Work Summary

Further result for modifying the fidelity expression. Examined the effects of iterations, evolution time, and J-χ ratio. The explicit expression of the evolution unitary is also shown for sanity check.

Main Work

1. Intuitive understanding of the Hilbert-Schmidt inner product as fidelity
2. Modifications on the fidelity expression
3. Explicit comparison of unitary evolution matrix
4. Results

1 Intuitive understanding of the Hilbert-Schmidt inner product as fidelity

Previously I don't quite understand the reasoning of using the the Hilbert-Schmidt inner product as the fidelity, i.e.

$$f_{HS} = \frac{Tr\left\{\hat{U}_{targ}^{-1}\hat{U}_f\right\}}{N} = |f_{HS}|e^{i\theta}$$

. A possible understanding can be as follows. Let the state vector be

$$\vec{s} = \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_N \end{pmatrix},$$

satisfying the normalization condition $\vec{s}^{\dagger}\vec{s} = 1$. If the evolution unitary is $\hat{U}$ and the initial state is $\vec{s}_i$, then the final state

$$\vec{s}_f = \hat{U}\vec{s}_i.$$

Starting from the initial state $\vec{s}_i$, assume the target unitary evolves the state to $\vec{s}_{targ} = \hat{U}_{targ}\vec{s}_i$ and the pulse-driven unitary evolves to $\vec{s}_{pulse} = \hat{U}_f\vec{s}_i$. Normally the fidelity of two states is defined to be

$$f = |\vec{s}_{targ}^{\dagger}\vec{s}_{pulse}|.$$

Expand the expression, sum over the N cases where the initial states are pure states, and average over the dimensions. Thus,

$$f = \frac{1}{N}\sum_i \left(\hat{U}_{targ}\vec{s}_i\right)^{\dagger}\hat{U}_f\vec{s}_i = \frac{1}{N}\sum_i \vec{s}_i^{\dagger}\left(\hat{U}_{targ}^{\dagger}\hat{U}_f\right)\vec{s}_i = \frac{Tr\left\{\hat{U}_{targ}^{-1}\hat{U}_f\right\}}{N}.$$

Thus recovering f_{HS}.

2 Modifications on the fidelity expression

As described by the previous notes, the fidelity expression f_{HS} can be modified to better suit our purposes. Following the conventions set before, in order for the fidelity to only include the entries that matters, the fidelity can be modified to be

$$f_2 = \frac{Tr\left\{\hat{I}_{6,g}\hat{U}_{targ}^{-1}\hat{U}_f\right\}}{6}, \text{ where } \hat{I}_{6,g} = \hat{I}_{6,cav} \otimes |g\rangle\langle g|$$

and here the $N_{cav} \times N_{cav}$ matrix $\hat{I}_{6,cav} = \begin{bmatrix} \hat{I} & \\ & \mathbf{0} \end{bmatrix}$ with the 6×6 identity matrix $\hat{I}$. Note that this f_2 expression is modified from the previous expression, but I think they should be equivalent, so this one is implemented for simplicity. Since this fidelity only concentrates on the 6×6 hopping unitary when the qubit starts and ends in the ground state, it is also referred as the ground-hop fidelity.

3 Explicit comparison of unitary evolution matrix

To check that the modification indeed produces better optimized unitary evolution, the absolute value of the 6×6 matrix of the unitary evolution is explicitly written out. In this case, let evolution time to be $\tau = 2.0\mu$s, then the target hopping unitary is

$$|U_{targ}| = |e^{-i\hat{H}_{hop}\tau}| = \begin{pmatrix} 0.5937 & 0. & 0.4911 & 0.4911 & 0. & 0.4063 \\ 0. & 0.5937 & 0.4911 & 0.4911 & 0.4063 & 0. \\ 0.4911 & 0.4911 & 0.1874 & 0. & 0.4911 & 0.4911 \\ 0.4911 & 0.4911 & 0. & 0.1874 & 0.4911 & 0.4911 \\ 0. & 0.4063 & 0.4911 & 0.4911 & 0.5937 & 0. \\ 0.4063 & 0. & 0.4911 & 0.4911 & 0. & 0.5937 \end{pmatrix}.$$

Then, apply the GRAPE algorithm with the fidelity defined by f_{HS} and f_2 respectively. For comparison sake, let the target fidelity to be 96%.

