QCHack 2021 - Q-CTRL Challenge

Team: Fizziqs

To implement a high fidelity NOT gate and Hadamard gate on the Q-CTRL qubit, we focused on four aspects: calibration of the qubit to determine the Rabi rate, parametrizing the pulse, smoothing the input pulse, and optimizing the controls through a closed-loop optimizer.

1. Calibration/Determination of Rabi Rate

To determine the Rabi rate, which maps the hardware amplitudes to the control pulse amplitudes, we performed a Rabi experiment for multiple input amplitudes (between 0 and 1). This was done with the help of the Opal Boulder tutorial on pulse calibration. For example, for the input amplitude 0.7, we sent varying durations of a constant rectangular pulse to the QCHack cloud qubit and then recorded the qubit population (figure 1). By fitting to the following cosine function, $A\cos(2\pi\Omega t + \phi)^2$, we can determine the Rabi rate for the associated input amplitude.

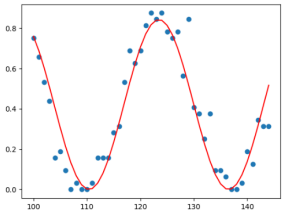

Figure 1: Qubit population over time (ns) for input amplitude 0.7.

After repeating for multiple input amplitudes, we find a linear relationship between the input amplitude and the Rabi rate.

$$rabi\,rate = 0.04787 * input\,amplitude + 0.002594$$

This allows us to create an initial pulse to send to the qubit from which we can then optimize. The initial pulse was determined using the rabi rate found as well as the Opal Boulder section on creating ideal plot pulses.

Ω_{NOT}:0.050466, duration=62.2514

Ω_H:-0.050466j, duration=31.12568

0.050466, duration=31.12568

0.050466, duration=31.12568

2. Pulse Parameterization

We split each control pulse into 18 complex control points with one additional parameter dictating the length of pulse. Since we use signal concatenation the duration of both pulses what set to the same number. This gives a parameter space of 18*2+1 = 37 real numbers in [-1,1]

3. Pulse Smoothing

The pulse from the pulse parameterization is resampled at a higher sample rate. It then performs a convolution of the discretized input pulse with the Hann window function of the time period of the pulse in order to create a smoother pulse curve and prevent large discontinuous jumps of the signal. We then apply a Planck-taper window filter to the pulse to smoothly bring the two endpoints to 0. This allows smooth concatenation of control pulses (see 4).

4. Controller optimization.

We chose to implement the closed-loop automated optimization approach because of the overall lack of understanding of the qubit system. We used Q-CTRL's closed-loop optimizer using a series of test controls to randomly generate target states which we can use to evaluate with the measurements. We decided to optimize both types of gates simultaneously, since we use gate composition to determine their performance.
We chose 5 input controls: the first two are the test NOT and Hadamard gates, and the last three are pulses of *n* sequential gates, where *n* is randomly chosen between 2 and 7.

Control pulse targets	**Example: n = 5**	**Expected P(|1>)**
1. A single NOT gate	N	1
2. A single Hadamard gate	H	0.5
3. A series of *n* NOT gates	N N N N N	1
4. A series of *n* Hadamard gates	H H H H H	0.5
5. A series of *n* NOT and Hadamard gates, ordered randomly	N H H N H	0.5

Table 1: Test suite of control pulses used in optimizer to generate infidelity values of NOT and Hadamard gates and their expected results

Note we choose to look for the Θ=0 version of the gates for ease of computation.

The error is then the absolute value difference between the probability of one in the measurement and the probability of one in the final ideal state, divided by *n* to normalize the error. From the entire test suite, the overall infidelity value of the NOT and Hadamard gates is the mean error from these 5 tests.

Overall we tried the gaussian process and cross entropy optimizers on Boulder Opal, due to the response time of the qubit (3-15s). They performed similarly with a final mean error of 0.05 across the tests.