Comprehensive Library of Variational LSE Solvers

Nico Meyer, Martin Röhn, Jakob Murauer, Axel Plinge, Christopher Mutschler, and Daniel D. Scherer
Fraunhofer IIS, Fraunhofer Institute for Integrated Circuits IIS, Nürnberg, Germany

***Abstract*—Linear systems of equations can be found in various mathematical domains, as well as in the field of machine learning. By employing noisy intermediate-scale quantum devices, variational solvers promise to accelerate finding solutions for large systems. Although there is a wealth of theoretical research on these algorithms, only fragmentary implementations exist. To fill this gap, we have developed the `variational-lse-solver` framework, which realizes existing approaches in literature, and introduces several enhancements. The user-friendly interface is designed for researchers that work at the abstraction level of identifying and developing end-to-end applications.**

***Index Terms*—quantum computing, variational quantum algorithms, quantum machine learning, quantum software library, quantum linear algebra, linear system of equations**

I. Introduction.

Quantum algorithms for matrix arithmetics, in particular solving linear system of equations (LSE), is a strongly studied subject in quantum computing. Most well-known might be the HHL algorithm [1], which promises a poly-logarithmic scaling in the system size, compared to the classical polynomial scaling [2]. However, this improvement is subject to a high sparsity and low condition number of the underlying system. Block encodings can be viewed as an generalization of this concept, with similar promises and requirements [3]. The application of these techniques has been studied in various fields, including training classical neural networks [4] and quantum reinforcement learning [5], [6]. However, executing these algorithms requires fault-tolerant and large-scale quantum devices.

With the advent of variational quantum algorithms (VQAs) [7], several variational LSE solvers have been proposed. The underlying idea is to train a variational quantum circuit (VQC) to prepare a state that is proportional to the solution of the system. For well-designed and shallow quantum circuit this can potentially be done on noisy intermediate-scale quantum (NISQ) devices. Our proposed `variational-lse-solver` library implements several bug fixes and improvements to the approach introduced by Bravo-Prieto et al. [8]. For systems up to a size of 50 qubits, they demonstrated a scaling behavior roughly resembling that of HHL. Other studies [9], [10] have expanded and reformulated this concept. The `variational-lse-solver` incorporates these approaches or has implementations that are almost identical. The application of the algorithms has been studied in several research efforts [11], [12], yet no implementation was made available.

The research was supported by the Bavarian Ministry of Economic Affairs, Regional Development and Energy with funds from the Hightech Agenda Bayern via the project BayQS.
Correspondence to: nico.meyer@iis.fraunhofer.de

II. Underlying Concept and Modifications

The objective is solving a $N \times N$ linear system, comprising N equations and N variables. This can be formulated compactly as finding $\boldsymbol{x}$, such that

$$A\boldsymbol{x} = \boldsymbol{b}, \tag{1}$$

where $A \in \mathbb{C}^{N\times N}$, and $\boldsymbol{x}, \boldsymbol{b} \in \mathbb{C}^N$. In practice, there often are additional demands on sparsity and condition number of the system matrix A. Bravo-Prieto et al. [8] propose to variationally prepare a state, that is proportional to the solution $\boldsymbol{x}$ of the LSE. This is formulated in a trainable manner, with $|V(\boldsymbol{\alpha})\rangle$ describing the VQC-based approximate solution for trainable parameters $\boldsymbol{\alpha}$. For the training routine, we need to implement a state $|b\rangle$ that is proportional to $\boldsymbol{b}$. Furthermore, a linear combination of unitaries representing A is required, i.e.

$$A = \sum_{k=0}^{L-1} c_k A_k, \tag{2}$$

with all A_k unitary and complex coefficients c_k.

A. Training with Modified Cost Function

Training $|V(\boldsymbol{\alpha})\rangle$ to be close to the ground truth $|x\rangle$ can be achieved by minimizing the *global* cost function

$$\hat{C}_G = \left\langle V(\boldsymbol{\alpha}) \left| A^\dagger \left(\mathbb{I} - |b\rangle\langle b|\right) A \right| V(\boldsymbol{\alpha}) \right\rangle. \tag{3}$$

In order to avoid unintentionally minimizing the norm of $|\psi\rangle := A|V(\boldsymbol{\alpha})\rangle$, it is common practice to employ a normalized version of the global cost function. Bravo-Prieto et al. suggest the formulation $C_G = \hat{C}_G / \langle\psi|\psi\rangle$ (see equation 5 in [8]). We instead propose

$$C_G = 1 - \frac{1-\hat{C}_G}{\langle\psi|\psi\rangle} \tag{4}$$

$$= 1 - |\langle b|\Psi\rangle|^2 \quad \text{with } |\Psi\rangle = \frac{|\psi\rangle}{\sqrt{\langle\psi|\psi\rangle}}, \tag{5}$$

which prevents division of the first summand of Equation (5) by $\langle\psi|\psi\rangle$ and therefore guarantees convergence of the cost function at a value of 0. With the same reasoning, we also implement an equivalent re-defined version of the normalized *local* cost function C_L (see equation 6 and 7 in [8]).

