FYP Report

Lin Zhonglin

March 26, 2024

Contents

1 Overview

1. Background:
2. Key part of my work: (depends on how far I go)
   (a) The optimised pulse for a particular state preparation
   (b) how it differs from other pulse (Chiyuan's alternate pulse way)
   (c) Go from model Hamiltonian to physical Hamiltonain, discuss the approximations' effect, compare fideilty

TODO:

1. Go through the Hamiltonian derivation → for comparison of fidelity later
2. does Qutip considers the cavity Hilbert space truncation discussed below?

Issues to be dealt with:

1. local minima / maxima in optimization algorithm
2. Reinhold: Optimizing a full unitary operation is sensible for qubits, or systems of a few qubits, but in the cavity manipulations to be performed in the following sections, this is not usually a sensible operation, since doing operations involving states at the border of the truncation (i.e. the maximum photon number state considered in the simulation), would typically require a larger truncation to simulate accurately (driving $|n_{ph}\rangle$) will almost always in part bring us partially to $|n_{ph}+1\rangle$.

2 Background, theory and motivation

content: introduce the idea of bosonic-modes-encoded qubits, give a simple example; motivate Bosonic-modes-encoded qubits -> that one physical device now can encode multiple qubits -> this also makes error correction more efficient -> easier to control multiple qubits

introduce cavity-transmon system motivate for cavity-DQD system

theory of cavity-DQD: original Hamiltonians -> rotation, approximaition, dispersive regime, etc... -> the Hamiltonian I used for pulse optimization

briefly bridge to and cover the overview of content of this report: 1. basic theory on optimization, QuTip, GRAPE, CRAB, etc... 2. pulse optimization for state preparation 3. compare fidelity from approximate Hamiltonian with fidelity from original Hamiltonian 4. possible future work: gate design, model noise

Quantum computing has emerged as a paradigm with the potential to solve problems that are intractable for classical computers. Among the various physical implementations of qubits, the cavity-coupled transmon system has become a leading platform due to its strong nonlinearity and large Hilbert space. This system leverages the coupling between the transmon qubit and the cavity to achieve robust qubit control and readout [**blais2004**, **houck2008**].

One of the key advantages of using a cavity for quantum computing is its potential for quantum error correction. By encoding a single logical qubit into multiple cavity Fock states, it is possible to protect against certain types of errors, thus enhancing the reliability of quantum computations [**girvin2014**, **ofek2016**].

Despite the success of the cavity-coupled transmon system, there is ongoing research into alternative qubit implementations that may offer further advantages. One such alternative is the cavity-coupled Double Quantum Dot (DQD) system. This system combines the advantages of semiconductor quantum dots, such as scalability and compatibility with existing semiconductor technology, with the strong coupling and large Hilbert space provided by the cavity. The DQD system has the potential to achieve high-fidelity qubit operations while maintaining the error correction capabilities of cavity-based systems [**petersson2012**, **mi2018**].

The control and manipulation of quantum systems are central to the realization of quantum computing. Quantum optimal control theory provides a framework for designing control protocols that achieve desired quantum operations with high fidelity. In this context, the goal is to find optimal control fields that drive the evolution of the quantum system from an initial state to a target state or implement a specific unitary transformation. Techniques such as Gradient Ascent Pulse Engineering (GRAPE) and Chopped Random Basis (CRAB) optimization are commonly used to find these optimal control fields [**khaneja2005**, **cai2011**]. These methods involve numerically optimizing a cost function, typically related to the fidelity between the achieved and desired quantum states, subject to constraints imposed by the physical system and control hardware.

In this report, we explore the application of quantum optimal control theory to the cavity-coupled DQD system for state preparation tasks. We compare the performance of different control optimization algorithms and investigate the effects of various approximations and constraints on the control fidelity. Through this study, we aim to

demonstrate the potential of the cavity-coupled DQD system as a platform for quantum computing and contribute to the development of robust control techniques for quantum technologies.

2.1 Hamiltonians of physical system

2.2 Quantum Optimal control

We assume we have a quantum system, whose Hamiltonian can be written in the following form:

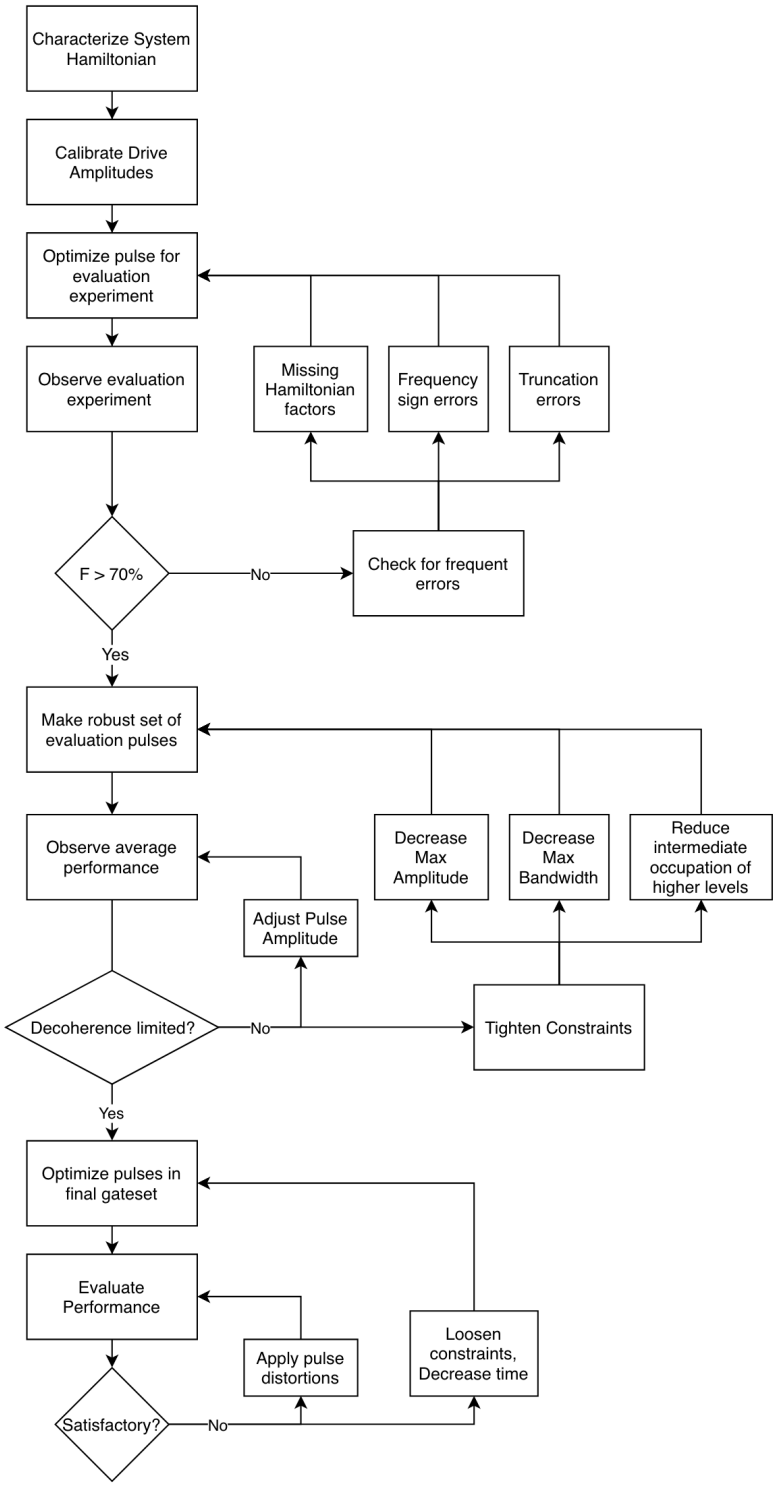

Figure 1: Flowchart for optimal control system

Overview of Quantum Optimal Control Quantum optimal control theory is a cornerstone of quantum technology, providing a systematic approach to manipulate quantum systems with high precision. The theory aims to determine the control strategies that steer a quantum system from an initial state to a desired final state with maximum

efficiency. This is crucial for the development and implementation of quantum computing, quantum sensing, and quantum communication technologies. The central challenge in quantum optimal control is to optimize control parameters, such as pulse shapes or sequences, to achieve high-fidelity quantum operations while minimizing errors and resource consumption [**glaser2015**, **brif2010**].

Basic Types of Quantum Optical Control Within the realm of quantum control, quantum optical control stands out as a key approach for manipulating quantum systems using electromagnetic fields. Coherent control and feedback control are two primary types in this domain. Coherent control involves shaping and timing electromagnetic fields to drive the quantum system's evolution coherently. This method is widely employed in controlling atomic and molecular systems, quantum dots, and superconducting qubits [**shapiro2012**, **chow2010**]. On the other hand, feedback control involves measuring the quantum system and using the outcomes to adjust the control fields in real-time, which is essential for error correction and stabilization in quantum systems [**wiseman2009**].

Overview and Basic Types of Quantum Pulse Optimization Quantum pulse optimization is a critical aspect of quantum control, focusing on finding the optimal control pulses for driving a quantum system's evolution. GRAPE, CRAB, and Krotov's methods are three widely used techniques in this area. GRAPE optimizes control pulses by numerically calculating the gradient of a fidelity measure concerning the control parameters. It is particularly effective for systems with a large number of control parameters [**khaneja2005**]. CRAB, on the other hand, reduces the complexity of the optimization problem by expanding the control pulses in a random basis and optimizing the expansion coefficients [**doria2011**]. Krotov's method is known for its monotonic convergence, making it suitable for problems where this property is crucial, such as open quantum systems [**reich2012**].

Python Packages for Quantum Optimal Control To facilitate quantum control simulations and optimizations, several Python packages have been developed. QuTiP is a comprehensive framework for simulating the dynamics of open quantum systems and includes tools for quantum control optimization [**johansson2013**]. The Krotov package, built on top of QuTiP, specifically implements Krotov's method for quantum optimal control, providing a user-friendly interface for solving quantum control problems [**goerz2019**].

2.3 Figure of merit, cost function

state transfer case single-state state transfer:

$$
\begin{aligned}
f(\vec{\epsilon}(t)) &= \mathcal{F}(\vec{\epsilon}(t)) \quad \text{state fidelity} \\
&\equiv \left|\langle\psi_{targ}| U(T,\vec{\epsilon}(t)) |\psi_{init}\rangle\right|^2 \\
&= \left|\langle\psi_{targ}| \mathcal{T} \exp\left\{-\frac{i}{\hbar}\int_0^T dt H(\vec{\epsilon}(t))\right\} |\psi_{init}\rangle\right|^2
\end{aligned}
$$

make the problem numerical, make $\vec{\epsilon}(t)$ a piece-wise constant function with $N = T/\delta t$,

δt is a parameter, usually set to time resolution of AWG.

$$
= \left|\langle\psi_{targ}| U_N U_{N-1} ... U_1 |\psi_{init}\rangle\right|^2, \quad \text{where, } U_k = \exp\left\{\frac{i\delta t}{\hbar} H(\vec{\epsilon}(k\delta t))\right\}
$$

multi-state state transfer:

$$
\begin{aligned}
f(\vec{\epsilon}(t)) &= \mathcal{F}(\vec{\epsilon}(t)) \quad \text{Fidelity} \\
&\equiv \begin{cases} \left|\sum_k \left\langle \psi_{targ}^{(k)} \right| U(T,\vec{\epsilon}(t)) \left|\psi_{init}^{(k)}\right\rangle\right|^2 & \text{coherent} \\ \sum_k \left|\left\langle \psi_{targ}^{(k)} \right| U(T,\vec{\epsilon}(t)) \left|\psi_{init}^{(k)}\right\rangle\right|^2 & \text{inoherent} \end{cases}
\end{aligned}
$$

for coherent case with number of state = dimension of Hilbert space,

$$
= \left|\mathrm{Tr}\left[U(T,\vec{\epsilon}(t)) U_{targ}^{\dagger}\right]\right|^2 \quad \text{unitary fidelity}
$$

open system case:

2.4 Pulse constraints

1. constraints come from AWG(Arbitrary Wave Generator)'s amplitude and bandwidth constraints → consider adding a set of constraint terms to the cost function

$$
f(\vec{\epsilon}(t)) = \mathcal{F}(\vec{\epsilon}(t)) + \sum_i \lambda_i g_i(\vec{\epsilon})
$$

2. pulse amplitude: the output power of our AWG is limited, and thus to be feasible, we need $\epsilon(t)\epsilon_{max}$ for all t. → hard cutoff / soft cutoff

   (a) Employ an optimization algorithm which naturally allows for such constraints → see K-BFGS-B in

scipy.optimize

(b) Parametrise optimization problen

$$\underset{\vec{\epsilon}(t)}{\text{maximize}} \quad \mathcal{F}(\vec{\epsilon}(t)) \rightarrow \underset{\vec{x}}{\text{maximize}} \quad \mathcal{F}(\vec{\epsilon}(\vec{x}))$$

$$\text{where, } \epsilon_k = \epsilon_{max} \tanh(x_k)$$

3. pulse bandwidth:

   - soft cutoff
   - hard cutoff
   - linear frequency-dependent penalty

2.4.1 limit the intermediate photon number

Since computer memory is finite, we are forced to choose a photon number truncation nph such that the operator a becomes a $n_{ph} \times n_{ph}$ matrix. When we do this, we are in effect replacing our infinite-dimensional oscillator with a finite-dimensional qudit. This replacement is only valid if all of the system dynamics relevant for the desired state transfers occurs within the $\{|0\rangle, \cdots, |n_{ph}-1\rangle\}$ subspace. For generic applied drives this is not the case. In order to enforce this property, we modify the optimization problem to find a solution which operates identically under several different values of n_{ph}. Writing the fidelity as computed with a truncation nph as $F_{n_{ph}}$, we have:

$$\underset{\vec{\epsilon}}{\text{maximize}} \left(\sum_k F_{n_{ph}+k}(\epsilon(t)) \right) - \left(\sum_i \lambda_i g_i(\epsilon(t)) \right) \tag{1}$$

- To enforce that the behavior is identical in the different truncations, we add the penalty term:

$$g_{\text{discrepancy}}(\boldsymbol{\epsilon}(\boldsymbol{t})) = \sum_{k_1 \neq k_2} \left(\mathcal{F}_{n_{\text{ph}}+k_1}(\boldsymbol{\epsilon}(t)) - \mathcal{F}_{n_{\text{ph}}+k_2}(\boldsymbol{\epsilon}(t)) \right)^2$$

  I think it's making the cost function symmetrical w.r.t. n_{ph}.

- A more recently developed, and more direct, method is to add a penalty term for any occupation of the final photon state ($|n_{ph}-1\rangle$) in the truncated Hilbert space at any time:

$$g_{\text{trajectory}} = \sum_{k=1}^{N} \left| \left\langle n_{\text{ph}} - 1 \mid \psi_{\text{fwd}}^{(k)} \right\rangle \right|^2$$

  Not sure what this is doing, putting here for possible future use.

Not sure whether this is implemented in QuTip.

2.4.2 Troubleshooting optimization convergence

Possible issue:

-

Check list:

1. Check that the time given $T = N\delta t$ is appropriate.
   - Specified in units that are consistent with the units specifying the Hamiltonian. For instance, if the Hamiltonian is specified in GHz, then the time step should be in units of ns.
   - N value appropriate
2. check constraints
   - Completely remove all constraints and penalties, and make sure that the algorithm works in this context before re-introducing them
3. check algorithm
   - check termination conditions
     Gradient based search algorithms usually have termination conditions specified in terms of the norm of the gradient. It is often necessary to lower the gradient norm threshold for termination to ensure that it does not give up -> gtol in scipy.minimize
4. check ionitial guess
   - Avoid special initial guesses, this depends on the algorithm. For gradient-based algo, make sure initial gradient is not vanishing.
5. miscellaneous checks
   - Truncation of cavity-state Hilbert space large enough?

2.5 Pulse optimization algorithms

- Quantum control
  - prepare some specific state
  - effect some state-to-state transfer
  - effect some transformation (or gate) on a quantum system

2.5.1 GRAPE: Gradient Ascent Pulse Engineering

GRAPE algorithm in QuTip:

1. overview:

   - The combined Hamiltonian is approximated as: $H(t) \approx H\left(t_k\right) = H_0 + \sum_{j=1}^{M} u_{jk} H_j$

     where, k is a timeslot index, j is the control index, and M is the number of controls. Hence t_k is the evolution time at the start of the timeslot, and u_{jk} is the amplitude of control j throughout timeslot k. Number of time steps $N = T/\delta t$.

   - The time evolution operator, or propagator, within the timeslot can then be calculated as: $U_k := e^{-iH(t_k)\Delta t_k}$

     where, Δt_k is the duration of the timeslot.

     The evolution up to (and including) any timeslot k (including the full evolution k=M) can the be calculated as $U\left(t_k\right) := U_k U_{k-1} \cdots U_1 U_0$

   - figure of merit: See Reinhold thesis section

   - optimization algorithm:

     (a) There are now $N \times M$ variables to minimize the figure of merit. The problem becomes a finite multi-variable optimiz

     (b) Optimization algorithm: (see Reinhold thesis section for details)

     Gradient and quasi-Newton method (considers Hessian) both implemented.

     (The default method in the QuTiP Qtrl GRAPE implementation is the L-BFGS-B method in Scipy)

     (c) calculating / approximating the gradient: see Reinhold thesis section

$$H(\vec{\epsilon}(t)) = H_0 + \sum_{k=1}^{m} \epsilon_k(t) H_k \tag{2}$$

Goal: Find optimal control field $\vec{\epsilon}(t)$ that optimizes some function $f(\vec{\epsilon}(t)) = f(H(\vec{\epsilon}(t)))$.

1. To optimize a large number of parameters, we need:

   (a) an efficient means of calculating the gradients of cost function w.r.t parameters

   (b) sub-optial local minima are sufficiently unlikely or close to global minima

2. Given methods to calculate cost function and gradients, we have two main classes of algorithm for performing optimization:

- line-search methods: one alternates between picking a direction in parameter space, radiating from our current point, and subsequently performing a 1-d minimization protocol to find the minimum along this line.

  → basic gradient descent: choose direction as gradient at the point.

  → Newton method: choose direction using both gradient and Hessian matrix at the point. → quasi-Newton, L-BFGS

- trust-region methods

3. calcualte gradient of $f(\vec{\epsilon}(t))$ with respect to $\vec{\epsilon}(t)$

   (a) For an analytical function $f(\vec{\epsilon})$, gradient is given by $\nabla f(\vec{\epsilon}) = \sum_{i=1}^{m} \frac{\partial f(\vec{\epsilon})}{\partial \epsilon_i} \hat{\epsilon}_i$

   (b) For discrete $\vec{\epsilon}$, gradient of $f(\vec{\epsilon})$ is given by the finite difference method.

   (c) approximate the gradient: skipping here.

   (d) When calculating / approximating gradient is too expensive, cosider optimization algorithm other than gradient descent.

4. take a step (step size as a parameter) in the direction.
   step size too small: optimization takes long
   step size too large: may never reach true optima

5. calcualte gradient again

6. repeat until convergence

7. issue of local minima / maxima:

   - Optimize for infidelity (1 - fidelity) instead of fidelity. Because infidelity is lower bounded by 0. Terminate optimization when infidelity reaches 0 or close to 0.

