May 2024

Comparison of two software toolboxes for the simulation of quantum systems in the context of coherent multidimensional spectroscopy

<u>Nils</u> Sebastian Martin Wilhelm Schneider

Division of Mathematical Physics
Department of Physics
Lund University

Thesis submitted for the degree of Bachelor of Science.
Supervised by Andreas Wacker

Abstract

Simulations are an indispensable tool in physics because they complement experiments, especially complex ones. Coherent multidimensional spectroscopy (CMDS) is such a complex experimental setup. In CMDS, the molecular system of interest interacts with two to four coherent laser pulses and the generated emission provides information about the energetic structure and the relaxation dynamics of the probed molecule.

The common technique to theoretically describe CMDS are perturbative expansions, in particular under the additional approximation that the pulses have zero width (duration). However, this approach doesn't capture some phenomena observed in real-world spectroscopic experiments, for example it doesn't account for pulse overlap artefacts. A more realistic modelling is achieved with explicitly propagating the density matrix of the system.

The issue with this, in turn, is that there exist many different simulation codes and toolboxes, which could impede the comparison and reproduction of simulation results. Kenneweg et al. (2024) therefore put forward their own *quantum dynamics toolbox (QDT)*, a *MATLAB*-based toolbox that provides tools for simulating light-matter interactions, especially non-linear spectroscopic experiments with explicit density matrix propagation. The rationale is that its modularity will allow for widespread adoption. To test this new toolbox, this thesis compares it to *QuTiP*-based simulation code, by attempting to reproduce a result of a simulation script (Hedse et al. 2023) that simulates pulse overlap artefacts in double quantum coherence spectroscopy (a type of CMDS).

It was only possible to reproduce the results from Hedse et al. qualitatively, but not quantitatively. Several factors that could have potentially been responsible for the variations were investigated, but they were ruled out as the cause. This sheds a light on the difficulties one encounters when trying to reproduce simulation results on a different code bases, especially when the code is to complex to allow for line-wise comparison.

Acknowledgements

First and foremost, I want to thank my supervisor Andreas Wacker. He took a lot of time to support me in my learning and research process, and he always had an open door and patience for questions. Our long discussions both made it possible for me to get familiar with all the underlying theoretical concepts, and gave me countless leads to try to track down differences between the codes as well as to understand why my plots look the way they do. Last but not least, he help me to understand the code from his group's previous publication (Hedse et al. 2023), and modified it to facilitate the comparison with the *QDT MATLAB* toolbox. Thank you!

I want to thank Armin and Akshat for being my examiners. Thank you Claudio for checking in on my progress.

I want to thank Cristina for many fruitful discussions that helped me organize my thought-s, especially when I struggled with nasty integrals and such. Thank you just for being with me and having a lot of patience with me when I'm frustrated.
I could not do it without you.

Last but not least, I am grateful to my parents for their feedback on the coherence of my text, and their overall support and patience.

Contents

List of Figures

List of Tables

Conventions, Terms, Abbreviations & Symbols

Conventions

citation style	(author(s) year, p. 42)
equation reference	(1)
figure reference	*Fig. 1a*
table reference	*Tab. 1*
section reference	*Sec. 1.1*
classes	`System`
variable names	`var1`

Technical Terms

envelope	"In physics and engineering, the envelope of an oscillating signal is a smooth curve outlining its extremes." (*Envelope (Waves) - Wikipedia* 2024)
DQC signal	The Double quantum coherence (DQC) signal

Abbreviations

2DES	2D electronic spectroscopy $\approx$ CMDS
a.u.	atomic units
arb. u.	arbitrary units
CMDS	coherent multidimensional spectroscopy
DQC	Double quantum coherence
eq(s).	equation(s)
FT	Fourier transform
DFT	discrete Fourier transform
FWHM	full width at half maximum (of a peak/ pulse)
LHS	left-hand side (of an equation)
LvNE	Liouville-von Neumann equation
ODE	ordinary differential equation
RHS	right-hand side (of an equation)
SE	Schrödinger equation
s.t.	such that
tot	total
vs	versus
w.r.t.	with respect to

