Development of an open-source calibration framework for superconducting qubits

Master's degree in Physics

Candidate:
Elisa Stabilini
28326A

July 4th 2025

Università degli Studi di Milano - Department of Physics

Supervisor:
Prof. Dr. Stefano Carrazza
Co-supervisors:
Dr. Alessandro Candido
Dr. Andrea Pasquale
Dott. Edoardo Pedicillo

Table of contents

1. Superconducting qubits

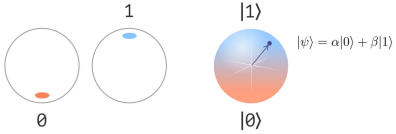
2. Quantum Middleware

3. Average Clifford gate fidelity optimization

4. New features in Qibo

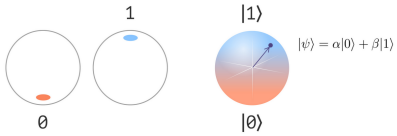
5. Conclusions & Outlooks

Quantum Computing, what for?



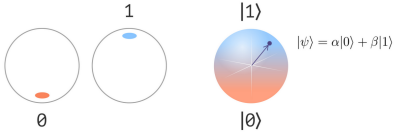
Quantum Computing, what for?

- Simulation of quantum system: classical computers can not efficiently simulate quantum staes. Classical computers can usally simulate a $\sim$ 30 qubits



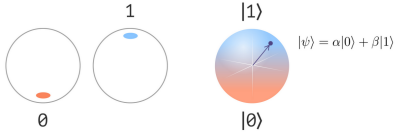
Quantum Computing, what for?

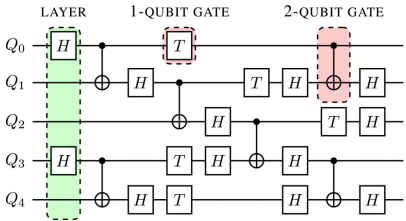
- Simulation of quantum system: classical computers can not efficiently simulate quantum staes. Classical computers can usally simulate a ~ 30 qubits
- Optimization and modeling (eg. direct annealing, adiabatic evolutions, QAOA) with applications in finance, traffic, weather...



Quantum Computing, what for?

- Simulation of quantum system: classical computers can not efficiently simulate quantum staes. Classical computers can usally simulate a ~ 30 qubits
- Optimization and modeling (eg. direct annealing, adiabatic evolutions, QAOA) with applications in finance, traffic, weather...
- Gate-based circuits (eg. Shor, Grover, Deutsch)

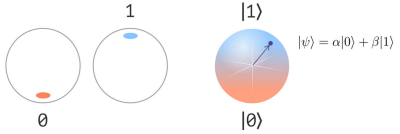


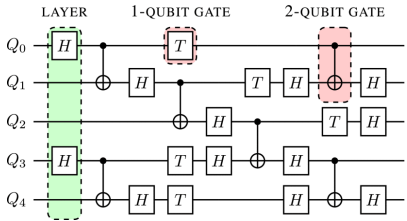


DOI: 10.1109/TQE.2021.3053921

Quantum Computing, what for?

- Simulation of quantum system: classical computers can not efficiently simulate quantum staes. Classical computers can usally simulate a ~ 30 qubits
- Optimization and modeling (eg. direct annealing, adiabatic evolutions, QAOA) with applications in finance, traffic, weather...
- Gate-based circuits (eg. Shor, Grover, Deutsch)
- AI-inspired algorithms (VQE, autoencoders, classifiers ...)





DOI: 10.1109/TQE.2021.3053921

Superconducting qubits

Artificial atoms

[arXiv:cond-mat/0703002]

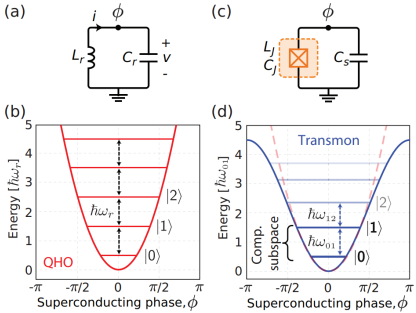
Qubit: two level quantum system

Artificial atoms

[arXiv:cond-mat/0703002]

Qubit: two level quantum system

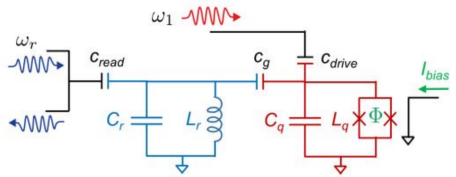
Superconducting qubits: use Josephson Junctions to build anharmonic oscillators

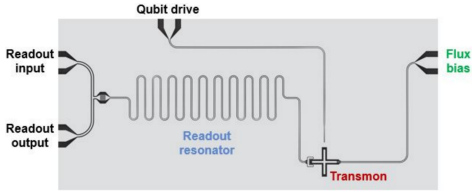


DOI: 10.1109/MAP.2022.3176593

Qubit control

[arXiv:2403.04383]





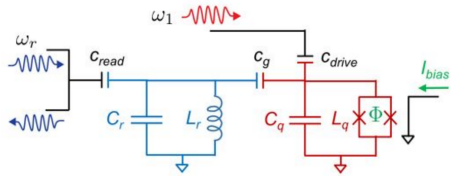
DOI: 10.1109/MAP.2022.3176593

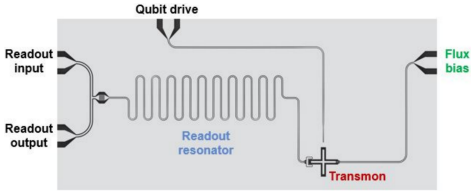
Qubit control

[arXiv:2403.04383]

