RESEARCH

Quantum–classical co-simulation for smart grids: a proof-of-concept study on feasibility and obstacles

Dominik Vereno[1*], Amin Khodaei[2], Christian Neureiter[1] and Sebastian Lehnhoff[3]

*Correspondence:
dominik.vereno@fh-salzburg.ac.at
[1] Josef Ressel Centre for Dependable System-of-Systems Engineering, Urstein Süd 1, 5412, Puch/Salzburg, Austria
Full list of author information is available at the end of the article

Abstract

With the rising complexity of our electricity infrastructure, smart grid simulations increasingly rely on co-simulation, which involves jointly executing independent subsystem simulations. However, in large-scale simulation scenarios, such as those involving costly power-flow analysis, co-simulation may experience computational-performance issues. Quantum computing offers a potential solution through quantum–classical co-simulation, in which one or more simulators of an otherwise classical co-simulation are executed on quantum hardware. However, there is no practical realization of this concept that establishes its feasibility. To address this gap, we integrate a quantum power flow simulator with a smart grid co-simulation and conduct an exploratory simulation study using a fictitious case-study scenario. The experiments demonstrate the feasibility of quantum–classical co-simulation; at the same time, they highlight four obstacles to the concept's realization in practice: 1) To use quantum computing for co-simulation, session-based scheduling is required. 2) Distributed simulation limits possible applications and requires proximity of computing resources. 3) For the efficient extraction of classical information from the quantum states, we need carefully designed operators. 4) Current hardware limitations—such as noise susceptibility and the lack of quantum random access memory—limit practical near-term uses of quantum power flow; therefore, attention should be turned to alternative applications that are more promising in the near term. These findings pave the way for future research on quantum–classical co-simulation and its potential applications in smart grids.

Keywords: quantum power flow; power-systems simulation; quantum computing; distributed simulation; HHL algorithm

Introduction

Modern electricity infrastructure faces various challenges, including the electrification of transport, and climate change. In response to these challenges, smart grids are being developed, which rely on Information and Communications Technology (ICT) for pervasive monitoring and automated control. These smart grids facilitate more efficient utilization of energy, reduce costs, and enable better integration of renewable energy sources, resulting in enhanced resilience and sustainability in electricity production and distribution [2].

Co-simulation has emerged as a promising paradigm for overcoming these challenges, especially the heterogeneity and trans-domain nature of subsystems, as well as their operational independence [3]. With co-simulation, independent subsystem simulators are coordinated to simulate the coupled system [4]. However, large-scale co-simulation with complex simulators can be computationally expensive. Some

power-system problems require repeated execution (e.g. for analyzing different contingency scenarios [5]), which further exacerbates the computational cost. While co-simulation can be parallelized and distributed [6], some subproblems may still be prohibitively expensive. For example, large-scale power-flow computations pose significant computational challenges [7].

Quantum computing may provide a solution. It is a novel computing paradigm that harnesses quantum-mechanical effects for information processing that promises drastic speed-up for many fundamental computational problems [8]. In recent years, research has identified the potential value of quantum computing for power systems [9]. For example, quantum mixed binary optimization was applied to unit commitment [10], and the viability for quantum annealing–based phasor-measurement unit placement was analyzed [11]. Furthermore, a quantum algorithm for solving linear systems of equations—the HHL algorithm [12]—promises exponential speed-up of power-flow analysis, both DC [13] and AC [14]. For more examples on the state of the art of quantum power-systems engineering, see [1] and [15].

Vereno *et al.* have proposed utilizing the potential of quantum computing in smart-grid co-simulation by introducing *quantum–classical co-simulation*, where "one or more simulators of an otherwise classical co-simulation are executed on quantum hardware" [16, p. 2]. They specifically propose the use case of applying quantum power flow in a smart grid co-simulation. Quantum–classical co-simulation can be seen as a form of hybrid quantum-classical approaches, as discussed in [17]. Ref. [16] only discusses the concept in theory, however, the authors highlight the need for a proof-of-concept study to determine its feasibility and to assess its usefulness. In our research, we address this need by conducting an exploratory case study–based proof-of-concept study where we integrate quantum power flow in a smart grid co-simulation. We opted for DC power flow in our experiments due to its simplicity and suitability for co-simulation scenarios involving quantum computing. DC power flow is attractive for applications requiring speed, which could benefit especially from quantum speedup. The limitations of quantum AC power flow, such as the need for Quantum Random Access Memory (QRAM) for efficient retrieval of intermediate results [15], further support our decision to use DC power flow instead.

The paper's two main goals and research contributions are:

1. We provide a proof of concept for quantum–classical co-simulation and its application to smart grids, using a quantum power flow simulator.
2. We identify and assess potential obstacles to the practical implementation of this concept, and their impact on its utility; we further provide recommendations on how to address them.

The next chapter offers background on the involved disciplines of quantum computing, power-systems engineering, and co-simulation. Chapter *Research Approach* describes how the proof-of-concept study is conducted. In *Experiments*, we lay out the experimental setup and show the results. After that, the chapter *Discussion of obstacles* presents four main issues we encountered in our experiments and ways of addressing them. Finally, we conclude the paper by highlighting the key takeaways and giving an outlook to future research.

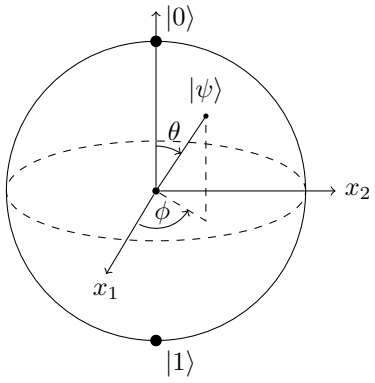

Figure 1 Representing the quantum state $|\psi\rangle$ in the Bloch sphere

Background

This chapter provides an overview of the key concepts and theories relevant to the research. First, a brief introduction to quantum computing is given; it forms the basis for the consequent section on the HHL algorithm, a quantum algorithm for solving systems of linear equations. Then, we cover how said algorithm is used to perform quantum DC power flow analysis. Finally, we explain co-simulation and its application in smart grid applications.

Quantum computing

Quantum computing leverages properties described by quantum mechanics to perform computation [18]. The fundamental unit of information in quantum computing is the quantum bit (*qubit* in short), which is analogous to the bit in classical computing. In contrast to a classical bit, a qubit can exist in a superposition of basis states $|0\rangle$ and $|1\rangle$ which can be described by a linear combination of the basis states with coefficients $\alpha,\ \beta \in \mathbb{C}$:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \quad \text{where } |0\rangle = \begin{pmatrix}1\\0\end{pmatrix} \quad \text{and} \quad |1\rangle = \begin{pmatrix}0\\1\end{pmatrix}. \tag{1}$$

When measuring the state of a quantum system, the superposition collapses, resulting in one of the basis states. According to the Born rule (2), the probability of the measurement yielding either basis state is proportional to the square of the corresponding coefficient in the state vector [19]. In other words, by estimating the probability one can make inferences about the superposition coefficients.

$$|\alpha|^2 + |\beta|^2 = 1 \tag{2}$$

To visualize the state of a qubit, the Bloch sphere can be used (see Figure 1). It is a complex unit sphere where the antipodes correspond to the basis states, and its surface represents all possible states.