B. Reduced Evaluation Overhead

It is possible to evaluate both the norm and the raw (i.e. not normalized) local and global cost functions using either the Hadamard or Hadamard-overlap test [13]. We developed two techniques that reduce the number of involved circuits by a factor of up to four:

TABLE I

FOR THE DIFFERENT METHODS WE DENOTE THE SPATIAL REQUIREMENTS AND THE NECESSARY INDIVIDUAL CIRCUIT EVALUATIONS FOR n DATA QUBITS AND m TERMS. BOTH THE RAW GLOBAL AND LOCAL LOSSES ARE SUBSEQUENTLY COMPOSED WITH THE NORM FOLLOWING EQUATION (4). THE DIRECT METHOD IS ONLY SUITABLE FOR SIMULATION BECAUSE OF ITS RELIANCE ON CLASSICAL NON-UNITARY MATRIX MANIPULATION.

Term	Method	Required Qubits	Evaluations[b]	NISQ Feasibility	Usage	
norm $\langle\psi\|\psi\rangle$	direct	n	1	–	*default*	
	Hadamard test	$n+1$	$1/2\,(m^2-m)$	$\surd$	*all non-default*	
global $\hat{C}_G$	direct	n	1	–		*default*
	Hadamard test[a]	$n+1$	m	$(\surd)$[c]	`local=False`	`method="hadamard"`
	Hadamard-overlap test	$2n+1$	$1/2\,(m^2+m)$	$\surd$		`method="overlap"`
	coherent[a]	$n+\lceil\log_2 m\rceil$	1	X		`method="coherent"`
local $\hat{C}_L$	direct	n	1	–		*default*
	Hadamard test	$n+1$	$n/2\,(m^2+m)$	$\surd$	`local=True`	`method="hadamard"`

[a] Implementation with hard-coded components: pennylane.ai/qml/demos/tutorial_vqls/, pennylane.ai/qml/demos/tutorial_coherent_vqls/
[b] Evaluating imaginary terms doubles the number of evaluations for all methods, except direct state evolution.
[c] Requires implementing multi-control gates, i.e. the NISQ-feasibility depends on the decomposition efficiency for a given instance.

The computation of the norm $\langle\psi|\psi\rangle$ makes use of the unitary decomposition of A from Equation (2). With $\beta_{kl} := \left\langle \mathbf{0}|V^\dagger(\boldsymbol{\alpha})A_l^\dagger A_k V(\boldsymbol{\alpha})|\mathbf{0}\right\rangle$ and starting from equation 14 of [8], we propose the decomposition

$$\langle\psi|\psi\rangle = \sum_k \sum_l c_k c_l^* \beta_{kl} \tag{6}$$

$$= \sum_k |c_k|^2 + \sum_k \sum_{l=k+1} \left(c_k c_l^* \beta_{kl} + c_l c_k^* \beta_{lk}\right) \tag{7}$$

$$= \sum_k |c_k|^2 + 2 \cdot \sum_k \sum_{l=k+1} \mathrm{Re}\left(c_k c_l^* \beta_{kl}\right), \tag{8}$$

where the second step employs $\beta_{kl} = \beta_{lk}^*$. As the terms β_{kl} have to be evaluated individually – using e.g. the Hadamard test for the real part, and a modified version for the imaginary one – realizing Equation (7) instead of Equation (8) saves half the terms. Additionally, we exploit that the imaginary part of β_{kl} can be ignored, if $c_k c_l^* \in \mathbb{R}$, leading to a further halving of the required Hadamard tests. An equivalent decomposition exploiting symmetries and real-valued coefficients is also implemented for the raw cost functions $\hat{C}_G$ and $\hat{C}_L$.

III. FRAMEWORK COMPONENTS AND USAGE

The proposed framework is based on `PennyLane` [13] and implements the routines proposed by Bravo-Prieto et al. [8], with the modifications discussed in Section II-A. The following section covers the details of the implementation, usage and potential extensions. An overview of the provided functionalities and respective constraints is provided in Table I. Additional usage information is also provided in the documentation of `variational-lse-solver`.

