

# Can we run quantum circuits on ultra-cold atom devices?

Fred Jendrzejewski<sup>1</sup> and Manuel Rudolph<sup>1</sup>

<sup>1</sup>Kirchhoff-Institut für Physik

May 28, 2020

## Abstract

In this blog-post, we present our path and thoughts towards using ultra-cold atom experiments for quantum computation. They are the result of a two month internship where we studied the feasibility of such an undertaking in our group. Many associate only universal devices, especially qubit devices, to be valid quantum computers. We show how we think of our ultra-cold atoms in terms of quantum circuits and implement first steps in the software framework [PennyLane](#).

## Motivation

Over recent years, interest in quantum information processing has increased tremendously. This rise was fueled by activities of massive commercial players like [IBM](#), [Google](#), [Microsoft](#) and large investments in startups like [D-Wave](#), [Rigetti](#), [Xanadu](#), [Zapata](#) and [Cambridge Quantum Computing](#). The work of these companies typically focuses on **universal quantum computers**, which hold the promise of exponential speed-up for a variety of NP-hard problems. However, this **universality usually comes at the price** of algorithmic overhead for a wide range of tasks which generally leads to worse overall fidelity. If we give up the constraint of a universal quantum computer, we can build **specialized quantum hardware** for the problems of interest. Such systems are called **quantum simulators**.

**Ultra-cold atoms have become a leading platform of such quantum simulators at a large scale.** Most importantly for us, they are the experimental platform our research group is working on. Already demonstrated applications involve an enormous variety of condensed-matter problems like e.g. the Hubbard model ([Bloch et al., 2008](#)), topological systems ([Goldman et al., 2014](#)), superfluidity ([Regal and Jin, 2007](#)) and disorder ([Lagendijk et al., 2009](#)). More recently, cold atoms started to find applications to Ising models ([Bernien et al., 2017](#)) and high-energy physics ([Zohar et al., 2016](#)).

However, **quantum simulators are currently limited to an academic environment** because of

1. the **complexity of constructing the experimental hardware**, which involves substantial technical know-how and funding.
2. the **specificity of knowledge and language required** to understand/communicate the experiment.
3. the need to **formulate the target problem as a Hamiltonian**.
4. the **manual compilation of programs** on this hardware class.

A **circuit-based approach to ultra-cold atom quantum simulators** would alleviate points 2) and 3) as we could agree on the quantum circuit, given the hardware-specific operations, and run it on the experiment. Integrating the system with a software stack like [PennyLane](#) would additionally allow to address and run the experiment on a higher level and **substantially lower the barrier of entry for users outside of the cold-atom and quantum physics community**.

These points motivated our research group to have a look how our experimental hardware, which is controlled by the [Labscript Suite](#), can be integrated into PennyLane to run quantum circuits on ultra-cold atom devices. Our group is part of [SynQS](#), working on two ultra-cold atom experiments, the NaLi and SoPa (named after the two atomic species used in the lab).

## Why use ultra-cold atoms as a quantum information platform ?

For ultra-cold atoms, the **processing unit is not a qubit** and as such, the **natural operations are not** generally the **commonly used and widely known qubit operations** such as  $X, Y, Z$  rotations, Hadamard  $H$  gates, entangling  $CNOT$  or  $XX$  gates with observables like  $\sigma^z$  expectations. Instead, our NaLi experiment implements a  $X(\theta)$  **rotation on a long spin** of many bosons on one optical lattice site. Additionally, the system naturally evolves under a many-body Hamiltonian  $H_{mb}$ , coupling the the atomic species.

What is the qubit equivalent of those operations? The relation is pretty complex - and that is the nice thing! **Cold atoms are different**. Given their success in the quantum simulation of many-body problems, they could make for an exciting quantum information processor that is **complementary to universal devices**. So as a general rule of thumb, it could be argued that cold atom machines often give up some control over individual particles, leading to much bigger systems. Additionally, because we work with a large number of atoms, we can measure obtain rather precise estimates of expectation values **with a single state preparation and one measurement**.

