Finding best pseudo-modes for HEOM from numerical spectral density

Junjie Wang, Xinxian Chen, Ignacio Franco
Department of Chemistry, University of Rochester
(Dated: August 12, 2024)

In this paper, we use supervised machine learning as the method for decomposing the Bath Correlation Function(BCF) between the open quantum system and the attached environment to series of exponential terms. The neural network model in machine learning provides opportunity of designing specific forms of the hidden layer, and focusing on the weight of the branches in the hidden layers give the indication of the parameter we are interested. This parameter fitting method allows approximating the bath correlation function efficient reduction of exponential terms, and so that improving the calculation efficiency.

Keywords: Quantum Dynamics, HEOM, Neural Network

I. INTRODUCTION

In open quantum systems, it is customary to divide the total Hamiltonian $H = H_{\rm S}+H_{\rm B}+H_{\rm SB}$ into a system part $H_{\rm S}$, a bath $H_{\rm B}$ and their interaction $H_{\rm SB}$. For simplicity, for one harmonic bath, $H_{\rm B} = \sum_j \omega_j a_j^\dagger a_j$ where a_j ($a_j^\dagger$) is the lowering (raising) operator of the j-th mode with characteristic frequency ω_j, and $H_{\rm SB} = Q_{\rm S}X_{\rm B}$ where $Q_{\rm S}$ is a system operator and $X_{\rm B} = \sum_j g_j(a_j^\dagger + a_j)$ is a bath operator. The influence of this bath on the system is completely characterized by its spectral density (SD) $J(\omega) = \sum_j g_j^2\delta(\omega-\omega_j)$, a quantity that summarizes the frequencies of the bath and the system-bath coupling strengths. In reality, the degrees of freedom (DOFs) of the bath part may be very large; and for characterizing the solvent part we often need a continuous SD, which makes those explicit quantum dynamics methods hard for these models. Hierarchical equations of motion (HEOM) is one of the most advanced implicit methods to capture the dynamics of the reduced density matrix σ_{ij} of open quantum systems.[1–4] The key in HEOM is to take the advantages of the bath correlation function (BCF) $C(t) = \langle X_B(t)X_B(0)\rangle$, and decompose it into a series of exponential terms such that $C(t) = \sum_{k=1}^K c_k e^{-\gamma_k t}$ and $C^\star(t) = \sum_{k=1}^K \tilde{c}_k e^{-\gamma_k t}$. Each exponential term gives the nature of a *pseudo-mode* (PM) of the bath. The number of PMs K needed depends on the structure and the temperature of the bath. For a K-PM bath, to capture the dynamics of σ_{ij}, the HEOM strategy is to introduce an *extended density tensor* $\varrho_{ijn_1\ldots n_K}$. Here $i,j \in \{1,\ldots,M\}$ with M the number of levels in the system, and $n_k \in \{0,\cdots,N_k-1\}$ for all k. The system density matrix is $\sigma_{ij} = \varrho_{ij0\ldots0}$.

The extended density tensor satisfies the HEOM[3, 5, 6]

$$\frac{\partial}{\partial t}\varrho(t) = -\mathbf{i}\left[H_{\rm S}^{>} - H_{\rm S}^{<} + \sum_{k=1}^{K}\Big(\Big(c_k Q_{\rm S}^{>} - \tilde{C}_k Q_{\rm S}^{<}\Big)\hat{\alpha}_k^\dagger + \big(Q_{\rm S}^{>} - Q_{\rm S}^{<}\big)\hat{\alpha}_k - \mathrm{i}\gamma_k\hat{\alpha}_k^\dagger\hat{\alpha}_k\Big)\right]\varrho(t) \equiv \mathcal{L}\delta\rho(t), \tag{1}$$

Here, the operator $O^>$ acts on the system index i and $O^<$ on index j. The creation operators $\hat{\alpha}_k^\dagger$ (and annihilation $\hat{\alpha}_k$) act on the k-th PM indexes n_k of $\varrho(t)$.

Obiviously, one expects an efficient way to get an accurate approximation of the BCF with fewer PMs. Traditionally, the PMs are calculated from the process of analytical integration of the equation

$$C(t) = \int_0^{+\infty} J(\omega)(\coth(\beta\hbar\omega/2)\cos(\omega t) - \mathbf{i}\sin(\omega t))\,\mathrm{d}\omega\,, \tag{2}$$

using the residues theorem and each pole of the integer and gives a PM. However, for a more general case, one expects to find a way to get the PM from any numerical spectral density without expansion to a complex plain. [7] indicates to use of a numerical way to find those PMs, a similar idea is also suggested for a special case in [8].