Microwave pulse applied to the qubit

$$V_d(t) = A\varepsilon(t)\sin(\omega_d t + \alpha);$$





DOI: 10.1109/MAP.2022.3176593

Qubit control

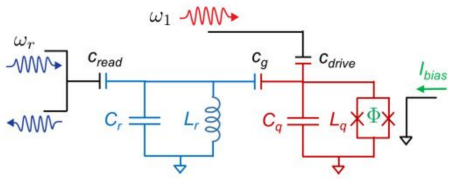
[arXiv:2403.04383]

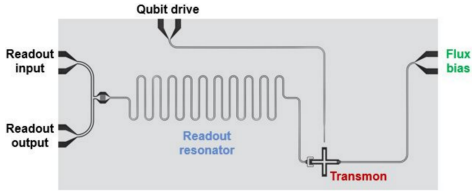
Microwave pulse applied to the qubit

$$V_d(t) = A\varepsilon(t)\sin(\omega_d t + \alpha);$$

Qubit - electric field Hamiltonian (RWA)

$$\hat{H} = -\frac{\hbar(\omega_q - \omega_d)}{2}\hat{\sigma}_z + \frac{\hbar\Omega}{2}\varepsilon(t)\left(\hat{\sigma}_x\cos\alpha + \hat{\sigma}_y\sin\alpha\right);$$





DOI: 10.1109/MAP.2022.3176593

Qubit control

[arXiv:2403.04383]

Microwave pulse applied to the qubit

$$V_d(t) = A\varepsilon(t)\sin(\omega_d t + \alpha);$$

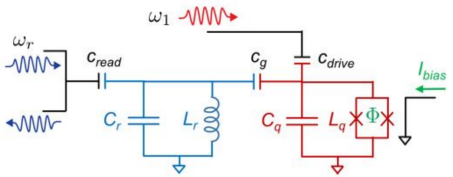
Qubit - electric field Hamiltonian (RWA)

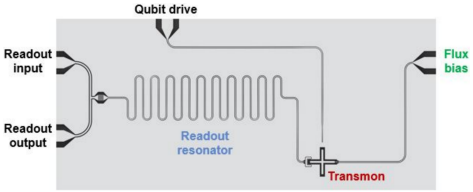
$$\hat{H} = -\frac{\hbar(\omega_q - \omega_d)}{2}\hat{\sigma}_z + \frac{\hbar\Omega}{2}\varepsilon(t)\left(\hat{\sigma}_x\cos\alpha + \hat{\sigma}_y\sin\alpha\right);$$

Qubit evolution under microwave pulse:

$$R_{\hat{n}(\alpha)}(\theta) = e^{-\frac{i}{2}\hat{n}(\alpha)\cdot\vec{\sigma}\theta} = e^{-\frac{i}{2}(\hat{\sigma}_x\cos\alpha + \hat{\sigma}_y\sin\alpha)\theta};$$

where $\theta = \Omega\int_0^{+\infty}\varepsilon(t')dt'$.





DOI: 10.1109/MAP.2022.3176593

State readout

[arXiv:cond-mat/0407325]

Qubit - resonator Hamiltonian:

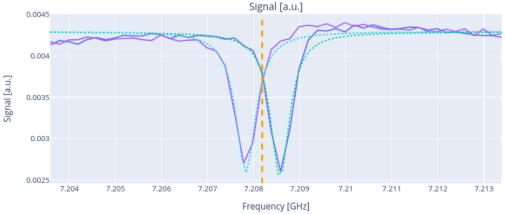
$$\hat{H} = \hbar\omega_r \hat{a}\hat{a}^\dagger - \frac{\hbar\omega_{01}}{2}\hat{\sigma}_z + \hbar g(\hat{\sigma}^+\hat{a} + \hat{\sigma}^-\hat{a}^\dagger);$$

State readout

[arXiv:cond-mat/0407325]

Qubit - resonator Hamiltonian:

$$\hat{H} = \hbar\omega_r \hat{a}\hat{a}^\dagger - \frac{\hbar\omega_{01}}{2}\hat{\sigma}_z + \hbar g(\hat{\sigma}^+\hat{a} + \hat{\sigma}^-\hat{a}^\dagger);$$



Dispersive regime ($g \ll \omega_q - \omega_r$):

$$\hat{H}_{disp} = \hbar(\omega_r - \chi\hat{\sigma}_z)\hat{a}^\dagger\hat{a} - \frac{\hbar}{2}(\omega_{01} + \chi)\hat{\sigma}_z;$$

dispersive shift: $\chi = \frac{g^2}{\Delta}$,
$\Delta = \omega_q - \omega_r$.