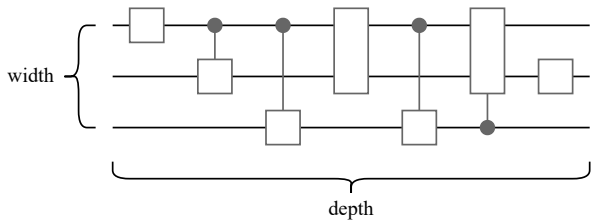

Figure 2 Illustration of width and depth of a quantum circuit

Alongside superposition, entanglement and tunneling are two fundamental quantum phenomena that have important applications in quantum computing. These principles are used in general-purpose circuit-based quantum computing and optimization-focused quantum annealing. In this paper, we deal with circuit-based computation. A circuit consists of a series of quantum operators (so-called *gates*) that act on the circuit's qubits. The width of a circuit refers to the number of qubits it contains, while the depth refers to the number of gates applied to these qubits (these dimensions are illustrated in Figure 2). As the width and depth of a circuit increase, so does its ability to perform complex computations; this, in turn, requires more precise control which is difficult to achieve in practice.

Recently, there has been a significant development in platforms and tools for quantum programming that provide access to cloud-based quantum hardware. Popular providers are IBM Quantum [20], Google Quantum AI [21], and D-Wave [22]. This development has facilitated researchers and practitioners to experiment with quantum algorithms and circuits. The steep growth of the discipline increases the need for software-engineering practices in quantum programming. This trend has led to the emergence of quantum software engineering [23] and architecture [24]. The development of standardized software development methodologies, design patterns, and software tools will be key to realizing the full potential of quantum computing.

HHL algorithm

The HHL algorithm is a quantum algorithm for solving linear systems of equations by Harrow, Hassidim, and Lloyd [12]. It has generated much excitement since it may provide exponential speedup for many problems in science and engineering. Its goal is to prepare a quantum state that encodes the solution to a system of equations.

Given the linear system

$$A\vec{x} = \vec{b}, \tag{3}$$

where $A \in \mathbb{C}^{N\times N}$ is a square matrix, and $\vec{b},\ \vec{x} \in \mathbb{C}^N$ are vectors. If A is a sparse, well-conditioned Hermitian matrix—meaning it is its own conjugate transpose—the algorithm prepares the normalized quantum state $|x\rangle$ that is an approximate solution to

$$A\,|x\rangle = |b\rangle\,, \quad \text{where } |.\rangle = \frac{.}{||.||}. \tag{4}$$

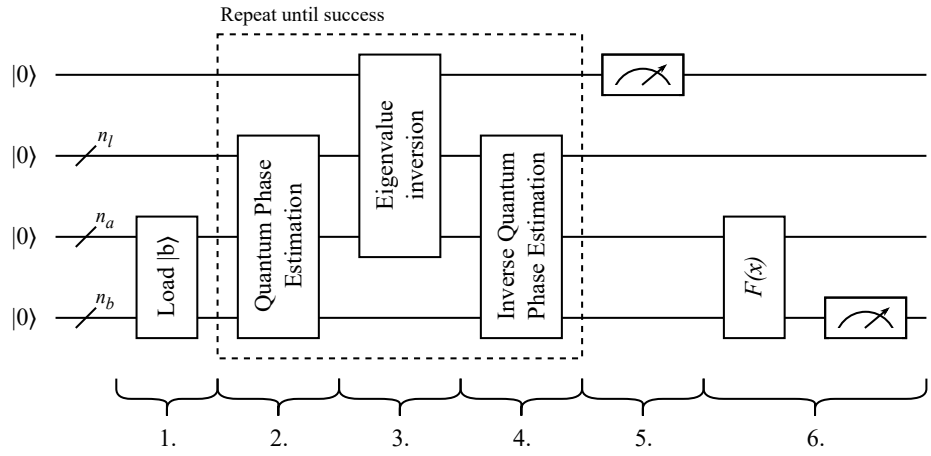

Figure 3 Quantum circuit for HHL algorithm

Figure 3 depicts the circuit for the HHL algorithm, including four quantum registers. It can be subdivided into six main steps [25]:

1 loading the input vector $|b\rangle$ to a quantum register,
2 estimating the eigenvalues of A using quantum phase estimation,
3 adding an ancilla qubit and applying conditioned rotation,
4 performing uncomputation using inverse quantum phase estimation,
5 determining whether the computation was successful by measuring the ancilla qubit, and
6 applying the observable M to query the output state.

The time complexity of the algorithm is $\mathcal{O}(\log(N)s^2\kappa^2/\varepsilon)$, where s is the sparsity, meaning at most s non-zero entries per row. Since A is a Hermitian matrix, the condition number κ represents the ratio of largest and smallest eigenvalue $\frac{|\lambda_{\max}|}{|\lambda_{\min}|}$. The precision is given by ε. Childs *et al.* [26] have further improved the scalability from poly$(1/\varepsilon)$ to poly$(\log 1/\varepsilon)$. The classical equivalent to the HHL algorithm is the conjugate gradient method; it also approximately solves a sparse, well-conditioned matrix, but exhibits a complexity of $\mathcal{O}(Ns\kappa\log(1/\varepsilon))$ [27].

Despite the potential advantage of the HHL algorithm, it has several caveats and limitations that restrict its practical utility (for a more detailed discussion, see [28]). For our research, four caveats are particularly relevant:

1 Expensive state preparation for input vector: To maintain the exponential advantage of HHL, we need a method to prepare the quantum state for $|b\rangle$ (Step 1) that scales at most logarithmically. In theory, quantum random access memory (QRAM)—as described in [29]—could help with this task [28], however, it was not yet realized in practice. Alternatively, the algorithm can be used as a subroutine where another component prepares $|b\rangle$ [12].
2 Precise eigenvalue representation: In Step 2, we apply quantum phase estimation to estimate eigenvalues of A by applying the operator $U = e^{iAt}$; determining a suitable time t can be a difficult task that diminishes the quantum speedup of the algorithm. An inappropriate value negatively affects both the performance and solution accuracy [30].

3 Circuit depth and noise susceptibility: The number of required operations and the circuit depth rises drastically as system size increases. Therefore, the noisy nature of current hardware makes larger systems infeasible [31].
4 Solution vector extraction: Reading all components of the solution vector would take linear time, which defeats the purpose of HHL's sublinear performance [12]. Often, however, only a subset of information is required. Consequently, we need a way to efficiently extract the required subset of information from the prepared quantum state $|x\rangle$ in Step 6.

Quantum power flow

Power-flow analysis is a fundamental tool in the operation and planning of power systems. It is used to compute the steady-state voltages, currents, and power flows in a network, given the network topology, component parameters, and load demand [32]. The analysis can be carried out for both AC and DC power systems.

$$P_i = \sum_{j=1}^{N} |V_i||V_j|(G_{i,j}\cos\theta_{i,j} + B_{i,j}\sin\theta_{i,j}) \tag{5}$$

$$Q_i = \sum_{j=1}^{N} |V_i||V_j|(G_{i,j}\cos\theta_{i,j} - B_{i,j}\sin\theta_{i,j}) \tag{6}$$

- P_i and Q_i are the injections of real and reactive power at bus i
- $|V_i|$ is the voltage magnitude at bus i
- $\theta_{i,j} = \theta_i - \theta_j$ is the voltage angle difference between bus i and j
- $G_{i,j}$ and $B_{i,j}$ are the conductance and susceptance of the line connecting bus i and bus j

In practice, the non-linear equations are solved approximately using approaches such as the Newton–Raphson method, where an initial estimate is iteratively refined using linear approximations. However, for some problems, such as transmission system planning and economic dispatch, the non-linear AC system can be treated as a DC system, under several simplifying assumptions, such as neglecting reactive power flows, voltage magnitudes, and line losses [33]. This results in the simplified power flow equation

$$P_i = \sum_{j=1}^{N} B_{i,j}\theta_{i,j}, \tag{7}$$

which can be formulated as an equivalent linear system of equations including the vector of nodal power injections $\mathbf{P}$, the nodal voltage angles $\boldsymbol{\theta}$ and the susceptance matrix $\mathbf{B}$. By removing the row and column corresponding to the slack bus, we make $\mathbf{B}$ invertible:

$$\mathbf{P} = \mathbf{B}\boldsymbol{\theta} \iff \mathbf{B}^{-1}\mathbf{P} = \boldsymbol{\theta}. \tag{8}$$

We can then use the voltage angles $\boldsymbol{\theta}$ to calculate the flow of a specific line $P_{i,j}$:

$$P_{i,j} = B_{i,j}(\theta_j - \theta_i). \tag{9}$$

Since both AC and DC power flow come down to solving systems of linear equations, they are amenable to be conducted using the HHL algorithm. For quantum AC power flow, Feng *et al.* [14] have proposed an approach that was later experimentally realized by Sævarsson *et al.* [31]. In contrast, Eskandarpour *et al.* [13] showed that the HHL algorithm can be used to perform quantum DC power flow. Their work was later expanded with a hybrid approach by Gao *et al.* [34].