## A concrete example from our group

Before we go into more detailed discussion of our implementation, we will present a **concrete example of our device**, where **we study lattice gauge theories**. These theories have become a popular benchmark for quantum simulators ([Kokail et al., 2019](#)) and quantum computers ([Martinez et al., 2016](#); [Mathis et al., 2020b](#)) in recent years. On our NaLi machine, we recently performed experiments on the building block for certain quantum simulators that would be suited for theories from high-energy physics ([Mil et al., 2020](#)). These experiments are performed with two atomic species, sodium and lithium, that can be prepared in two internal spin states.

**Re-thinking our analog quantum simulation** for the purpose of quantum circuits, we can **represent the sodium atoms as a very long spin** state  $|\psi_N\rangle$ . The observable outcomes are  $L_z = -\frac{N}{2}, \dots, \frac{N}{2}$ , where  $N \sim 10^5$  is the number of sodium atoms. On a quantum computer this should be compared to a qubit, for which we can only measure two outcomes  $L_z = \pm \frac{1}{2}$ .

The **lithium atoms** on the other hand can be **described by the states on two independent sites**  $|\psi_p\rangle$  and  $|\psi_v\rangle$ . For each site we can observe that number of atoms that sit on it. The observable outcomes are then  $n_{p/v} = 0, \dots, n$  where  $n \sim 10^4$  is the number of lithium atoms.

In this formulation and **analogous to other circuit-based quantum computing devices**, the NaLi experiment then consists of three main stages:

1. The atoms are prepared in some initial state...
2. controlled through a set of operations/gates...
3. and then measured to evaluate the operators  $L_z$  and  $N_{p,v}$

Without having to dig through all the physical details to understand the publication ([Mil et al., 2020](#)), the figures in it intuitively open themselves up to **readers with varying backgrounds** when plotting the

corresponding circuits next to them as shown in Fig. 1.

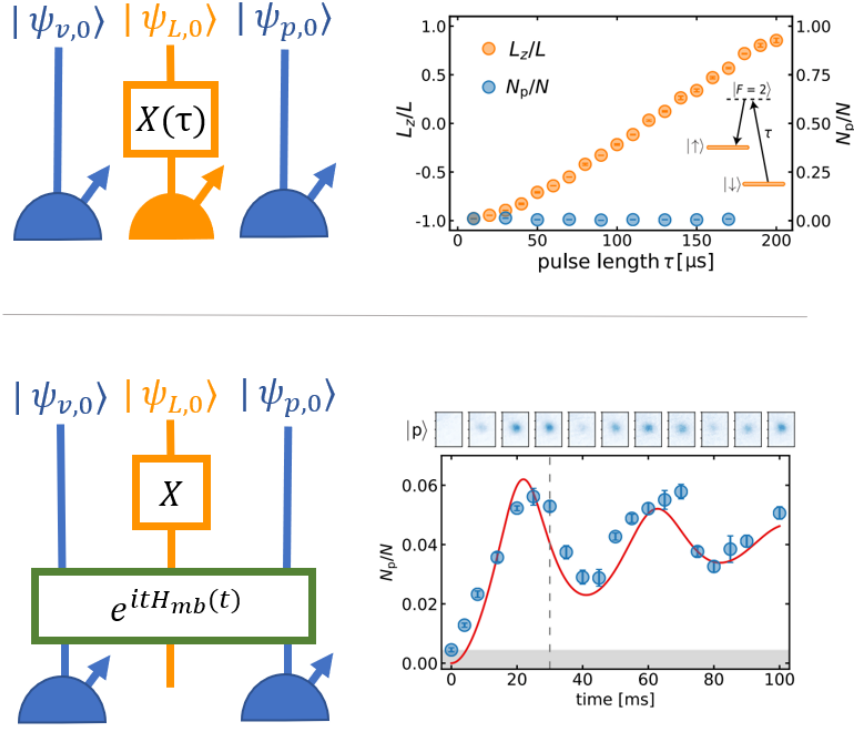


Figure 1: Sketched quantum circuits that correspond to selected figures from Mil et al. (Mil et al., 2020). The results gain a clear intuitive meaning through the quantum circuits even though the system that is being simulated is not known. The color blue signifies the Sodium atoms and the orange color Lithium.