Here, taking the advantage of the symmetry properties of a BCF, we assume that its approximation has the form

$$\tilde{C}(t) = \sum_i c_i e^{-\eta_i t} + \sum_j d_j e^{-\gamma_j t} + \bar{d}_j e^{-\bar{\gamma}_j t} \tag{3}$$

and leave complex parameters $\{c_i, d_j, \bar{d}_j\}$ and real parameters $\{\gamma_i, \eta_j, \theta_j\}$ as undetermined coefficients. The we try to find the best approximation $\tilde{C}(t)$ for $C(t)$ by minimize the "distance" of the two functions. However, this process in obviously not unique, and hence need further investigations.

II. METHODS

The correlation function calculated from spectrum density has symmetry properties, which allows us to assume that it has the form in Equation3. Having the approximation form, we can use a neural network model as the technique to do the fitting.

A. Developing NN model

Firstly, we should determine the number of exponential pairs we are going to use. Given a reduction number j of terms we need, the linear combination of those terms can be represented as j number of nodes in the model, which indicates the dimension of the model is j.

Thus, the input of the model becomes j copies of the time series, and the output of the model becomes the one dimension of the number series from a linear combination of the nodes passing the defined activation function, which is the exponential function in this condition.

Given the linear model is defined by:

$$\vec{y} = \vec{w} \cdot \vec{x} + \vec{b}.$$

The parameters of the model can be written in the form:

$$e^{\vec{w}\cdot\vec{x}+\vec{b}} = \vec{c} \cdot e^{\vec{w}\cdot\vec{x}}.$$

The form of Equation3 shows the BCF is decaying overtime. At the t = 0, the exponential term is 1, and the BCF is decaying from:

$$\tilde{C}(0) = \sum_i c_i + \sum_j d_j + \bar{d}_j = \sum_i^N c_i, \tag{4}$$

the RHS generalize the term, which does not distinct whether those coefficient is for term $e^{-\eta_i t}$ $e^{-\gamma_j t}$ or $e^{-\bar{\gamma}_j t}$ for simplicity.

Equation4 can be seen as a restriction to the coefficients during the fitting process. During the autogradient descent in an N term model, withholding the value of $\tilde{C}(0)$, the model can update the coefficients of the first N-1 term and update the last term by setting $c_N = \tilde{C}(0) - \sum_i^{N-1} c_i$ in order to set $\tilde{C}(0)$ fixed during all the update iterations.

Furthermore, with the special target and the artificially defined activation function, the randomly chosen parameter will likely make the loss diverge during the optimization process. Thus, the initialization plays an important role in this process.

B. Initialization

Empirically, the optimization process in Neutral Network largely depends on the initial condition and learning rate in the process. Many scholars are suffering from trying different initial parameters. Due to the specific form of the activation function in the hidden layer in the model, the importance of the initialization process becomes much more important.

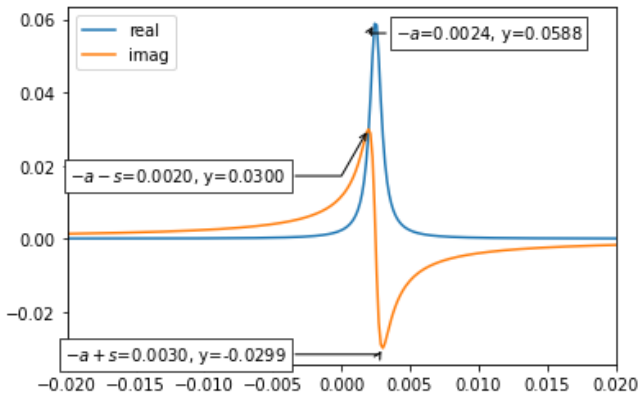

Figure 1: Plotting the real and imaginary part of a sample $\frac{d_j}{s+ia+ik}$, where $a = 0.0025, s = 0.0005$. Based on the plot and analytical function, we can infer that a is the peak of the real part, and $2s$ is the distance of two peaks in the imaginary part.

Since we know that the Fourier Transformation of the function e^{ax} gives a delta function $\delta(x-a)$ peaked at x = a. The former case works for a different domain, where $x \in \mathbb{R}$, but the time domain cannot take value from the negative region, so in our case, $t \in \mathbb{R}_+$.

$$\int_0^\infty e^{ik\cdot t} e^{(s+ia)t} dt = \frac{1}{s+ia+ik} \tag{5}$$

With $s, a, k \in \mathbb{R}$. With that being said, the Fourier Transformation of our approximation form can be written as:

$$\tilde{C}(t) \xrightarrow{\mathcal{F}} \tilde{C}(k) := \sum_j \frac{d_j}{s_j + ia_j + ik} + \frac{\bar{d}_j}{s_j - ia_j + ik}. \tag{6}$$

The transformed function has different real and imaginary components, and its shapes can help us to extract the necessary information to initialize.