When optimizing f_{HS} in GRAPE, the fidelity $f_{HS} = 96.00\%$ after 800 iterations and around 10 minutes. As a reference, the resulting pulse-driven evolution unitary has $f_2 = 80.32\%$. The absolute values are

$$|U_{evol}| = \begin{pmatrix} 0.7838 & 0.2171 & 0.434 & 0.1461 & 0.1534 & 0.1697 \\ 0.1209 & 0.7264 & 0.5172 & 0.2137 & 0.2405 & 0.1143 \\ 0.5311 & 0.4691 & 0.3692 & 0.1313 & 0.3279 & 0.341 \\ 0.1443 & 0.2755 & 0.1848 & 0.8418 & 0.2621 & 0.197 \\ 0.0501 & 0.1958 & 0.3937 & 0.2955 & 0.8313 & 0.038 \\ 0.0915 & 0.1088 & 0.3309 & 0.2265 & 0.1191 & 0.8795 \end{pmatrix}.$$

When optimizing f_2 in GRAPE, the fidelity $f_2 = 96.04\%$ after 117 iterations and around 1 minute. As a reference, the resulting pulse-driven evolution unitary has $f_{HS} = 18.24\%$. The absolute values are

$$|U_{evol}| = \begin{pmatrix} 0.6438 & 0.0639 & 0.4331 & 0.399 & 0.0742 & 0.3769 \\ 0.0393 & 0.6448 & 0.4548 & 0.425 & 0.3842 & 0.085 \\ 0.4542 & 0.476 & 0.1754 & 0.0473 & 0.5159 & 0.4529 \\ 0.432 & 0.4329 & 0.0165 & 0.2627 & 0.4738 & 0.5263 \\ 0.0895 & 0.3655 & 0.4936 & 0.5009 & 0.5375 & 0.0468 \\ 0.337 & 0.0468 & 0.5019 & 0.5114 & 0.0172 & 0.5606 \end{pmatrix}.$$

It is clear that the result obtained by the modified GRAPE algorithm produces evolution unitary closer to the target unitary when the fidelities defined are at about the same value. As a further example, apply the modified GRAPE algorithm for 4000 iterations produces a pulse-driven evolution unitary of $f_2 = 99.995\%$ and $f_{HS} = 11.68\%$. The absolute values are

$$|U_{evol}| = \begin{pmatrix} 0.5937 & 0.0005 & 0.4909 & 0.4916 & 0.0005 & 0.4058 \\ 0.0002 & 0.5935 & 0.4909 & 0.4911 & 0.4068 & 0.0003 \\ 0.4912 & 0.4911 & 0.1874 & 0.0006 & 0.4909 & 0.4914 \\ 0.4906 & 0.4914 & 0.0003 & 0.1872 & 0.4913 & 0.4912 \\ 0.0005 & 0.4062 & 0.4918 & 0.4908 & 0.5934 & 0.0003 \\ 0.4067 & 0.0007 & 0.4909 & 0.491 & 0.0001 & 0.5937 \end{pmatrix}.$$

4 Results

Following most conventions set last time, the only difference is that the results are shown for both cases of fidelity f_{HS} and f_2, and in the plots f_{HS} is called full fidelity and f_2 is called ground-hop fidelity.

4.1 Effect of iterations

The number of iterations is the maximum limit set for the number of updates the pulse sequence will go through before stopping. The fidelity error is plotted against the max iteration number in Figure 1 and 2. Both ground-hop fidelity and full fidelity are calculated and labelled in the graphs, but Figure 1 and 2 uses f_{HS} and f_2 in the algorithm's definition of fidelity respectively.

Worth noting that as shown in the previous section, f_2 matters more than f_{HS}. With 2000 iterations, f_2 is 98.0% in Figure 1 and 99.985% in Figure 1, indicating a better performance in the modified fidelity expression.

