Optimizer

The problem

- The execution of the circuit is accompanied by errors. We would like to reconstruct the circuit in order to reduce it.

- Obviously, the order of the gates that we have does not uniquely define the execution of a main unitary operation. However, finding the best order from this unitary is a difficult problem.
- We have some $transformations$ on two or three gates that do not change the main unitary, but they can help to improve the fidelity of the circuit.
- Let's apply these transformations and see how they affect the $function$ of our circuit.

[The main article (RL) (https://arxiv.org/pdf/2103.07585.pdf)](https://arxiv.org/pdf/2103.07585.pdf)

The function

- Obviously, the great function is a **fidelity** between the ideal circuit and the noisy circuit. But it is difficuilt to solve it on a large number of qubits.
- For inspiration [of this article (https://arxiv.org/abs/2306.15020)](https://arxiv.org/abs/2306.15020) we can assume that the accuracy of a similar scheme consisting only of gates of the Clifford group can be correlated with the accuracy of a given circuit
- A simple function is a length of the circuit.

Native gates

The simpest form of transformation can be written in basis set - [RX, RZ, CZ]

Transformations

As in the main article, we have two types of transformations:

1) Instant conversions - the conversion that we apply at any stage of optimization - gate fusion and merge CZ 2) Not instant - the conversion that we may be apply to our circuit

```
"""
───@──@───     ───
   │  │     ≡
───@──@───     ───
"""
```

```
"""
─R_─R_─R_─R_─@─R_─      ≡      ─RZ─RX─@─RZ─RX─
             │                        │
─────────────@────      ≡      ───────@───────
"""
```

```
"""
───RX(a + pi)───@───       ─RX(a)──@──X────
                │      ≡           │
────────────────@───       ────────@───Z──
"""
```

```
"""
────@────RX(a + pi)─      ───X───@────RX(a)───
    │                  ≡         │
────@───────────────      ─────Z─@────────────
"""
```

```
"""
───RZ───@───       ───@───RZ───
        │      ≡      │
────────@───       ───@───────
"""
```

```
"""
────RX(pi + b)──────RZ(a)─   ≡   ─RX(b)──RZ(-a)───X───
"""
```

```
"""
───RZ(a)───RX(pi + b)─── ≡ ────X──────RZ(-a)──────RX(b)────
"""
```

Simulated annealing

On each stage k we define a decreasing temerature T_k :

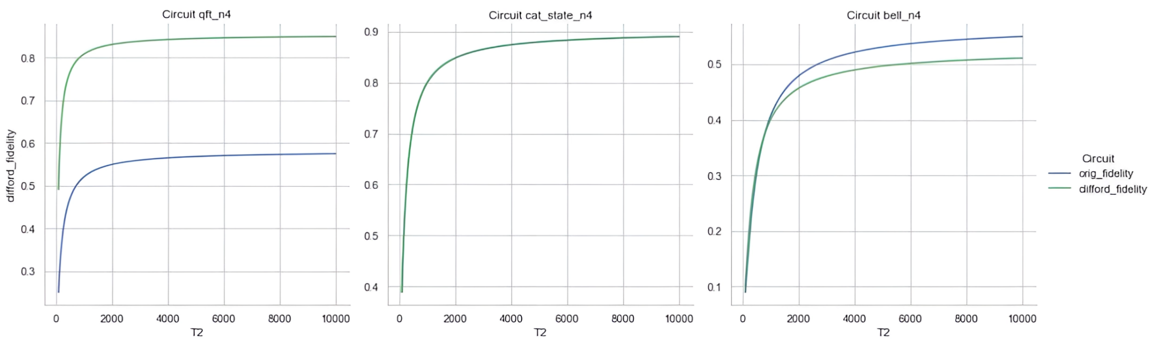
1) Choose random n_k - number of transformations steps of this stage - this number decrease with T_k 2) On each step: 1. choose random transformer 2. find all possible transformation of this kind 3. choose random transformation, apply it 3) Solve the loss function of the transformed circuit and - if it increase, accept this transformation - otherwise, accept this transformation with probability depending on T_k

