

# **LibKet: Cross-Platform Library for Running Quantum Algorithms on NISQ processors**

IEEE Quantum Week 2022  
September 18-23, 2022

**Matthias Möller<sup>1</sup> and Carmen G. Almudever<sup>2</sup>**

<sup>1</sup>Delft University of Technical (m.moller@tudelft.nl)

<sup>2</sup>Technical University of Valencia (cargara2@disca.upv.es)

# Who we are

## *Carmen G. Almudever*



Distinguished Researcher – Quantum Computing Architectures  
Computer Engineering Department, Technical University of  
Valencia (UPV)

- Quantum software, mapping of quantum algorithms
- QEC and FT quantum computation
- Architecting and benchmarking of quantum computers
- Scalability of quantum computing systems

# Who we are

## *Matthias Möller*



Associate Professor of Numerical Analysis,  
Department of Applied Mathematics  
Delft University of Technology, NL

- Heterogeneous high-performance scientific computing
- Quantum software and early applications
- Quantum machine learning

# Funding and support





<https://tinyurl.com/3vw4zdc8>



## Tutorial at IEEE QCE22, September 18-23, 2022

LibKet: A Cross-Platform Library for Running Quantum Algorithms on NISQ Processors

Organizers: Carmen G. Almudever, Matthias Möller

### Session 1: Sunday, September 18, 10:00 AM – 11:30 AM MDT (UTC-6)

Time	Content	Lecturer	Slides	Binder
10:00-11:00 am	Hands-on Introduction to Quantum Computing	Carmen	<a href="#">slides</a>	tutorial 01
11:00-11:30 am	Libket - The Basics	Matthias	<a href="#">slides</a>	tutorial 02

### Session 2: Sunday, September 18, 12:00 AM – 1:30 PM MDT (UTC-6)

Time	Content	Lecturer	Slides	Binder
1:00-1:45 pm	LibKet - Advanced Features	Matthias	<a href="#">slides</a>	tutorial 03
1:45-2:30 pm	Variational Quantum Algorithms	Carmen/Matthias	<a href="#">slides</a>	tutorial 04

Click **now** since first launch can take some time



<https://tinyurl.com/3vw4zdc8>



Starting repository: mmoelle1/LibKet/master

Your session is taking longer than usual to start!  
Check the log messages below to see what is happening.

Build logs

[hide](#)

```
---> 29bf3cb5b44f
Step 48/51 : RUN ./postBuild
---> Running in 9345aed0e8bf
Cloning into '/home/jovyan/LibKet'...
warning: redirecting to https://gitlab.com/mmoelle1/LibKet.git/
Submodule 'external/OpenQL' (https://github.com/QE-Lab/OpenQL.git) registered for path 'external/OpenQL'
Submodule 'external/QuEST' (https://github.com/QuEST-Kit/QuEST.git) registered for path 'external/QuEST'
Submodule 'external/armadillo' (https://gitlab.com/conradsnicta/armadillo-code.git) registered for path 'external/armadillo'
Submodule 'external/optim' (https://github.com/kthohr/optim.git) registered for path 'external/optim'
Submodule 'external/pegtl' (https://github.com/taocpp/PEGTL.git) registered for path 'external/pegtl'
```



<https://tinyurl.com/3vw4zdc8>

jupyter qce22-tutorial01 (autosaved)

Visit repo

Copy Binder link

FileEditViewInsertCellKernelWidgetsHelp

Not TrustedPython 3 (ipykernel)

Markdown

Memory: 154 MB / 2 GB

Tutorial #1: Hands-on Introduction to Quantum Computing

In this first part of the tutorial you will learn the fundamentals of quantum computing including superposition of states, destructive measurement, entanglement, quantum gates and circuits. You will be able to understand and calculate the output of a quantum circuit.

Getting started

To get a better understanding of how quantum computing works from the mathematical point of view we implement a couple of quantum gates by hand and investigate how they act on quantum states. This is all done using the Python package `numpy` whose functionality we make available under the alias `np` with the following command

```
In [2]: import numpy as np  
import math
```

Let us represent the single-qubit state

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad |\alpha|^2 + |\beta|^2 = 1$$

by its two complex-valued coefficients  $\alpha$  and  $\beta$  of the state vector as a vector.

As we just shown, the quantum state  $|0\rangle$  is represented by the vector  $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ , whereas the quantum state  $|1\rangle$  corresponds to the vector  $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ , which can be defined as: