Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Comparing the Effects of Markovian and Non-Markovian Dynamics on Average Gate Fidelity Using Process Tensors

A Dissertation Submitted to Trinity College Dublin
in partial fulfilment of the requirements for the degree of
Quantum Science and Technology (M.Sc.)

Abstract

Within the study of realistic quantum technologies, interaction with the environment is inevitable which leads to the introduction of dissipation and decoherence into the system. While Markovian models assume that dynamics are memoryless, non-Markovian processes take environmental memory into account enabling information to flow back into the system. This dissertation seeks to explore if the effects of memory-backflow can be harnessed in such a way that the average gate fidelity of the system with a target unitary gate is increased. Using the OQuPy simulation framework, we first construct a process tensor that allow us to retain the environments memory, and then connect it to a bath of bosonic harmonic oscillators to mimic real world noise. From here superoperators are then extracted in matrix form which allows for the conversion into the Choi matrix on which we can perform Kraus decomposition. This allows us to compute the average gate fidelity of our noisy quantum system with a target unitary gate. Throughout our results it is shown that the introduction of non-Markovian memory significantly enhances both the dynamics of the Bloch vector components, allowing them to remain more stable under decoherence, but also the average gate fidelity between our quantum channel and the target unitary gate, allowing the channel to more consistently mimic the desired operation for longer periods of time. These results are consistent on all time scales tested, though Lamb-Shifts and biased memory-backflow may dampen these effects in more complex scenarios. These findings suggest that non-Markovian dynamics, when properly modelled and controlled, could serve as a useful resource when trying to improve the performance of quantum gates in more realistic, noise prone quantum technologies.

Keywords:
Open quantum systems, Markovian dynamics, non-Markovian dynamics, process tensors, memory backflow, average gate fidelity, Kraus decomposition, Choi matrix, superoperators, decoherence, bosonic bath, quantum noise, OQuPy simulations, Bloch sphere dynamics, Lamb shift.

Author: Aidan Doyle

Supervisor: Prof. Paul Eastham

School of Physics
Trinity College Dublin
July 2025

Declaration

I hereby declare that this dissertation is entirely my own work and that it has not been submitted as an exercise for a degree at this or any other university.

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at http://www.tcd.ie/calendar.

I have completed the Online Tutorial on avoiding plagiarism *Ready Steady Write*, located at http://tcd-ie.libguides.com/plagiarism/ready-steady-write.

I consent / do not consent to the examiner retaining a copy of the thesis beyond the examining period, should they so wish (EU GDPR May 2018).

I agree that this thesis will not be publicly available, but will be available to TCD staff and students in the University's open access institutional repository on the Trinity domain only, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

Signed: Aidan Doyle Date: 18/07/2025

I. INTRODUCTION

A. Open Quantum Systems

Studying open quantum systems is incredibly important for bringing the idealized theory behind quantum science closer to the incredible complexity of developing a real world quantum computer. No real world quantum system is going to be perfectly isolated and interactions with a system's environment are going to introduce dissipation, decoherence and noise [5]. Thus being able to understand, model and manipulate these effects is crucial for research and practical quantum engineering.

To go about solving an open quantum system, we will need to introduce the Hamiltonian that gives us information about the system H_S, information about the environment H_B, and information about any system-environment interaction H_I. This gives us a total Hamiltonian [3] which can be written as:

$$H_{total} = H_S + H_B + H_I. \tag{1}$$

The system we will be looking at will always be a two level system with two pseudo spin levels $|\uparrow\rangle$ and $|\downarrow\rangle$ [1]. This corresponds to a system described by a two dimensional Hilbert space. The system Hamiltonian will always be some variation of the form:

$$H_S(t) = h_x(t)\hat{S}_x + h_y(t)\hat{S}_y + h_z(t)\hat{S}_z, \tag{2}$$

where h_x, h_y and h_z are all some pre-specified time dependent control fields [1] that guide the evolution of the quantum state, and $\hat{S}_x$, $\hat{S}_y$ and $\hat{S}_z$ represent the spin half operators along the x, y and z directions. $H_S(t)$ is the part that we are most interested in, as its the part that we have the most control over and it directly determines how the system interacts.

The environment I will be using throughout this paper will be a large reservoir of bosonic harmonic oscillators, represented by the Hamiltonian:

$$H_B = \sum_q \omega_q b_q^\dagger b_q, \tag{3}$$

where $b_q^\dagger$ and b_q are the real space bosonic creation and annihilation operators and ω_q is the frequency of each given harmonic oscillator [6].

The system-bath interaction Hamiltonian, as the name suggests, describes how the system couples to the surrounding environment. For our standard bosonic bath, this takes the form:

$$H_{SB} = \hat{O}\sum_q (g_q b_q + g_q^\star b_q^\dagger) \tag{4}$$

where $\hat{O}$ is a Hermitian operator acting on the system, and the constants g_q quantify the strength at which each of the oscillators couples to the system [7]. For the rest of this paper, we choose the coupling of the system and the environment to be through the operator $\hat{S}_z$.

When simulating the influence that an environment has on a system, it is typical not to track the individual g_q constants [7], and instead we replace the discrete sum with an integral weighted by the spectral density $J(\omega)$:

$$J(\omega) = \sum_q \left|g_q\right|^2 \delta(\omega - \omega_q). \tag{5}$$

This equation encodes both the distribution of the bath modes but also how strongly each bath mode couples to the system. Rather than specifying each individual g_q and ω_q, it is instead easier to redefine $J(\omega)$ in such a way that it represents their collective behaviour. For this the spectral density form is:

$$J(\omega) = \alpha \frac{\omega^\zeta}{\omega_c^{\zeta-1}} \times f_c, \tag{6}$$

where α represents the coupling constant, ω_c is the cutoff frequency, ζ is the cutoff sharpness and f_c defines the cutoff function for the spectral density [7]. For this paper we will be using the Gaussian cutoff function which can be shown as:

$$f_c = \exp\left(-\frac{\omega^2}{\omega_c^2}\right). \tag{7}$$

B. Markovian Systems

Markovian and non-Markovian systems represent two different ways that an open quantum system could evolve. In the study of open-quantum systems, Markovianity implies that the environment connected to an open Markovian system retains no memory. A Markovian system evolves in such a way that every future state has no dependence on any of its previous states [8], and only looks at the current state for evolution. The Markovian approximation allows us to simplify the complicated dynamics of an open quantum system interacting with its environment into a realistic framework where the evolution can be described by time-independent equations [3]. The fact that the system does not retain information creates an idealized description of the dynamics within open quantum system which allows for the development and derivation of master equations with a much neater structure and widespread application.

The focal point of Markovian dynamics is the divisibility condition, that states that for all times $t_2 > t_1 > t_0$, [9] the evolution between two points in time can be described by the successive application of Completely Positive Trace Preserving (CPTP) maps:

$$\mathscr{E}(t_2, t_0) = \mathscr{E}(t_2, t_1)\mathscr{E}(t_1, t_0). \tag{8}$$

This requirement guarantees that the system remains within physical expectations, and a system that satisfies this requirement is said to undergo "Markovian evolution".

The most general form for the Markovian master equation is of the form:

$$\frac{d\rho(t)}{dt} = -i\left[H, \rho(t)\right] + \sum_k \gamma_k \left(L_k \rho(t) L_k^\dagger - \frac{1}{2}\left\{L_k^\dagger L_k, \rho(t)\right\}\right), \tag{9}$$

known as the Gorini-Kossakowski-Sudarshan-Lindblad equation [3]. Here we have H as the self-adjoint Hamiltonian, L_k are the Lindblad operators and γ_k are the rates that determine the strength of each process. This equation ensures that the evolution map of a density matrix $\rho(t) = \mathscr{E}_t[\rho(0)]$ is CPTP for all time t [3].

A key part of Markovian evolution is its irreversibility [10]. Although the entire system evolves unitarily, the reduced dynamics of the open system produce loss of coherence and decay in quantum correlations. This allows us to capture behaviour of the system without the need to track the combined system-environment effects, and instead allows us to describe the dynamics solely in terms of the density matrix. This irreversibility does mean however that any coherence or entanglement lost to the environment is permanently irretrievable, creating effectively an exponential relaxation towards a steady state [11][12]. This highlights a major challenge with Markovian systems in regards to building quantum technologies, that being the issue of how we protect the sensitive quantum information from the effects of Markovian noise.

While Markovian models are widely used in fields like quantum optics, quantum measurement theory and quantum information, the simplicity of Markovian systems can be misleading. Examples such as spontaneous emission into a vacuum show when a Markovian approximation holds as the environment has no memory and information is lost permanently [13]. In an environment with longer memory times however, like a structured reservoir or a spin environment, or when the coupling between the system and environment is strong, the assumptions around the Markovian model collapse. In these cases we can have information backflow, which will lead to the breakdown of the divisibility condition and thus non-Markovian behaviour [14]. For the design of realistic quantum technologies, using a non-Markovian description will produce a more accurate interpretation of the system.

C. Non-Markovian Systems

In practice, as much as we would like our systems to be closed, isolated and real, quantum systems are often noisy and suffer from decoherence. Non-Markovianity arises naturally in a physical setting, particularly when the coupling to the surrounding environment is strong, or when the environment retains information for significant periods of time [14].

In contrast to Markovian systems, non-Markovian evolution is inherently subject to memory effects. The future state of the system depends on both the current state of the system, but also on its history and the previous interactions with the environment. This implies that memory that has been lost to the environment can return to the system, leading to a potential uptick in coherence or entanglement [15]. One of the main components of a non-Markovian system is a phenomenon known as information backflow [16]. Information backflow can be observed in quantities like the coherence, the purity or the distinguishability between quantum states. Within a Markovian system, these quantities tend to decay uniformly. In non-Markovian systems however, these quantities can periodically increase (or decrease) as the system receives information back from the environment.