Error model

1. *Simple* - after each gate we apply Z-error on operation's qubits with probability = 0.05
2. *Hard* - (more realistic, but it took so long to solve it, and we didn't use it) - the probability depends of type of gates:

    - $RX(\theta)$, $RZ(\theta)$ gates : $p_{error} = e^{-\frac{t(\theta)}{T_{2_{RX,RY}}}}$
    - CZ gate: $p_{error} = e^{-\frac{t_{CZ}}{T_{2_{CZ}}}}$
    - Idle: $p_{error} = e^{-\frac{t_{idle}}{T_{2_{idle}}}}$
    - Also, it takes $t_{prepare}$ to prepare the gate

    It can be shown that fidelity and clifford fidelity is stably dependent on T_2



Circuit Tests

As in the main article we tests the scheme on a random circuits

Random circuit generating pseudocode:

0. Circuits consists of 3-7 qubits
1. Repeat d times:
    A. choose random n_1 , n_2 - number of qubits
    B. apply CZ on n_1 , n_2
    C. {choose random θ and random axis from x and z} $- \times 2$
    D. apply $R_{axis_1}(\theta_1)$ on n_1 and $R_{axis_2}(\theta_2)$ on n_2
2. Apply instant conversions

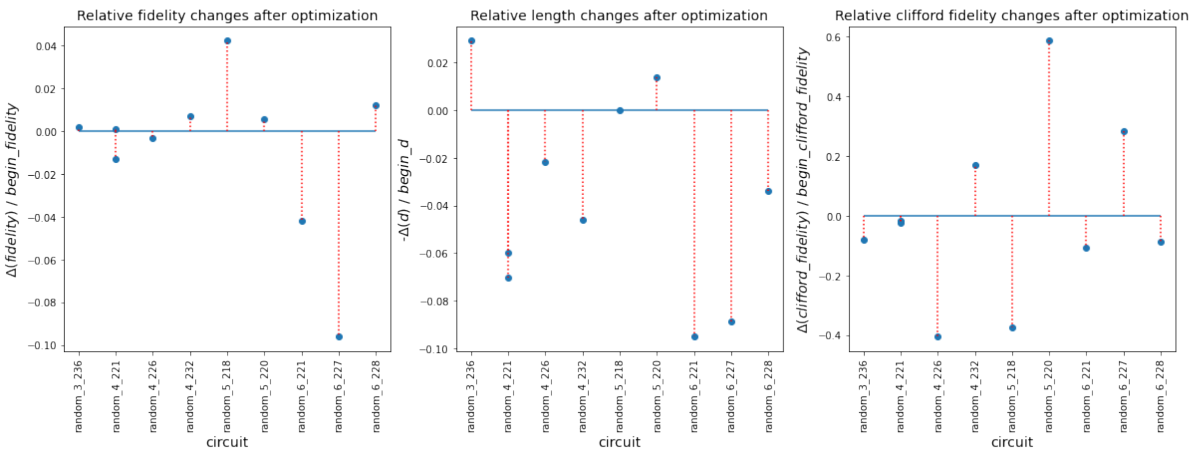
Results

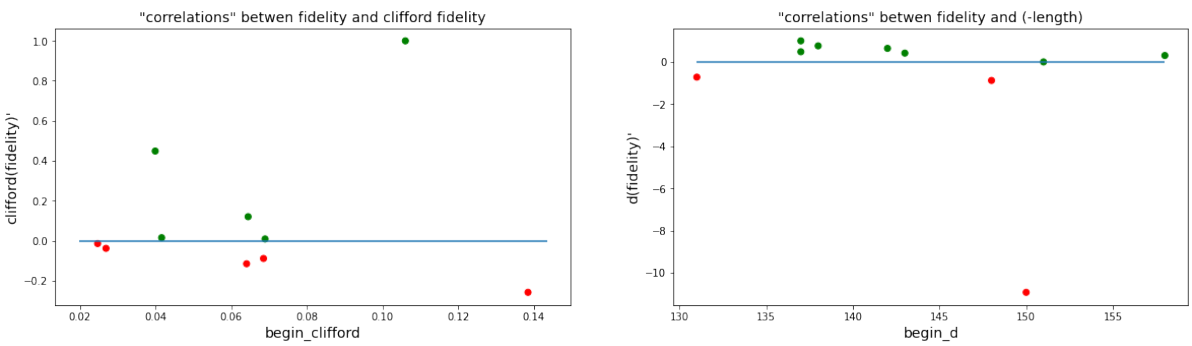
Execute the optimizer with the *infidelity* as a loss function. Then we will get the best transformation our model can get. After that, we will compare the other functions of the initial and final circuit and will be able to see how they correlate with fidelity.