Smart grid co-simulation

In a co-simulation, independent simulators are coupled that differ regarding their simulation tool, solver algorithm, or step size [35]. The paradigm allows for jointly executing independent simulations to simulate a larger system. It is therefore possible to perfrom the modeling on the subsystem level "without having the coupled problem in mind" [36, p. 516]. The coupling of the simulators can either be done bilaterally or they can be connected to a central orchestrating framework, yielding *orchestrated co-simulation*. When a larger number of simulators is involved, this simplifies the simulation architecture [37]. The orchestrating framework has three main tasks: initializing the simulators, synchronizing them, and facilitating data exchange between them [38]. A simulator comprises a simulation model together with a simulation kernel for executing it [39]. For this study, we assume simulators to have the capability of instantiating multiple homogeneous entities; for example, a power-plant simulator can handle multiple simulated power plants, each connected to a different bus in the grid. The topology of data exchange between simulated entities is specified in the simulation scenario.

The emergence of complex cyber-physical systems and systems of systems has brought with it an increased research interest in co-simulation. A broad state-of-the art analysis can be found in [35] whereas Gomes *et al.* [4] provide an in-depth technical discussion. The simulation paradigm has proven promising in various application domains, among them maritime and automotive engineering as well as robotics [40]. The most prominent domain for co-simulation seems to be power grids. Palensky *et al.* [38] provide an extensive primer on co-simulation of power systems together with ICT. For an empirical analysis of smart grid co-simulation see [41] and for a literature review see [42].

Co-simulation requires synchronizing simulators with varying step sizes and even time paradigms, e.g. discrete-event or continuous time. It further requires facilitating complex, at times asynchronous, data exchange. Therefore, frameworks are usually used that take care of these difficult tasks and provide interfaces to both connect already developed simulators and also define the scenario. A comprehensive overview of different frameworks can be found in [43]. One critical aspect of such frameworks is their adherence to the two most important co-simulation standards: First, the High-Level Architecture enables the reuse and interoperation of simulations [44]. Second, the Functional Mock-Up Interface (FMI) allows for interchange of dynamic co-simulation models [45].

Research Approach

This study aims to demonstrate the technical feasibility of quantum–classical co-simulation and identify potential issues. The integration of quantum DC power flow with smart grid co-simulation is the primary area of focus. The evaluation of the integration's scalability and usability will determine its practical applicability.

Case-study scenario selection

To conduct the study, an exploratory approach was adopted, which relies on a fictitious case-study scenario. The scenario involves a highly simplified load-shedding situation, in which a power line in a transmission system is monitored for potential overloads. If an overload is detected, the power of a solar farm is throttled and instead provided by a different generator. The scenario is implemented on a simple 5-bus test system, details of which are provided in the next chapter. The scenario was chosen based on the following criteria:

1. Small scale: A 5-bus transmission system with a low number of interacting sub-systems was used. The limited scale makes it easier to find freely available quantum hardware that is compatible.
2. Low complexity: The scenario was simplified to be suitable for a proof-of-concept study; domain-specific aspects are of little relevance. For example, time series replay was used for the solar generation and aggregate loads.
3. Suitability of DC approximation: DC simplification is appropriate for frequently computed line overload detection in transmission systems.
4. Sign negligibility: The direction of flow is not important when monitoring overload. This is necessary since we have no way of efficiently recovering sign information when using HHL.
5. Bi-directionality: The computed line flow is influenced by the other simulators, and vice versa.

Assessment criteria

The study involves the implementation of the chosen case-study scenario, followed by simulation runs in three main configurations: classical power flow, HHL run on simulated quantum hardware, and HHL run on real quantum hardware. The results obtained from each configuration will be compared to evaluate the accuracy and usefulness of quantum–classical co-simulation for real-world applications. Although the primary motivation for quantum–classical co-simulation is to make smart grid co-simulations faster, the timing analysis is not the main criterion at this stage of research. This is because of the overhead introduced by cloud-based quantum computing in a small-scale scenario, which vastly outweighs the potential advantage of improved scalability. Furthermore, the current implementation does not fully realize the logarithmic scalability promised by HHL due to inefficient circuit preparation (HHL caveat 1 in the background section on HHL algorithm). In addition to discussing timing aspects, the evaluation will also qualitatively assess how well the quantum solutions correspond to the classical ones, with an emphasis on feasibility not quantitative analysis. The main focus of the study is to use the implementation and simulation runs to identify and assess potential issues to determine the utility of the concept for near- to mid-term applications with real-world grid sizes.

Designing observables for information extraction

A central goal of DC power flow is computing the flow of active power for transmission lines based on grid characteristics and nodal injections. This computation is usually done via the voltage angles, which are then used to compute line flows. Since (at least) one bus—in our case Bus 0—serves as a slack bus and is therefore our reference angle with $\theta_0 = 0$, our goal is to find the remaining angles

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 & \theta_4 \end{bmatrix}^{\mathrm{T}}. \tag{10}$$

With the HHL algorithm we can approximately prepare a normalized quantum state

$$|x\rangle = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix}^{\mathrm{T}} := \frac{\boldsymbol{\theta}}{||\boldsymbol{\theta}||}. \tag{11}$$

In order to use the results of the algorithm in any classical routine, we have to extract information from the quantum state via measurement. Specifically, we have to apply an operator M (represented by a square matrix) to compute

$$F(x) = \langle x| M |x\rangle . \tag{12}$$

To retrieve the k^{th} element of $|x\rangle$ (which corresponds to θ_k) we can use a diagonal matrix with a single 1 at position k:

$$M_{i,j} = \begin{cases} 1 & \text{if } i = j = k \\ 0 & \text{otherwise} \end{cases}. \tag{13}$$

For example, to retrieve x_3 we can perform the computation

$$F(x) = \langle x| M |x\rangle = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = x_3^2, \tag{14}$$

yielding the square of x_3. Note that the information on the sign is lost. Consequently, we cannot distinguish a negative or positive voltage angle in relation to the reference angle θ_0. This is a critical issue, since we cannot get an accurate estimate of a line flow if the angles of the involved buses differ in sign. Therefore, we use an alternative approach: We design the observable in a way that extracts the differences of two coefficients in the state vector $|x\rangle$. Consider how to compute the power flow $P_{k,l}$ of the transmission line connecting buses k to l:

$$P_{k,l} = B_{k,l}(\theta_l - \theta_k) = \frac{B_{k,l}}{||\boldsymbol{\theta}||}(x_l - x_k). \tag{15}$$