#### Additional benefits of using the circuit language include

- a **more uniform definition of fidelity** for ultra-cold atom experiments relative to fidelity in qubit devices where each operation and the final result can be quantified.
- **improved communication with and feedback** of theorists involved in the project by working on the same level of resolution.
- more easily **generalizing the capabilities of the device** to work on **other problems**.
- general accessibility and a **path towards using ultra-cold atoms experiments in the quantum computing community**.

## Choice of framework software for our quantum circuits

The choice of which framework software to use for our circuit-based approach was not a particularly hard one for us. We have our own experiment in the lab which implements a non-standard device for quantum simulation and thus some of our requirements are (roughly in order of importance):

1. Can we **integrate our own experimental hardware** into the framework and potentially run it from there?
2. How strong of an **assumption** is made **about the computational unit** of the hardware? Does it have to be strictly qubits/photons etc.?
3. Is the framework **open-source** and shows **openness to the academic community**?
4. Can also **integrate classical simulators** that are based on our hardware?
5. Does it provide some sort of **user management and safety measures** to avoid unqualified users from killing the system?

Lets give a brief overview of how we answered those questions with regard to PennyLane. **PennyLane is designed for users to write and use their own *plugin***, which is what they call the integration of quantum computing devices into the framework. Officially, it only supports *qubit* and *continuous variable* devices and not more general approaches but we may be able to **integrate bosonic and fermionic atoms flexibly** as the code itself is not very restricted. Other big platorms like [Qiskit](#), [Forest](#) and [Strawberry Fields](#) focus exclusively on their own quantum computing hardware or simulator without the possibility to integrate your own. PennyLane is **open-source and Python-based**, which is perfect as the *Labscript Suite* controlling our experiment is so, too. Additionally, the company behind PennyLane, [Xanadu](#), is a partly academic player who is very open in their publications and **supports a great community** around the software. The framework allows to add simulators to the framework device which mirror the functionality of the hardware. It performs various parameter range checks for the system and the operations used. Not built-in is user a system for management which allows only registered users to run the experiment from a queue. We would have to implement this ourselves.

All-in-all, **PennyLane seems to be the best and likely only fit** for our endeavors. Let's talk about it in more detail.

## Pennylane

*A cross-platform Python library for quantum machine learning, automatic differentiation, and optimization of hybrid quantum-classical computations.*

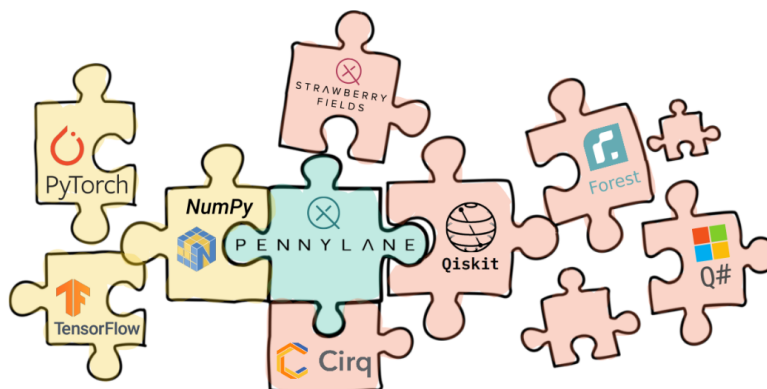


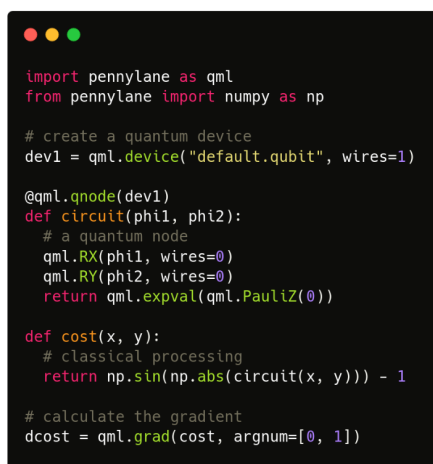
Figure 2: PennyLane as the connecting puzzle piece. [Source](#)