Important general symbols

$=$	equal to
$\equiv$	equal by definition
$[a,b]$	$= ab - ba$, i.e. commutator of a and b
$\vec{a}$	vector a
A	matrix A
A_{ij}	element in the ith row and the jth column of the matrix A
a^*	complex conjugate of a
$\vec{a}^{\mathrm{T}}$	transpose of $\vec{a}$
A^{T}	transpose of A
$A^\dagger$	Hermitian conjugate of A, equal to $(A^*)^{\mathrm{T}}$
$\hat{A}$	operator A
a_i	ith eigenvalue of some operator $\hat{A}$
$\hat{H}$	Hamilton operator
ε_i	ith eigenvalue of $\hat{H}$
$\dot{f}$	$=\frac{\mathrm{d}}{\mathrm{d}t} f$
h	Planck's constant, equal to $6.62607015 \cdot 10^{-34}$ Js (BIPM 2018, p. 203)
$\hbar$	reduced Planck's constant, (set) equal to $h/2\pi$
δ_{ij}	Dirac delta function, equal to 1 if $i = j$, 0 otherwise
$\|\psi(t)\rangle$	state ψ
$\psi(\vec{r},t)$	wavefunction associate with state ψ
$\|\phi_i\rangle$	ith eigenstate of some operator
$\phi_i(\vec{r})$	wavefunction associate with state ϕ_i
$c_i(t)$	(time-dependent) coefficient of $\|\phi_i\rangle$
$\hat{\rho}(t)$	density operator
$\mathrm{Tr}(A)$	trace of matrix A
W	a probability

Symbols for CMDS

N_k	number of modulation phases to be probed for kth pulse
Φ_k	modulation frequency of the kth pulse
φ_k	modulation phase of the kth pulse
$\Delta\varphi$	modulation phase difference between first and second pulse
n_t	number of the pulse train
N_t	total number of pulse trains simulated per value of τ
Δt_t	time difference between to pulse trains
t_0	start time of the individual pulse train
t_k	time of the kth pulse
τ	$= t_2 - t_1$
ω	angular frequency of the laser

1 Introduction

Simulations are an indispensable tool in physics, not to replace experiments and observations necessarily - even though some phenomena might only be accessible by simulations - but to complement experiments. This is done in two main ways: Firstly, simulations can produce testable predictions from abstract theories, which makes it possible to experimentally scrutinize these theories and thus makes them falsifiable. Secondly, comparing experimental and simulation results can also help to interpret the experimental results for complex setups, since in simulations one can control all the parameters freely and thus potentially narrow down the effect(s) that caused certain pattern observed.

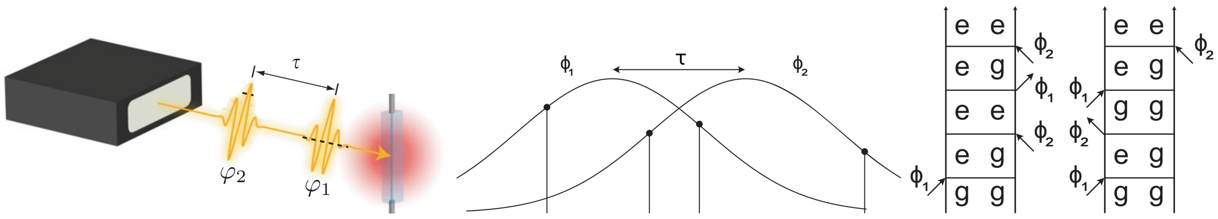
Coherent multidimensional spectroscopy (CMDS) is such a complex experimental setup.[1] It is "an ultrafast optical technique that can study relaxation dynamics with femtosecond time resolution" (Fresch et al. 2023). In CMDS, the molecular system of interest interacts with two to four coherent laser pulses and the generated emission is plotted w.r.t. the excitation and detection frequencies as well as the major time delay, giving information about the life-times of superpositions and exited states in the probed molecule. (Minhaeng Cho 2019, p. 8) While perturbative expansions leading to nth-order non-linear response functions are a common technique to theoretically describe CMDS, a more realistic modelling is achieved with explicit, "brute-force" propagation of the density matrix of the molecular system at hand (Kenneweg et al. 2024).

For this purpose, there exists a variety of toolboxes, as is pointed out by Kenneweg et al. (2024). A major one is *QuTiP* (Johansson et al. 2012, Johansson et al. 2013), a *python*-based toolbox. However, the Hamiltonian has to be constructed by the user.

Kenneweg et al. put forward their own toolbox, the *quantum dynamics toolbox (QDT)*, a *MATLAB*-based toolbox that provides tools for simulating light-matter interactions, especially non-linear spectroscopic experiments with explicit density matrix propagation. As the authors state, their goal is to provide a user-friendly, easy to use modular toolbox, that handles the construction of the Hamiltonian in the background and that is capable of replacing individual custom implementations of different research groups, and which would thereby reduce the likelihood of errors occurring and increase the reproducibility of results, since different research groups could use a unified basis for their simulation codes.

[1] In this thesis, we are mainly concerned with Two-dimensional electronic spectroscopy (2DES), that is, two-dimensional CMDS in the optical and UV range that transfers electrons between molecular energy levels, because it is one of the most widely used techniques. (Minhaeng Cho 2019, p. 3). The mathematical formalism is largely the same for 2D-spectroscopy in the infrared-regime.

To put this idea into practice, however, is first necessary to test the *QDT* code for the kind of simulations frequently performed by a research group to see if it is capable of achieving the same results as the previous code base of the group, and to compare the performances. This was attempted in this thesis.



(a) Mueller et al. 2019 **(b)** Hedse et al. 2023 **(c)** Hedse et al. 2023

Figure 1: (a) Schematic experimental setup. (b) Electric field vs time plot when for overlapping pulses. (c) Double-sided Feynman diagrams for artefactual signal due to the pulse overlap. Image sources see sub-figures.

In particular, a script using the *QDT* toolbox will be compared to a script from Hedse et al. 2023, which uses *QuTiP*. The code, including the construction of the Hamiltonian was written by Alex Arash Sand Kalaee (see Hedse et al. 2023). The simulation chosen for testing the two codes is the simulation of pulse overlap artefacts in double quantum coherence spectroscopy, a type of CMDS (see Hedse et al. 2023). *Fig. 1* tries to schematically illustrate the setup. The motivation for choosing this test system/ simulation task is that pulse overlap artefacts cannot be modelled using common perturbative approaches where the pulses are approximated to have zero pulse width, as discussed in Kenneweg et al. 2024. Thus, pulse overlap artefacts are a suitable testing ground for the two code bases, since the simulation of pulse overlap artefacts requires explicit simulation of the density matrix propagation. The first metric/benchmark was set to be reproducing the results shown in *Fig. 2* below, which is taken from Hedse et al. 2023. It is a plot of the isolated signal components with a frequency that is the difference between modulation frequency of the first and the second pulse as well as double that frequency, for different values of the inter-pulse time delay τ.

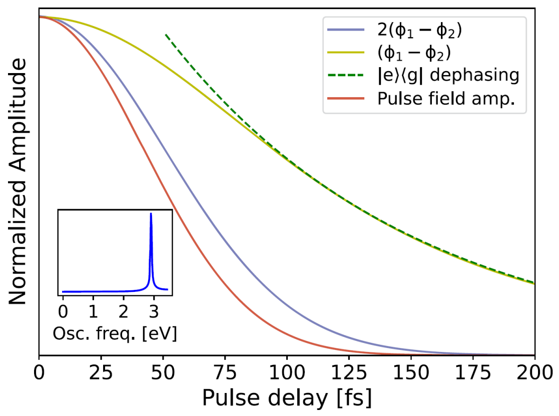


Figure 2: FIG. 4 in Hedse et al. 2023

A secondary objective was to potentially compare the results from both codes, which are obtained by numerical integration, with calculations based on the perturbative expansion of the density matrix (see Mukamel 1995, pp. 23-31; or section 7.1.2 for an introduction).

The main limitations of this work are time constraints, especially considering the of the code bases. This means in particular, that is was not possible to look at the code of the open-source *QuTiP* toolbox that is the foundation of the code for Hedse et al. 2023. Also, there was no time to look at the `System` class of *QDT*. Therefore, the code that does the actual density matrix propagation, i.e., solves the Lindblad master equation, was not investigated directly.

Furthermore, the reproducibility of results between the code basis was only investigated for one particular simulation, namely a toy model of pulse overlap artefacts in coherent multidimensional spectroscopy (CMDS) that used only two pulses. The generalizability of our results is therefore limited. In particular, *QDT* also allows for much more complex systems with several coupled n-level systems and reservoirs to be subjected to simulated spectroscopic measurements. These simulations of many-body effects have not been subjected to testing. A minor point is that *MATLAB* itself is not open source, so any issues stemming from there could not be investigated at all. Lastly, a single, ordinary consumer laptop was used when probing run times.

In the next section, the theory underlying the toolboxes used will be summarized. In section 3, details of the performed simulations will be discussed. In section 4, the results from trying to reproduce the results from Hedse et al. with the *MATLAB*-based *QDT* code will be presented and discussed. Finally, in section 5, the findings will be summarized and it will be hinted at possible angles for further investigation. The Appendix includes some derivations of equations discussed in section 2.