State readout

[arXiv:cond-mat/0407325]

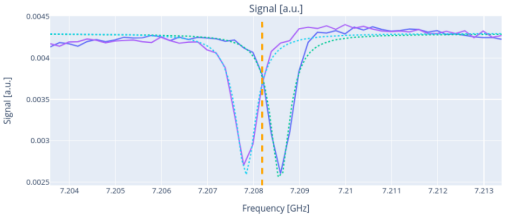
Qubit - resonator Hamiltonian:

$$\hat{H} = \hbar\omega_r \hat{a}\hat{a}^\dagger - \frac{\hbar\omega_{01}}{2}\hat{\sigma}_z + \hbar g(\hat{\sigma}^+\hat{a} + \hat{\sigma}^-\hat{a}^\dagger);$$

Dispersive regime ($g \ll \omega_q - \omega_r$):

$$\hat{H}_{disp} = \hbar(\omega_r - \chi\hat{\sigma}_z)\hat{a}^\dagger\hat{a} - \frac{\hbar}{2}(\omega_{01} + \chi)\hat{\sigma}_z;$$

dispersive shift: $\chi = \frac{g^2}{\Delta}$,
$\Delta = \omega_q - \omega_r$.



Quantum Middleware

Superconducting qubit calibration

Circuit: sequence of quantum gates applied to qubits to perform a computation.

Circuit with
logical gates

Superconducting qubit calibration

Circuit: sequence of quantum gates applied to qubits to perform a computation.

Gates: abstract unitary operations (logical or native).

We need to calibrate *only* native gates.

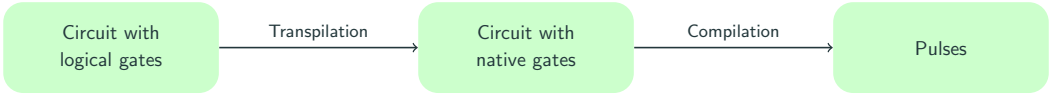
Native single qubit gates on superconducting qubits: $R_X(\pi)$, $R_X(\frac{\pi}{2})$, $R_Z(\theta)$.

Superconducting qubit calibration

Circuit: sequence of quantum gates applied to qubits to perform a computation.

Gates: abstract unitary operations (logical or native).

Pulses: physical signal (control field) implementing a gate.



We need to calibrate *only* native gates.

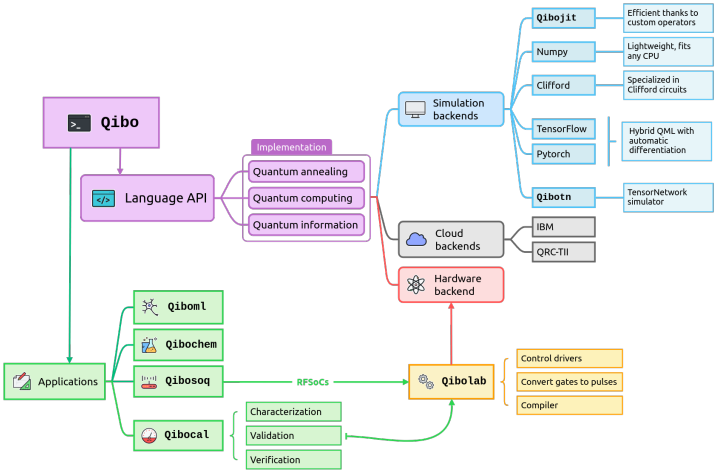
Native single qubit gates on superconducting qubits: $R_X(\pi)$, $R_X(\frac{\pi}{2})$, $R_Z(\theta)$.

For each pulse we calibrate frequency, duration, power, shape.

Qibo framework

[arXiv:2308.06313]

Qibo: a modular and open-source framework for quantum computing, control and calibration.



https://qibo.science/

Average Clifford gate fidelity optimization

Randomized Benchmarking

[arXiv:0707.0963v1]

Randomized benchmarking estimates average gate fidelity by applying random sequences of Clifford gates followed by an inverting gate.

Randomized Benchmarking

[arXiv:0707.0963v1]

Randomized benchmarking estimates average gate fidelity by applying random sequences of Clifford gates followed by an inverting gate.

Randomization with Clifford gates provides a depolarized noise channel:
$\rho \to d\frac{\mathbb{I}}{2} + (1-d)\rho$.

Randomized Benchmarking

[arXiv:0707.0963v1]

Randomized benchmarking estimates average gate fidelity by applying random sequences of Clifford gates followed by an inverting gate.

Randomization with Clifford gates provides a depolarized noise channel:
$\rho \to d\frac{\mathbb{I}}{2} + (1-d)\rho$.

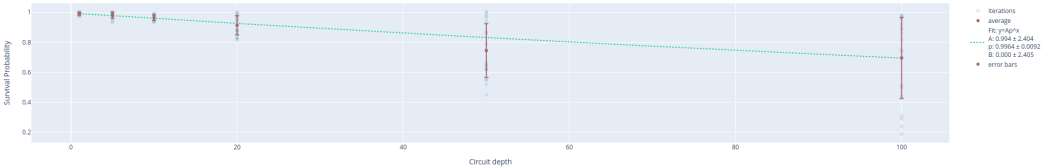
Randomized Benchmarking protocol:

1. Initialize the system in the ground state
2. For each sequence length m draw a sequence of Clifford group elements
3. Calculate the inverse gate
4. Measure complete sequence
5. Repeat the process for multiple sequences of the same length while varying the length

Randomized Benchmarking

[arXiv:0707.0963v1]

The survival probability decays exponentially with the number of Clifford gates $F(m) = Ap^m + B$ where $1 - p$ is the depolarization rate.