2.5.2 CRAB: Chopped RAndom Basis

CRAB-Chopped RAdom Basis algorithm in QuTip:

1. rough overview:

2. Since the pulse complexity is usually very low, it is sufficient to transform the optimal control problem to a few parameter search by introducing a physically motivated function basis that builds up the pulse. Compared to the number of time slices needed to accurately simulate quantum dynamics (often equals basis dimension for Gradient based algorithms), this number is lower by orders of magnitude, allowing CRAB to efficiently optimize smooth pulses with realistic experimental constraints.

3. choosing functional basis:

   - Consider a priori knowledge of the system

     → such as symmetries, magnitudes of scales,...

     → integrate experimental constraints such as maximum frequencies allowed, maximum amplitude, smooth ramping up and down of the pulse ...

   - Consider expected solution (e.g. sign, smoothness, bang-bang behavior, singularities, maximum excursion or rate of change,....).

4. where CRAB differs from GRAPE:

   - Optimized pulse from CRAB is a smooth function.
   - CRAB optimizes pulse function basis coefficient instead of amplitude of poulse at time slices.
   - CRAB considers time slices only when calculating fidelity for a set of function basis coefficients.

5. "dressed" CRAB: escaping local optima

3 State preparation pulse optimization

3.1 Starting with a simple diagnostic pulse, single qubit system

To verify that I have a working code that can optimize a pulse, I will start with a simple qubit filp operation of a single-qubit system. The purpose of this section is to show that the algorigm is indeed running optimization by recovering analytically solvable state-to-state transer using pulse optimization.

In the following example, we consider using the CRAB algorithm as implemented in the qutip python library. In qutip, the ctrlpulseoptim.optimize$_pulse_unitary function is used to optimize pulse shapes to minimize the fidelity error, which is equi$

The Hamiltonian of a single qubit system with arbitrary control is give by,

$$H(t) = \frac{\hbar\omega_0}{2}\sigma_z + \epsilon_x(t)\sigma_x + \epsilon_y(t)\sigma_y \tag{3}$$

where, $\epsilon_x(t), \epsilon_y(t)$ are control pulses.

Consider a special case of the above general case.

$$H_1(t) = \frac{\hbar\omega_0}{2}\sigma_z + \frac{\hbar\omega_1}{2}(\sigma_x \cos\omega t + \sigma_y \sin\omega t) \tag{4}$$

From Valerio's note, via frame rotation, starting from $|\psi_{init}\rangle = |+\hat{z}\rangle$, we have evoloved state,

$$|\psi(t)\rangle = e^{-i\omega t/2}[\cos(\Omega t/2) - i\cos\theta\sin(\Omega t/2)]|+\hat{z}\rangle - ie^{i\omega t/2}\sin\theta\sin(\Omega t/2)|-\hat{z}\rangle \tag{5}$$

where,

$$\Omega = \sqrt{(\omega_0 - \omega)^2 + \omega_1^2}\cos\theta = \frac{\omega_0 - \omega}{\Omega}$$

Suppose we want to achieve a state-to-state transfer from $|+\hat{z}\rangle$ to $|-\hat{z}\rangle$,

$$P_{+\hat{z}\to-\hat{z}}(t) = \sin^2\theta\sin^2(\Omega t/2) = \left(\frac{\omega_1}{\Omega}\right)^2\sin^2(\Omega t/2)$$

With $H_1(t)$, $P_{+\hat{z}\to-\hat{z}}(t)$ only goes to 1 if

1. $\omega_1 = \Omega$, i.e. $\omega = \omega_0$.
2. $\omega_1 = \frac{\pi}{T}$ (with evolution time T fixed, smallest frequency needed)

Now we run the pulse optimization and compare with analytical result.

Let t = evo$_t ime = T$

$Defining the time evolution parameters : -To solve the evolution the control amplitudes are considered constant within piecewise tin$

$expm(-iH(t_k)dt). - Combining these for all the time slots gives the approximation to the evolution from an initial state 0 at t =$

$0 to U(T) 0 at the t = evo_t ime. The number of time slots and evo_t ime have to be chosen such that the time slot durations (dt) are small comp$

$set drift frequency \omega_0$ to some number, at resonance $\omega = \omega_0$, for state to state transfer to happen at the end of pulse, require $\omega_1 = \frac{\pi}{T}$.

Next, we set the initial guess pulses where the pulse optimization algorithm starts. During the each iteration of the optimization, the Nelder-Mead algorithm calculates a new set of coefficients that improves the currently worst set among all set of coefficients. For details see [1,2] and a textbook about static search methods. The algorithm continues until one of the termination conditions defined above has been reached. If undesired results are achieved, rerun the algorithm and/or try to change the number of coefficients to be optimized for, as this is a very crucial parameter.

Though the exact analytical solution couldn't be recovered from from numetical optimization, the algorithm can be crudely verified to be running by a crude self-implemented forward evolution simulation code snippet.

3.2 Diagnostic pulse: cavity vacuum to cat state

Having tested the code on a single qubit system, we now move on to a more complex system that I am eventually interested in. In this code, we consider a cavity coupled to a single qubit. For this physical system, we have drift Hamiltonian, from Reinhold PhD thesis (2.13) [**reinhold2019**]:

$$H_d = \omega_q a^\dagger a + \frac{\omega_z}{2}\sigma_z + \frac{\chi}{2}a^\dagger a\sigma_z \tag{6}$$

It can be shown that it's equivalent to:

$$\begin{aligned} H_d &= g\left(\hat{\sigma}_+\hat{\sigma}_- + \hat{a}^\dagger\hat{a}\hat{\sigma}_z\right) \\ &= g\left(\frac{1}{2}(1+\hat{\sigma}_x) + \hat{n}\hat{\sigma}_z\right) \end{aligned}$$

where, g is the cavity-qubit coupling strength which is set to be $50 * 2\pi MHz$ here, as given by experiment

For this physical system we have both cavity and auxiliary qubit contorl, in the form of control Hamiltonian:

$$H_c = \epsilon_c(t)\hat{a} + \epsilon_T(t)\hat{\sigma}_- + h.c. \tag{7}$$

where,

- $\epsilon_c(t)$ is the contorl signal to cavity
- $\epsilon_T(t)$ is the control signal to ancilliary qubit

with $\hat{\sigma}_- = \frac{1}{2}\hat{\sigma}_x - \frac{i}{2}\hat{\sigma}_y$, rewrite H_c as following for ease of implementation using QuTip:

$$H_c = \epsilon_c(t)\hat{a} + \epsilon_T(t)(\frac{1}{2}\hat{\sigma}_x - \frac{i}{2}\hat{\sigma}_y) + h.c. \tag{8}$$