The presence of non-Markovianity in quantum systems has significance both physically and theoretically. On the one hand it makes theoretical models and simulations far more complicated [20]. A system cannot be described by a simple time-local equation anymore, rather it requires approaches that account for memory, like the process tensor formalisms [21]. Non-Markovian systems also require a far higher computational cost when compared to their Markovian counterparts. Simulating non-Markovian dynamics involves storing and updating information about a system's state over a long period of time, which increases memory demands and makes such simulations computationally expensive. On the other hand, non-Markovianity can be harnessed to create temporary revivals in coherence or entanglement by exploiting information backflow, and enabling a more realistic modeling of a quantum computer where ignoring the environmental memory isn't possible.

D. Process Tensors

It is often difficult to model memory effects of non-Markovian dynamics on a system, especially since they aren't accounted for in standard approaches (like time-local master equations). Thus a mathematical framework known as process tensors [17] was developed to address this challenge, and create an operationally defined description of non-Markovian dynamics.

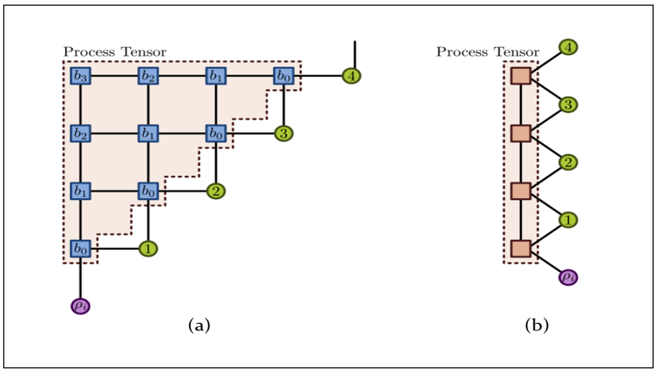

Fig. 1. (*a*): A graphical depiction of how a process tensor is constructed by contracting the set of influence tensors b_k into a two dimensional tensor network. Each horizontal row of influence tensors corresponds to a single time step. (*b*): The final contracted form of the process tensor represented as a Matrix Product Operator (MPO). Adapted from [1].

The process tensor as shown above builds in the influence of an environment over time on a quantum system. It is constructed by contracting the set of influence tensors b_k, which encode the environment [22] into a two dimensional tensor network in which all the dynamics are represented. As is visible, at each time step there is a layer of influence tensors that connect to its recent history, and the number of those connections is determined by the memory cutoff, which denotes how far back in time an environment retains memory.

It should be noted that the process tensor itself is independent of the system and its control operations. More specifically, the influence tensors b_k which encode the bath correlations can be separated from the system and pre-contracted into a reusable object, allowing for it to be plugged into any relevant simulation. The final output is a vertical stack of tensors, forming a Matrix Product Operator (MPO), as shown in Figure 1, which neatly represents the baths influence across every time step.

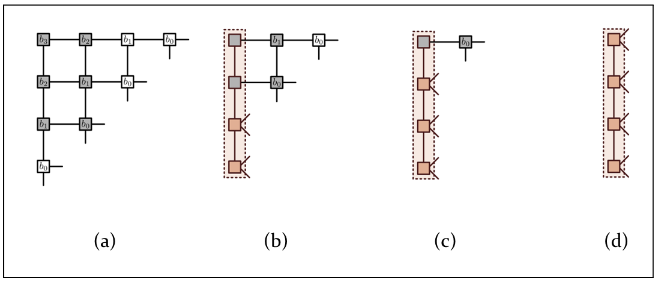

Fig. 2. Shows the contraction of a process tensor at each time step, starting with an uncontracted influence tensor network. After each contraction, a Singular Value Decomposition (SVD) is performed on the area highlighted in brown to prevent exponential growth of the bond dimensions and keep the process tensor computationally efficient. Adapted from [1].

To contract the process tensor, the influence tensors from the next time step are absorbed into the network, contracting iteratively as shown above in Figure 2 (a-d). There is the possibility however that the legs connecting each tensor, known as the bond dimension, can grow exponentially when directly stacking these tensors, and thus a compression step needs to be applied after each contraction. This compression is called Singular Value Decomposition (SVD), and is performed vertically on the system at each time step to remove redundant correlations. The areas where this truncation is applied is shown in brown in Figure 2. By the final step we obtain the vertical Matrix Product Operator (MPO) that encodes the influence of the bath up to a desired time step. Then this process tensor can be reused over and over again testing different control sequences in non-Markovian systems where the dynamics depend on past interactions.

II. Methods

A. Building The Process Tensor

For this work, our process tensors were generated using the OQuPy code [23]. In this code, there are a number of constants that need to be defined and objects that need to be created before a process-tensor can be built, such as the bath coupling correlations and the various parameters.

To build a process tensor using OQuPy, one of the main components needed is the bath and the bath correlations. As stated before our system is coupled to an environment modelled by a bosonic bath. This bath is made up of an infinite collection of independent quantum harmonic oscillators that together encode the degrees of freedom of the environment and how they interact with the system [24]. We defined our bath Hamiltonian and our interaction previously in Equation 3 and Equation 4. The correlations function makes use of OQuPy's `PowerLawSD()` attribute which creates a power-law spectral density defined in Equation 6. It takes five important inputs which define how the environment interacts with the system. Firstly, there is the coupling constant, denoted by α. This defines how the simulated environment will inject noise into the system over time. To decouple the system from the environment entirely, we can set $\alpha = 0$, which in effect gives us a closed quantum system to be used for verification purposes. Next, there is the cutoff frequency, denoted ω_c, which limits the maximum possible frequency of bath modes that can couple meaningfully to the system. Then there is the cutoff type which defines what the cutoff function looks like. OQuPy takes many variations of the cutoff function, however the one we are interested in is the one shown in Equation 7 which is of the Gaussian form. Furthermore, we have the cutoff sharpness, denoted by ζ. This is here to help control how smoothly or sharply the bath stops contributing beyond a certain energy point. Finally, there is the temperature T. This controls how "hot" the bath is which is important for creating realistic experimental conditions.

The correlation function is then plugged into OQuPy's `bath()` attribute, along with the system coupling operator $\hat{S}_z$, which couples the bath along the z-axis of the Bloch sphere. This tells OQuPy how the system couples to the environment, representing the interaction Hamiltonian.

The final components needed before creating the process tensor are the so-called tempo parameters. Using OQuPy's `TempoParameters()` attribute, these parameters decide how the Time-Evolution Matrix Product Operator (TEMPO) algorithm, partitions and compresses the influence of the bath over time. It takes three inputs, the first of those being the time step dt. This is used to discretize the system's evolution, with a smaller dt leading to more accurate dynamics but also an increase in memory and thereby increase in the cost of computation. Secondly we have the cutoff memory time t_{cut}. This defines how far back in time the system's memory is kept in the simulation. The smaller the cutoff, the closer the system gets to Markovian behaviour. Thirdly (and lastly) we have the relative error tolerance. As the name would suggest, it describes the relative error tolerance for tensor compression in the tempo algorithm, and is there to control the trade-off between performance and accuracy. Lower values increase precision in the results but likewise will increase the total computational cost. Together, these parameters control both the memory depth and the numerical stability of the simulation, which allows the TEMPO algorithm to capture the effects of non-Markovian dynamics efficiently while balancing the computational costs.

To build the process tensor, we used the OQuPy attribute named `pt_tempo_compute()` which uses the tempo algorithm to effectively mimic the dynamics of a non-Markovian system. The function takes the bath and the tempo parameters as defined before, and also takes the total time interval (the start and the end time). Internally, `pt_tempo_compute()` divides the evolution into a number of different time steps, each of length dt. From here it calculates the effect that the environment has on the system at each time step (it does this using the baths correlation function), and compresses this data into a collection of influence tensors b_k. These are then combined using the system controls , to build a `SimpleProcessTensor` which can be exported and reused as a representation of the environments actions over time.

B. Calculating The Fidelity

Within the study quantum information, the fidelity of a quantum channel serves as a great measure to show how closely it replicates a desired operation like a quantum gate. This is particularly important within a non-Markovian system when trying to compare an often noisy or decoherent evolution to the respective ideal gate. To calculate the average gate fidelity between a given dynamical map $\mathcal{E}(\rho)$ and a target unitary U_t, we need to recall that the Kraus representation theorem states that any dynamical CPTP map can be expressed in the form:

$$\mathcal{E}(\rho) = \sum_{i=1}^{D} K_i \rho K_i^\dagger, \tag{10}$$

where the Kraus operators $\{K_i : i = 1, ..., D\}$ satisfy the completeness relation:

$$\sum_{i=0}^{D} K_i^\dagger K_i = \mathbb{1}. \tag{11}$$

Given this, as derived in Pederson et al. [2], we are now able to calculate the average fidelity of a quantum channel with a target unitary gate such that:

$$\begin{aligned} F &= \int_S \langle\psi| U_t^\dagger \mathscr{E}(|\psi\rangle\langle\psi|) U_t |\psi\rangle \, dv, \\ &= \sum_i \int_S \left|\langle\psi| M_i |\psi\rangle\right|^2 dv, \\ &= \frac{1}{D(D+1)} \left\{ Tr\left(\sum_i M_i^\dagger M_i\right) + \sum_i |Tr\,(M_i)|^2 \right\}, \end{aligned} \tag{12}$$

where $M_i = U_t^\dagger K_i$, and K_i are the Kraus Operators for the map $\mathscr{E}(\rho)$.

To obtain the relevant Kraus operators from a given OQuPy simulation, I first had to extract the super-operator that represented the given quantum channel. This could then be converted into the Choi matrix, which is then diagonalized to perform Kraus decomposition and obtain the Kraus operators.

A super-operator is a representation of how a quantum state described by a density matrix evolves due to a quantum process. When the system is coupled to an environment, the super-operator includes the regular unitary evolution, but also incorporates decoherence, dissipation and entanglement. To give it a more formal definition, a super-operator is a linear operator that produces an output matrix ρ_{out} for a given input density matrix ρ_{in} such that $\mathscr{E}(\rho_{in})$ = ρ_{out}.