Therefore, if we are able to compute $|x_l - x_k|$ we are able to estimate the magnitude of the line flow $|P_{k,l}|$. To extract the difference, we define the matrix

$$M_{i,j} = \begin{cases} 1 & \text{if } (i = j = k) \vee (i = j = k) \\ -1 & \text{if } (i = k \wedge j = l) \vee (i = l \wedge j = k) \\ 0 & \text{otherwise} \end{cases} . \tag{16}$$

If we want to compute the voltage-angle difference between the slack bus and any other bus, we end up with a matrix as defined in (13). To illustrate, let us estimate the magnitude of the line flow between Bus 2 and Bus 3 which correspond to x_2 and x_3; therefore, $k = 2$ and $l = 3$. Computing $F(x)$ yields

$$\begin{aligned} F(x) = \langle x| M |x\rangle = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \\ = x_2^2 - 2x_2x_3 + x_3^2 = (x_3 - x_2)^2, \end{aligned} \tag{17}$$

which we can take the square root of to receive $|x_3 - x_2|$. As in (15) we can multiply by $B_{2,3}/||\boldsymbol{\theta}||$ to get the magnitude of the line flow $|P_{2,3}|$. In contrast to (14), the sign information we lose here is that of the differences and not that of the components, which allows for a proper estimation of the magnitude of the angle difference and consequently the line flow. However, the information on line-flow direction is lost; we are therefore limited to applications that are not dependent on it.

Tool selection

Selecting appropriate tools is crucial for quantum–classical co-simulation. To support quantum computing and co-simulation, tools that can abstract low-level complexities and provide high-level programming are necessary. Additionally, the quantum-computing platform and co-simulation framework must be free and open-source to facilitate reproducibility and further research. The quantum-computing tool must provide local and cloud-based quantum simulators to enable efficient development and testing. Furthermore, it must allow simulators for both noise-free and noisy hardware, which is valuable in the Noisy Intermediate-Scale Quantum (NISQ) era—where noise is a critical aspect—to evaluate algorithm performance. Providing free access to real quantum computers is another essential criterion. Furthermore, the tool should offer session-based computing job scheduling to avoid queuing for each simulation step, prioritizing subsequent jobs after the initial wait. We chose IBM's Qiskit platform [46], specifically Qiskit Runtime [47]. For the co-simulation framework, in addition to being free and open-source, it needs to have an easy-to-use programming interface to connect simulators to the framework, and a programmatic interface to specify simulation topology. We have determined the framework Mosaik (originally introduced by [48] and [49]), specifically Version 3.0 [50]. The framework is designed for large-scale smart grid scenarios and uses the python programming language, which makes it particularly suitable for use with Qiskit.

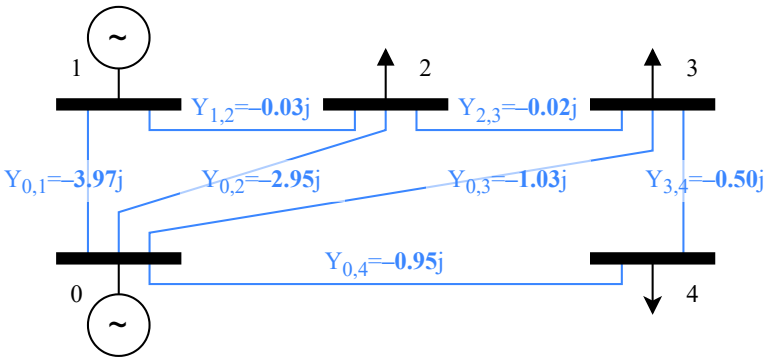

Figure 4 5-bus test system

Experiments

Based on the determined research approach, simulation experiments were conducted. We describe the details of the case-study scenario and how it was implemented, including co-simulation architecture and quantum computing–specific configurations. Then, we present the simulation results.

Case-study scenario

As a basis for our experiments we have chosen a 5-bus test system. Our choice of test-system size hinges on us being able to execute it on freely available quantum hardware. The test system is derived from one used by Sævarsson *et al.* [31] in their quantum AC power flow experiments; however, we use it in a DC setting. It is depicted in Figure 4. Accounting for the slack bus, the system results in a 4×4 modified susceptance matrix:

$$B' = \begin{bmatrix} 4 & -0.03 & 0 & 0 \\ -0.03 & 3 & -0.02 & 0 \\ 0 & -0.02 & 1.55 & -0.5 \\ 0 & 0 & -0.5 & 1.45 \end{bmatrix}. \tag{18}$$

Just as in [31], the admittances are chosen not to correspond to a realistic network, but to yield eigenvalues that can accurately be represented using a small number of bits. Therefore, any inaccuracies of eigenvalue representation do not influence the simulation results. In real-world applications, accurate eigenvalue representation requires careful consideration. However, the accuracy can be improved drastically by just a few additional qubits [31]. Here, the eigenvalues of B' are $\{1, 2, 3, 4\}$.

In our case-study scenario, a slack generator is connected to Bus 0. It simply matches the slack power, be it positive or negative; in this simplified example, it does not have limitations such as a maximum output. On Bus 1 is a solar farm with a peak generation of 3 MW. It is implemented as a time series that is based on the synthetic load profiles for a PV module provided by Austrian Power Settling and Clearing (APSC) [51] for July 1st 2022. The other buses—2, 3, and 4—are aggregate loads with a peak consumption of 0.5 MW, 0.5 MW, and 1.5 MW respectively. As with the solar farm, the loads are based on the APSC synthetic load profile of a household for the same day; it is scaled to the respective maximum power.

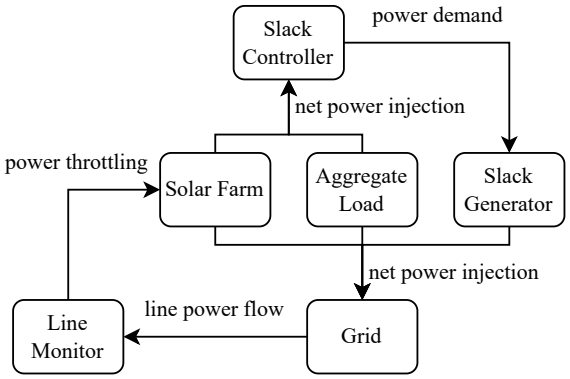

Figure 5 Co-simulation architecture: simulators and their information flows

Co-simulation architecture

For this study, we have chosen the smart grid co-simulation framework Mosaik (Version 3.0). It provides two programming interfaces: One is responsible for the interaction between the orchestrator and a simulator. The other specifies how a simulation scenario can be defined, including instantiating and connecting entities. A central aspect of co-simulation is the selection of participating simulators and the nature of their information exchange. Here, we briefly describe each simulator and Figure 5 shows their connections. Please note that for all but one simulator, there is only one instance. For example, in our small-scale case-study scenario, there is only one solar farm even though the solar-farm simulator is capable of handling multiple instances. In contrast, the aggregate-load simulator handles three instances.

The simulators are:

- Grid: It contains the grid topology and line admittances, and executes power-flow simulation using information on nodal injection. The simulator is capable of quantum power flow computation.
- Slack generator: An idealized slack generator that compensates for generation–consumption mismatch.
- Solar farm: The simulator outputs a scaled pre-recorded generation profile. The simulator has an input for the amount of power shedding.
- Aggregate load: Like the solar-farm simulator, this simulator also outputs a scaled-up, pre-recorded time series, specifically of household consumption.
- Slack controller: It aggregates all power generation and consumption to tell the slack generator how much load to generate or absorb.
- Line monitor: This simulator monitors a transmission line and compares the power flow to a predetermined threshold and outputs a power-shedding amount.
- Collector: The co-simulation contains a simulator for retrieving all relevant data and recording it in a format suitable for analysis. It is an example of a simulator that has no equivalent in the real-world system but exists purely for the simulation study.