Due to the special form of the unit pair of the summation function, as shown in the figure 1, we can precisely locate the dominant term of a value, which indicates the peak's location in the real part of the transformed function. After getting a, we can get s by locating the distance between the pair of peaks in the imaginary part of the function.

d_j and $\bar{d}_j$ can sequentially be estimated by calculating the peak height. Plugging in $k = -a$ into $\frac{d_j}{s+ia+ik}$, we will get the value $y = \frac{d_j}{s}$ in the real space; similarly, plugging in $k = -a \pm s$, we will get the get the value $y = \pm\frac{d_j}{2s}$ in the imaginary space.(For example $k = -a+s$, $y = \frac{d_j}{s+is} \times \frac{s-is}{s-is} = \frac{d_j}{2s}(1-i)$).

Since we are looking for the function with the linear combinations of multiple exponential terms (PM), which will result in a linear combination of transformed terms, the height of those peaks may be higher than the actual value, and the locating of the peaks will be harder to manually distinguished or slightly shifted. Then, the preceding methods can only be a rough estimation, which should be corrected by the machine learning model.

However, for the initialization part, we can assume each peak as a delta function $\delta(k)$, which means the width of each peak is small enough to affect other peaks.

To do that, we have to start by focusing on the real and imaginary parts of the BCF separately, and so does each of their transformation. For one exponential term unit, we can separate their real and imaginary part as follows:

$$\begin{aligned}(d_r + id_i)e^{(s+ia)t} &= (d_r + id_i)e^{st}(cos(at) + isin(at)) \\ &= e^{st}(d_r cos(at) - d_i sin(at)) \\ &\quad + ie^{st}(d_i cos(at) + d_r sin(at)) \\ &= (\frac{d_r}{2}(e^{(s-ia)t} + e^{(s+ia)t}) - i\frac{d_i}{2}(e^{(s-ia)t} - e^{(s+ia)t})) \\ &+ i(\frac{d_i}{2}(e^{(s-ia)t} + e^{(s+ia)t}) + i\frac{d_r}{2}(e^{(s-ia)t} - e^{(s+ia)t}))\end{aligned} \tag{7}$$

The above form works well for an exponential conjugate pair since the third line is a linear combination of complex conjugate terms. Then, adding its complex conjugate term makes the third line of Equation 7 looks like:

$$\begin{aligned}&(\frac{d_r + \bar{d_r}}{2}(e^{(s-ia)t} + e^{(s+ia)t}) \\ &\qquad + i\frac{\bar{d_i} - d_i}{2}(e^{(s-ia)t} - e^{(s+ia)t})) \\ &+i(\frac{d_i + \bar{d_i}}{2}(e^{(s-ia)t} + e^{(s+ia)t}) \\ &\qquad + i\frac{d_r - \bar{d_r}}{2}(e^{(s-ia)t} - e^{(s+ia)t}))\end{aligned}$$

For the function above, the first line is the real part and the second line is the imaginary part. Thus, the expected formula after transformation will be:

$$\begin{aligned}&\tilde{C_r}(t) \xrightarrow{\mathcal{F}} \tilde{C_r}(k) := \\ &\sum_j \frac{d_{rj} + \bar{d_{rj}}}{2}(\frac{1}{s_j - ia_j + ik} + \frac{1}{s_j + ia_j + ik}) \\ &\quad +i\frac{\bar{d_{ij}} - d_{ij}}{2}(\frac{1}{s_j - ia_j + ik} - \frac{1}{s_j + ia_j + ik})\end{aligned} \tag{8}$$

$$\begin{aligned}&\tilde{C_i}(t) \xrightarrow{\mathcal{F}} \tilde{C_i}(k) := \\ &\sum_j \frac{\bar{d_{ij}} + d_{ij}}{2}(\frac{1}{s_j - ia_j + ik} + \frac{1}{s_j + ia_j + ik}) \\ &\quad +i\frac{d_{rj} - \bar{d_{rj}}}{2}(\frac{1}{s_j - ia_j + ik} - \frac{1}{s_j + ia_j + ik})\end{aligned} \tag{9}$$

After the Fourier Transformation, each real and imaginary part of $\tilde{C}(k)$ will have its own real and imaginary part of the transformed function. That provides a convenient way for initialization of d and $\bar{d}$ to some degree, since for each j's term in the summation series, they share s value and a value. Thus, we can choose the part with the most oblivious features for us to obtain information from peaks. For d_{rj} and $\bar{d_{rj}}$, we calculate the height of the peaks from the real part of the transformation of $\tilde{C_r}(k)$ and the real part of the transformation of $\tilde{C_i}(k)$; For $d_{ij} and \bar{d_{ij}}$, we calculate the height of the peaks from the imaginary part of the transformation of $\tilde{C_r}(k)$ and the imaginary part of the transformation of $\tilde{C_i}(k)$.