[PennyLane](#)([Bergholm 2018](#)) is created by [Xanadu](#) and presents itself as the **connecting puzzle piece between powerful established python libraries for machine learning, and quantum computing platforms**. It provides readily implemented quantum algorithms for quantum machine learning, quantum chemistry and optimization. A particular algorithm provided in PennyLane is automatic differentiation of

quantum circuits which allows to perform shot-efficient application of classical-quantum algorithms such as the *Variational Quantum Eigensolver* (VQE).

Existing popular quantum computing platforms such as [Qiskit](#) for qubits and [Strawberry Fields](#) (also created by *Xanadu*) for photons have written plugins for PennyLane which make their **quantum simulators or quantum hardware accessible through it** and with that PennyLane’s quantum algorithms and automatic differentiation feature.

In fact, **any quantum lab can adapt the plugin template for their own experiment and simulators** to assimilate their own specific workflow with PennyLane where users are generally able to use the same workflow accross platforms and know what to expect. Fig. 3 shows a typical way of using PennyLane in a classical-quantum hybrid algorithm. Here, the default qubit simulator by PennyLane is used. After writing our own plugin, we can call that device, change the quantum circuit to contain the supported operations and the workflow is otherwise the same. All-in-all, **the interface is very robust towards changing up the quantum backend** (especially when they support the same operations).



```
import pennylane as qml
from pennylane import numpy as np

# create a quantum device
dev1 = qml.device("default.qubit", wires=1)

@qml.qnode(dev1)
def circuit(phi1, phi2):
    # a quantum node
    qml.RX(phi1, wires=0)
    qml.RY(phi2, wires=0)
    return qml.expval(qml.PauliZ(0))

def cost(x, y):
    # classical processing
    return np.sin(np.abs(circuit(x, y))) - 1

# calculate the gradient
dcost = qml.grad(cost, argnum=[0, 1])
```

Figure 3: Typical simple way of using PennyLane by instantiating the device, defining the parametrized quantum circuit and the cost, and calculating the gradient for the classical parameters. [Source](#)

## Our workflow and connecting PennyLane to our hardware

The current state of our *pennylane\_ls* plugin for ultra-cold atom experiments can be viewed in [this public github repository](#). It provides examples from our labs and can guide you to adapting it to your own experiment and needs.

Our group uses the [Labscript suite](#) to coordinate and execute experimental sequences with cold atoms.

*The labscript suite is a collection of programs, which work together to form a control system for autonomous, hardware timed experiments.*

On the deepest level it is very nuts and bolts with plenty of commands in the style of Fig. 4 below.

```

191 #lets start the experiment!!!
192 start()
193 #initialise all the channels
194 Experiment.SetChannels()
195 #load the MOTs
196 Experiment.loadSodiumMOT(dt_RampMOT)
197 Experiment.t += dt_NaMOT
198 #take the picture of the MOT while loading using mako
199 mako.expose(Experiment.t-3,'MOTimage',trigger_duration=1e-3)
200 #start optical pumping to spin polarize the sample
201 Experiment.UMPUMP()

```

**Experiment.py**

```

2 def UMPUMP(self):
3     if use_Umpump == True:
4         D3_20_atomic_beam_shutter.go_low(self.t-100*1e-3)
5         D3_25_MOT_IGBT.go_low(self.t)
6         D2_3_Na_slower_AOM.go_low(self.t-5*1e-3)
7         D1_6_Na_repumper_shutter.close(self.t)
8         D1_4_slower_shutter.go_low(self.t)
9         A2_1_curvature_power_supply.constant(self.t,value = 0)
10        A2_6_Offset_Y_fast.constant(self.t-1e-3,value = 0)
11        A3_2_Offset_Y.constant(self.t-1e-3,value = 0)
12        A3_3_Offset_Z.constant(self.t-1e-3,value = 0)

```

**NaLiFunctions.py**

Figure 4: The *Experiment.py* (left) defines the experimental sequence of a Labscript-controlled lab experiment, here by calling functions from *NaLiFunctions.py* at given times. The function *UMPUMP* (right) shows the exact low level hardware instructions that are to be executed.