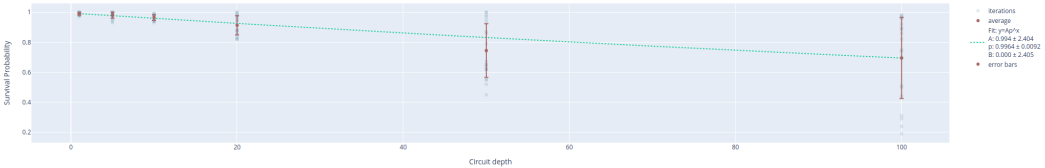


Randomized Benchmarking

[arXiv:0707.0963v1]

The survival probability decays exponentially with the number of Clifford gates $F(m) = Ap^m + B$ where $1 - p$ is the depolarization rate.

From p we can extract the average error per Clifford gate.

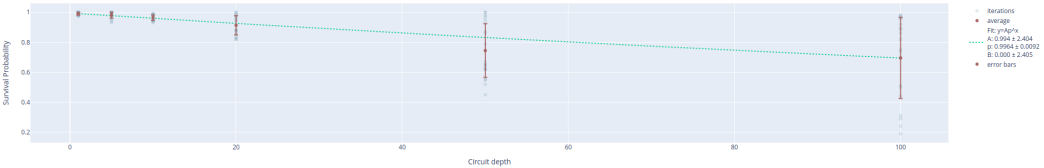


Randomized Benchmarking

[arXiv:0707.0963v1]

The survival probability decays exponentially with the number of Clifford gates $F(m) = Ap^m + B$ where $1 - p$ is the depolarization rate.

From p we can extract the average error per Clifford gate.



Can we optimize the average Clifford gate fidelity to automate $R_X(\pi)$ gate recalibration?

RB optimization

[arXiv:1403.0035]

Test closed-loop optimization with modern optimization libraries.

Optimization parameters:

- Pulse amplitude
- Pulse frequency
- Pulse shape (through β DRAG parameter)

RB optimization

[arXiv:1403.0035]

Test closed-loop optimization with modern optimization libraries.

Pre-calibrated platform

Optimization parameters:

- Pulse amplitude
- Pulse frequency
- Pulse shape (through β DRAG parameter)

RB optimization

[arXiv:1403.0035]

Test closed-loop optimization with modern optimization libraries.

Optimization parameters:

- Pulse amplitude
- Pulse frequency
- Pulse shape (through β DRAG parameter)

RB optimization

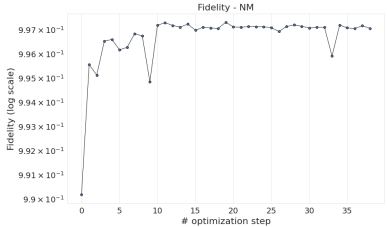
[arXiv:1403.0035]

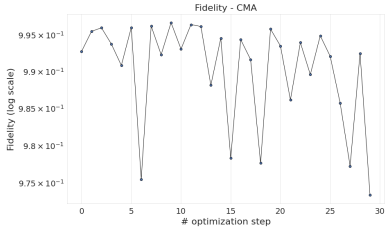
Test closed-loop optimization with modern optimization libraries.

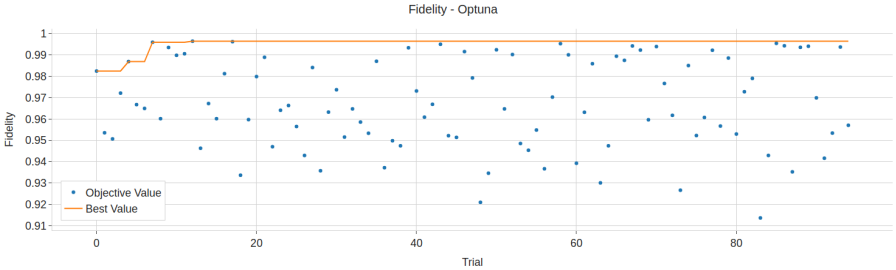
Optimization parameters:

- Pulse amplitude
- Pulse frequency
- Pulse shape (through β DRAG parameter)

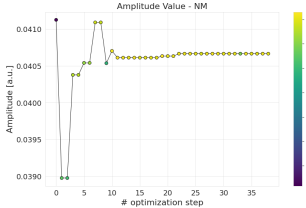
Average Clifford gate Fidelity

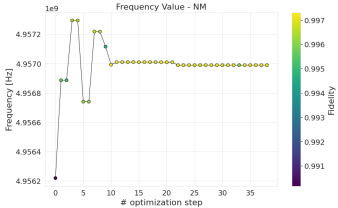


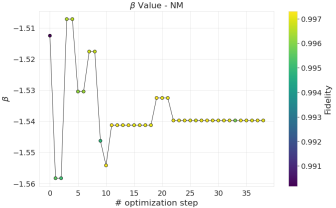


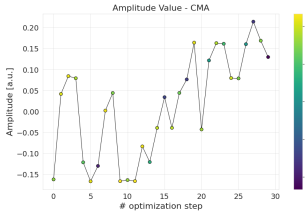


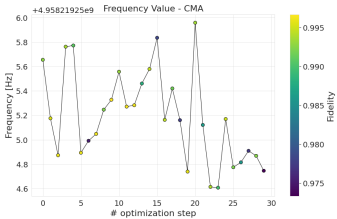
Parameters evolution

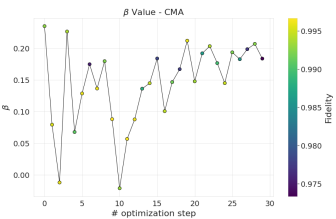












New features in Qibo

Improving Qubit Control for Calibration

Single qubit gates:

Qibo already implement many calibration routines for $R_X(\pi)$.