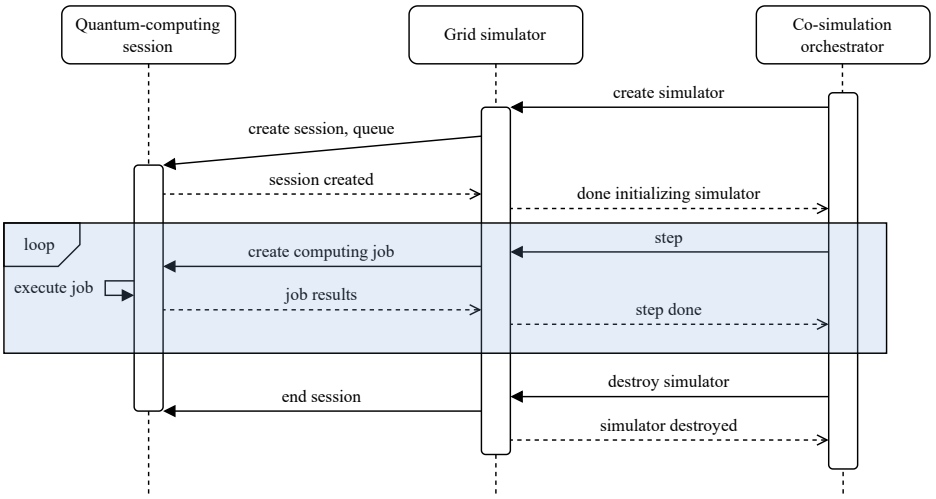

Figure 6 Control sequence

Quantum configuration

We utilize Qiskit as our quantum-computing platform and the IBM QASM simulator as our simulated quantum hardware. Using simulated hardware reduces queuing times and provides a noise-free simulation of a quantum machine. We select the IBM Oslo quantum computer, which employs the Falcon r5.11H processor, for our HHL implementation since it requires seven qubits to solve the 4×4 system of equations. In quantum computing, estimating coefficients—α, β in (1)—involves repeatedly preparing a quantum state and measuring its outcome, which yields the underlying coefficients via Born's rule (2) from the estimated probability. The number of iterations or shots performed impacts the accuracy of the estimation, but also affects computational expense. For our experiments, we use 10^5 shots, but the optimal number of shots is case-dependent and must be evaluated accordingly.

Implementation

The implementation of the HHL algorithm is derived from the one used in [31]. However, we employ Qiskit Runtime primitives, i.e. elementary subroutines. Specifically, we use the Estimator primitive where an operator M is applied to a repeatedly prepared quantum state to estimate the expected value. The operator definition is described in *Designing observables for information extraction*.

One significant benefit of Qiskit Runtime is session-based scheduling, which avoids the need to queue each time for every simulation step. In a co-simulation scenario, for each time step, a new computing job is created, which is then submitted to the hardware in the cloud. Queuing for each simulation step would be completely unworkable, and Qiskit Runtime enables starting a session for the entire co-simulation, prioritizing each job once the session has started. This control sequence is illustrated in Figure 6. The orchestrator initializes the simulator which in turn queues on the quantum-computing platform to create a new session, then the actual co-simulation may start. For each of the simulator's time steps, a computing job is submitted to the session with prioritized access to the quantum resources. Only after the entire simulation run is ended, the session is closed.

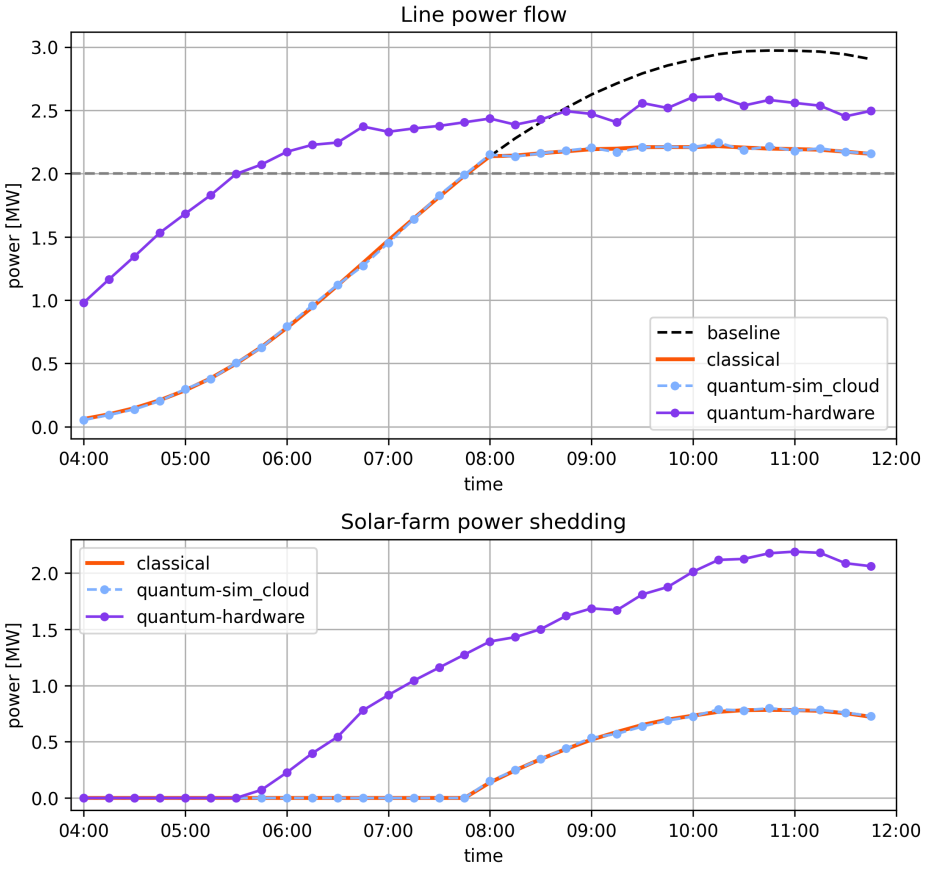

Figure 7 Simulation results

Simulation results

Multiple simulation experiments were conducted to allow for a comparison between different ways of computing power flow: classical solution, simulated quantum computer, and real quantum hardware. The focal point of this comparison was the load-shedding behavior of our case-study scenario, determining whether quantum computing could be a valuable substitute for classical computation.