Assuming that the control signals $\epsilon_c(t), \epsilon_T(t)$ are real, and ignoring scalar constants, we have

$$H_c = \epsilon_c(t)(\hat{a} + \hat{a}^\dagger) + \epsilon_T(t)\hat{\sigma}_x$$

Initial and target states:

$$|\psi_{\text{initial}}\rangle = |\alpha\rangle$$

$$|\psi_{\text{final}}\rangle = |\alpha\rangle + |-\alpha\rangle$$

For testing the code, consider

$$|\psi_{\text{initial}}\rangle = |0\rangle$$

$$|\psi_{\text{final}}\rangle = |\alpha\rangle$$

The optimization algorithm settings used in this example are:

The initial guess pulse:

The optimised pulse:

To verify whether the cavity state basis truncation N is large enough, i.e. whether the optimziation result has converged with respect to N, consider:

1. Run the optimization with the same algorithm settings and same initial guess, but with different N values. Then plots the optimized pulses ran with different N values on the same plot.

2. Simulate the evolution of the system with the optimized pulses, but with a range of higher N values.

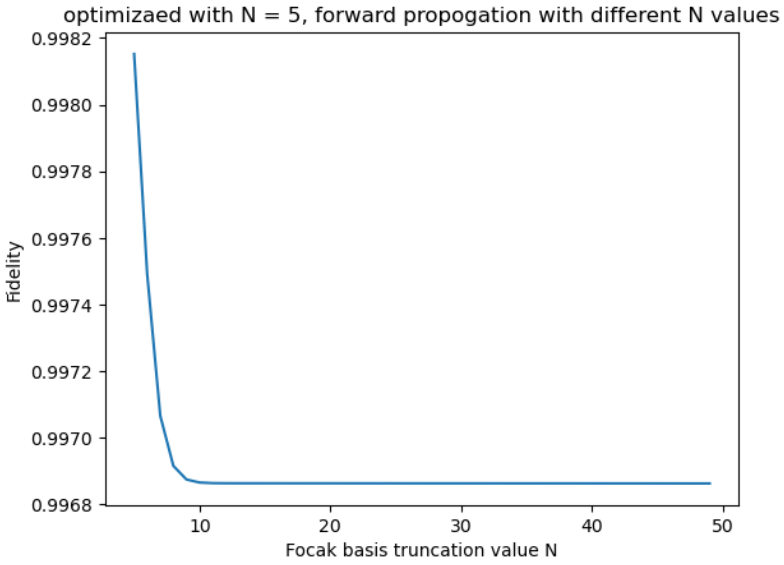

Figure 2: cavity state Fock basis truncation N convergence

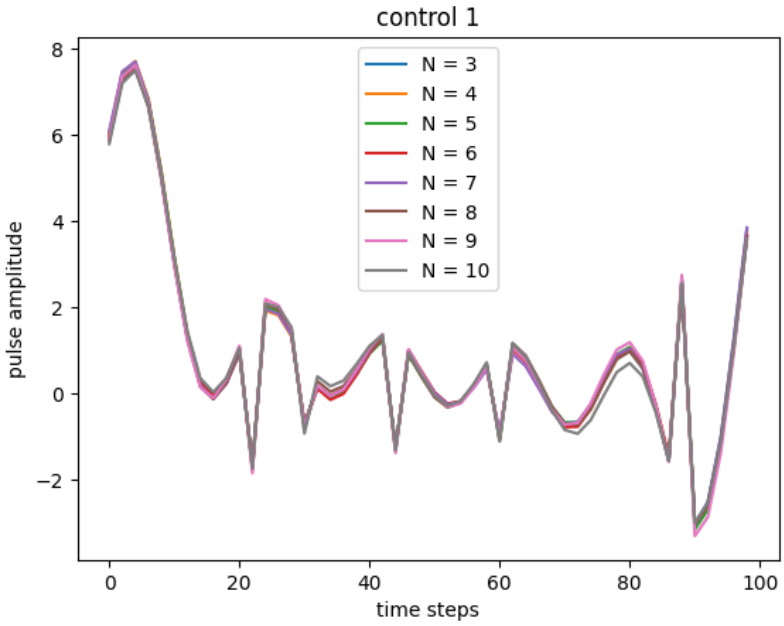

Figure 3: cavity state Fock basis truncation N convergence

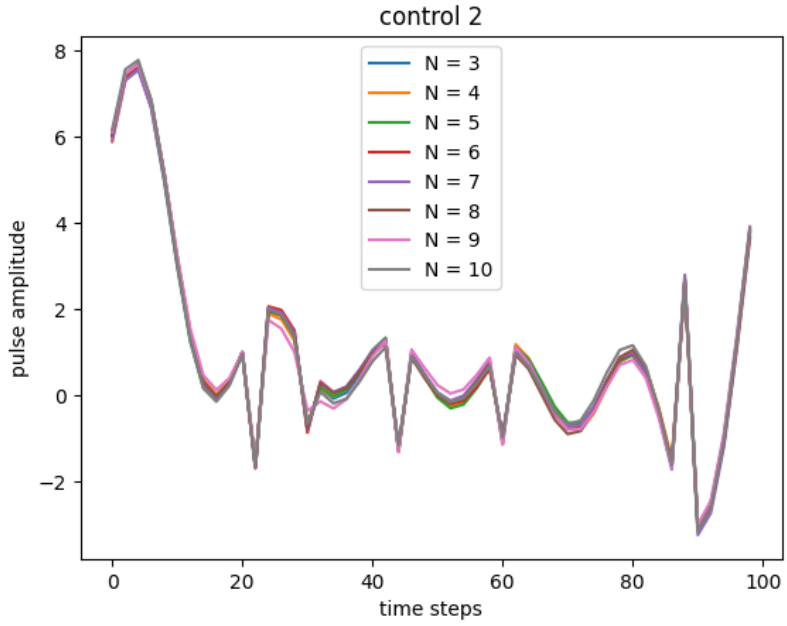

Figure 4: cavity state Fock basis truncation N convergence

3.3 Diagnostic pulse: unselective spin flip, cavity coupled with single qubit(effective)

Going from a cavity cavuum state to a cavity coherent state analytically requires only a displacement operation which requires only a control pulse to the cavity. Hence, in the previous disagnostic scenario the full Hamiltonia with all three (real) control channels were not used.