¡3-¿Introduce support for $R_X(\pi/2)$ as native gate in Qibolab.

Improving Qubit Control for Calibration

Single qubit gates:

Qibo already implement many calibration routines for $R_X(\pi)$.

We can try to reduce residual erros by adding calibrating $R_X(\pi/2)$ as native gate.

¡3-¿Introduce support for $R_X(\pi/2)$ as native gate in Qibolab.

Native RX90

Qibolab single-qubit native gates are $R_X(\pi)$ and MZ.

Native RX90

Qibolab single-qubit native gates are $R_X(\pi)$ and MZ.

$R_X(\frac{\pi}{2})$ gate is implemented by halving the calibrated values for $R_X(\pi)$.

Native RX90

Qibolab single-qubit native gates are $R_X(\pi)$ and MZ.

$R_X(\frac{\pi}{2})$ gate is implemented by halving the calibrated values for $R_X(\pi)$.

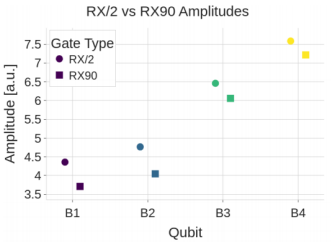
Problem: there may be nonlinearities and distortions in signals that degradete the $R_X(\frac{\pi}{2})$ fidelity.

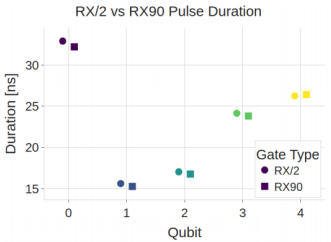
Native RX90

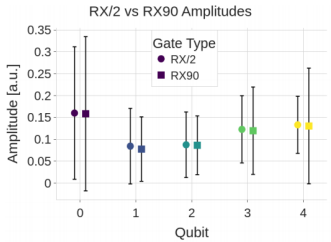
Qibolab single-qubit native gates are $R_X(\pi)$ and MZ.

$R_X(\frac{\pi}{2})$ gate is implemented by halving the calibrated values for $R_X(\pi)$.

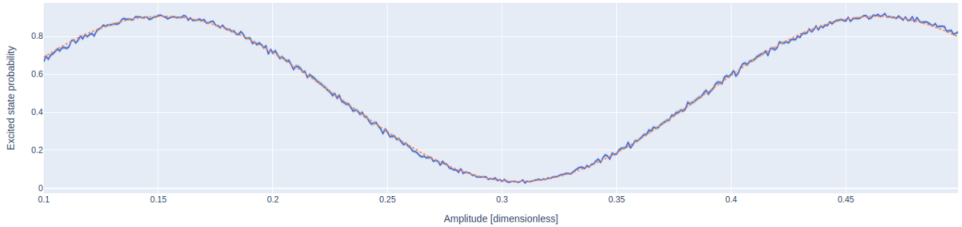
Problem: there may be nonlinearities and distortions in signals that degradete the $R_X(\frac{\pi}{2})$ fidelity.



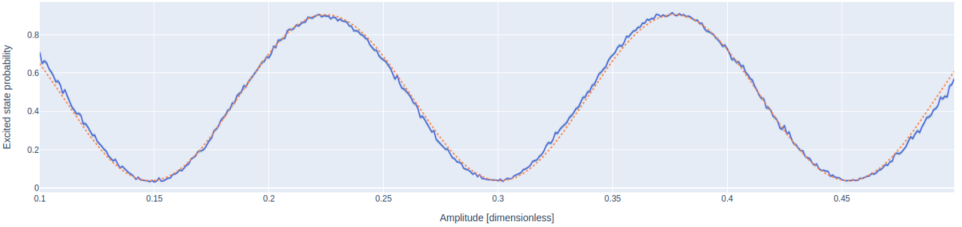




Rabi amplitude experiment



$R_X(\pi)$



$R_X\left(\frac{\pi}{2}\right)$

Improving Qubit Control for Calibration

Single qubit gates:

Qibo already implement many calibration routines for $R_X(\pi)$.

We can try to reduce residual erros by adding directly calibrating $R_X(\pi/2)$.

Introduce support for $R_X(\pi/2)$ as native gate in Qibolab.

Two qubit gates:

¡4-¿Add Cryoscope portocol to improve control over flux pulses.

Improving Qubit Control for Calibration

Single qubit gates:

Qibo already implement many calibration routines for $R_X(\pi)$.

We can try to reduce residual erros by adding directly calibrating $R_X(\pi/2)$.

Introduce support for $R_X(\pi/2)$ as native gate in Qibolab.

Two qubit gates:

Two-qubit gates typically require bringing qubits into resonance.

¡4-¿Add Cryoscope portocol to improve control over flux pulses.

Improving Qubit Control for Calibration

Single qubit gates:

Qibo already implement many calibration routines for $R_X(\pi)$.

We can try to reduce residual erros by adding directly calibrating $R_X(\pi/2)$.

Introduce support for $R_X(\pi/2)$ as native gate in Qibolab.

Two qubit gates:

Two-qubit gates typically require bringing qubits into resonance.