4.2 Effect of evolution time

Evolution time set to the algorithm is equivalent to the length of the pulse subject to optimization. Assume the target total evolution time to be $T = 10\mu$s such that the electron state changes effectively. To check, the absolute value of the 6×6 unitary describing the evolution after T is

$$|U_{hop}| = |e^{-i\hat{H}_{hop}T}| = \begin{pmatrix} 0.9045 & 0. & 0.2939 & 0.2939 & 0. & 0.0955 \\ 0. & 0.9045 & 0.2939 & 0.2939 & 0.0955 & 0. \\ 0.2939 & 0.2939 & 0.809 & 0. & 0.2939 & 0.2939 \\ 0.2939 & 0.2939 & 0. & 0.809 & 0.2939 & 0.2939 \\ 0. & 0.0955 & 0.2939 & 0.2939 & 0.9045 & 0. \\ 0.0955 & 0. & 0.2939 & 0.2939 & 0. & 0.9045 \end{pmatrix}.$$

There is the choice to set the pulses to length T/n, where n is any integer. The algorithm outputs the final optimized unitary driven by the pulse sequences U_f. Define the single run fidelity to be the fidelity comparing U_f and $U_{targ}(t = T/n)$, and define the evolution fidelity to be the one comparing $(U_f)^n$ and $U_{targ}(t = T)$. When the different pulse lengths are applied, the time duration of each time slice, where the pulse stays constant, is 10ns.

Figure 3, 4, 5, and 6 plots the single/evolution fidelity error vs the number of pulse sequence that needs to be applied to reach T. As usual, the scenarios are calculated for both cases when the GRAPE algorithm applys the fidelity expression f_{HS} and f_2.

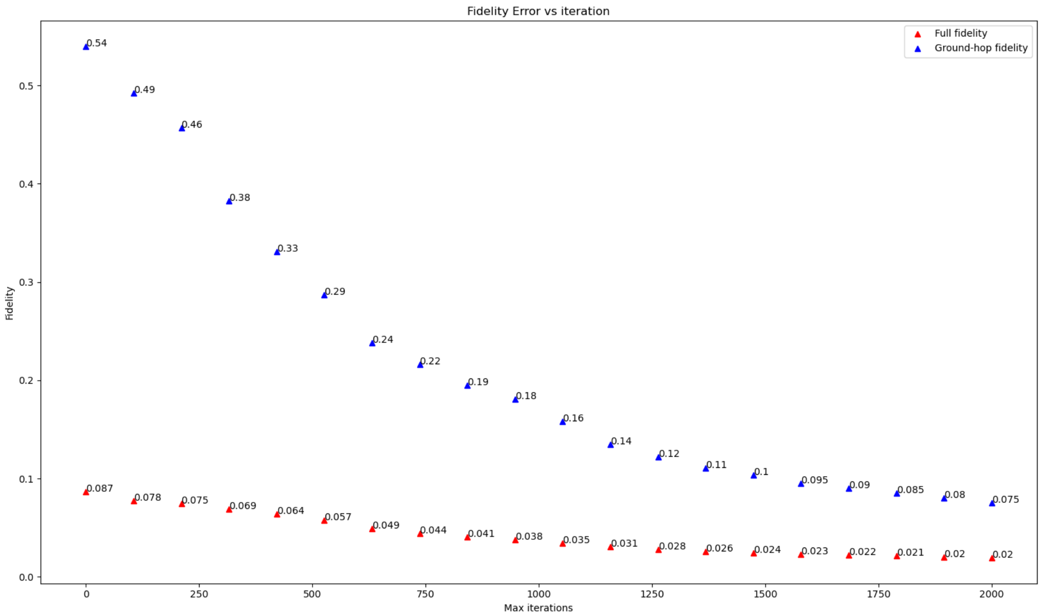

Figure 1: Fidelity error vs iteration number. Fidelity calculation in algorithm using f_{HS}.

