Acknowledgements This work is based upon research supported by the South African Research Chair Initiative, Grant No. 64812 of the Department of Science and Innovation and the National Research Foundation of the Republic of South Africa Support from the NICIS (National Integrated Cyber Infrastructure System) e-research grant QICSI7 is kindly acknowledged.

Funding Open access funding provided by University of KwaZulu-Natal.

Data Availability Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study. The source code implementing of the algorithms analyzed here can be found in the Github repository (github.com/aidanpellow/vqls).

Declarations

Conflict of interest Aidan Pellow-Jarman is Quantum Scientist at QUNOVA computing. Francesco Petruccione is the Chair of the Scientific Board and Co-Founder of QUNOVA computing. The authors declare no other competing interests.

References

1. Harrow, A.W., Hassidim, A., Lloyd, S.: Quantum algorithm for linear systems of equations. Phys. Rev. Lett. **15**(103), 150502 (2009)
2. Dervovic, D., Herbster, M., Mountney, P., Severini, S., Usher, N., Wossnig, L.: Quantum linear systems algorithms: a primer (2018) arXiv:1802.08227
3. Preskill, J.: Quantum Computing in the NISQ era and beyond. Quantum **2**, 79 (2018)
4. McClean, J.R., Boixo, S., Smelyanskiy, V.N., Babbush, R., Neven, H.: Barren plateaus in quantum neural network training landscapes. Nat. Commun. **9**, 4812 (2018)
5. Peruzzo, A., McClean, J., Shadbolt, P., Yung, M.-H., Zhou, X.-Q., Love, P.J., Aspuru-Guzik, A., O'Brien, J.L.: A variational eigenvalue solver on a quantum processor. Nat. Commun. **5**, 4213 (2014)
6. Bravo-Prieto, C., LaRose, R., Cerezo, M., Subasi, Y., Cincio, L., Coles, P.J.: Variational quantum linear solver: a hybrid algorithm for linear systems (2019). https://doi.org/10.48550/arXiv.1909.05820
7. Huang, H.-Y., Bharti, K., Rebentrost, P.: Near-term quantum algorithms for linear systems of equations. New J. Phys. **23**, 113021 (2021)
8. Xu, X., Sun, J., Endo, S., Li, Y., Benjamin, S.C., Yuan, X.: Variational algorithms for linear algebra. Sci. Bull. **66**(21), 2181–2188 (2021)
9. O'Malley, D., Subasi, Y., Golden, J., Lowrie, R., Eidenbenz, S.: A near-term quantum algorithm for solving linear systems of equations based on the Woodbury identity (2022) arXiv:2205.00645
10. Huggins, W.J., Lee, J., Baek, U., O'Gorman, B., Whaley, K.B.: A non-orthogonal variational quantum eigensolver. New J. Phys. **22**, 073009 (2020)
11. Rattew, A.G., Hu, S., Pistoia, M., Chen, R., Wood, S.: A Domain-agnostic, noise-resistant, hardware-efficient evolutionary variational quantum eigensolver (2020). arXiv:1910.09694
12. Mitarai, K., Negoro, M., Kitagawa, M., Fujii, K.: Quantum circuit learning. Phys. Rev. A **98**, 032309 (2019)
13. Pellow-Jarman, A., Sinayskiy, I., Pillay, A., Petruccione, F.: A comparison of various classical optimizers for a variational quantum linear solver. Quantum Inf. Process. **20**, 202 (2021)
14. Albash, T., Lidar, D.A.: Adiabatic quantum computing. Rev. Mod. Phys. **90**, 015002 (2018)
15. Fletcher, R.: A new approach to variable metric algorithms. Comput. J. **13**(3), 317–322 (1970)
16. Spall, J.C.: Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. IEEE Trans. Autom. Control **37**(3), 332–341 (1992)
17. Powell, M.J.D.: An efficient method for finding the minimum of a function of several variables without calculating derivatives. Comput. J. **7**(2), 155–162 (1964)
18. Powell, M.J.D.: A direct search optimization method that models the objective and constraint functions by linear interpolation. In: Gomez, S., Hennart, J.P. (eds.) Advances in Optimization and Numerical Analysis. Mathematics and Its Applications, vol 275. Springer, Dordrecht (1994). https://doi.org/10.1007/978-94-015-8330-5_4