To be able to make use of this super-operator, and to be able to turn it into the Choi matrix, we need to be able to express it in matrix form. This matrix representation relies on a process known as vectorization, where a D^2 dimensional column vector is created by flattening a $D \times D$ dimensional density matrix by stacking the rows (or columns) on top of each other. The linear map $\mathscr{E}$ then acts on the newly flattened vector as a $D^2 \times D^2$ matrix denoted by $\hat{S}$, known as the super-operator matrix, which satisfies:

$$Vec(\rho_{out}) = \hat{S} \cdot Vec(\rho_{in}). \tag{13}$$

Using this I attempted to extract the super-operator matrix for the given quantum channel that I simulated using OQuPy. The first step involved constructing a complete and linearly independent basis set of density matrices for any single qubit system. They must span the full operator space to fully capture the the action of the quantum channel. Incomplete or non-independent density matrices fail to properly comb through all the required degrees of freedom, making it impossible to properly reconstruct the superoperator. We choose our basis such that:

$$\begin{aligned} \rho_{00} &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad \rho_{01} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \\ \rho_{10} &= \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \quad \rho_{11} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}. \end{aligned} \tag{14}$$

This is a complete and linearly independent basis that spans all 2×2 Hermitian matrices and allows for any arbitrary input state to be represented as some linear combination of basis elements. Then using OQuPy's `compute_dynamics()` function, the process tensor is applied to each of the input basis states. This evolves each of the basis states from an initial time to a final time, where for each input I extracted the final output density matrix which are then vectorized.

By stacking these resulting output vectors as columns, I was able to construct a 4×4 matrix that represents the super-operator $\hat{S}$ in its vectorized form. This matrix allows us to see how the quantum channel evolves any input density matrix to its corresponding output state, but more importantly for us, grants us the ability to create the Choi matrix out of the super-operator. This gets us one step closer to Kraus decomposition, and there with the ability to calculate the average fidelity of a quantum channel with a target unitary gate.

The Choi matrix is a mathematical structure that is derived from the Choi-Jamiolkowski isomorphism, which defines an isomorphic relationship between quantum channels and operators on a doubled Hilbert space. The Choi matrix can be constructed by applying the designated quantum channel to half of a maximally entangled state, and applying the identity to the other half. The outputted operator encodes the action of the whole channel and can be used to analyze its properties.

The super-operator and the Choi matrix are effectively equivalent under index shuffling [4], which can be seen in Figure 3. Thus to obtain the Choi matrix from the super-operator $\hat{S}$, which is currently in the form of a 4×4 matrix, we first need to reshape $\hat{S}$ into a rank-4 tensor with the indices (i, j, k, l) These indices correspond to the matrix elements of the

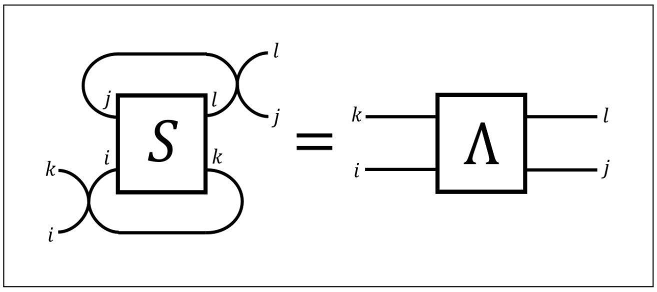

Fig. 3. A visual representation of the index shuffling that defines super-operator Choi matrix equivalence, adapted from [4].

input and output operators. Then, the indices of the tensors are permuted such that:

$$\begin{matrix} S \\ i \\ j \\ k \\ l \end{matrix} \quad \longrightarrow \quad \begin{matrix} \Lambda \\ i \\ k \\ j \\ l \end{matrix} \tag{15}$$

The reshuffling step reorders the rank-4 tensor such that $(i,j,k,l) \to (i,k,j,l)$. Then finally the reshaped tensor is re-flattened back into a 4×4 matrix to produce the Choi matrix.

A Kraus operator is an element of a set of matrices that help to represent the action of a quantum channel, and how it will evolve a given density matrix. We know from Equation 10 that we can represent a quantum channel using the Kraus operators. Now that we have constructed the Choi matrix, we can start to perform Kraus decomposition on it.

The Choi matrix fully characterizes the given quantum channel, and thus by diagonalizing it, we can obtain the eigenvalues and eigenvectors that define the set of Kraus operators [4]. Each of the Choi eigenvectors $|K_i\rangle$ correspond to a flattened version of the Kraus operator K_i, which represents a specific linear transformation that the quantum channel can apply to a system. The correlated eigenvalue λ_i then shows the relative strength of the related Kraus operator, illustrating how much the particular transformation contributes to the whole channel. For eigenvectors with an eigenvalue of $\lambda_i = 0$, their Kraus operators do not provide any meaningful transformation to the quantum channel, and are thus discarded.

Mathematically, given a Choi matrix Λ, with eigenvalues $\{\lambda_i : i = 1,...,D\}$ and eigenvectors $\{|K_i\rangle, i = 1,...D\}$, the Kraus operators are given as [4]:

$$K_i = \sqrt{\lambda_i}\, unvec(|K_i\rangle). \tag{16}$$

To implement this equation, I started by calculating the eigenvalues and eigenvectors using a Hermitian eigen-decomposition. All Choi matrices are guaranteed to be positive semi-definite for any valid quantum channel (find a paper to reference here), and thus we do not need to worry about imaginary or negative eigenvalues. Next, I looped over all the separate eigenpairs and discarded any with an eigenvalue of zero (in reality it was below a computational threshold to account for numerical errors). Each of the leftover eigenvectors were then reshaped into square matrices (stacking their columns as before) to recover the corresponding Kraus operators. Finally they are then each scaled by the square root of the eigenvalue as required, returning the set of Kraus operators that construct the quantum channel.

Now with a full set of Kraus operators, using Equation 12 we are able to work out the average gate fidelity between a simulated quantum channel and a target unitary gate. The result F, shows how closely the action the quantum channel is to the ideal target unitary gate. An average fidelity of 1 would indicate that the quantum channel is replicating the ideal gate perfectly, while lower values will show to what degree the noise in the system has degraded the operation of the gate.

C. Overall Methods Summary

In summary, to measure how closely a non-Markovian quantum channel can replicate a target unitary gate, we start by constructing a process tensor in OQuPy. This process tensor describes the effect of the bosonic harmonic oscillator environment on the system over time. The time dependent Hamiltonian defined earlier, which incorporates time-dependent control fields to guide the system's evolution, then has the process tensor applied to it to evolve the system. By simulating the evolution on a complete set of input density matrices, a corresponding super-operator that describes the quantum channel can be extracted. This is then turned into the Choi matrix which then has Kraus decomposition performed on it. The outputted Kraus operators then give us the ability to compute the average gate fidelity, providing a measure of how good the noisy channel is at approximating the ideal target unitary gate.

III. Results and Discussion

A. Algorithm Testing

To be able to go through and discuss the results of our fidelity calculations, and go through the possible implications of any of the results, we first need to be sure that the method and algorithm being used is correct and produces reasonable results that we can interpret. In order to test the algorithm coded in python, we can input a known evolution with calculable outcomes and compare the outputs.

Our general parameterization of the time-dependent system Hamiltonian throughout this paper has been of the form:

$$H_S(t) = h_x(t)S_x + h_y(t)S_y + h_z(t)S_z. \tag{17}$$

This Hamiltonian however has three time-dependent control fields which makes the verification process significantly more complex and introduces extra computation which is not necessary here. To combat this we will start by looking at a simpler time-independent system Hamiltonian with verifiable answers to the fidelity calculation. For this case we set:

$$H_S(t) = S_x. \tag{18}$$

Here we have set $h_x = 1$ and have set $h_y = h_z = 0$.

To get an idea of what this Hamiltonian will physically be doing to the system, we will take a moment to describe the motion on the Bloch sphere. It should be noted that $\hat{S}_x = \hat{\sigma}_x/2$ and thus we can show that the unitary evolution of this Hamiltonian as time progresses is:

$$\begin{aligned} U(t) &= \exp(-iHt), \\ &= \exp(-iS_x t), \\ &= \exp(-i\hat{\sigma}_x t/2) = \exp(-i\hat{\sigma}_x \theta), \end{aligned} \tag{19}$$

with $\hbar = 1$. Physically, this corresponds to a rotation by an angle $\theta = t/2$ around the x-axis. This implies that any initial Bloch vector $\overrightarrow{r}(0)$ will be evolved according to:

$$\overrightarrow{r}(t) = R_x(\theta)\overrightarrow{r}(0), \tag{20}$$

where here, R_x is the 3-dimensional rotation matrix about the x-axis:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\left(\frac{t}{2}\right) & -\sin\left(\frac{t}{2}\right) \\ 0 & \sin\left(\frac{t}{2}\right) & \cos\left(\frac{t}{2}\right) \end{pmatrix}, \tag{21}$$

and thus the quantum state evolves by rotating around the x-axis of the Bloch sphere with a constant angular velocity.

Setting our initial state to the down state, the evolution around the Bloch sphere can be pictured as in Figure 4:

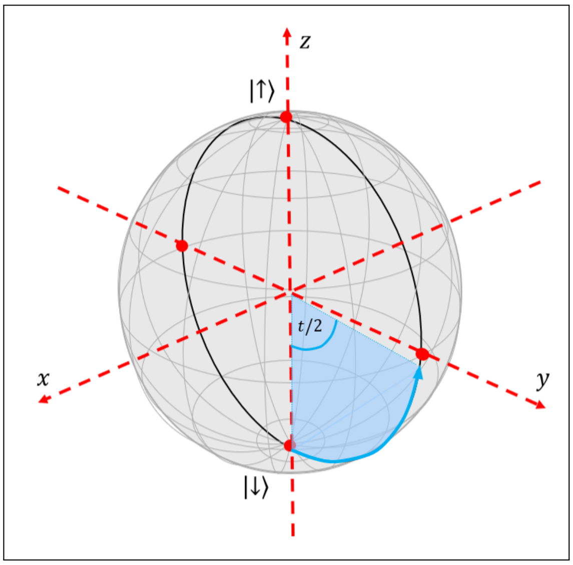

Fig. 4. A visual representation of the behaviour of the chosen Hamiltonian on the Bloch sphere. Starting in the initial state $|\downarrow\rangle$ and rotating around the x-axis.

Here the initial state is rotated by an angle $t/2$ around the x-axis, staying constant in the yz-plane.

Before checking that our fidelity calculation produces reasonable results, we must first check whether the dynamics produced by the system output the expected behaviour. We want to check that from our initial state $|\downarrow\rangle$, that our values for $\langle\hat{\sigma}_y\rangle$ and $\langle\hat{\sigma}_z\rangle$ behave has we would expect. To do this we must work out what $\langle\hat{\sigma}_y\rangle$ and $\langle\hat{\sigma}_z\rangle$ are expected to be at any point in time. We know that:

$$\begin{aligned} \overrightarrow{r}(t) &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\left(\frac{t}{2}\right) & -\sin\left(\frac{t}{2}\right) \\ 0 & \sin\left(\frac{t}{2}\right) & \cos\left(\frac{t}{2}\right) \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}, \\ &= \begin{pmatrix} 0 \\ \sin\left(\frac{t}{2}\right) \\ -\cos\left(\frac{t}{2}\right) \end{pmatrix}, \end{aligned} \tag{22}$$

and thus with zero bath coupling, we should expect to see that $\langle\hat{\sigma}_x\rangle$ stays constant at 0, $\langle\hat{\sigma}_y\rangle$ should behave with a motion $\sin\left(\frac{t}{2}\right)$, and $\langle\hat{\sigma}_z\rangle$ should behave like $-\cos\left(\frac{t}{2}\right)$. Using OQuPy's `compute_dynamics()`, we are now able to plot the behaviour of each of the Bloch vector components (at zero bath coupling) on a graph for comparison.

As is clear from the Figure 5, we have that $\langle\hat{\sigma}_x\rangle = 0$, $\langle\hat{\sigma}_y\rangle = \sin\left(\frac{t}{2}\right)$ and $\langle\hat{\sigma}_z\rangle = -\cos\left(\frac{t}{2}\right)$ for all t. Therefore

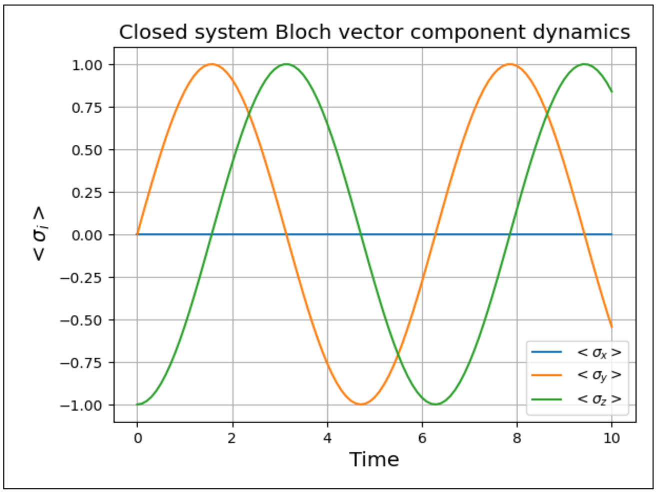

Fig. 5. Shows the closed system dynamics of the Bloch vector components $\langle\hat{\sigma}_x\rangle$, $\langle\hat{\sigma}_y\rangle$ and $\langle\hat{\sigma}_z\rangle$. The x-component stays constant at $\langle\hat{\sigma}_x\rangle = 0$, while both the y-component and z-component exhibit oscillatory behavior such that $\langle\hat{\sigma}_y\rangle = \sin\left(\frac{t}{2}\right)$ and $\langle\hat{\sigma}_z\rangle = -\cos\left(\frac{t}{2}\right)$ as expected.

showing us that at zero coupling our algorithm is outputting the correct dynamics.

To calculate the gate fidelity, we must have a target unitary gate for our fidelity calculations to try and mimic. We know from Equation 19 that the unitary evolution of our Hamiltonian can be expressed as $U(t) = \exp(-i\hat{\sigma}_x t/2)$. This can be expanded and rearanged such that we express it in terms of sine and cosine and thus get our target unitary matrix to be:

$$\begin{aligned} U(t) &= \cos\left(\frac{t}{2}\right)\mathbb{1} - i\sin\left(\frac{t}{2}\right)\hat{\sigma}_x, \\ &= \begin{pmatrix} \cos\left(\frac{t}{2}\right) & -i\sin\left(\frac{t}{2}\right) \\ -i\sin\left(\frac{t}{2}\right) & \cos\left(\frac{t}{2}\right) \end{pmatrix}. \end{aligned} \tag{23}$$

For simplicity reasons, I will set the time at which we are looking for our unitary gate to $t = \pi$. This is going to make our fidelity calculations significantly easier and less time consuming. Thus our target unitary gate for testing purposes becomes:

$$\begin{aligned} U(\pi) &= \begin{pmatrix} \cos\left(\frac{\pi}{2}\right) & -i\sin\left(\frac{\pi}{2}\right) \\ -i\sin\left(\frac{\pi}{2}\right) & \cos\left(\frac{\pi}{2}\right) \end{pmatrix}, \\ &= \begin{pmatrix} 0 & -i \\ -i & 0 \end{pmatrix}. \end{aligned} \tag{24}$$

Putting this into OQuPy, the fidelity at each timestep is then extracted with our target gate, and is then calculated and plotted:

From Figure 6 it is clear that our algorithm is outputting that the fidelity with the target unitary gate is maximized when $t = \pi$, as expected, and then oscillates with a period of $t_{period} = 2\pi$. There also seems to be no coupling with the system as we are able to reach a perfect fidelity.

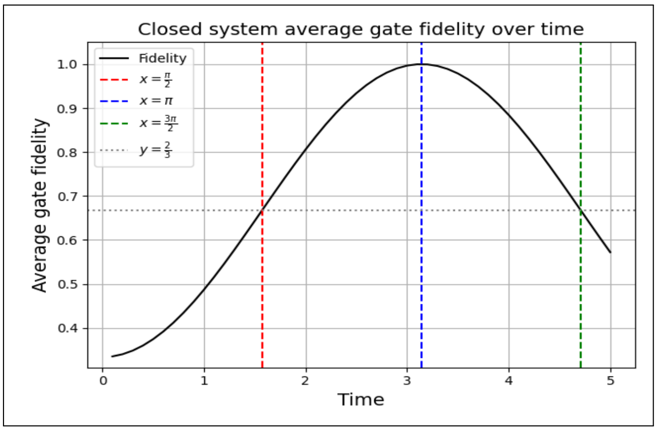

Fig. 6. The average gate fidelity with the closed system Hamiltonian with the target unitary gate. Fidelity oscillates between $F = 1/3$ and $F = 1$ as expected. The graph also contains four further constant lines at $x = \pi/2, \pi, 3\pi/2$ and $y = 2/3$ which are used for verification of results.

To check if the graph is correct, I explicitly calculated the fidelity at the maximum point, and two other points that create nice unitary matrices to see if the values matched up, however this process should also work for any other time t.

If a quantum operation is completely unitary, with no input from the environment, we can represent the quantum channel such that [3]:

$$\mathcal{E} = U\rho U^\dagger, \tag{25}$$

and thus the only non-zero Kraus operator for a unitary transformation is the unitary matrix itself. Therefore implying that if we can find the unitary matrix for the maximum and minimum fidelity times, we can substitute them into the fidelity equation and compare answers.

Substituting $t = \pi$ into Equation 23 we get our Unitary matrix and thus our singular Kraus operator to be:

$$U = \begin{pmatrix} 0 & -i \\ -i & 0 \end{pmatrix}. \tag{26}$$

Given this, we can now sub this matrix back into Equation 12, where we have $D = 2$ and:

$$\begin{aligned} M &= U_t^\dagger K \\ &= \begin{pmatrix} 0 & i \\ i & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & -i \\ -i & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \end{aligned} \tag{27}$$

Which means that $\sum_i M_i^\dagger M_i$ is just the identity, allowing us to calculate:

$$\begin{aligned} F &= \frac{1}{2(2+1)}\left\{2+|(2)|^2\right\}, \\ &= \frac{6}{6} = 1. \end{aligned} \tag{28}$$

Giving us a perfect fidelity at $t = \pi$ as expected.

In the same way, we can also use this to calculate if the fidelity at the other two chosen points is correctly placed on the graph. It should be clear that a fidelity of $F = 2/3$ occurs at $t = \pi/2$ and at $t = 3\pi/2$. For $t = \pi/2$, we get our unitary matrix and our required Kraus operator as:

$$K = \begin{pmatrix} \frac{\sqrt{2}}{2} & -i\frac{\sqrt{2}}{2} \\ -i\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}. \tag{29}$$

This again allows us to calculate:

$$\begin{aligned} M &= U_t^\dagger K, \\ &= \begin{pmatrix} 0 & i \\ i & 0 \end{pmatrix} \cdot \begin{pmatrix} \frac{\sqrt{2}}{2} & -i\frac{\sqrt{2}}{2} \\ -i\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} & i\frac{\sqrt{2}}{2} \\ i\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}, \end{aligned} \tag{30}$$

giving:

$$M^\dagger M = \begin{pmatrix} \frac{\sqrt{2}}{2} & -i\frac{\sqrt{2}}{2} \\ -i\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix} \cdot \begin{pmatrix} \frac{\sqrt{2}}{2} & i\frac{\sqrt{2}}{2} \\ i\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{31}$$

However, we have that $Tr(M) = \sqrt{2}$, and thus we get out that our fidelity to be:

$$F = \frac{1}{2(2+1)}\left\{2+\left|\left(\sqrt{2}\right)\right|^2\right\} = \frac{2}{3}, \tag{32}$$

which is the fidelity that is presented on our graph.

Repeating this process for $t = 3\pi/2$, we get:

$$K = \begin{pmatrix} -\frac{\sqrt{2}}{2} & -i\frac{\sqrt{2}}{2} \\ -i\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{pmatrix}. \tag{33}$$

Hence:

$$\begin{aligned} M &= U_t^\dagger K, \\ &= \begin{pmatrix} 0 & i \\ i & 0 \end{pmatrix} \cdot \begin{pmatrix} -\frac{\sqrt{2}}{2} & -i\frac{\sqrt{2}}{2} \\ -i\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} & -i\frac{\sqrt{2}}{2} \\ -i\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}, \end{aligned} \tag{34}$$

which gives:

$$M^\dagger M = \begin{pmatrix} \frac{\sqrt{2}}{2} & i\frac{\sqrt{2}}{2} \\ i\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix} \cdot \begin{pmatrix} \frac{\sqrt{2}}{2} & -i\frac{\sqrt{2}}{2} \\ -i\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \tag{35}$$

and again we have that $Tr(M) = \sqrt{2}$, giving us he exact same fidelity calculation as in Equation 32. Therefore we can conclude with confidence, that our algorithm is correctly extracting the superoperator, and calculating the fidelity when the system coupling is at $\alpha = 0$.

B. *Introducing Noise to Markovian dynamics*

The easiest way to observe how the environment could affect an open quantum system, is by increasing the bath coupling to the qubit such that the effects of noise and decoherence are observable. To do this, we start by setting $\alpha = 0.126$ [1]. This increases the environment coupling constant allowing OQuPy to introduce noise into the quantum system. A value of $\alpha = 0.126$ is chosen for our constant as it introduces a weak to middling environment coupling to the system. This value is large enough so that we can see visible decoherence, but also weak enough so that the system doesnt immediately thermalize and lose all quantum coherence. Initially we will also set the memory cutoff $t_{cut} = 0.1$ to introduce this noise into a Markovian like system. A value of $t_{cut} = 0.1$ is chosen as it minimizes the memory time of the bath as much as possible within OQuPy, which is essential for creating Markovian dynamics. We should observe a continual loss of information to the system which cannot be retrieved through phenomena such as memory-backflow.

First we will look at how our dynamics behave under this coupling constant using the same system Hamiltonian as defined before:

$$H_S(t) = S_x. \tag{36}$$

This will allow us to make comparisons and observe any changes in the behaviour of the system. With the defined parameters, our graph of the dynamics takes the form shown in Figure 7. It should be clear from Figure 7 that the behaviour of the Bloch vector

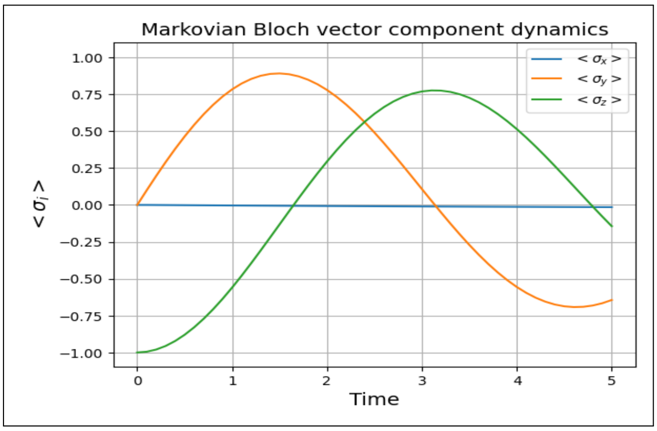

Fig. 7. Shows the Bloch vector component dynamics of a Markovian system coupled to a bosonic bath. Both $\langle\hat{\sigma}_y\rangle$ and $\langle\hat{\sigma}_z\rangle$ show significant signes of decoherence from their closed system counterparts. There is also a slight deviation in the x-component dynamics $\langle\hat{\sigma}_x\rangle$.

components differs dramatically from the behaviour seen when we extracted them from a closed system. Rather than behaving like a constant wave matching $\sin(\frac{t}{2})$ as before, we have that $\langle\hat{\sigma}_y\rangle$ exhibits amplitude damping as it no longer reaches ±1. Similarly, the dynamics of $\langle\hat{\sigma}_z\rangle$ also present a large amount of amplitude damping as they stray from the previously constant oscillation $-\cos(\frac{t}{2})$. There is also a small but noticeable deviation to the dynamics of the x component $\langle\hat{\sigma}_x\rangle$. This is insignificant however and is likely due to the memory cutoff not being completely set to zero because of restrictions within OQuPy. Take note however of the fact that there is no change in the period time, and rather the period of the oscillation stays constant between the closed quantum system and the Markovian system. This will be discussed later.

Graphically, on the Bloch sphere this would look something like Figure 8, where rather than a constant loop around the Bloch sphere in the yz-plane, the dynamics slowly dissipate information over time until all the quantum information is lost completely.

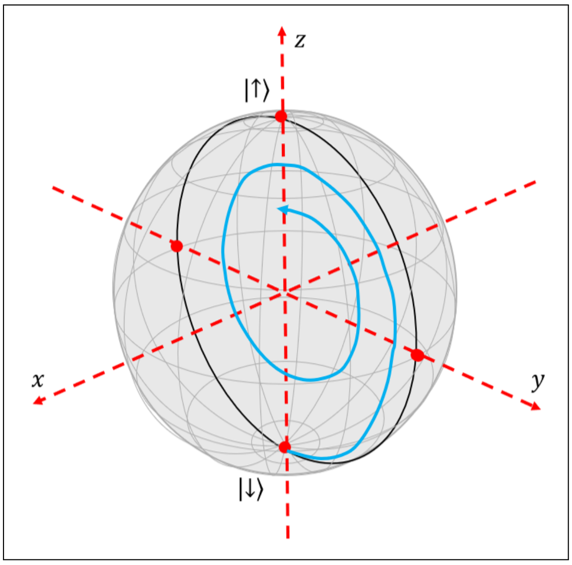

Fig. 8. A visual representation of the Bloch vector component dynamics for the Markovian system. As the system loses coherence, the individual components decrease which presents itself in the form of a slow spiral towards the centre of the Bloch sphere.

There is a similar trend when comparing the average gate fidelity of the closed quantum system to the one coupled to the environment. To give a better overall view of the change in fidelity, both graphs have been extended to $t = 10$.

As can be seen in Figure 9, there is a clear decrease in maximum average gate fidelity when coupling with the environment. This is in line with the change in dynamics as the system loses information to the environment and slowly diverges further and further from the behaviour of the target unitary gate. Interestingly enough however, there is an increase in the minimum average gate fidelity as time goes on, whereas the closed quantum system has a constant minimum fidelity of $F = 1/3$ which oscillates with a period of 2π. This makes perfect sense however. The minimum fidelity occurs when the quantum channel that is implemented by the system is maximally distinguishable from the chosen target unitary. In the case of a closed system, this takes place when the input gate that is being implemented rotates to a position that is orthogonal to the target unitary, giving the minimum average fidelity for single qubit unitaries. In the open quantum system however, the increase in the damping by the environment effectively shrinks the distance between the noisy channel and the target unitary when they are supposed to be orthogonal, which in turn increases the minimum average fidelity.

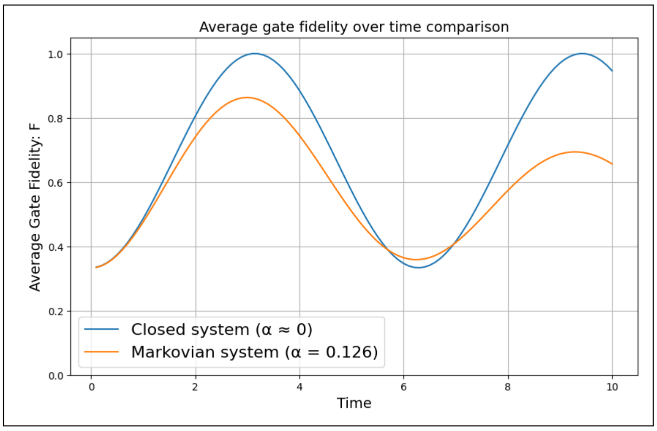

Fig. 9. Compares the average gate fideliy of the Markovian system coupled to the bosonic bath with the target unitary gate to the fidelity of the closed system with the target unitary gate. The Markovian system shows significant deviation from the ideal behaviour, with the introduction of noise causing significant dips in the maximum fidelity peaks.

C. Introducing Noise to non-Markovian dynamics

Having compared the effects of introducing noise into a Markovian system to the closed system results, we now will look at how introducing noise effects a non-Markovian system. To mimic non-Markovianity within OQuPy, and inject memory-backflow, we increase our memory cutoff time to $t_{cut} = 5.0$. This tells the TEMPO algorithm how far back in time it needs to keep track of the influence of the environment, which as established earlier is stored in the influence tensors b_k within the process tensor. We keep our coupling constant at a value of $\alpha = 0.126$ as before and we also use the same system Hamiltonian:

$$H_S(t) = S_x. \tag{37}$$

which is maintained for consistency purposes. The comparison between the non-Markovian system and the closed system will bare significant resemblance to the comparison of the Marokovian to the closed, and thus from here onwards we will be mostly focused on the differences between the Markovian and non-Markovian dynamics and average fidelities.

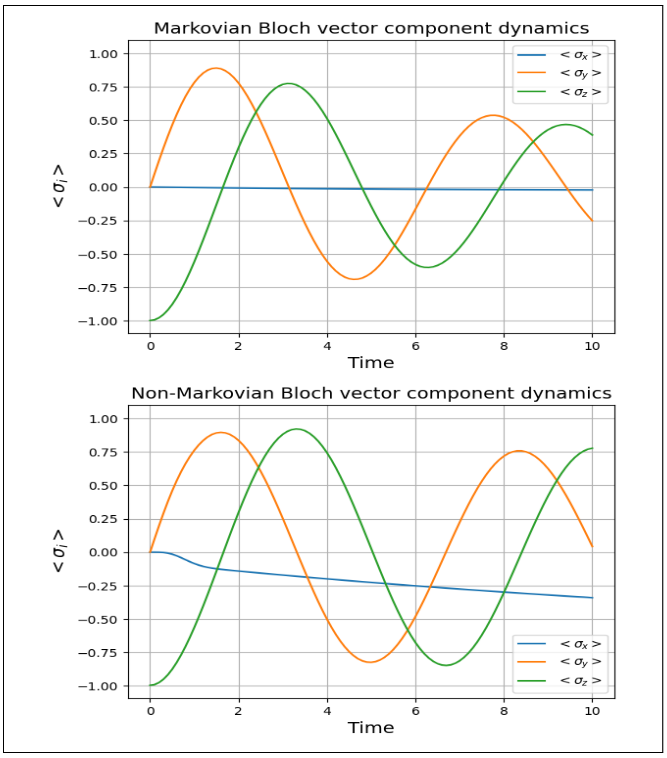

Fig. 10. Compares the dynamics of the Bloch vector components of the Markovian system to those of the non-Markovian system. The dynamics of the non-Markovian system increase significantly due to memory-backflow when compared to the Markovian. There is also the introduction of non-trivial dynamics to $\langle\hat{\sigma}_x\rangle$ in the non-Markovian system likely due to the influx of information previously lost to the environment.

Looking at Figure 10, it is clear to see that there is a significant difference in the behaviour of the Bloch vector components as time progresses. The length of time that the dynamics are simulated for is once again increased to $t = 10$ in order to make the visual differences more obvious. Starting with the behaviours of $\langle\hat{\sigma}_y\rangle$ and $\langle\hat{\sigma}_z\rangle$ respectively, with the introduction of non-Markovianity and memory-backflow, the amplitude damping is noticeably decreased. This diminished amplitude damping also seems to have a more significant effect on the z-component of the dynamics. This is likely because the system is coupled to the environment via $\hat{S}_z$, making the decoherence stronger in the z-direction. Thus any memory-backflow from the environment will have a proportionally greater effect on reviving $\langle\hat{\sigma}_z\rangle$.

Although small, both $\langle\hat{\sigma}_y\rangle$ and $\langle\hat{\sigma}_z\rangle$ exhibit a noticeable shift in the the period of their oscillation. This is likely due to a phenomenon known as a Lamb-Shift [18][19]. This is where for a qubit coupled to a bath, the Lamb-Shift induces a shift in the frequency of a wave, and thus the slight change in period. The reason the Lamb-Shift doesn't show up in the Markovian dynamics is because in Markovian scenarios, the Lamb-Shift is commuted out by the Hamiltonian of the system. In non-Markovian systems however, the Hamiltonian often doesn't commute thus introducing non trivial effects as shown.

The graph also displays that the introduction of memory-backflow has induced non-trivial dynamics in the x-component of the Bloch vector which were previously neglidgeable in the Markovian system. This jump in dynamics is not present until $t \approx 0.5$ as $\langle\hat{\sigma}_x\rangle$ initially behaves similarly to the Markovian dynamics, likely because there is initially no memory for the environment to return to the system. The comparison shows that the introduction of non-Markovianity and memory-backflow greatly increases the quantum memory retention of the system. This is most clear when comparing the length of the Bloch-vector as time progresses.

In Figure 11, we can clearly see that the Markovian Bloch vector is affected far more by the decoherence and dissipation of the environment coupling, thus suggesting that the non-Markovian system is far better at retaining quantum coherence over time.

Next, we will go about comparing the average gate fidelit between the Markovian and non-Markovian systems. No changes are made here to either the system or the constants mentioned before. In Figure 12 the graph of the two fidelies over time are plotted together.

As is clearly visible from the average gate fidelities presented in the plot, the maximum fidelity peaks in the non-Markovian average gate fidelity calculations are higher at the first and second maximum. For the first peak there is a difference of roughly 0.05 in fidelity, however by the second peak this gap in fidelity has opened to more than 0.12. It should be

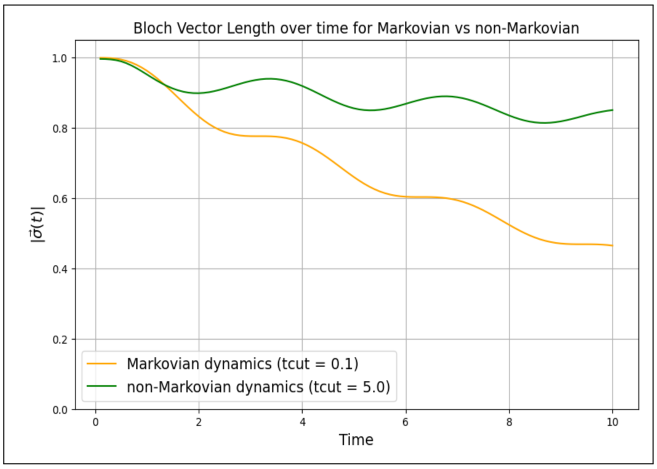

Fig. 11. Comparison of the Bloch vector length between the Markovian and non-Markovian systems, which indicates how quickly the dynamics of each system decay. The non-Markovian system retains it's Bloch vector length for a longer period of time due to memory-backflow.

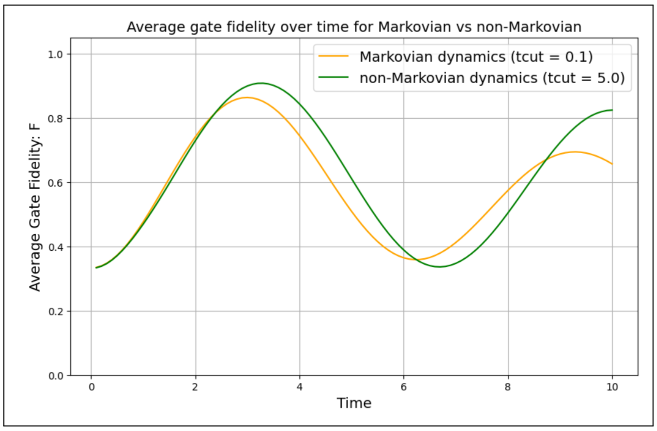

Fig. 12. Looks at the difference between the average gate fidelity of the non-Markovian system and the Markovian system with the target unitary gate. The comparison shows clearly that as time progresses the non-Markovian system is significantly less affected by decoherence. The non-Markovian system does however exhibit a change in frequency, likely caused by a Lamb-shift [18][19].

noted that the local minimum of the non-Markovian fidelity is also closer to the absolute minimum average gate fidelity than that of the Markovian system.

When combining these two observations, it seems to suggest that the introduction of non-Markovian dynamics and memory-backflow significantly increase the ability of the system to best mimic the target unitary gate. We can explicitly show this by changing the target unitary gate to a time dependent one that mimics the system Hamiltonian. Thus we set our new target Hamiltonian to be:

$$U_t(t) = \exp\left(-i\hat{\sigma}_x t/2\right). \tag{38}$$

This will allow us to visualize the effects of noise and decoherence over time on the Markovian and non-Markovian systems. The graph for the average gate fidelity will not be included here as it is just a straight line at $F = 1$. Again, it is clearly visible that the non-Markovian average gate fidelity decreases significantly slower than that of the Markovian system. While the non-Markovian system has a larger initial dip, the subsequent memory-backflow causes a shift in the rate of decoherence, leading to better retention of quantum information. This does not even account for the shift in frequency caused by the Lamb-Shift in the non-Markovian dynamics. Accounting for this shift would likely increase the disparity between the two fidelities even further. Effectively, this shows that the non-Markovian system used here is far better at mimicking the actions of a target unitary quantum gate than its Markovian counterpart.

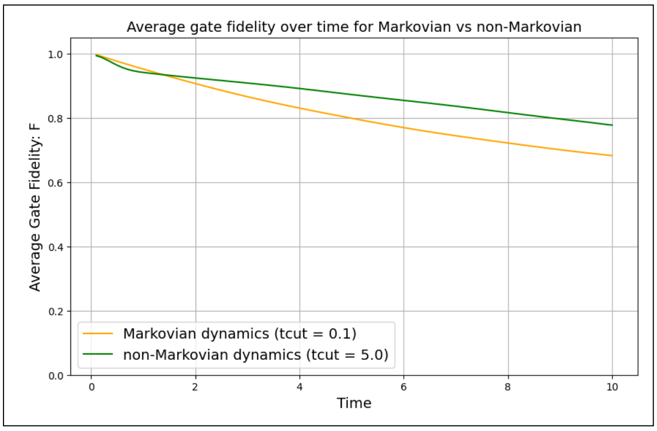

Fig. 13. Compares the average gate fidelity of the Markovian and non-Markovian system to a target unitary gate that attempts to match the behaviour of the system Hamiltonian. This shows how the non-Markovian system is better able to replicate the ideal gate behaviour, even with its frequency being Lamb-shifted.

Our next set of results will be to test and observe the effect on the fidelity as the memory cut-off t_{cut} is increased. This is done by taking values for t_{cut} between 0.1 and 5, when the system is at its most Markovian and most non-Markovian respectively, in increments of 0.1 and looping over them to calculate the average gate fidelity of the system with the target unitary gate. We choose the target unitary gate here to be as before when we showed the dissipation of the fidelity over time:

$$U_t(t) = \exp\left(-i\hat{\sigma}_x t/2\right). \tag{39}$$

The times chosen to take samples of the fidelity at are $t = 1, 3, 5$. This is so that we can assess how the change in t_{cut} affects the fidelity at three different stages of our system-environment coupling. Right at the start when the environment has only had time to put a small amount of noise in the system, then a little later when there is a bit more decoherence and then finally a time where there is noticeable decoherence in the system.

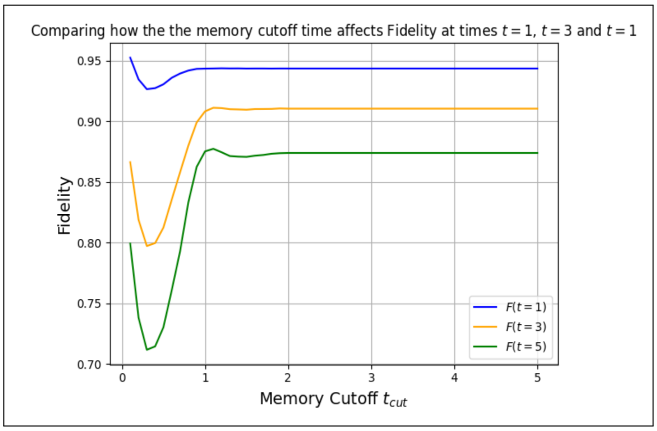

Fig. 14. Shows how the average gate fidelity with the target unitary gate $U_t(t) = \exp(-i\hat{\sigma}_x t/2)$ changes when compared across a range of possible memory cutoff times $0.1 < t_{cut} < 5$, at three different points in the evolution of the system $t = 1, 3, 5$. The plots all consistently show that an initial dip in fidelity followed by a significant spike when memory-backflow kicks in, followed by a platue likely caused by the chosen frequency cutoff $\omega_c = 3.04$.

The plot in Figure 14 gives us a clearer picture of what is happening to the average gate fidelity with the target unitary gate as the memory cutoff is increased. At all points in time, the fidelity starts off relatively high for $t_{cut} = 0.1$ but then takes an immediate plunge as the memory cutoff is initially increased to $t_{cut} \approx 0.3$. From there however it then drastically increases until roughly $t_{cut} \approx 1.1$ before it stabilizes and stays approximately constant. Comparing the graph for $t = 1$ against the two graphs for $t = 3$ and $t = 5$, we can propose that early on in a quantum system's lifetime, there is no real advantage to memory-backflow in the system. The fidelity at $t = 1$ is at its highest when $t_{cut} = 0.1$, although the difference is small. However when we get to the graphs for $t = 3$ and $t = 5$, the spike in fidelity due to the introduction of memory-backflow is undeniably significant, jumping up well past where the fidelity began at low values for t_{cut}. Thus from this we can conclude that early in a system's lifetime non-Markovianity has no real effect on the behaviour of the system. However as time progresses, the introduction of non-Markovian dynamics and memory-backflow help keep the system coherent and stop it from losing quantum information permanently.

Looking at the change in fidelity at all three points in time, it appears that a memory cutoff value of $t_{cut} \approx 1.1$ is optimal for this given system. This is long enough so that the memory-backflow has ample time to take affect and return lost information to the system, but small enough as to not cause a noticeable lamb-shift in the dynamics that will disrupt the fidelity.

After $t \approx 2$ at all three points in the system's evolution, the average gate fidelity is noticeably contant. This is caused by our choice of frequency cutoff ω_c which limits the maximum possible frequency of the bath modes that can couple to the system. The correlation function, which effectively describes the amount of time that the bath retains memory for, is directly proportional to the cutoff frequency. This means that our choice of cutoff frequency $\omega_c = 3.04$ allows the bath to retain memory for a period of $t \approx 2$, and thus increasing t_{cut} past that point has no further affect on the fidelity as there is no extra memory to introduce. An increase in the cutoff frequency would induce a faster decay in the correlation function and thus the environment would retain memory for less time. The opposite would be true if we decreased the cutoff frequency.

D. Comparison With More Complicated Systems

For comparison, I will now introduce a slightly more complicated system Hamiltonian, to see if the effect of introducing noise has any significant change from the slightly simpler system, and to check if non-Markovianity is more or less effective at preserving the average gate fidelity of the system with the target unitary gate.

We will define our slightly more complicated Hamiltonian to be of the form:

$$H_S(t) = S_x + S_y. \tag{40}$$

This Hamiltonian introduces more complicated dynamics than the rotation around a single axis while still producing predictable closed system behaviour which can be easily tested. The Dynamics created by this Hamiltonian are displayed in Figure 15 and show distinct oscillations in all three Bloch vector components. For the sake of convenience, in this section we will be setting the target unitary gate to be the identity, which represents the gate required

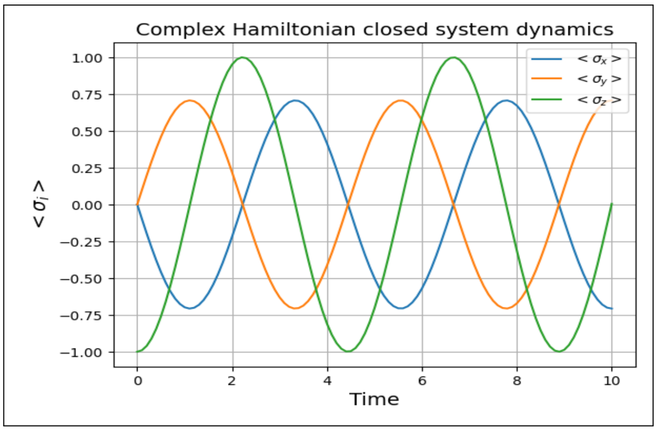

Fig. 15. Shows the closed system dynamics of the more complicated Hamiltonian $H_S(t) = S_x + S_y$.

to turn our initial state $|\downarrow\rangle$ back into itself:

$$U_t = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{41}$$

The system-environment coupling factor will remain unchanged from our previous calculations $\alpha = 0.126$, and thus introducing noise into the system gives us Figure 16, where the first plot shows us how the dynamics are affected in a Markovian system with $t_{cut} = 0.1$, and the second plot showing non-Markovian dynamics with the memory cutoff once again set to $t_{cut} = 5$. Similarly to previous in the slightly simpler system, there is a clear decrease in dissipation and decoherence in the non-Markovian system as memory-backflow reintroduces lost information to the system. Once again, the z-component of the Bloch vector also seems to be the greatest beneficiary of the of the memory-backflow as it disproportionately affected in the Markovian system for the same reasons as before. Curiously, both $\langle\hat{\sigma}_x\rangle$ and $\langle\hat{\sigma}_y\rangle$ exhibit a negative shift introduced by the memory-backflow. This suggests that the bath is feeding the information back in a biased way, likely caused because the environment is coupled to the system via $\hat{\sigma}_z$, but rotates in the xy-plane, which could potentially create a preferred damping direction. However this will need further investigation.

There is also a noticeable shift in the oscillation period of the Bloch vector components for the non-Markovian dynamics once again. Similarly to before, this is likely caused by the phenomenon known as a lamb-shift [18][19] which induces a shift in the frequency of a wave for a qubit coupled to a bath.

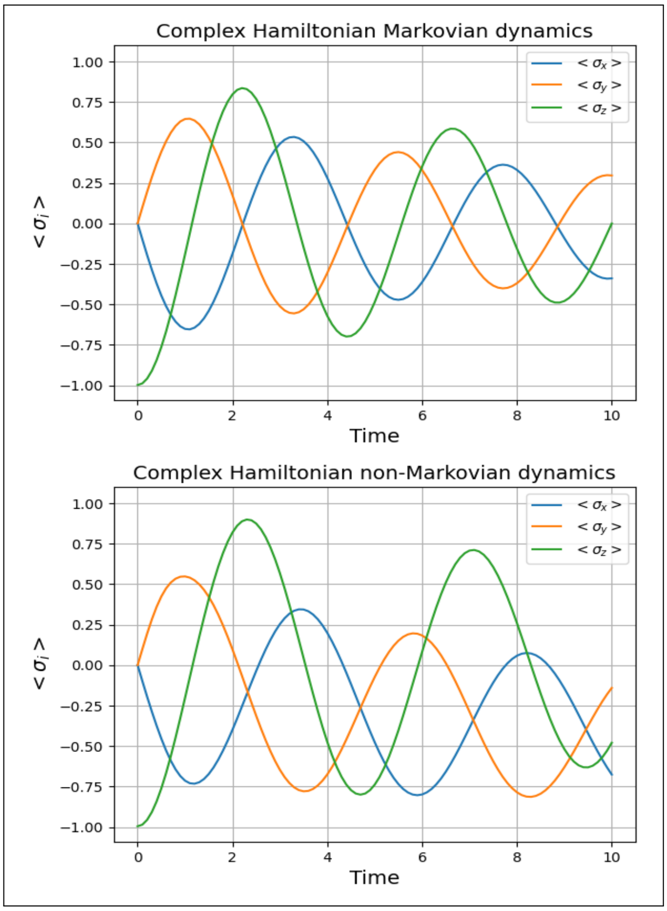

Fig. 16. Shows the dynamics of the Bloch vector components of the more complicated Markovian and non-Markovian systems connected the a bosonic bath simulating the environment. The Markovian system, as before, shows significantly more decoherence in its dynamics than the non-Markovian system, particularly in the z-component. While the memory-backflow does seem to preserve the dynamics of the non-Markovian system better, both $\langle\hat{\sigma}_x\rangle$ and $\langle\hat{\sigma}_y\rangle$ show a negative shift suggesting that the memory is being re-introduced biasedly by the environment.

When comparing the average gate fidelities between the Markovian and non-Markovian systems, we can see similar trends in Figure 17 to what we have observed before.