A. Modes of Loading System Matrix

One integral part of the routine is encoding the system matrix as unitary operations. While there is considerable research effort on the synthesis of circuits given an unitary, we approach this topic from a more abstracted view. Our framework allows to provide the decomposition of the system matrix in three different modes:

- A `circuit` implementing each unitary A_k. This allows to integrate custom unitary synthesis techniques.
- Explicit `unitary` representations of A_k. This internally employs circuit synthesis tools from `PennyLane`.
- A `pauli` decomposition $A = \sum_k c_k \bigotimes_i P_i$, with $P_i \in \{I, X, Y, Z\}$. This potentially leads to more terms, but simplifies the internally realized implementation of controlled versions of A_k.

Additionally, one can use a direct evaluation of the loss function described in Section III-D by only providing the potentially non-unitary `matrix` A.

B. Usage Example for Recreating Simple Experiment

To demonstrate the straightforward usability, we recreate one of the illustrative experiments from Bravo-Prieto et al. [8]. The task is to variationally solve the linear system defined by

$$A = I_0 I_1 I_2 + 0.2 \cdot X_0 Z_1 I_2 + 0.2 \cdot X_0 I_1 I_2 \tag{9}$$

$$b = H_0 H_1 H_2 \,|000\rangle\,, \tag{10}$$

where we omit the explicit tensor product symbol $\otimes$ for readability. Using the proposed `variational-lse-solve` framework it is possible to set up and replicate this experiment in just a few lines of code:

```
from variational_lse_solver import VarLSESolver

lse = VarLSESolver(["III", "XZI", "XII"],
                   np.ones(8)/np.sqrt(8),
                   coeffs=[1.0, 0.2, 0.2],
                   method="hadamard",
                   local=False,
                   lr=0.01,
                   steps=50)
solution, _ = lse.solve()
```

Instead of providing the explicit vector representation of b it also would be possible to submit a quantum circuit applying Hadamard gates to the individual qubits. The experimental results depicted in Figure 1 demonstrate the effectiveness of the framework. More detailed usage information is provided in the `README` file.

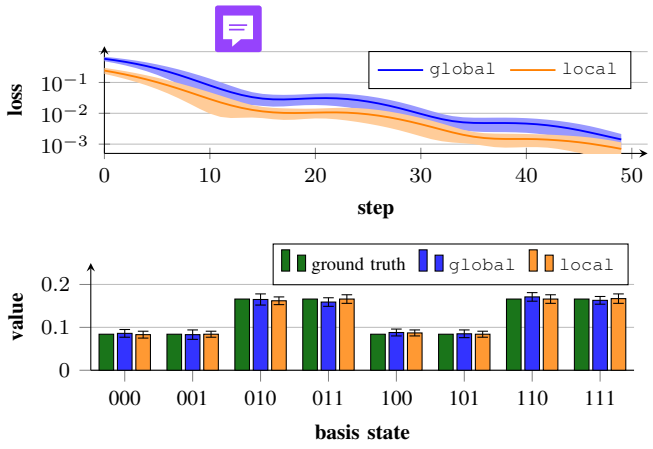

Fig. 1. Results produced with the proposed framework on the LSE described in Equations (9) and (10). The variational ansatz consists of a depth $d = 1$ version of Figure 2. The training for 50 steps in the upper plot averaged over 100 random initializations shows smooth convergence for both loss functions. The evaluation of the final results with 1000 shots in the lower plot is in good agreement with the normalized ground truth solution. The error bars denote the 25th and 75th percentile over the 100 trained parameter sets.

C. Dynamic Circuit Ansatz

In the standard setup the circuit ansatz $|V(\boldsymbol{\alpha})\rangle$ is pre-defined and only the parameters $\boldsymbol{\alpha}$ are optimized. To allow for a more flexible design, it is possible to use dynamic circuit creation [10]. The proposed framework realizes this strategy using the dynamic ansatz in Figure 2, with the adjustable parameter of depth d potentially increasing during training. Once the value of the loss function stagnates an additional layer is appended, which parameter initialization realizing a warm start.

This technique is always applied by default, with the possibility to define early stopping criteria, initialization instructions, maximal depth, and more. Additionally, it is also possible to provide custom circuit layouts.

D. Additional Evaluation Methods

We propose and implement additional techniques for the actual evaluation of the loss functions, an overview is provided in Table I. First of all, we explicitly realize the Hadamard-overlap test for evaluation of the local cost function, which was briefly mentioned in [8]. Second, we implement a version of the global cost function that exploits setups with *coherent* access to the unitary decomposition of A [13], [14]. Last but not least, we provide a fully differentiable *direct* computation of the losses C_G and C_L following Equation (4). This method is not feasible on actual quantum hardware, but allows for fast validation and prototyping in simulation.