The *Experiment.py* file (Fig. 4) determines the exact sequence of hardware controls that is executed during the measurement. It contains Labscript syntax and is fed into the Labscript Python compiler *Runmanager*.

Our approach for combining Labscript with PennyLane is summarized in Fig. 5.

With the help of the plugin functions *pre\_apply()*, *apply()* and *post\_apply()*, we generate a new *Experiment.py* on the fly depending on the quantum circuit programmed through the PennyLane interface.

```

start()

## Begin of Preparation ##
Experiment.prepare_initial()
## End of Preparation ##

## Begin Sequence of Gates ##
Experiment.X(1e-05)
Experiment.H_mb(1)
## End Sequence of Gates ##

## Finishing ##
Experiment.finishing()

stop(Experiment.t+1)

```

Figure 5: The functions *pre\_apply()*, *apply()* and *post\_apply()* are used to generate a new *Experiment.py* on the fly for each quantum circuit in PennyLane. *apply()* loops through the operations in the circuit while the other functions always write the same necessary code for the compilation.

In other words, our PennyLane plugin translates a quantum circuit to an *Experiment.py* which our control system can interpret.

The *expval()* method from the plugin is then used to send the newly generated file to the *Runmanager* and engage the sequence. The output shots are then evaluated and the method returns an expectation.

A detailed discussion of the technical details can be found in the README.md of the [public repository](#). The **features of our plugin** currently include:

- **Proposing a higher-level control of experiments** that are run controlled by the Labscript Suite through PennyLane.
- A **framework device that provides a template for labs with Labscript** to quickly adapt code for their own setup.
- Including **two devices of the SynQS group at the Kirchhoff-Institute for Physics** of Prof. Fred Jendrzejewski.

The framework device in *SynQSDevice.py* is written such that **a group which uses the Labscript Suite** to run their experiment **can adapt the module easily to their own setup**.

Finishing up this project, we have **succeed in providing first circuit control to our experiments** through PennyLane and extract the results. However, much **further automation will be needed** to fully leverage the possibilities of the quantum circuits that we can run on our ultra-cold atom devices.

## Future steps

To get the project **towards a full PennyLane plugin**, we have to:

1. Give **wires** a meaning.
2. **Operations need to be restricted to certain wires**, especially when different atomic species are involved.
3. Implement **true remote control** of the lab through the *pennylane\_ls* module.
4. Implement **full result post-processing** that is currently used in the lab.
5. Implement some kind of **user management**.

While it seems quite clear how we can implement points 1 - 4, it seems to be a whole new problem to implement some kind of user management for our systems.

In any case the **first steps are remarkably promising** and we will see how the quantum circuits can be efficiently employed in ultra-cold atom experiments.

## References

- Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, M. Sohaib Alam, Shahnawaz Ahmed, Juan Miguel Arrazola, Carsten Blank, Alain Delgado, Soran Jahangiri, Keri McKiernan, Johannes Jakob Meyer, Zeyue Niu, Antal Száva, and Nathan Killoran. PennyLane: Automatic differentiation of hybrid quantum-classical computations. 2018. URL <https://arxiv.org/abs/1811.04968v3>. Accessed on Thu, May 28, 2020.
- H Bernien, S Schwartz, A Keesling, H Levine, A Omran, H Pichler, S Choi, AS Zibrov, M Endres, M Greiner, V Vuletić, and MD Lukin. Probing many-body dynamics on a 51-atom quantum simulator. *Nature*, 551: 579–584, Nov 2017.
- Immanuel Bloch, Jean Dalibard, and Wilhelm Zwerger. Many-body physics with ultracold gases. *Reviews of Modern Physics*, 80(3):885–964, jul 2008. doi: 10.1103/revmodphys.80.885. URL <https://doi.org/10.1103%2Frevmodphys.80.885>.
- N Goldman, G Juzeliūnas, P Öhberg, and I B Spielman. Light-induced gauge fields for ultracold atoms. *Reports on Progress in Physics*, 77(12):126401, nov 2014. doi: 10.1088/0034-4885/77/12/126401. URL <https://doi.org/10.1088%2F0034-4885%2F77%2F12%2F126401>.