Here we consider the full system Hamiltonian (cavity + auxiliary qubit + coupling terms) drift Hamiltoinan is given by,

$$H_d = \text{wr} * n_{\text{cavity}} + \text{Ez} * \sigma_z + \text{chi} * \sigma_z * (n_{\text{cavity}} + 1/2) \tag{9}$$

where parameters of the physical system are:

(notet that energy terms have unit GHz and time terms have unit ns)

- cavity frequency, wr $= 2\pi$
- qubit anharmonicity Ez $= 0.44\pi$
- qubit-cavity coupling strength chi $= 0.007$

Real control channels in the Hamiltonain are given by (each term comprises a channel):

$$H_c = a + a^\dagger + -1j * (a - a^\dagger) + \sigma_x \tag{10}$$

The initial and target states for this pulse optimization code are:

$$\psi_0 = \frac{1}{\sqrt{2}}(|0\rangle_{\text{cavity}} \otimes |0\rangle_{\text{qubit}} + |1\rangle_{\text{cavity}} \otimes |0\rangle_{\text{qubit}})$$
$$\psi_{\text{targ}} = \frac{1}{\sqrt{2}}(|0\rangle_{\text{cavity}} \otimes |1\rangle_{\text{qubit}} + |1\rangle_{\text{cavity}} \otimes |1\rangle_{\text{qubit}})$$

where the qubit state is flipped irrepsective of the cavity state

The final optimization algorithm parameters are:

- optimization algorithm: GRAPE
- Number of time slots, n_ts = 500
- Time allowed for the evolution, evo_time = 1
- Fidelity error target, fid_err_targ = 1e-3
- Maximum iterations for the optisation algorithm, max_iter = 1000
- Maximum (elapsed) time allowed in seconds, max_wall_time = 120
- initial guess pulse type, p_type = 'LIN'

The inital and optimized pulses are:

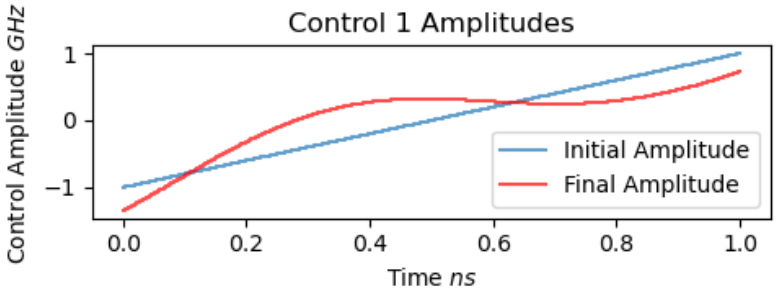

Figure 5: unselective spin flip control 1 pulse amplitudes

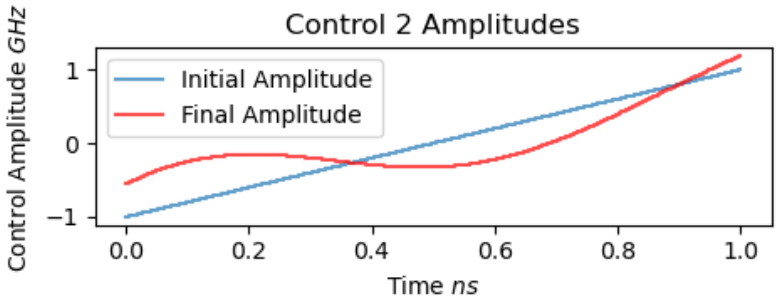

Figure 6: unselective spin flip control 2 pulse amplitudes

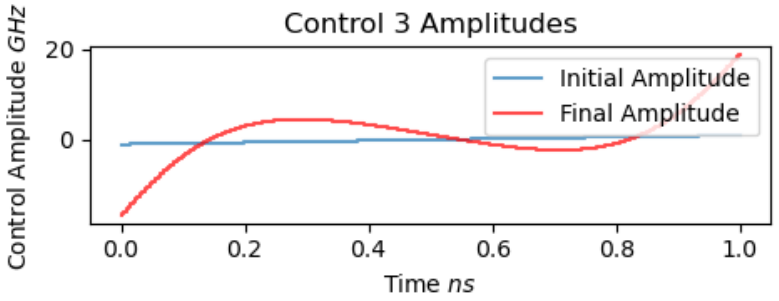

Figure 7: unselective spin flip control 3 pulse amplitudes

Forward simulation is given below to ensure that pulse optimization has run properly. Simulation gives:

- sesolve final fidelity: 0.9995869703894827
- self-implemented final fidelity: 0.9995575205270999

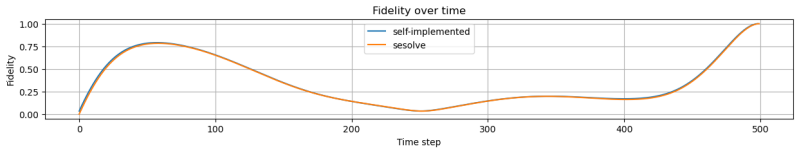

Figure 8: unselective spin flip forward simulation

3.4 Diagnostic pulse: selective spin flip, cavity coupled with single qubit(effective)

For this scenario, the drift and contorl Hamiltonian terms are same as in the previous section. We are interested in this state preparation because as achieved by Krastanov et al.(2015) [**Krastanov2015**] which gave a contruction for how to achieve arbitrary operation on dispersively coupled cQED systems using a set of two opeartions: displacements and selective number-dependent arbitrary phase (SNAP) operations. SNAP operations allow an arbitrary

set of relative phases to be applied to different photon number states, and can be represented with the form

$$S(\vec{\theta}) = \sum_k e^{i\theta_k} |k\rangle \langle k| \tag{11}$$

Clearly, SNAP operations require selective spin rotations of which selective spin flip is a special case.

State preparation optimization target is as follow:

$$\psi_0 = \frac{1}{\sqrt{2}}(|0\rangle_{\text{cavity}} \otimes |0\rangle_{\text{qubit}} + |1\rangle_{\text{cavity}} \otimes |0\rangle_{\text{qubit}})$$
$$\psi_{\text{targ}} = \frac{1}{\sqrt{2}}(|0\rangle_{\text{cavity}} \otimes |0\rangle_{\text{qubit}} + |1\rangle_{\text{cavity}} \otimes |1\rangle_{\text{qubit}})$$

(where qubit spin flip is selective on cavity state)

The best optimization result that was obtained by tuning the optimization script from previous section is as follow.