IV. Concluding Remarks

We introduce `variational-lse-solver`, a comprehensive library for research on variational linear system of equations (LSE) solvers. The framework allows researchers to abstract away technical details and focus on developing and refining end-to-end applications. We believe this tool will greatly benefit the community and contribute to the progress of quantum software development.

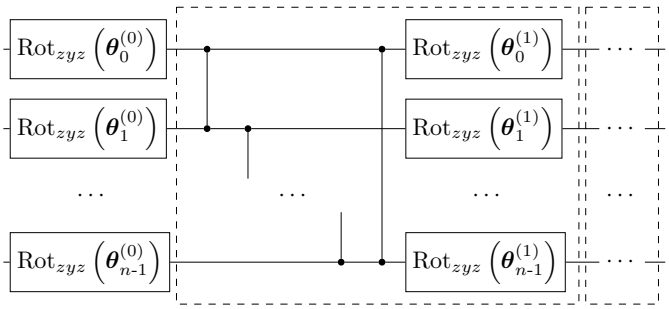

Fig. 2. The dynamic ansatz used in the proposed framework, a modified version of [10]. The parameterized gates are realized as $\mathrm{Rot}_{zyz}(\boldsymbol{\alpha}) = R_z(\alpha_2)R_y(\alpha_1)R_z(\alpha_0)$, entanglement is created with nearest-neighbor CZ-gates. The initial parameter set $\boldsymbol{\theta}^{(0)}$ is drawn u.a.r. from $[0, 2\pi]$. For increasing the depth, the new parameters are selected as $\boldsymbol{\theta}_i^{(d+1)} = (-\alpha, 0, \alpha)$, for all qubits i, and with α u.a.r. as above. This ensures that the best solution to the LSE found until that point is not worsened by the modification itself. For depth d, the number of trainable parameters therefore is $3n(d+1)$.

Code Availability

The described library can be installed via `pip install variational-lse-solver`. The code with details on requirements and usage is also available at https://github.com/nicomeyer96/variational-lse-solver. Further information and data is available upon reasonable request.

References

[1] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations," *Phys. Rev. Lett.*, vol. 103, no. 15, p. 150502, 2009.

[2] V. Strassen, "Gaussian elimination is not optimal," *Numerische Mathematik*, vol. 13, no. 4, pp. 354–356, 1969.

[3] G. H. Low and I. L. Chuang, "Hamiltonian simulation by qubitization," *Quantum*, vol. 3, p. 163, 2019.

[4] J. Liu, M. Liu, J.-P. Liu, Z. Ye, Y. Wang, Y. Alexeev, J. Eisert, and L. Jiang, "Towards provably efficient quantum algorithms for large-scale machine-learning models," *Nat. Commun.*, vol. 15, no. 1, p. 434, 2024.

[5] E. A. Cherrat, I. Kerenidis, and A. Prakash, "Quantum reinforcement learning via policy iteration," *Quantum Mach. Intell.*, vol. 5, no. 2, p. 30, 2023.

[6] N. Meyer, C. Ufrecht, M. Periyasamy, D. D. Scherer, A. Plinge, and C. Mutschler, "A Survey on Quantum Reinforcement Learning," *arXiv:2211.03464*, 2022.

[7] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio *et al.*, "Variational quantum algorithms," *Nat. Rev. Phys.*, vol. 3, no. 9, pp. 625–644, 2021.

[8] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, and P. J. Coles, "Variational quantum linear solver," *Quantum*, vol. 7, p. 1188, 2023.

[9] X. Xu, J. Sun, S. Endo, Y. Li, S. C. Benjamin, and X. Yuan, "Variational algorithms for linear algebra," *Science Bulletin*, vol. 66, no. 21, p. 2181–2188, 2021.

[10] H. Patil, Y. Wang, and P. S. Krstić, "Variational quantum linear solver with a dynamic ansatz," *Phys. Rev. A*, vol. 105, no. 1, p. 012423, 2022.

[11] C. J. Trahan, M. Loveland, N. Davis, and E. Ellison, "A variational quantum linear solver application to discrete finite-element methods," *Entropy*, vol. 25, no. 4, p. 580, 2023.

[12] T. Yoshikura, S. L. Ten-No, and T. Tsuchimochi, "Quantum inverse algorithm via adaptive variational quantum linear solver: applications to general eigenstates," *J. Phys. Chem. A*, vol. 127, no. 31, pp. 6577–6592, 2023.

[13] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, S. Ahmed, V. Ajith, M. S. Alam, G. Alonso-Linaje, B. AkashNarayanan, A. Asadi *et al.*, "Pennylane: Automatic differentiation of hybrid quantum-classical computations," *arXiv:1811.04968*, 2022.

[14] R. Kothari, "Efficient algorithms in quantum query complexity," Ph.D. dissertation, University of Waterloo, 2014.