In Figure 7, we have the power flow on the monitored line on top, and the power-shedding amount on the bottom. In both diagrams, one can see the classical solution and the two quantum solutions, one with simulated and one with real hardware. For the power flow, we also include a baseline reference that shows the power flow in the observed line without having load-shedding measures that curtail overload. One can observe that the noise-free quantum simulator corresponds closely to the classical result. However, the flow computed with real quantum hardware deviates so drastically from the actual solution as to be unusable in practice. The severe effects from noise become more apparent when looking at the output distribution of measuring the state of an exemplary HHL circuit: Figure 8 illustrates estimated probability distributions, which correspond to the squares of the coefficients, and thus the normalized solution vector—see Borne rule (2). Whereas the noise-free quantum simulator yields clearly distinguishable values, both the noisy quantum

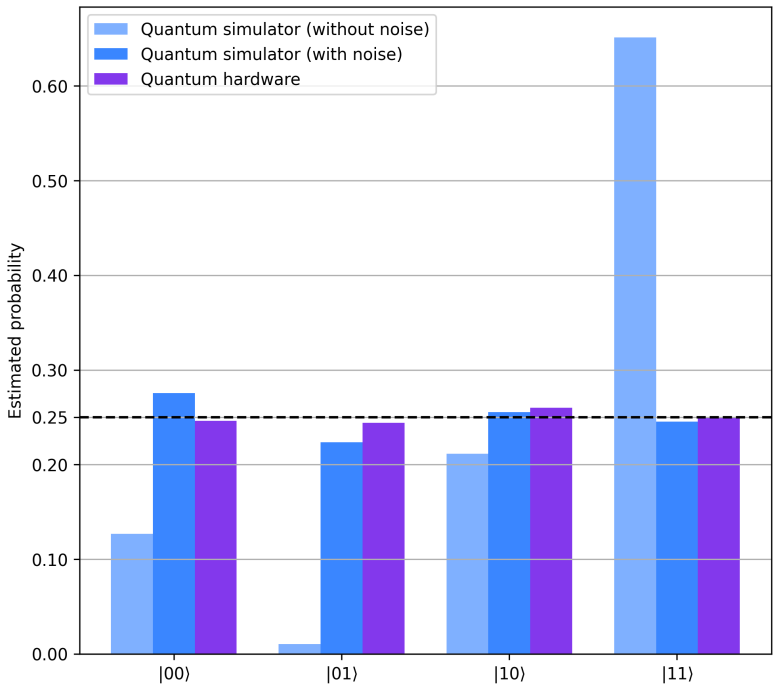

Figure 8 Comparison of estimated probabilities

simulator and the real quantum hardware result almost in uniform distributions. This is to be expected when dealing purely with noise. It is highly likely, that the large circuit depth of 2620 leads to an overwhelming accumulation of noise and an execution time that well exceeds the coherence time of the quantum computer.

Finally, we should address the timing aspects. The central motivating factor for quantum–classical co-simulation is accelerating large-scale smart grid co-simulations. However, for this early feasibility study, we do not focus on timing improvement. The simulation experiments have shown that both runs using simulated and real quantum hardware take orders of magnitude longer than the classical solution. Even with a perfect quantum implementation that preserves the potential logarithmic scalability of the HHL algorithm, we would expect the overhead introduced by the quantum solution to far outweigh the scalability advantages at a small scale. Future research should address these issues in an in-depth quantitative way.

Discussion of obstacles

Our experiments have exposed four major obstacles related to quantum–classical co-simulation and quantum power flow for smart grid co-simulation. In this chapter, we describe each of the obstacles and their impact, and discuss how they can be addressed. This is summarized in Figure 9.

Queuing time

Processing time on real quantum hardware is a high-demand, limited resource. A low number of quantum computers is accessed by numerous users via the cloud. Except

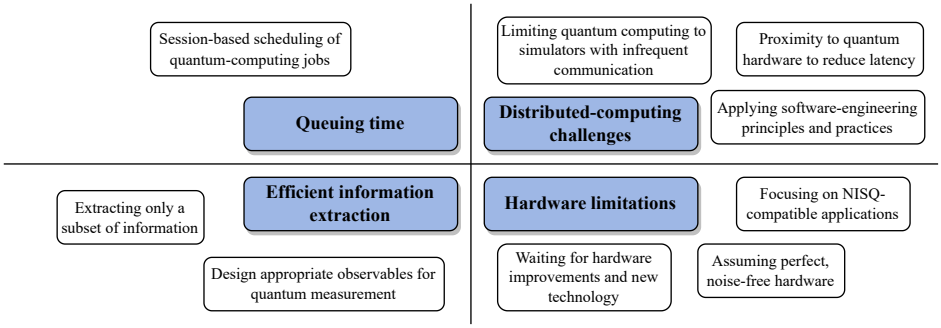

Figure 9 Overview of obstacles and ways to address them

for specialized scenarios, this paradigm will likely remain for the near- to mid-term future. As a result, scheduling and prioritizing computing jobs is a constant challenge. For co-simulations, submitting a single large computing job is not possible since there are constant interactions between the simulators. Instead, each time step creates a new computing job, making queuing for each job impractical.

In our study, we have found a simple, easy-to-implement way to deal with this issue: Using session-based scheduling, we queue only once to start the session and then submit computing jobs to it. This way, the jobs get prioritized, which minimizes wait time for all jobs but the first. The session is started when instantiating the simulator and is ended once it is destroyed; this sequence is illustrated in Figure 6. Qiskit Runtime has proven to be an effective facilitator for this scheduling paradigm.

Distributed-computing challenges

In the context of quantum computing, cloud-based access is common. Integrating a quantum computing–based simulator in a co-simulation results in distributed simulation, where the execution of a simulation is distributed across multiple processors [52]; specifically, we are dealing with distributed *co*-simulation. As communication relies on the internet, high-speed, low-latency connections, such as InfiniBand, are not viable [53]. Therefore, significant overhead and latency are introduced, which can either occur within a simulator (e.g. when only the quantum processing is done on the cloud) or between the orchestrator and a purely cloud-run simulator.

To mitigate these issues, one strategy is to restrict the application of cloud-based resources to simulators with infrequent interaction, where the data and time overhead is less critical than for simulators with high-frequency communication. Palenksy *et al.* [38] suggest running closely coupled simulators on the same machine. Another way to address the problem is through geographical proximity. According to Mirz *et al.* [53], geographical distance is the main factor contributing to round-trip time, indicating that it is preferable to have quantum resources located as close as possible. In an ideal scenario, a local or even an on-site quantum computer laboratory is available for some industrial or academic applications.

Efficient information extraction

Our quantum algorithm for solving power flow, the HHL algorithm, promises logarithmic scalability, in principle. However, there are various limitations and caveats

that threaten this advantage. Importantly, we cannot retrieve all components of the solution vector, since that would require a linear number of steps. Therefore, we need an efficient means of extracting the necessary information from the quantum state that does not diminish the algorithms scalability. Furthermore, when measuring the quantum state, we lose sign information, meaning we cannot distinguish positive and negative coefficients; this requires further consideration on how to compute the required quantities.

The main way to deal with this issue is to only extract a subset of information from the prepared state. If the subset's size scales at most logarithmically with system size, we can theoretically maintain the exponential speed-up of HHL. With power-flow analysis, for example, often only the power flow in a few lines is of interest. Efficiently extracting the line flow requires designing suitable observables that can be implemented in a quantum circuit (for more details, see *Designing observables for information extraction*). In our case study, we are only interested in the magnitude of the line flow, not having its direction is therefore irrelevant. For other applications, this issue could potentially be a decisive factor.

Hardware limitations

We are currently in the era of NISQ hardware; therefore, we have to deal with noise and decoherence impeding computation. The impact of these limitations is evident for the HHL algorithm; it results in excessively large circuits whose size scales poorly for larger systems. The coherence time of quantum hardware may be much too short to reliably execute a full circuit, and the noise accumulates over many gates in a large-depth circuit. In our experiments, the effects of noise and decoherence make the results basically unusable, resulting in output distributions (Figure 8) barely distinguishable from pure noise. Currently, quantum error correction and mitigation techniques are not nearly advanced enough to compensate for this. Although the number of qubits has been touted as an indicator for the improvement of quantum computers, for quantum power flow with HHL, it is not a major issue. The number of required qubits rises only logarithmically; therefore, computing large-scale systems potentially only requires a few qubits. However, error mitigation and correction may require additional qubits, increasing the need for quantum hardware with a large qubit count. Additionally, the lack of QRAM poses a challenge for efficiently preparing the input vector; it is also a likely prerequisite for an efficient HHL-based implementation of quantum AC power flow [15].