- C. Kokail, C. Maier, R. van Bijnen, T. Brydges, M. K. Joshi, P. Jurcevic, C. A. Muschik, P. Silvi, R. Blatt, C. F. Roos, and P. Zoller. Self-verifying variational quantum simulation of lattice models. *Nature*, 569 (7756):355–360, may 2019. doi: 10.1038/s41586-019-1177-4. URL <https://doi.org/10.1038%2Fs41586-019-1177-4>.
- Ad Lagendijk, Bart van Tiggelen, and Diederik S. Wiersma. Fifty years of Anderson localization. *Physics Today*, 62(8):24–29, aug 2009. doi: 10.1063/1.3206091. URL <https://doi.org/10.1063%2F1.3206091>.
- Esteban A. Martinez, Christine A. Muschik, Philipp Schindler, Daniel Nigg, Alexander Erhard, Markus Heyl, Philipp Hauke, Marcello Dalmonte, Thomas Monz, Peter Zoller, and Rainer Blatt. Real-time dynamics of lattice gauge theories with a few-qubit quantum computer. *Nature*, 534(7608):516–519, jun 2016. doi: 10.1038/nature18318. URL <https://doi.org/10.1038%2Fnature18318>.
- Simon V. Mathis, Guglielmo Mazzola, and Ivano Tavernelli. Toward scalable simulations of Lattice Gauge Theories on quantum computers. 2020a. URL <https://arxiv.org/abs/2005.10271v1>. Accessed on Thu, May 28, 2020.
- Simon V. Mathis, Guglielmo Mazzola, and Ivano Tavernelli. Toward scalable simulations of Lattice Gauge Theories on quantum computers. 2020b. URL <https://arxiv.org/abs/2005.10271v1>. Accessed on Thu, May 28, 2020.
- A Mil, TV Zache, A Hegde, A Xia, RP Bhatt, MK Oberthaler, P Hauke, J Berges, and F Jendrzejewski. A scalable realization of local U(1) gauge invariance in cold atomic mixtures. *Science*, 367:1128–1130, Mar 2020. doi: 10.1126/science.aaz5312.
- C.A. Regal and D.S. Jin. Experimental Realization of the BCS-BEC Crossover with a Fermi Gas of Atoms. In *Advances In Atomic Molecular, and Optical Physics*, pages 1–79. Elsevier, 2007. doi: 10.1016/s1049-250x(06)54001-7. URL <https://doi.org/10.1016%2Fs1049-250x%2806%2954001-7>.
- Maria Schuld Christian Gogolin M. Sohaib Alam Shahnawaz Ahmed Juan Miguel Arrazola Carsten Blank Alain Delgado Soran Jahangiri Keri McKiernan Johannes Jakob Meyer Zeyue Niu Antal Száva Ville Bergholm, Josh Izaac and Nathan Killoran. PennyLane: Automatic differentiation of hybrid quantum-classical computations. 2018. URL <https://arxiv.org/abs/1811.04968v3>. Accessed on Thu, May 28, 2020.
- Maria Schuld Christian Gogolin M. Sohaib Alam Shahnawaz Ahmed Juan Miguel Arrazola Carsten Blank Alain Delgado Soran Jahangiri Keri McKiernan Johannes Jakob Meyer Zeyue Niu Antal Száva Ville Bergholm, Josh Izaac and Nathan Killoran. PennyLane: Automatic differentiation of hybrid quantum-classical computations. 2018. URL <https://arxiv.org/abs/1811.04968v3>. Accessed on Thu, May 28, 2020.
- E Zohar, JI Cirac, and B Reznik. Quantum simulations of lattice gauge theories using ultracold atoms in optical lattices. *Rep Prog Phys*, 79:014401, Jan 2016. doi: 10.1088/0034-4885/79/1/014401.