Since we want to capture the most dominant terms, we selected the terms with highest peak value and leave the correction work for the machine learning model. Note that Equation3 contains both real exponential terms $c_i e^{-\eta_i t}$ and complex pairs, if the real part of the transformed function contains the peak centered at $a = 0$, we can count it first and rank the rest of the peaks.

C. Optimization

PyTorch provides a convenient way to do the gradient descent for us.

In our model, we use Adam as our optimizer. Due to the problem we are facing in atomic scale, we convey all the units into the atomic units, when we solve our problems. Thus, the learning rate is usually settled somewhere between 10^{-8}.

During the optimization process, we want to minimize loss using PyTorch Auto-Gradient.

Lagrangian Optimization

(Not sure if that is useful) An alternative optimization way is to apply the concept of Lagrangian Multiplier to optimize the process. We want to manipulate regularization terms by trying Lagrangian-based constrained optimization:

$$\frac{\partial}{\partial t}\begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_k \end{pmatrix} - \mathbb{M}\begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_k \end{pmatrix} = \begin{pmatrix} G_1 \\ G_2 \\ \vdots \\ G_k \end{pmatrix} \to 0 \tag{10}$$

$$G_0 + \lambda_1 G_1 + \cdots + \lambda_k G_k = \mathcal{L} \to 0 \tag{11}$$

$$\tag{12}$$

We are starting from a relatively simple matrix with diagonal values and off-diagonal γ value on one side of the diagonal, which the matrix $\mathbb{M}$ becomes:

$$\mathbb{M} = \begin{pmatrix} \gamma_{11} & \gamma_{12} & \gamma_{13} & \cdots & \gamma_{1k} \\ 0 & \gamma_{22} & \gamma_{23} & \cdots & \gamma_{2k} \\ \vdots & & \ddots & & \\ 0 & 0 & 0 & \ldots & \gamma_{kk} \end{pmatrix} \tag{13}$$

Then, the corresponding function sets become:

$$C(t) - \sum_{i=1}^{k} c_i f_i(t) = G_0 \tag{14}$$

$$\partial_t f_1 - (\gamma_{11} f_1 + \gamma_{12} f_2 + \gamma_{13} f_3 + \cdots + \gamma_{1k} f_k) = G_1 \tag{15}$$

$$\partial_t f_2 - (0 f_1 + \gamma_{22} f_2 + \gamma_{23} f_3 + \cdots + \gamma_{2k} f_k) = G_2 \tag{16}$$

$$\vdots \tag{17}$$

$$\partial_t f_k - (0 f_1 + 0 f_2 + 0 f_3 + \cdots + \gamma_{kk}) f_k = G_k \tag{18}$$

$$\tag{19}$$

In order to use this optimization process, we have to change our basis functions from only complex exponential terms to a more restricted function sets:

$$\sum_j d_j e^{-\gamma_j t} \to \sum_i f_i \tag{20}$$

$$\partial_t f_i = \sum_{k=1}^{i-1} c_k f_k \tag{21}$$

During the optimization process, we want to minimize $\mathcal{L}$ using PyTorch Auto-Gradient, with Lagrangian multiplier λ_i.

Sanity Check

After doing the fitting, we calculate the decoherence based on the approximated function using Padé Method[9] and compared it with the Pure Dephasing decoherence mode.

$$r(t) = exp[-\sum_i \frac{|g_i|^2}{\omega_i^2}(1 - cos\omega_i t) coth(\frac{\omega_i}{2k_B T})] \equiv e^{\Gamma(t)} \tag{22}$$

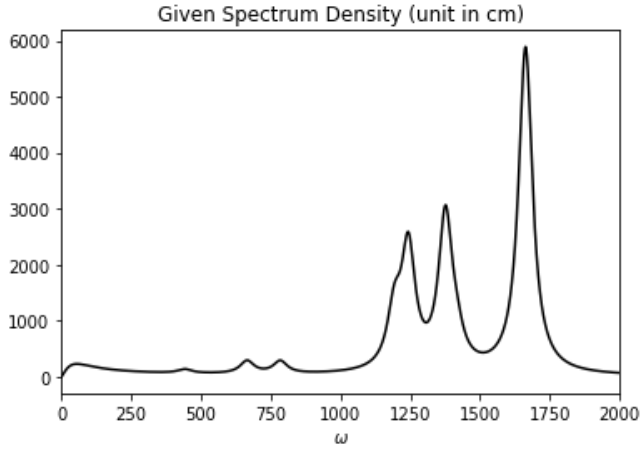

Figure 2: The figure shows an example of spectrum density by setting the scale of both axis in centimeters.

$$\Gamma(t) = -\int_0^\infty \mathrm{d}\omega \frac{J(\omega)}{\omega^2}(1 - cos\omega t) coth(\frac{\omega_i}{2k_B T}) \tag{23}$$