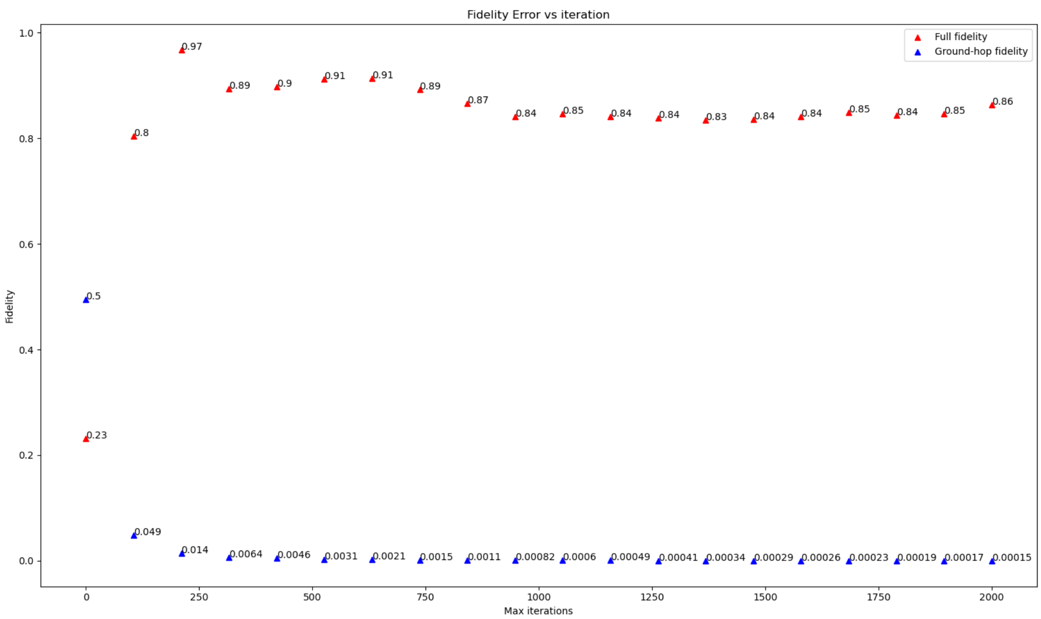

Figure 2: Fidelity error vs iteration number. Fidelity calculation in algorithm using f_2.

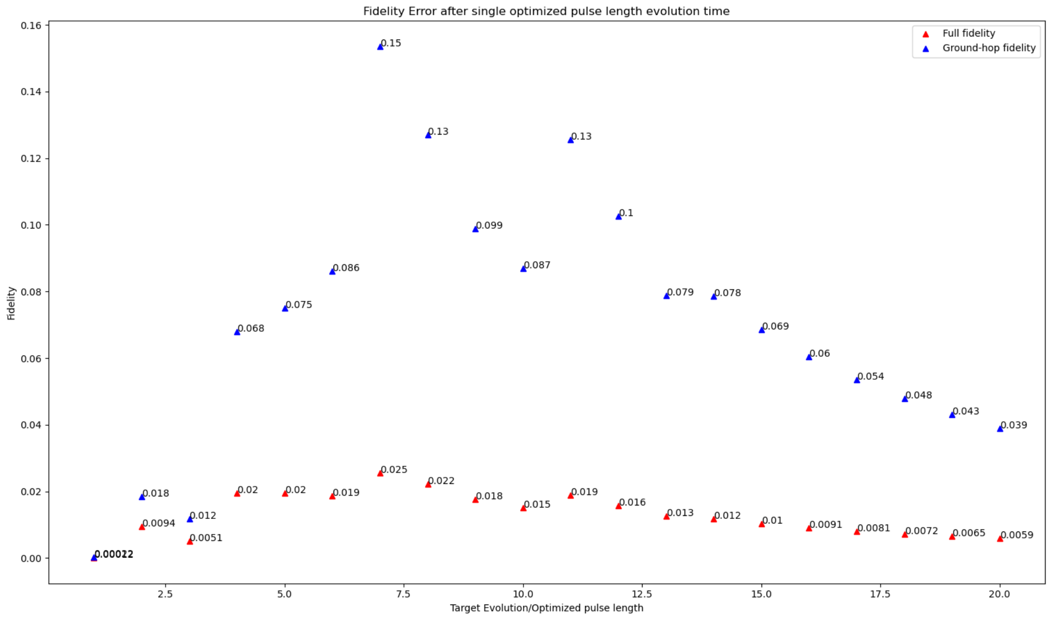

Figure 3: Single run fidelity error vs $n = T/t$. Fidelity calculation in algorithm using f_{HS}.