Similar to before in the less complicated system, the peaks in average gate fidelity once again show an increase for the non-Markovian system from that of the Markovian. While the difference is less significant than before, there is still a noticeable increase in maximum peaks and decrease in minimum peaks, allowing the non-Markovian dynamics to better mimic the identity which was set as our target unitary gate for these calculations.

While the more complicated system does present similar results as captured in the system with the simpler Hamiltonian, further research and testing needs to be done on whether complicating the

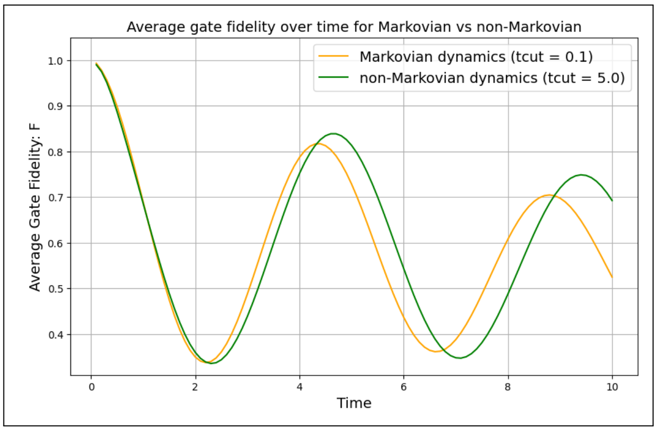

Fig. 17. Compares the average gate fidelity of Complex Hamiltonian Markovian and non-Markovian systems with the unitary target gate chosen here to be the identity. The non-Markovian system still exhibits less decoherence however the effect is diminished from the simpler system suggesting that the added complexity could mitigate the effects of the memory-backflow.

Hamiltonian can introduce enough Lamb-shifting of the frequency or biased memory-backflow to offset the benefits of non-Markovian dynamics.

IV. Conclusion

Overall, we successfully showed that the introduction of non-Markovian dynamics and memory-backflow increases the average gate fidelity of the system with a target unitary gate. By constructing a process tensor using OQuPy, we were able to model memory effects for different memory cutoff times and thus giving us the ability to test the effects of Markovianity and non-Markovianity against a closed quantum system and against each other. By extracting the output states of a complete set input density matrices, we were able to reconstruct the super-operator matrix representation of a quantum channel, allowing us to get the Choi matrix through index shuffling, and then perform eigen decomposition to produce the Kraus operators. This then enabled us to calculate the fidelity of a quantum system with a target unitary gate, giving us a metric for comparison between a real system and an ideal quantum operation.

Our results in this report continually showed that in non-Markovian environments where memory is retained, there is a greater preservation of quantum coherence. This can be seen not only by looking at the dynamics of the Bloch vector components, which decayed slower with the presents of memory-backflow, but is also present in the Fidelity which consistently remained higher in non-Markovian systems across all simulated time scales. We directly compared closed, Markovian, and non-Markovian dynamics under the same control Hamiltonian, and plotted fidelity versus time across these cases where we saw consistently that the non-Markovian system experienced slower decline in fidelity. Additionally we also plotted the affect on the Fidelity of simulating with different values for the memory cutoff time t_{cut} at three different time points in the evolution of the system. We observed that not only the optimal value for t_{cut} given our system, but how the memory effects continue to have a greater effect on the system as time progresses.

In all cases studies, the non-Markovian system consistently outperformed the Markovian system. One stipulation to this however is in the case of a more complicated Hamiltonian. While the fidelity was still higher and the dynamics decayed slower for the non-Markovian case, the effects of memory-backflow seemed to be significantly diminished, likely due to the phenomenon known as the Lamb-shift [18][19] where the frequency of a system coupled to a bath is increased, and also due to the environment returning memory to the system in a seemingly biased fashion.

In conclusion, our work highlights the usefulness of process tensors in performing realistic quantum operations. We have also seen throughout that non-Markovian system, through the memory-backflow that they introduce, can mitigate decoherence more effectively and invariably produce higher average gate fidelity values when compared to a target unitary gate. If properly understood and controlled, in the future non-Markovianity could be leveraged as a resource for enabling quantum gates in a noisy environment, and could be used for the development of realistic quantum devices where managing environmental interactions is key to maintaining reliable quantum computing.

References

[1] E. P. Butler. (2024). *Optimisation of Non-Markovian Systems Using Tensor Networks*, PhD Thesis, Trinity College Dublin.

[2] L. H. Pedersen, N. M. Møller, and K. Mølmer. (2007). Fidelity of quantum operations. Physics Letters A, vol. 367, no. 1–2, pp. 47–51. Available: https://doi.org/10.1016/j.physleta.2007.01.069

[3] M. A. Nielsen and I. L. Chuang. (2010). *Quantum Computation and Quantum Information*, 10th Anniversary Edition. Cambridge University Press.

[4] C. J. Wood, J. D. Biamonte, and D. G. Cory. (2015). A Graphical Calculus for Open Quantum Systems. Quantum Information and Computation, vol. 15, no. 9–10, pp. 759–811. Available: https://arxiv.org/abs/1111.6950

[5] H.-P. Breuer and F. Petruccione. (2002). *The Theory of Open Quantum Systems*. Oxford University Press.

[6] U. Weiss. (2012). *Quantum Dissipative Systems*, 4th ed. World Scientific.

[7] A. J. Leggett, S. Chakravarty, A. T. Dorsey, M. P. A. Fisher, A. Garg, and W. Zwerger. (1987). Dynamics of the dissipative two-state system. *Reviews of Modern Physics*, vol. 59, no. 1, pp. 1–85. Available: https://doi.org/10.1103/RevModPhys.59.1

[8] Á. Rivas, S. F. Huelga, and M. B. Plenio. (2014). Quantum non-Markovianity: characterization, quantification and detection. *Reports on Progress in Physics*, vol. 77, no. 9, 094001. Available: https://doi.org/10.1088/0034-4885/77/9/094001

[9] M. M. Wolf and J. I. Cirac. (2008). Dividing Quantum Channels. *Communications in Mathematical Physics*, vol. 279, no. 1, pp. 147–168. Available: https://doi.org/10.1007/s00220-008-0411-y

[10] G. P. Beretta. (2009). Nonlinear quantum evolution equations to model irreversible adiabatic relaxation with maximal entropy production and other nonunitary processes. Reports on Mathematical Physics, vol.64, no.1, pp.139–168. Available: https://doi.org/10.1016/S0034-4877(09)90024-6

[11] H. Spohn. (1980). Kinetic equations from Hamiltonian dynamics: Markovian limits. *Reviews of Modern Physics*, vol. 52, no. 3, pp. 569–615. Available: https://doi.org/10.1103/RevModPhys.52.569

[12] R. Alicki and K. Lendi. (1987). Quantum Dynamical Semigroups and Applications. Lecture Notes in Physics, vol. 286. Springer. Available: https://doi.org/10.1007/3-540-18276-4

[13] C. W. Gardiner and P. Zoller. (2000). *Quantum Noise: A Handbook of Markovian and Non-Markovian Quantum Stochastic Methods with Applications to Quantum Optics*. Springer. Available: https://doi.org/10.1007/978-3-540-47620-7

[14] H.-P. Breuer, E.-M. Laine, and J. Piilo. (2009). Measure for the Degree of Non-Markovian Behavior of Quantum Processes. *Physical Review Letters*, vol. 103, no. 21, 210401. Available: https://doi.org/10.1103/PhysRevLett.103.210401

[15] E.-M. Laine, J. Piilo, and H.-P. Breuer. (2010). Measure for the non-Markovianity of quantum processes. *Physical Review A*, vol. 81, no. 6, 062115. Available: https://doi.org/10.1103/PhysRevA.81.062115

[16] D. Khurana, B. K. Agarwalla, and T. S. Mahesh. (2019). Experimental emulation of quantum non-Markovian dynamics and coherence protection in the presence of information backflow. Phys. Rev. A, vol. 99, no. 022107.

[17] F. A. Pollock, C. Rodríguez-Rosario, T. Frauenheim, M. Paternostro, and K. Modi. (2018). Operational Markov condition for quantum processes. *Physical Review A*, vol. 97, no. 1, 012127. Available: https://doi.org/10.1103/PhysRevA.97.012127

[18] J. Ilves. (2019). *Bath-induced persistent Rabi oscillations in a qubit*, Master's Thesis, Aalto University, School of Science. Available: https://aaltodoc.aalto.fi/handle/123456789/37206

[19] G. Suárez and M. Horodecki. (2025). *Making Non-Markovian master equations accessible with approximate environments*, arXiv preprint arXiv:2506.22346. Available: https://doi.org/10.48550/arXiv.2506.22346

[20] R. Trivedi. (2022). Description and complexity of non-Markovian open quantum dynamics. arXiv preprint arXiv:2204.06936. Available: https://doi.org/10.48550/arXiv.2204.06936

[21] C. Guo. (2022). Memory complexity of quantum processes. arXiv preprint arXiv:2203.01492. Available: https://doi.org/10.48550/arXiv.2203.01492

[22] M. R. Jørgensen and F. A. Pollock. (2022). Exploiting the causal tensor network structure of quantum processes to efficiently simulate non-Markovian path integrals. *Quantum*, vol. 6, 803. Available: https://doi.org/10.22331/q-2022-05-31-803

[23] G. E. Fux, P. Fowler-Wright, J. Beckles, E. P. Butler, P. R. Eastham, D. Gribben, J. Keeling, D. Kilda, P. Kirton, E. D. C. Lawrence, B. W. Lovett, E. O'Neill, A. Strathearn, and R. de Wit. (2024). A Python package to efficiently simulate non-Markovian open quantum systems with process tensors. *Journal of Chemical Physics*, vol. 161, no. 12, 124108. Available: https://doi.org/10.1063/5.0225367

[24] K. Liu and J. Lu. (2025). Error bounds for open quantum systems with harmonic bosonic bath. arXiv preprint arXiv:2408.04009. Available: https://arxiv.org/abs/2408.04009