Since we know that both the real and imaginary part of the BCF has the form of the Fourier sine integral, which indicates the imaginary part can be treated as the positive portion of the odd function after transforming from the Spectrum Density $J(\omega)$ as the positive portion of another odd function:

$$\hat{f}_{odd}(\gamma) = 2\int_0^\infty f(x) sin(2\pi\gamma x)\mathrm{d}x \tag{24}$$

$$f(x) = 2\int_0^\infty \hat{f}_{odd}(\gamma) sin(2\pi\gamma x)\mathrm{d}\gamma, \tag{25}$$

and the real part is the positive portion of the even transformation using the cosine kernel:

$$\hat{f}_{even}(\gamma) = 2\int_0^\infty f(x) cos(2\pi\gamma x)\mathrm{d}x \tag{26}$$

$$f(x) = 2\int_0^\infty \hat{f}_{even}(\gamma) cos(2\pi\gamma x)\mathrm{d}\gamma, \tag{27}$$

Specifically, the transformation from the real and the imaginary of the BCF to the spectrum density is:

$$J(\omega) = \int_0^\infty \tilde{C}_r(t) \tanh(\beta\hbar\omega/2) cos(2\pi t\omega)\mathrm{d}t \tag{28}$$

$$J(\omega) = -\int_0^\infty \tilde{C}_i(t) sin(2\pi t\omega)\mathrm{d}t, \tag{29}$$

with $\tilde{C} = \tilde{C}_r(t) + i\tilde{C}_i(t)$. Thus, we can either choose to recover the $J(\omega)$ using 29 and 28 and plug into the Equation22 or use the Padé Method to get the decoherence.

Additionally, calculating the FFT using may result in a scaling problem, as shown in Figure3. To solve the issue,

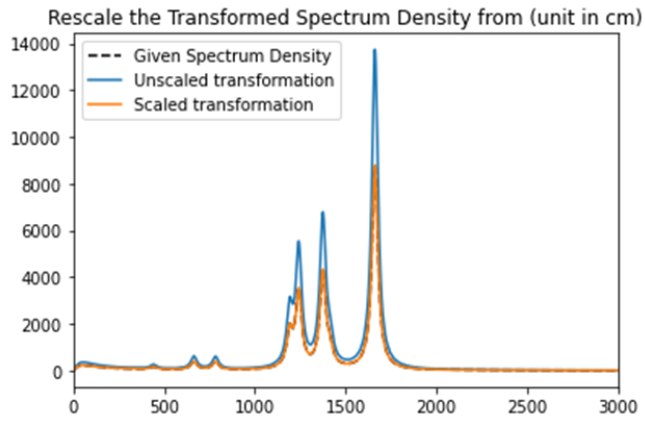

Figure 3: Rescaling the Transformation. The dash-line is the given spectrum density shown in Figure2. The blue line is getting from applying the given spectrum density twice, and people should expect the same spectrum density, but it results in stretching the amplitude of the peaks. The orange line shows the blue line after multiplying the rescaling factor.

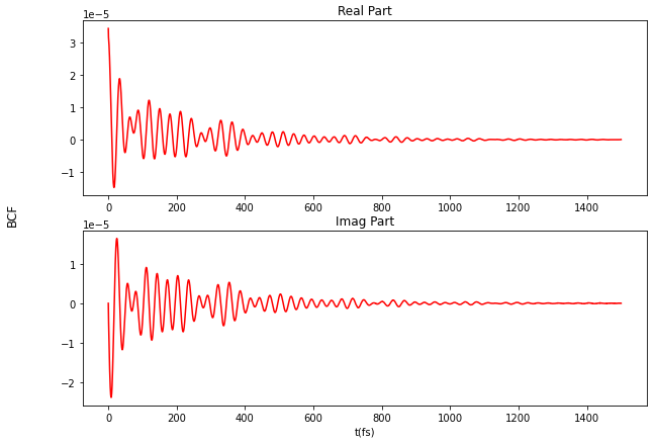

Figure 4: The BCF, with the real (upper) and imaginary (lower) part, is calculated based on the given spectrum density $J(\omega)$ in Figure2 using Equation2.

we should rescale the spectrum density after applying the FFT by multiplying the rescaling factor by the calculated spectrum density, which is determined by the ratio of the highest peak of the calculated spectrum density to that of the given spectrum density.

$$r = \frac{max(J_{given})}{max(J_{calculated})}$$

III. RESULT OF AN EXAMPLED BCF

Starting from a given spectrum density $J(\omega)$, the BCF can be calculated by using Equation2 in a specific temperature (T = 300K). As Figure4 shows, the real and