Even though hardware development is progressing at a rapid pace [54], useful quantum power flow—in the context of quantum–classical co-simulation or not—is unlikely in the near-term future. Therefore, a degree of patience is required until we have large-scale fault-tolerant quantum computing and new technologies like QRAM. Until then, we can focus on quantum approaches that are more suitable for NISQ hardware, such as variational algorithms (e.g. Variational Quantum Eigensolver) and quantum annealing. Alternatively, Bertels *et al.* [55] suggest assuming perfect noise-free qubits for algorithm and application development, to have them ready once the hardware is sufficiently advanced.

Conclusion

Co-simulation is a powerful paradigm to address smart grid simulation challenges, but large-scale scenarios suffer from performance issues. Quantum–classical co-simulation can alleviate these issues by using quantum computing to run constituent simulators. Although the approach was previously postulated, a proof-of-concept study was missing. This study demonstrates the technical feasibility and identifies potential issues, using a case study–based approach focused on integrating quantum DC power flow with smart grid co-simulation. Four major obstacles are uncovered, varying in relation to quantum–classical co-simulation, HHL-based quantum power flow, and quantum computing generally: queuing delays, distributed-computing challenges, the need for efficient information extraction, and hardware limitations.

The key findings from our proof-of-concept experiments are:

- It is feasible to integrate a quantum computing–based simulator with smart grid co-simulation using commonly available software tools with moderate integration effort.
- Session-based scheduling is critical for quantum–classical co-simulation to avoid significant performance penalties; there are tools that enable this.
- All distributed, network-based co-simulation challenges are inherent to quantum–classical co-simulation and must be addressed.
- The HHL-based quantum DC power flow works in principle and may be applicable in some co-simulation scenarios given sufficient hardware advancement.
- Efficient solution extraction of a subset of the HHL solution is achievable by designing appropriate observables without compromising scalability.
- HHL faces significant issues on NISQ hardware and may require large-scale fault-tolerant quantum computing for practical use.

In conclusion, our study demonstrates the technical feasibility of quantum–classical co-simulation for smart grid applications, and identified several key challenges that need to be addressed. While the practical realization of quantum power flow on NISQ hardware appears unlikely in the near to mid-term, the potential for using quantum computing to address optimization problems in smart grid co-simulation is promising. In particular, variational algorithms and quantum annealing show promise for improving the scalability and efficiency of smart grid co-simulation. We anticipate that further research at the intersection of quantum computing, co-simulation, and power-systems engineering will open up new opportunities for solving complex problems in large-scale smart grid systems.

Funding

The financial support by the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development and the Christian Doppler Research Association as well as the Federal State of Salzburg is gratefully acknowledged.

Availability of data and materials

The data as well as the code for simulation and data analysis can be found in https://github.com/DominikVereno/quantum-classical-co-simulation-proof-of-concept.

Author's contributions

Dominik Vereno: Conceptualization, Methodology, Software, Investigation, Writing - Original Draft, Visualization. Amin Khodaei: Conceptualization, Methodology, Writing - Review & Editing. Christian Neureiter: Resources, Funding acquisition. Sebastian Lehnhoff: Conceptualization, Supervision

Competing interests
The authors declare that they have no competing interests.

Author details
[1] Josef Ressel Centre for Dependable System-of-Systems Engineering, Urstein Süd 1, 5412, Puch/Salzburg, Austria. [2] Electrical and Computer Engineering Department, University of Denver, 2155 E Wesley Ave, 80208, Denver, CO, USA. [3] R&D Division Energy, OFFIS, Escherweg 2, 26121 , Oldenburg, Germany.

References
1. M. H. Ullah, R. Eskandarpour, H. Zheng, and A. Khodaei, "Quantum computing for smart grid applications," *IET Generation, Transmission & Distribution*, vol. 16, no. 21, pp. 4239–4257, 2022.
2. H. Farhangi, "The path of the smart grid," *IEEE Power and Energy Magazine*, vol. 8, no. 1, pp. 18–28, 2010.
3. D. Vereno, J. Harb, and C. Neureiter, "Paving the way for reinforcement learning in smart grid co-simulations," in *6th Workshop on Formal Co-Simulation of Cyber-Physical Systems*, 2023.
4. C. Gomes, C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe, "Co-simulation: A survey," *ACM Computing Surveys*, vol. 51, no. 3, 2018.
5. R. Eskandarpour, P. Gokhale, A. Khodaei, F. T. Chong, A. Passo, and S. Bahramirad, "Quantum computing for enhancing grid security," *IEEE Transactions on Power Systems*, vol. 35, no. 5, pp. 4135–4137, 2020.
6. C. Steinbrink, F. Schlögl, D. Babazadeh, S. Lehnhoff, S. Rohjans, and A. Narayan, "Future perspectives of co-simulation in the smart grid domain," in *IEEE International Energy Conference (ENERGYCON)*, 2018, pp. 1–6.
7. D.-H. Yoon and Y. Han, "Parallel power flow computation trends and applications: A review focusing on GPU," *Energies*, vol. 13, no. 9, 2020.
8. M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge, UK: Cambridge University Press, 2010.
9. R. Eskandarpour, K. Ghosh, A. Khodaei, A. Paaso, and L. Zhang, "Quantum-enhanced grid of the future: A primer," *IEEE Access*, vol. 8, pp. 188 993–189 002, 2020.
10. S. Koretsky, P. Gokhale, J. M. Baker, J. Viszlai, H. Zheng, N. Gurung, R. Burg, E. A. Paaso, A. Khodaei, R. Eskandarpour, and F. T. Chong, "Adapting quantum approximation optimization algorithm (QAOA) for unit commitment," in *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)*, 2021, pp. 181–187.
11. E. B. Jones, E. Kapit, C.-Y. Chang, D. Biagioni, D. Vaidhynathan, P. Graf, and W. Jones, "On the computational viability of quantum optimization for PMU placement," in *2020 IEEE Power & Energy Society General Meeting (PESGM)*, 2020, pp. 1–5.
12. A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations," *Physical Review Letters*, vol. 103, no. 15, 2009.
13. R. Eskandarpour, K. Ghosh, A. Khodaei, and A. Paaso, "Experimental quantum computing to solve network DC power flow problem," 2021. [Online]. Available: https://arxiv.org/abs/2106.12032
14. F. Feng, Y. Zhou, and P. Zhang, "Quantum power flow," *IEEE Transactions on Power Systems*, vol. 36, no. 4, pp. 3810–3812, 2021.
15. S. Golestan, M. R. Habibi, S. Y. Mousazadeh Mousavi, J. M. Guerrero, and J. C. Vasquez, "Quantum computation in power systems: An overview of recent advances," *Energy Reports*, vol. 9, pp. 584–596, 2023.
16. D. Vereno, A. Khodaei, C. Neureiter, and S. Lehnhoff, "Exploiting quantum power flow in smart grid co-simulation," in *11th Workshop on Modeling and Simulation of Cyber-Physical Energy Systems*. San Antonio, Texas, USA: IEEE, 2023.
17. S. Endo, Z. Cai, S. C. Benjamin, and X. Yuan, "Hybrid quantum-classical algorithms and quantum error mitigation," *Journal of the Physical Society of Japan*, vol. 90, no. 3, p. 032001, 2021.
18. J. D. Hidary, *Quantum Computing: An Applied Approach*, 2nd ed. Cham, Switzerland: Springer International Publishing, 2021.
19. M. Born, "Zur Quantenmechanik der Stoßvorgänge," *Zeitschrift für Physik*, vol. 37, no. 12, pp. 863–867, Dec. 1926.
20. IBM, "Ibm quantum," https://www.ibm.com/quantum, [Accessed April 15th, 2023].
21. Google, "Google quantum ai," https://quantumai.google, [Accessed April 15th, 2023].
22. D.-W. Systems, "D-wave," https://www.dwavesys.com/, [Accessed April 15th, 2023].
23. J. Zhao, "Quantum software engineering: Landscapes and horizons," 2020. [Online]. Available: https://arxiv.org/abs/2007.07047
24. A. A. Khan, A. Ahmad, M. Waseem, P. Liang, M. Fahmideh, T. Mikkonen, and P. Abrahamsson, "Software architecture for quantum computing systems – a systematic review," 2022. [Online]. Available: https://arxiv.org/abs/2202.05505
25. I. Quantum, "Solving linear systems of equations using hhl and its qiskit implementation," https://learn.qiskit.org/course/ch-applications/solving-linear-systems-of-equations-using-hhl-and-its-qiskit-implementation, [Accessed February 8th, 2023].
26. A. M. Childs, R. Kothari, and R. D. Somma, "Quantum algorithm for systems of linear equations with exponentially improved dependence on precision," *SIAM Journal on Computing*, vol. 46, no. 6, pp. 1920–1950, 2017.
27. J. R. Shewchuk, "An introduction to the conjugate gradient method without the agonizing pain," Carnegie Mellon University, USA, Tech. Rep., 1994.
28. S. Aaronson, "Read the fine print," *Nature Physics*, vol. 11, no. 4, pp. 291–293, 2015.
29. V. Giovannetti, S. Lloyd, and L. Maccone, "Quantum random access memory," *Physical Review Letters*, vol. 100, p. 160501, 2008.
30. C. Shao, "Reconsider hhl algorithm and its related quantum machine learning algorithms," 2018. [Online]. Available: https://arxiv.org/abs/1803.01486