Frequency tuning is achieved via flux pulses applied to tunable qubits.

¡4-¿Add Cryoscope portocol to improve control over flux pulses.

Flux pulse reconstruction

[arXiv:1907.04818]

Transmon frquency dependence on magnetic flux:

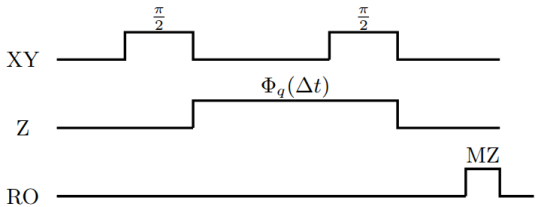
$$f_q(\Phi_q) \approx \left(\sqrt{8 E_J E_C \left| \cos\left(\pi \frac{\Phi_q}{\Phi_0} \right) \right|} \right).$$

Flux pulse reconstruction

[arXiv:1907.04818]

Transmon frquency dependence on magnetic flux:

$$f_q(\Phi_q) \approx \left(\sqrt{8E_J E_C \left| \cos\left(\pi \frac{\Phi_q}{\Phi_0} \right) \right|} \right).$$

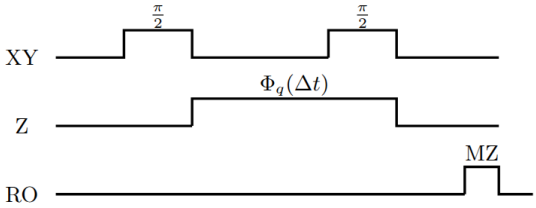


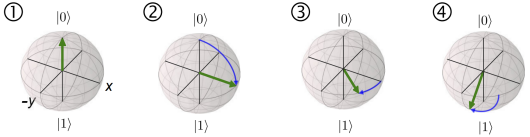
Flux pulse reconstruction

[arXiv:1907.04818]

Transmon frquency dependence on magnetic flux:

$$f_q(\Phi_q) \approx \left(\sqrt{8E_J E_C \left| \cos\left(\pi \frac{\Phi_q}{\Phi_0} \right) \right|} \right).$$





DOI: 10.1039/D2TC01258H

Flux pulse reconstruction

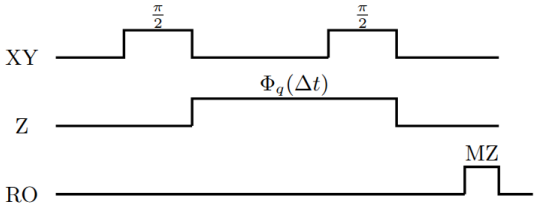
[arXiv:1907.04818]

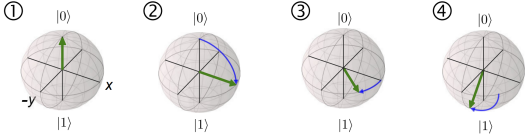
Transmon frquency dependence on magnetic flux:

$$f_q(\Phi_q) \approx \left(\sqrt{8E_J E_C \left| \cos\left(\pi \frac{\Phi_q}{\Phi_0} \right) \right|} \right).$$

Detuning as function of the flux pulse:

$$\Delta f_q = \frac{\varphi_{\tau+\Delta\tau} - \varphi_\tau}{2\pi} \approx \frac{1}{\Delta\tau} \int_\tau^{\tau+\Delta\tau} \Delta f_q(\Phi_{q,\tau+\Delta\tau}(t)).$$





DOI: 10.1039/D2TC01258H

Filter determination

1. Determine exponential correction

Filter determination

1. Determine exponential correction
2. Obtain coefficients for the Infinite Impulse Response from exponential correction

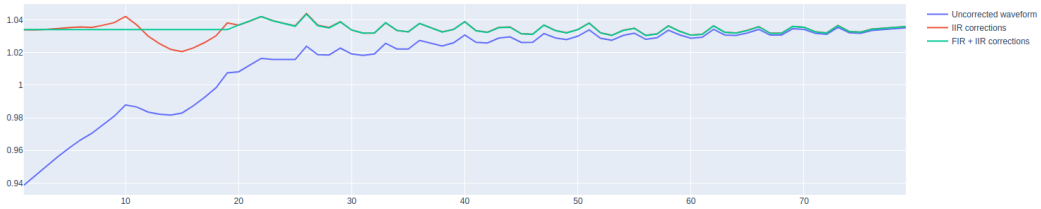
Filter determination

1. Determine exponential correction
2. Obtain coefficients for the Infinite Impulse Response from exponential correction
3. Determine Finite Impulse Response coefficients

Filter determination

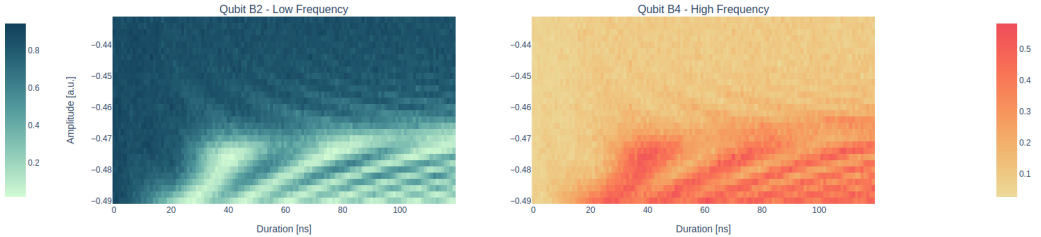
1. Determine exponential correction
2. Obtain coefficients for the Infinite Impulse Response from exponential correction
3. Determine Finite Impulse Response coefficients
4. Obtain coefficients for real-time correction