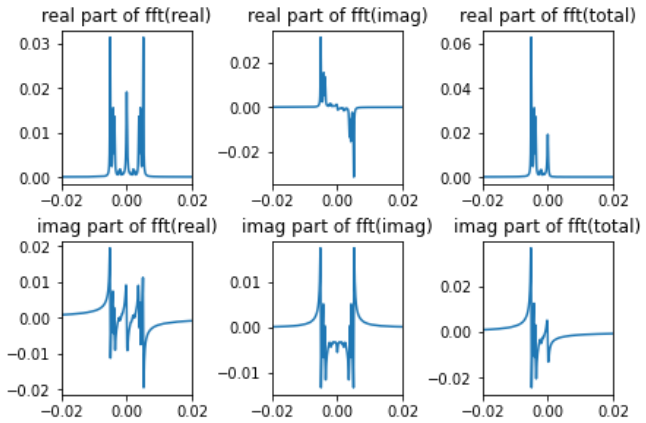

Figure 5: After applying fast Fourier transformed the BCF from time-space to k-space. The first two columns show the Fourier transformation of the real part and the imaginary part of the BCF function. The Third column shows the direct Fourier Transformation of the BCF in complex form, which is the addition of the two columns on its left. The first row shows the real part after the transformation, and the second row shows the imaginary.

the imaginary part are calculated separately and result in two data sets.

Figure2shows the $J(\omega)$ generated from 17 terms, and the peak at roughly$\omega = 50(cm^{-1})$ indicates there is a Drude term, and we should expect a single peak at k = 0 after applying the FFT to the BCF, and the peak in the real part of the transformation in Figure5 aligns our anticipation.

As described in the initialization process, we pick the three most peaked frequencies as the representative terms of Brownian modes to capture the most vibration features in the BCF and the peak at the k = 0 to represent the drude mode, which captures the decaying feature in the BCF.

Fitting with time-space in equal distance point distribution

In this scenario, the data sets of BCF shown in Figure4 is spanned in linear space, which means the δt is the same for all the t point, with a total of 2000 points.

After enough iterations of the auto-gradient descent, we get the fitting result as shown in Figure6. The imaginary part of the BCF fits perfectly. However, the fitting result of the real part of the BCF vibrates in a greater amplitude at the beginning. The reason might be my restriction in Equation4.

Since the fitted imaginary part overlaps perfectly with the imaginary part of the given BCF, I choose the imaginary part to calculate the $J(\omega)$ as shown in Figure7, and the calculated $J(\omega)$ shows a similar shape with the given

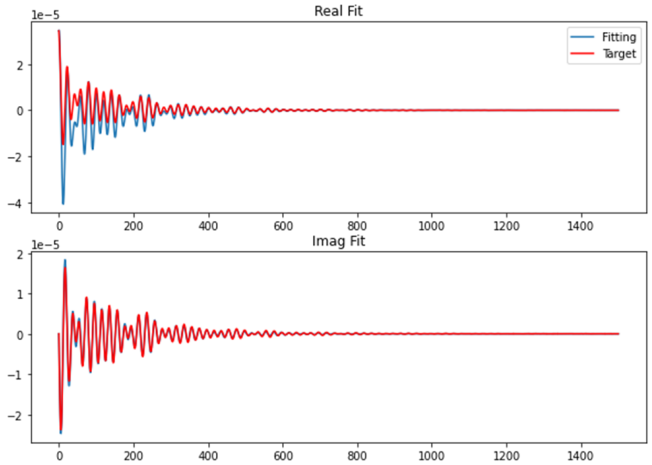

Figure 6: Linear Space Fitting Result: The red line is the fitting target BCF, which is shown in Figure4, and the blue line is the fitting result. In this picture, the real part captures the frequency but fails to capture the amplitude feature, but the imaginary part is perfectly fitted.

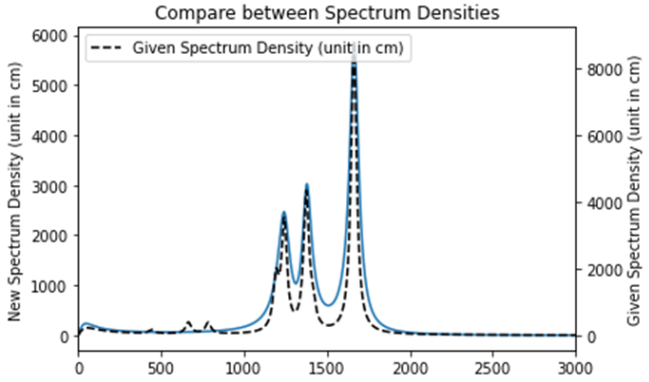

Figure 7: Linear Space Spectrum Density Result: Since the perfect fitting in the imaginary part is shown in Figure6, the calculated $J(\omega)$ is given by the blue line multiplied by the rescaling factor. The blue line seems the simplification form of the $J(\omega)$ with wider peaks.

$J(\omega)$ but with lower peak heights and wider peak widths, which is reasonable since it can be perceived as an approximation of the convolution product by the Gaussian kernel and the given $J(\omega)$. Then, the decoherence over time overlaps perfectly due to the perfect matching concerning the $J(\omega)$.