31. B. Sævarsson, S. Chatzivasileiadis, H. Jóhannsson, and J. Østergaard, "Quantum computing for power flow algorithms: Testing on real quantum computers," 2022. [Online]. Available: https://arxiv.org/abs/2204.14028
32. J. Grainger and W. Stevenson, *Power System Analysis*. New York: McGraw-Hill Education, 1994.
33. K. Purchala, L. Meeus, D. Van Dommelen, and R. Belmans, "Usefulness of DC power flow for active power flow analysis," in *IEEE Power Engineering Society General Meeting*, 2005, pp. 454–459 Vol. 1.
34. F. Gao, G. Wu, S. Guo, W. Dai, and F. Shuang, "Solving DC power flow problems using quantum and hybrid algorithms," 2022. [Online]. Available: https://arxiv.org/abs/2201.04848
35. I. Hafner and N. Popper, "An overview of the state of the art in co-simulation and related methods," *SNE Simulation Notes Europe*, vol. 31, no. 4, pp. 185–200, 2021.
36. F. Schloegl, S. Rohjans, S. Lehnhoff, J. Velasquez, C. Steinbrink, and P. Palensky, "Towards a classification scheme for co-simulation approaches in energy systems," in *International Symposium on Smart Electric Distribution Systems and Technologies*, 2015, pp. 516–521.
37. V. H. Nguyen, Y. Besanger, Q. T. Tran, and T. L. Nguyen, "On conceptual structuration and coupling methods of co-simulation frameworks in cyber-physical energy system validation," *Energies*, vol. 10, no. 12, 2017.
38. P. Palensky, A. A. van der Meer, C. D. Lopez, A. Joseph, and K. Pan, "Cosimulation of intelligent power systems: Fundamentals, software architecture, numerics, and coupling," *IEEE Industrial Electronics Magazine*, vol. 11, no. 1, pp. 34–50, 2017.
39. J. Denil, B. Meyers, P. De Meulenaere, and H. Vangheluwe, "Explicit semantic adaptation of hybrid formalisms for fmi co-simulation," ser. DEVS '15. San Diego, CA, USA: Society for Computer Simulation International, 2015, p. 99–106.
40. C. Gomes, C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe, "Co-simulation: State of the art," 2017. [Online]. Available: https://arxiv.org/abs/1702.00686
41. G. Schweiger, C. Gomes, G. Engel, I. Hafner, J. Schoeggl, A. Posch, and T. Nouidui, "An empirical survey on co-simulation: Promising standards, challenges and research needs," *Simulation Modelling Practice and Theory*, vol. 95, pp. 148–163, 2019.
42. P. Mihal, M. Schvarcbacher, B. Rossi, and T. Pitner, "Smart grids co-simulations: Survey & research directions," *Sustainable Computing: Informatics and Systems*, vol. 35, p. 100726, 2022.
43. M. Vogt, F. Marten, and M. Braun, "A survey and statistical analysis of smart grid co-simulations," *Applied Energy*, vol. 222, pp. 67–78, 2018.
44. J. S. Dahmann, "High level architecture for simulation," in *Proceedings of the 1st International Workshop on Distributed Interactive Simulation and Real Time Applications*, 1997, pp. 9–14.
45. T. Blochwitz, M. Otter, M. Arnold, C. Bausch, C. Clauß, H. Elmqvist, A. Junghanns, J. Mauss, M. Monteiro, T. Neidhold, D. Neumerkel, H. Olsson, J.-V. Peetz, and S. Wolf, "The functional mockup interface for tool independent exchange of simulation models," in *Proceedings of the 8th International Modelica Conference*, 2011, pp. 105–114.
46. Qiskit contributors, "Qiskit: An open-source framework for quantum computing," 2023.
47. I. Quantum, "Qiskit runtime overview," https://quantum-computing.ibm.com/lab/docs/iql/runtime/, [Accessed April 2, 2023].
48. S. Schütte, S. Scherfke, and M. Tröschel, "Mosaik: A framework for modular simulation of active components in smart grids," in *2011 IEEE First International Workshop on Smart Grid Modeling and Simulation (SGMS)*, 2011, pp. 55–60.
49. S. Rohjans, S. Lehnhoff, S. Schütte, S. Scherfke, and S. Hussain, "mosaik - a modular platform for the evaluation of agent-based smart grid control," in *IEEE PES ISGT Europe 2013*, 2013, pp. 1–5.
50. A. Ofenloch, J. S. Schwarz, D. Tolk, T. Brandt, R. Eilers, R. Ramirez, T. Raub, and S. Lehnhoff, "Mosaik 3.0: Combining time-stepped and discrete event simulation," in *2022 Open Source Modelling and Simulation of Energy Systems (OSMSES)*, 2022, pp. 1–5.
51. APCS - Austrian Power Clearing and Settlement AG, "Synthetic load profiles," https://www.apcs.at/en/clearing/physical-clearing/synthetic-load-profiles, 2023, [Accessed March 15, 2023].
52. R. M. Fujimoto, *Parallel and Distributed Simulation Systems*, ser. Wiley Series on Parallel and Distributed Computing. Nashville, TN: John Wiley & Sons, Dec. 1999.
53. M. Mirz, S. Vogel, B. Schäfer, and A. Monti, "Distributed real-time co-simulation as a service," in *2018 IEEE International Conference on Industrial Electronics for Sustainable Energy Systems (IESES)*, 2018, pp. 534–539.
54. M. Brooks, "What's next for quantum computing," https://www.technologyreview.com/2023/01/06/1066317/whats-next-for-quantum-computing/, MIT Technology Review, 2023, [Accessed March 18, 2023].
55. K. Bertels, A. Sarkar, and I. Ashraf, "Quantum computing—from NISQ to PISQ," *IEEE Micro*, vol. 41, no. 5, pp. 24–32, 2021.