Filter determination

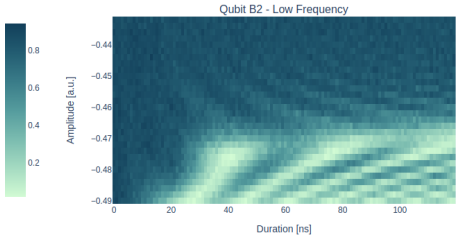


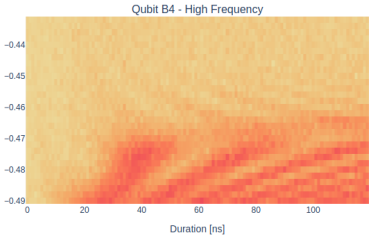
1. Determine exponential correction
2. Obtain coefficients for the Infinite Impulse Response from exponential correction
3. Determine Finite Impulse Response coefficients
4. Obtain coefficients for real-time correction

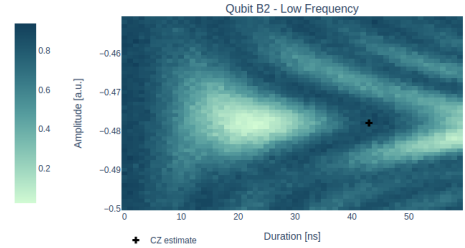
Impact of correction on chevron plots

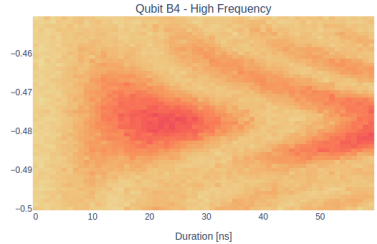


Impact of correction on chevron plots









Cryoscope routine

`relaxation_time`: `float`

Wait time for the qubit to decohere back to the *gnd* state.

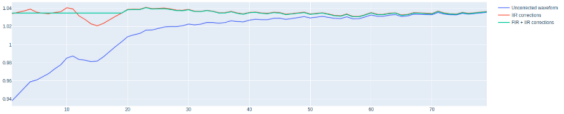
Example

A possible runcard to launch a Cryoscope experiment could be the following:

```
- id: cryoscope

  operation: cryoscope
  parameters:
    duration_max: 80
    duration_min: 1
    duration_step: 1
    flux_pulse_amplitude: 0.7
    relaxation_time: 50000
```

The expected output is the following:



Note

In the case where there are no filters the protocol will compute the FIR and the IIR filters. If the filters are already present the computation of the filters will be skipped and only the reconstructed waveform will be shown.

Requirements

- Single Shot Experiments

ON THIS PAGE

Cryoscope

Parameters

Example

Requirements

Conclusions & Outlooks

Gate recalibration by RB optimization

- ✓ We can always find parameters configuration with average Clifford gate fidelity $> 99.5\%$.
- ✓ Does **not rely on manual tuning** by an operator.
- ✗ RB evaluation is **computationally expensive** ($\sim$ 30 minutes).
- ✗ More stable methods (eg. Nelder-Mead) require many cost function evaluation.
- ✗ Parameters drift makes optimization unstable.

Possible future work: **optimize RB parameters** to allow a faster and more reliable optimization.

Library additions

- ✓ Extended Qibolab and Qibocal libraries to support **native** $R_X(\pi/2)$ **gates** with dedicated calibration routines.
- ✓ Implemented and added the **Cryoscope** calibration experiment to Qibocal library to correct flux-pulse distortions (average NMSE improvement $\sim 70\%$).
- ✗ Study long-time distortions of the flux-pulse.

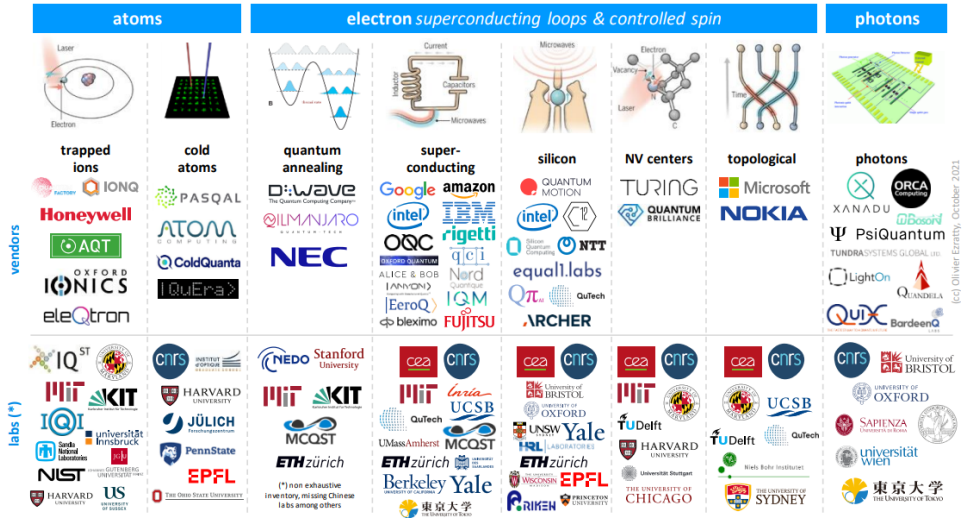
Possible future extensions:

- Readout optimization protocols
- Active qubit reset schemes
- Implement leakage mitigation strategies

Thank you

Backup slides

Qubit platforms



Calibration workflow & assessment

Procedure:

1. Resonator characterization
2. Qubit characterization
3. Gate calibration
4. Gate set characterization

Metrics example:

- readout & assignment fidelity
- relaxation time T_1
- decoherence time $T_2 2$
- gate fidelity

Calibration workflow & assessment

Main steps:

1. Resonator characterization
2. Qubit characterization
3. Gate calibration
4. Gate set characterization

Calibration quality metrics:

- readout & assignment fidelity
- relaxation time T_1
- decoherence time $T_2 2$
- gate fidelity

Qubit	**Readout Fidelity**	**Assignment Fidelity**	**T1** [μs]	**T2** [μs]	**Gate infidelity** ($\cdot 10^{-3}$)
D1	0.876	0.938	26.4 ± 0.4	13.0 ± 2.0	27.0 ± 21.0
D2	0.945	0.973	14.9 ± 0.1	18.0 ± 10.0	20.5 ± 6.2
D3	0.905	0.952	23.6 ± 0.3	28.0 ± 12.0	7.0 ± 18
D4	0.929	0.964	20.6 ± 0.2	38.0 ± 5.2	4.4 ± 4.8
B2	0.902	0.951	17.5 ± 0.2	27.5 ± 5.2	18.0 ± 16

Clifford gates

- Special subset of quantum gates that map Pauli operators to Pauli operators under conjugation
- Clifford gates group is generated by H, S, ($CNOT$) gates
- Quantum circuits that consist of only Clifford gates can be efficiently simulated with a classical computer (Gottesman–Knill theorem)

Hadamard (H)

$$\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}$$

Phase (S, P)

$$\begin{bmatrix}1 & 0\\ 0 & i\end{bmatrix}$$

Controlled Not (CNOT, CX)

$$\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix}$$

RB optimization summary

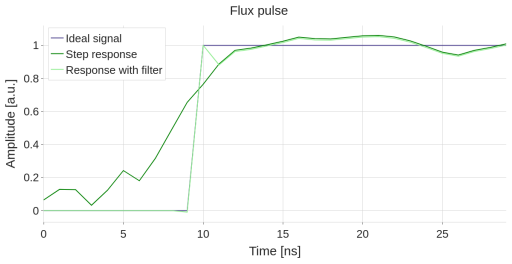
Analysis Name	Iterations	Time [s]	Fidelity Best	Amplitude Best [a.u.]	Frequency Best [$\times 10^9$ Hz]	β Best
Nelder_Mead	40	2892	0.99731	0.04063	4.95701	-1.53253
init_simplex_1	40	3096	0.99533	0.04069	4.95820	-
init_simplex_2	40	3128	0.99564	0.04131	4.95817	-
init_simplex_3	40	3190	0.99554	0.04058	4.95819	-
SLSQP	15	3873	0.99518	0.04058	4.95822	-0.00115
CMA-ES	30	4536	0.99665	-0.16634	4.95822	0.08802
optuna_30	30	763	0.99583	0.28059	4.95601	2.75519
optuna_100	100	2257	0.99645	0.29026	4.955279	-1.46504
optuna_1000	1000	21811	0.99847	-0.00051	4.961836	0.2222
optuna no β	1000	21820	0.99846	0.04104	4.957374	-

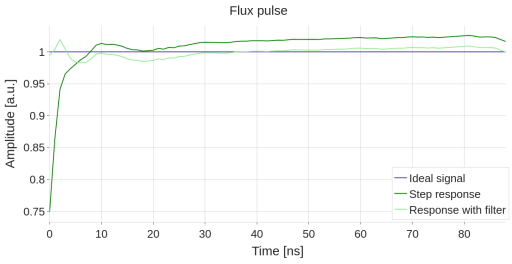
Filter determination details

Compensate distortion using a digital IIR filter based on an inverted exponential model

$$s_{\text{corr}}(t) = \frac{s(t)}{g(1 - Ae^{-t/\tau})}$$

parameters g, A, and τ are estimated via least-squares minimization using `scipy.optimize`





Filter determination details

Digital IIR filter implemented in control hardware using discretized coefficients.

$$a_0 y[n] = \sum_{i=0}^{N} b_i x[n-i] - \sum_{i=1}^{M} a_i y[n-i],$$

Filter coefficients derived from fitted parameters (1st order IIR):

$$\begin{aligned} a_0 &= 1, \\ a_1 &= -(1-\alpha), \\ b_0 &= 1 - k + k\alpha, \\ b_1 &= -(1-k)(1-\alpha), \end{aligned}$$

where $\alpha = 1 - \exp\left(-\frac{1}{f_s \cdot \tau(1+A)}\right)$ and

$$k = \begin{cases} \frac{A}{(1+A)(1-\alpha)}, & \text{if } A < 0, \\ \frac{A}{1+A-\alpha}, & \text{if } A \geq 0, \end{cases}$$

FIR filters:

$$y[n] = \sum_{i=0}^{N} b_i x[n-i],$$

FIR filter coefficients are optimized using CMA-ES to minimize the average relative deviation between the FIR-filtered IIR-corrected signal and the ideal step response.