That is, we calculate the initial and final length of the scheme, and also the initial and final Clifford fidelity

Out[17]:

	circ	n	begin_fidelity	end_fidelity	begin_d	end_d	begin_clifford	end_cliffor
0	random_3_236	3.0	0.128828	0.129058	137.0	133.0	0.138436	0.12721
1	random_4_221	4.0	0.063896	0.063943	150.0	159.0	0.064028	0.06299
2	random_4_221	4.0	0.068775	0.067883	142.0	152.0	0.068916	0.06737
3	random_4_226	4.0	0.066574	0.066348	138.0	141.0	0.106013	0.06323
4	random_4_232	4.0	0.068662	0.069139	131.0	137.0	0.064438	0.07532
5	random_5_218	5.0	0.036296	0.037830	151.0	151.0	0.068504	0.04277
6	random_5_220	5.0	0.050490	0.050765	143.0	141.0	0.039875	0.06322
7	random_6_221	6.0	0.037091	0.035543	137.0	150.0	0.041567	0.03706
8	random_6_227	6.0	0.027473	0.024839	158.0	172.0	0.024644	0.03162
9	random_6_228	6.0	0.026833	0.027160	148.0	153.0	0.026859	0.02451

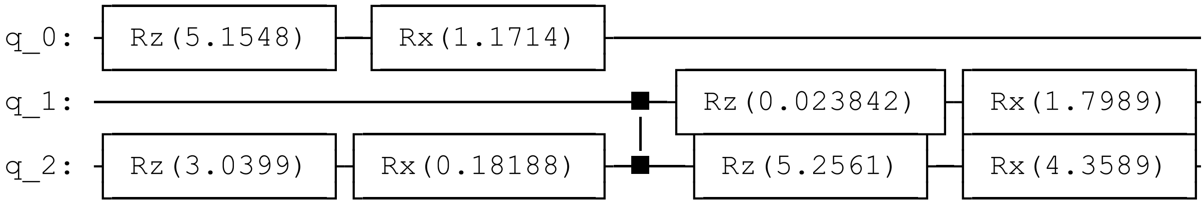




Example

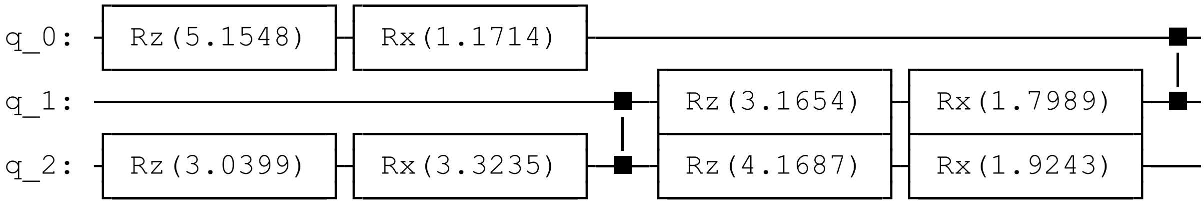
before:

Out[20]:



after:

Out[21]:



Main Result We see that in random schemes fidelity doesn't increases after optimization. At the same time, the increase and decrease in fidelity calculated on the clifford gates, as a length of the circuit, does not carry information about the real fidelity