However, due to the different performance of the real and imaginary part in the fitted BCF, the $J(\omega)$ calculated by Equation28 differs a lot from the result calculated by Equation29. To solve this question, I tried to more focus on the time period in the beginning.

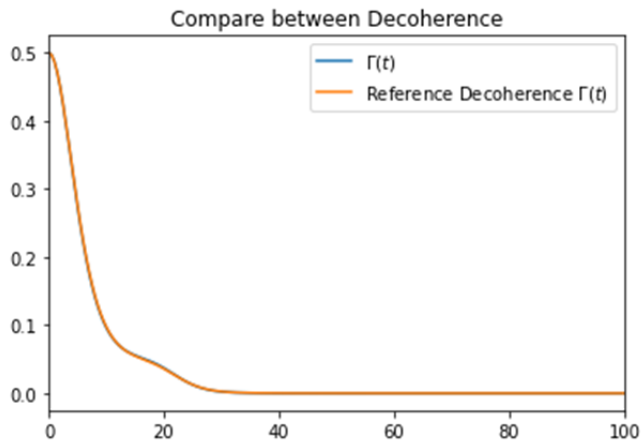

Figure 8: Linear Space Decoherence Fitting Result: The orange line shows the decoherence propagating calculated from the given $J(\omega)$, and the blue line is calculated from the calculated $J(\omega)$ by the Equation22. Due to the similarity between the given and the calculated spectrum density, the coherence between the system and the bath aligns perfectly.

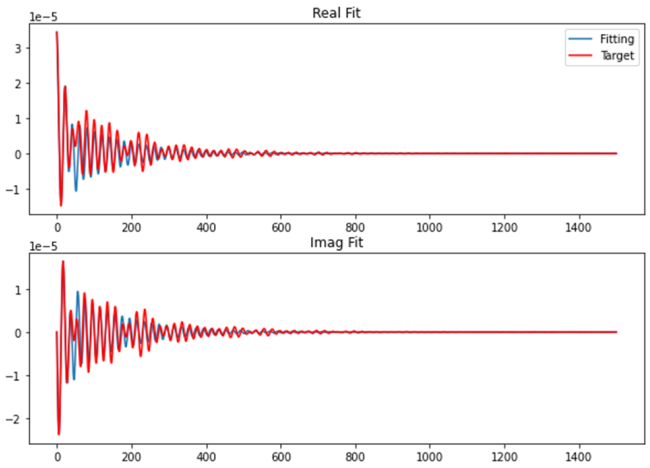

Figure 9: Log Space Fitting Result: The red line is still the fitting target BCF, which is shown in Figure4, and the blue line is the fitting result. In this picture, both the real part and the imaginary part capture the major shape of the BCF and its vibration frequency, but there is a small difference in matching amplitude of each vibration.

Fitting with time-space in log space point distribution

In this scenario, the data sets of BCF showed in Figure4 is spanned in log space, which means the δt is smaller in the beginning time and larger in the later time, with a total of 3000 points. The reason to make the data points at the beginning time period denser is to make the loss value greater and the model more sensitive to the smaller

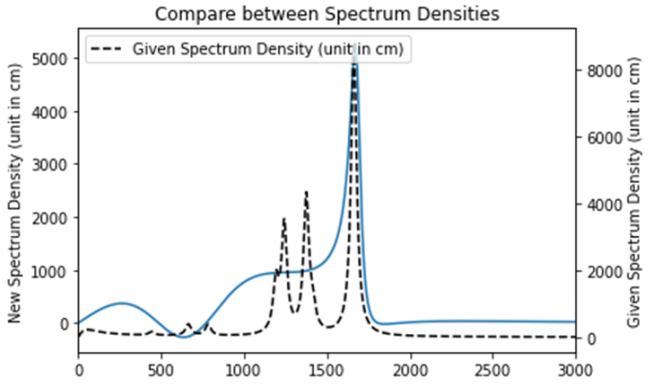

Figure 10: Log Space Spectrum Density Result: Similar structure as in Figure7, the calculated $J(\omega)$ with multiplied by the rescaling factor is given by the blue line. This time the blue line does not look like an over simplified shape of given $J(\omega)$, instead it differs a lot from the given $J(\omega)$.