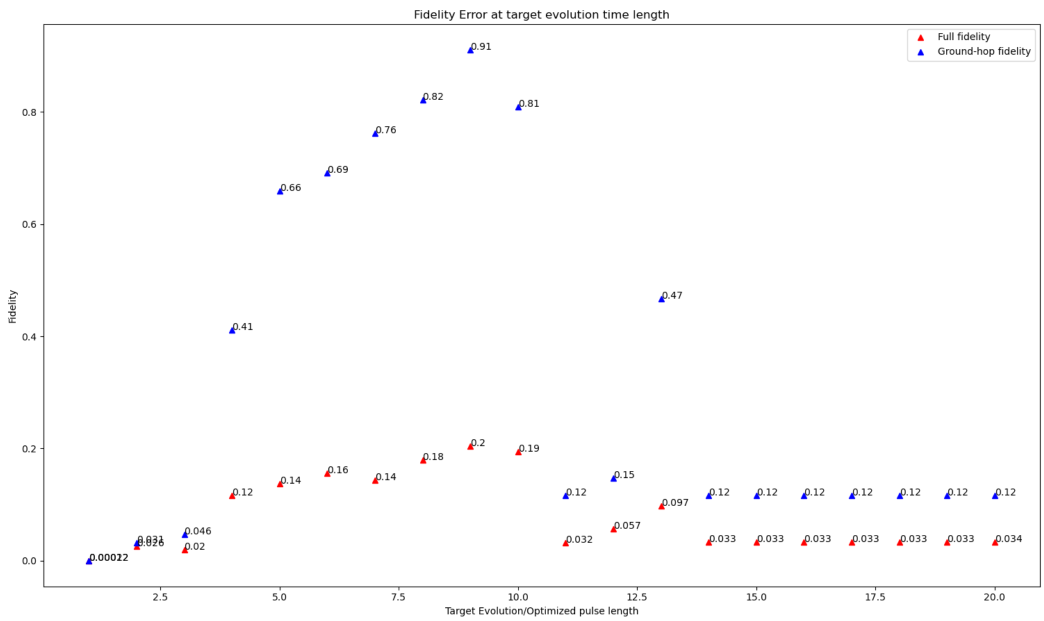

Figure 4: Evolution fidelity error vs $n = T/t$. Fidelity calculation in algorithm using f_{HS}.