The final optimization algorithm parameters are:

- optimization algorithm: GRAPE
- Number of time slots, n_ts = 100000
- Time allowed for the evolution, evo_time = 500
- Fidelity error target, fid_err_targ = 1e-8
- Maximum iterations for the optisation algorithm, max_iter = 10000
- Maximum (elapsed) time allowed in seconds, max_wall_time = 7200
- initial guess pulse type, p_type = 'SINE'

The inital and optimized pulses are:

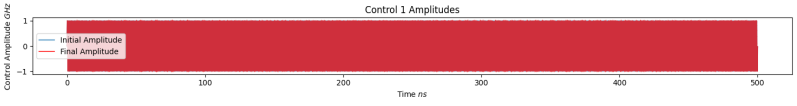

Figure 9: selective spin flip control 1 pulse amplitudes

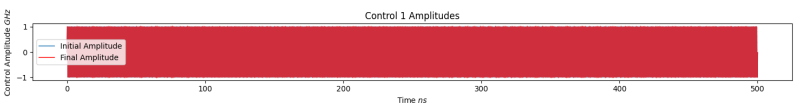

Figure 10: selective spin flip control 2 pulse amplitudes

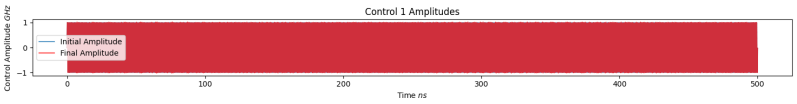

Figure 11: selective spin flip control 3 pulse amplitudes

The control pulses have very fast oscillations due to the large number of time slices. Zooming into control 1 pulse amplitudes, as shown in fig **??**, we see that the pulse is fairly smooth. However, this fast oscillation is still not desirable and preferably gotten rid of. Another reason that might have contributed to this fast oscillation is the background frequency of the cavity and qubit of which the control pulses are trying to cancel out. One way to partially deal with this fast oscillation issue is to work with the Hamiltonian in the rotating frame which will be tried out in the next section when cavity vacuum state to cavity cat state is being optimized.

selective_spin_flip_GRAPE_control1_zoomin.png

Figure 12: selective spin flip control 1 pulse amplitudes zoomed in

Forward simulation is given below to ensure that pulse optimization has run properly.
Simulation gives:

- sesolve final fidelity: 0.9994876128337873
- cavity ptrace fidelity: 0.9997184822307845
- qubit ptrace fidelity: 0.999999719275808

- self-implemented final fidelity: 0.9997005493228243

Figure 13: selective spin flip forward simulation

A particular analytical solution as given by Chiyuan already gives fidelity of 0.98. We expect a higher fidelity from a numerical optimization solution.

However, this numerical solution has the folowing issues. The optimzied pulses go beyong the amplitude constrains that required for the effective Hamiltonian to be valid. When the pulse amplitdes constrained are supplied to the optimziation algorithms, the results are not very satisfactory.

The following run has two modifications:

- initial guess pulse is changed from sindusoidal to linear to see whether this improves the fast oscillation issue
- Pulse amplitude constraints are added to the optimization algorithm

However, as can be seen from the optimized pulses below, the fast oscillation issue is not resolved well. This is especially so for control 3.

The inital and optimized pulses are:

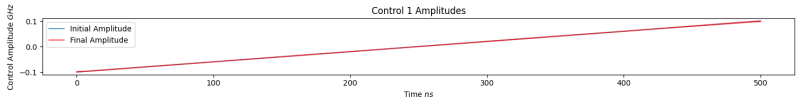

Figure 14: selective spin flip control 1 pulse amplitudes

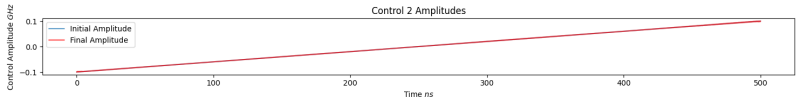

Figure 15: selective spin flip control 2 pulse amplitudes

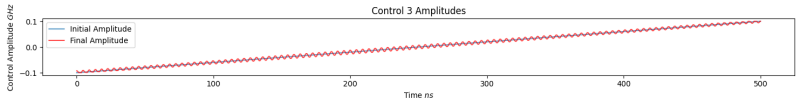

Figure 16: selective spin flip control 3 pulse amplitudes

Forward simulation is given below to ensure that pulse optimization has run properly.

Simulation gives:

- sesolve final fidelity: 0.9992259750482354
- cavity ptrace fidelity: 0.9992525011412332
- qubit ptrace fidelity: 0.9992706948956261
- self-implemented final fidelity: 0.9999937323329505

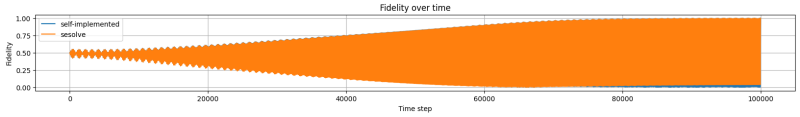

Figure 17: selective spin flip forward simulation

At this point, we continued varying the parameters (puse length, number of slices, initial guess pulse, etc.). However, none gave optimization results where fidelity, pulse amplitude, pulse smoothness and fast oscillation issue are all accounted for satisfactorily.

3.5 vacuum to cat state, cavity coupled with single qubit(effective)

In this section, we finally get to do a state preparation optimization of cavity vacuum state to cavity cat state. The most efficient way to do this is to optimize for the cavity state alone by tracing out the qubit state when calculating state fidelity. However, the QuTip package does not support this feature. I attemped to modify QuTip source code to allow for this feature but failed. One might naively think that QuTip calculates state fidelity at the end of each optimization iteration. Intead, QuTip calculates the state fidelity throughout the entire control pulse evolution for some caching purposes that improves the overall performance of the optimization algorithm. Here, a problem arises as tracing out qubit state in the middle of a pulse is not valid.
Hence, composite state of the cavity and qubit was still used for optimization which restricts the search space to smaller than ideal. However, the optimization results as shown below were not very satisfacotry.
The final optimization algorithm parameters are:

- optimization algorithm: GRAPE
- Number of time slots, n_ts = 100000
- Time allowed for the evolution, evo_time = 5000
- Fidelity error target, fid_err_targ = 1e-8
- Maximum iterations for the optimization algorithm, max_iter = 10000
- Maximum (elapsed) time allowed in seconds, max_wall_time = 21600

- initial guess pulse type, p_type = 'SIN'

The inital and optimized pulses are:

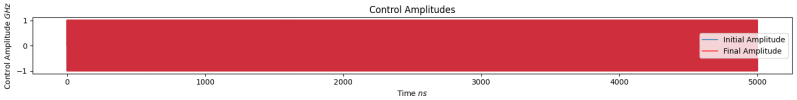

Figure 18: vacuum to cat state control 1 pulse amplitudes

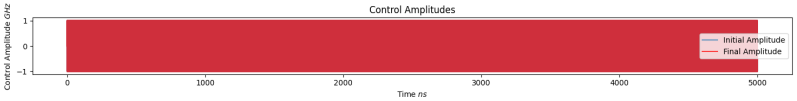

Figure 19: vacuum to cat state control 2 pulse amplitudes

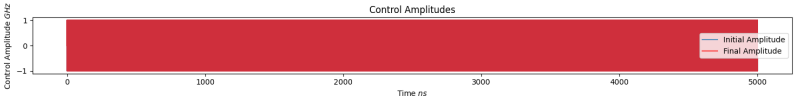

Figure 20: vacuum to cat state control 3 pulse amplitudes

Forward simulation is given below to ensure that pulse optimization has run properly. Simulation gives:

- sesolve final fidelity: 0.4390558062780975
- cavity ptrace fidelity: 0.44320315443027297
- qubit ptrace fidelity: 0.9871678390509413
- self-implemented final fidelity: 0.9529111010142354

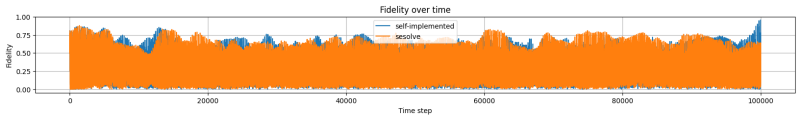

Figure 21: vacuum to cat state forward simulation

It can be seen that the optimization results are not very satisfactory. Many tunings were attempted but none gave satisfactory results.

At this point, we turned to look at a new python quantum control optimziation library, Krotov which builds on top of QuTip, to see whether it can give better optimization results. Krotov's optimization method is a gradient-based optimization algorithm like GRAPE. Krotov's method distinguishes itself by guaranteeing monotonic convergence for near-continuous control fields. Besides the difference in optimization algorithm used, the Krotov package has the following incentives to be tried:

- Krotov is much better implemented than QuTip. For instance, Krotov allows display of optimization progress for every iteration.

- Krotov allows inbuilds better customizability than QuTip. For instance, Krotov allows for the optimization of the fidelity to be optimized.

Besides switching from QuTip to Krotov, the Hamiltonians are also redefined to be an interaction picture. The drift Hamiltonian (H_0) and control Hamiltonians (H_1, H_2, H_3) are defined as follows:

1. **Drift Hamiltonian** (H_0):

$$H_0 = \hbar\omega_r a^\dagger a + \frac{E_s}{2}\sigma_z + \frac{E_m}{2}\tau_z - \chi_m\tau_z\left(a^\dagger a + \frac{1}{2}\right) - \chi_s\sigma_z\left(a^\dagger a + \frac{1}{2}\right) - \frac{\chi_0}{2}\sigma_z\tau_z, \tag{12}$$

where ω_r is the resonator frequency, a and $a^\dagger$ are the annihilation and creation operators of the cavity, E_s and E_m are the energy levels of the qubit and the molecular orbital, σ_z and τ_z are the Pauli Z operators for the qubit and the molecular orbital, respectively, and χ_m, χ_s, and χ_0 are the dispersive shifts.

2. **Control Hamiltonians**: - H_1: This control Hamiltonian corresponds to the real part of the cavity drive. It is given by

$$H_1 = \left(a + a^\dagger\right), \tag{13}$$

where a and $a^\dagger$ are the annihilation and creation operators of the cavity.

- H_2: This control Hamiltonian corresponds to the imaginary part of the cavity drive. It is given by

$$H_2 = -i\left(a - a^\dagger\right), \tag{14}$$

where a and $a^\dagger$ are the annihilation and creation operators of the cavity.

- H_3: This control Hamiltonian corresponds to the drive on the qubit. It is given by

$$H_3 = \sigma_x, \tag{15}$$

where σ_x is the Pauli X operator for the qubit.

The optimized results are as follow.

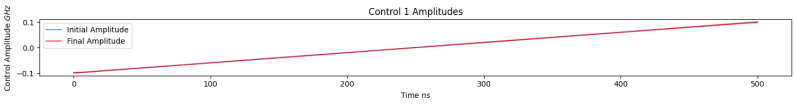

Figure 22: selective spin flip control 1 pulse amplitudes

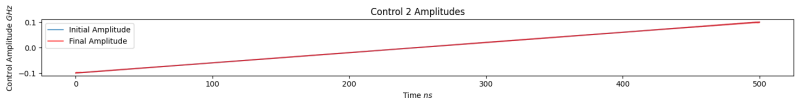

Figure 23: selective spin flip control 2 pulse amplitudes

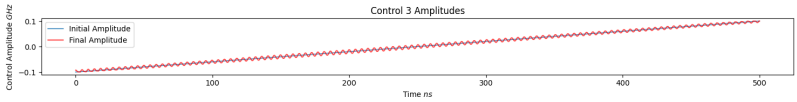

Figure 24: selective spin flip control 3 pulse amplitudes

Forward simulation is given below to ensure that pulse optimization has run properly. Simulation gives:

- sesolve final fidelity: 0.9992259750482354
- cavity ptrace fidelity: 0.9992525011412332
- qubit ptrace fidelity: 0.9992706948956261
- self-implemented final fidelity: 0.9999937323329505

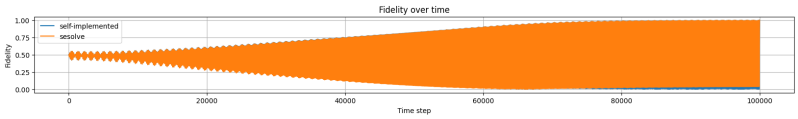

Figure 25: selective spin flip forward simulation

3.6 Further discussions reagarding numerical pulse optimzation

1. choosing time parameters:
   - evolution time (evo_time): pulse length needed can be estimated
   - number of time slices (n_ts): n_ts that is too small will lead to large error; n_ts that is too large will result in long computation time; hence an appropriate n_ts value needs to be tested via trial and error.
2.

4 Physical realizability

go back the Hamiltonian massagiing: Perturbation, frame change, throwing away small cross terms, exact compare fidelity

5 Future work

6 Conclusion