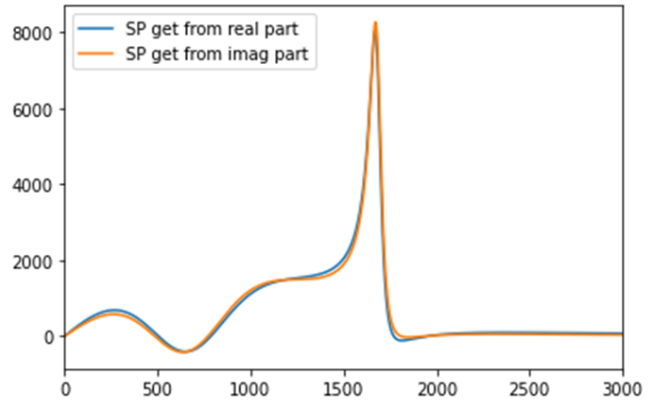

Figure 11: Spectrum Density Result for both parts: Comparing the calculated $J(\omega)$ from both real and imaginary parts of the fitted BCF using Equation28 and Equation28 respectively. The result shows these two $J(\omega)$ do not differ a lot, which shows the problem in the previous scenario is solved.

time interval.

The calculated $J(\omega)$ can also help diagnose the fitting results. In this data point distribution, the real and imaginary part in the fitted BCF has a similar performance, but there is still some mismatches in the amplitude concerning the given $J(\omega)$. Comparing the results in Figure7, the highest peak in Figure10 seems still to have a similar shape besides a lower height, and the Drude mode becomes more dominant with a higher peak than in Figure7. Those two relatively lower peaks in Figure7 become too wide that they merge with the highest peak and become a "step". Thus, I anticipate that the mismatches in the amplitude result from the too-quick decay caused by the width of the peak in Figure10.

Even though the mismatch in the $J(\omega)$, the decoherence over time overlaps with the decoherence calculated from the given $J(\omega)$, which is a kind of success.

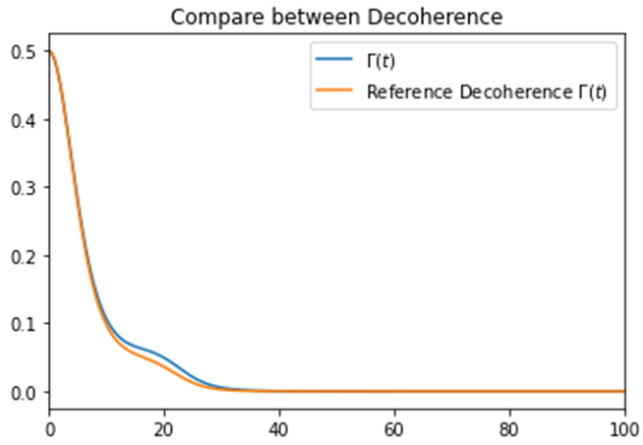

Figure 12: Log Space Decoherence Fitting Result: Even though the calculated $J(\omega)$ differs a lot from the given $J(\omega)$ as shown in Figure10, the decoherence propagation (Blue line) does not differs a lot from the one get from the given $J(\omega)$ (Orange line).

IV. DISCUSSION

The Method of using machine learning to reduce the representative terms has features to capture the main shape of the BCF; however, they cannot cover every detail of the original fitting target given our aim to be efficient, since the fitting results is a linear combination of a smaller number of terms which contributes to the superposition from the reduced number of waves.

The Lagrangian optimization method is useful but does not make significant improvements during the optimization process in this particular scenario, which means the fitting results do not become better but the model requires more terms. The Lagrangian optimization works better in the condition where the off-diagonal element $\gamma_{i\neq j}$ in the matrix $\mathbb{M}$.

In the BCF fitting, the major vibration frequency and the initial state $\tilde{C}(0)$ can be captured during the initialization process and restriction by Equation4. The pros of those initial works will help the machine learning models give repetitive results instead of giving good results in some uncontrollable probabilities. However, that restriction will result in getting some mismatches in vibration amplitude when $t \neq 0$.

Comparing the decoherence solution with the analytical result, we can successfully replicate the results with a reasonably small difference, but the spectrum density $J(\omega)$ calculated by the BCF fitting results can diverge a lot.

The future effort may be trying with denser data point distribution and different learning rates for the best op-

timization in auto-gradient descent.

[1] Y. Tanimura and R. Kubo, J. Phys. Soc. Jpn. **58**, 101 (1989).
[2] Y. Tanimura, J. Phys. Soc. Jpn. **75**, 82001 (2006).
[3] Y. Tanimura, J. Chem. Phys. **153**, 20901 (2020).
[4] T. Ikeda and G. D. Scholes, J. Chem. Phys. **152**, 204101 (2020).
[5] Y. Yan, M. Xu, T. Li, and Q. Shi, J. Chem. Phys. **154**, 194104 (2021).
[6] R. Borrelli and S. Dolgov, J. Phys. Chem. B **125**, 5397 (2021).
[7] N. Lambert, S. Ahmed, M. Cirio, and F. Nori, Nature Communications **10**, 3721 (2019).
[8] C. Duan, Z. Tang, J. Cao, and J. Wu, Phys. Rev. B **95**, 214308 (2017).
[9] J. Hu, R.-X. Xu, and Y. Yan, The Journal of Chemical Physics **133**, 101106 (2010), https://doi.org/10.1063/1.3484491.