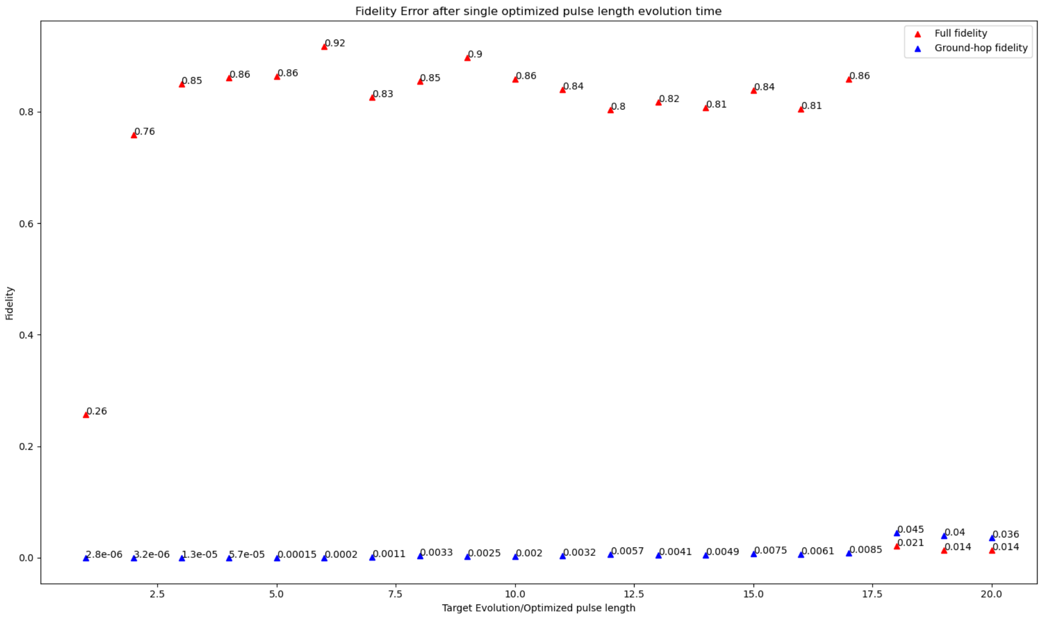

Figure 5: Single run fidelity error vs $n = T/t$. Fidelity calculation in algorithm using f_2.

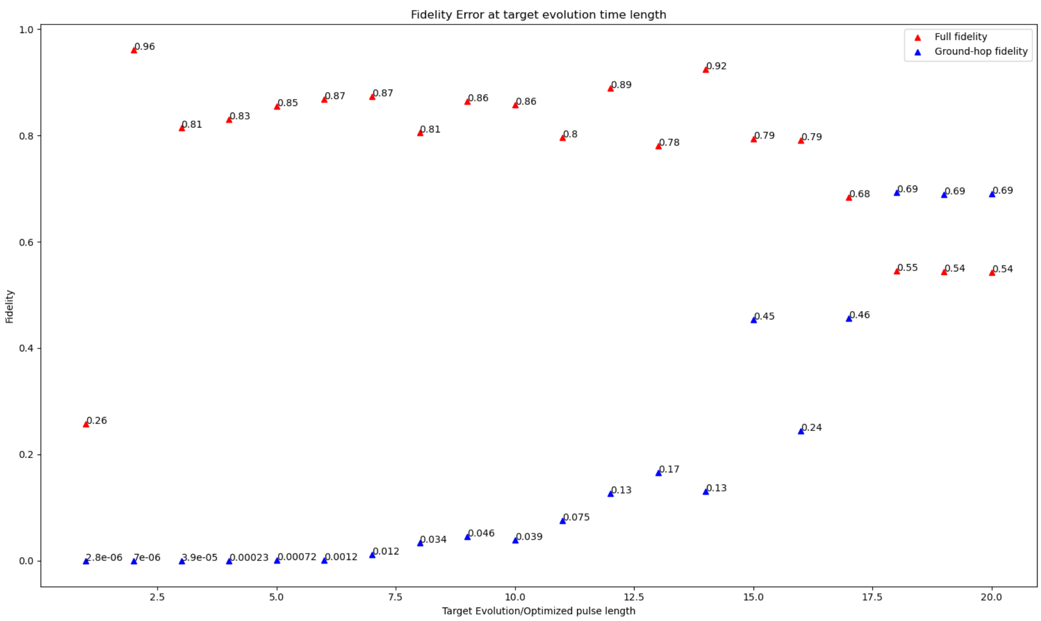

Figure 6: Evolution fidelity error vs $n = T/t$. Fidelity calculation in algorithm using f_2.

4.3 Effect of J-χ ratio

With the value of qubit-cavity interaction fixed to be $\chi = 2\pi \times 2.2$MHz, χ determines the capability of the system to control within a time frame. Therefore, the ratio of J with χ should critically affect the fidelity achieved. On the other hand, for simulation purposes, only the product of JT, where T is the pulse length, matters.

The resulting period from the qubit-cavity interaction is $T_{int} = \frac{2\pi}{\chi} \approx 0.5\mu$s. With JT fixed, the ratio of J and χ can be translated to the ratio of T and T_{int}.

Considering practical limitation, take values T=0.5T_{int} to T=10T_{int}, keeping the product $JT = 4.0$. This is equivalent of scanning through values of $J/\chi \approx 1$ to $\frac{1}{20}$. In this case, the phase of electron dynamics is fixed.

The resulting fidelity calculation from the original GRAPE algorithm and the modified GRAPE algorithm are shown in Figure 7 and 8 respectively. A more focused graph is shown in Figure 9 where only the values of f_2 are shown.

It is clear that only the result from the modified GRAPE algorithm works as expected such that the longer the evolution time, the lower the fidelity error since the qubit-cavity interaction works for a longer time. However, the values from the original GRAPE algorithm seems to imply the opposite. While I don't quite understand why, the value of f_{HS} has been showed in previous sections that it can't accurately represent how close the unitary is to the target, so the results from the modified GRAPE matters more.

The results from the modified GRAPE indicates that with the constants set, as long as the evolution time is more than 4 T_{int}, the fidelity error is low enough to simulate a phase change of around $JT = 4.0$.

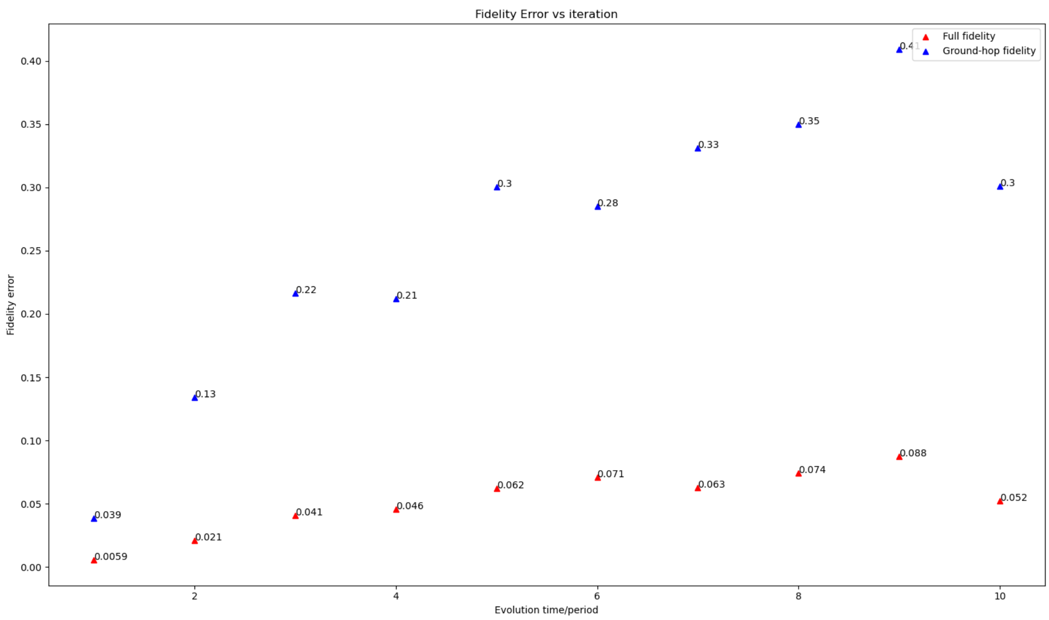

Figure 7: Fidelity error vs $n = T/T_{int}$. Fidelity calculation in algorithm using f_{HS}.

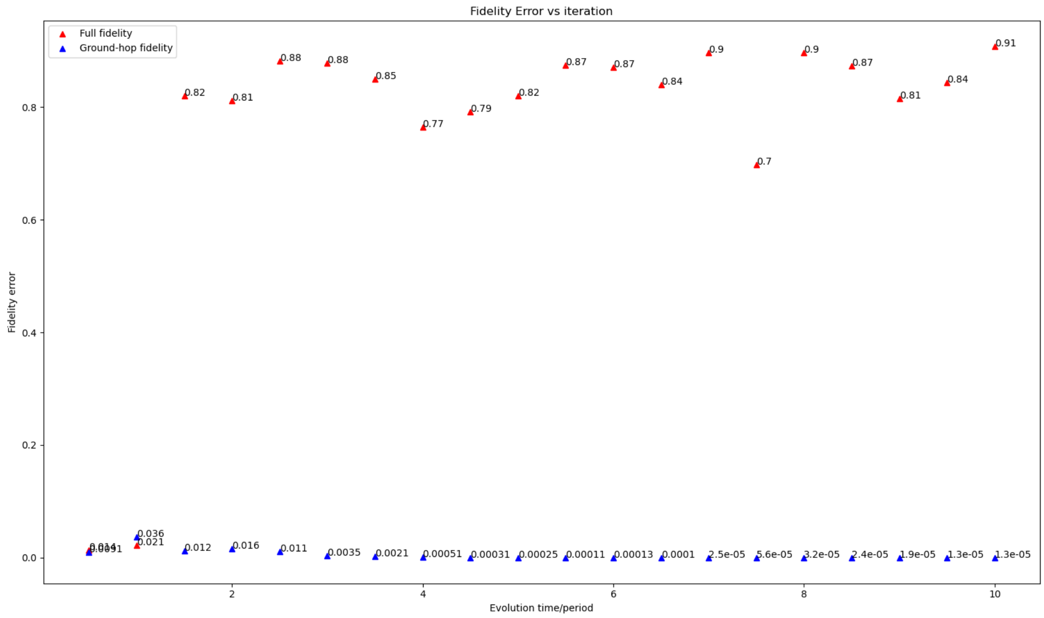

Figure 8: Fidelity error vs $n = T/T_{int}$. Fidelity calculation in algorithm using f_2.

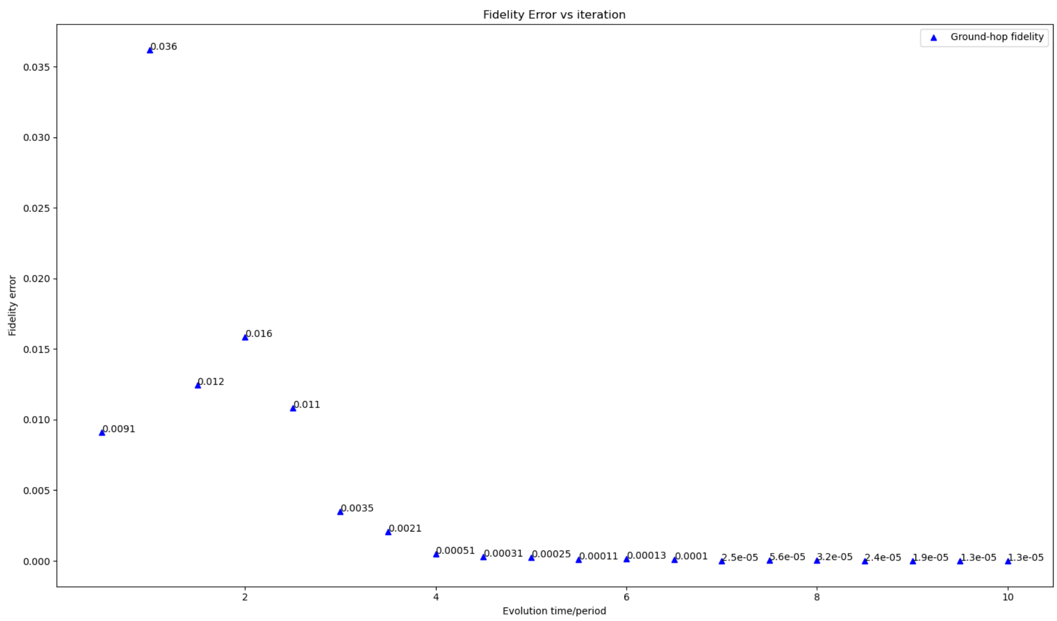

Figure 9: Fidelity error vs $n = T/T_{int}$. Fidelity calculation in algorithm using f_2. Only showing values of f_2.