2 Theory

In quantum mechanics, regardless of the specific atom or molecule whose electronic states one is concerned with, one commonly idealizes the object of study, in such a way that we assume that the electrons are confined to occupy one of a few energy levels or states with defined energies. Also, we usually do not concern ourselves with the spatial dependence of the wavefunctions associated with these energy levels. In the following, we refer to this set of possible energies that the electrons can have as the *system*, and we are going to explore various mathematical tools and techniques to analyse the systems' behaviour in experiments. In particular, we want to try to understand how we can model the time evolution of the . That is, understanding the change in occupation of the energy levels over time, especially when an external potential is added and/or if we consider the possible loss of energy from the system to the environment.

2.1 Density Operator

If the initial state of a system is known, the system is said to be in a single *pure state* $|\psi(t)\rangle$ (Hamm and Zanni 2011, p. 48) and the time evolution of the system can described by inserting the wavefunction $\psi(\vec{r}, t)$ associated with the state into the Schrödinger equation (Rand 2016, p. 13):

$$i\hbar \frac{\mathrm{d}}{\mathrm{d}t} \psi(\vec{r}, t) = \hat{H} \psi(\vec{r}, t) \tag{SE}$$

However, often the initial state of the system is not fully known experimentally, but instead only the average values or the probability distributions of certain variables such as positions and momenta are known. Thus, one needs to combine methods from quantum mechanics and statistical mechanics to describe the (average) time evolution of the system (Blum 2012, p. 35; Wong 2022, p. 125).

These states are called *mixed states* or *statistical averages* and cannot be described by a single state vector (Hamm and Zanni 2011, p. 50f.; Blum 2012, p. 38), in its place we have to introduce a new representation of a quantum systems state, the density operator $\hat{\rho}(t)$, that can describe pure and mixed states.

For pure states, the use of the density operator can be motivated in the following way: Consider a pure quantum state described by the state vector $|\psi(t)\rangle$. We can write $|\psi(t)\rangle$ as the sum of i orthonormal eigenstates $|\phi_i\rangle$ that are eigenvectors/ -functions of an Hermitian[2] operator $\hat{A}$, since for all Hermitian operators there exits a complete orthonormal

[2] I.e., $\hat{A}^\dagger = \hat{A}$

basis of eigenvectors, i.e., the operator is diagonalizable (Neumann 1950):

$$\begin{cases} |\psi(t)\rangle = \sum_i c_i(t)\,|\phi_i\rangle \text{ s. t.} \\ \hat{A}\,|\phi_i\rangle = a_i\,|\phi_i\rangle \end{cases} \tag{1}$$

where $c_i(t) = \langle\phi_i|\psi(t)\rangle$ since the $|\phi_i\rangle$ are orthonormal, i.e., $\langle\phi_i|\phi_j\rangle = \delta_{ij}$. We have thus expressed the state vector w.r.t. a convenient basis, which allows us to apply algebraic methods to quantum mechanical problems, in particular, the (SE) is transformed from a differential to an eigenvalue equation. Next, we can rewrite the expectation value of the observable A, i.e., the expectation value of the operator $\hat{A}$, in the following way (Hamm and Zanni 2011, p. 48f.):

$$\begin{aligned} \langle\hat{A}\rangle &= \langle\psi(t)|\hat{A}\psi(t)\rangle \\ &= \sum_i\sum_j c_i(t)^* c_j(t)\,\underbrace{\langle\phi_i|a_j\phi_j\rangle}_{\equiv A_{ij}} \\ &\equiv \sum_i\sum_j c_i(t)^* c_j(t) A_{ij}, \end{aligned} \tag{2}$$

where the definition of A_{ij} in the last row essentially means that we represent the operator $\hat{A}$ as a matrix in its eigenvector-basis. In the following, we can omit taking the spacial dependence of the eigenfunctions into account, and we treat them purely as orthonormal basis vectors to the state vectors and operators. Similarly, we will also ignore the spatial dependence of $\hat{\rho}(t)$ and of other operators $\hat{A}$, since we will treat them largely from an algebraic view point as matrices in the basis formed by the eigenfunctions.

Equation (2) can be written more neatly if we introduce the density operator (Hamm and Zanni 2011, p. 49; Hamm 2005, p. 1):

$$\begin{aligned} \hat{\rho}(t) &\equiv |\psi(t)\rangle\langle\psi(t)| \\ &= \sum_{i,j} c_i(t)^* c_j(t)\,|\phi_i\rangle\langle\phi_j| \ \text{(by: (1))} \end{aligned} \tag{3}$$

$$\Rightarrow\ \rho_{ij}(t) \equiv \langle\phi_i|\hat{\rho}(t)|\phi_j\rangle = c_i(t)^* c_j(t) \tag{4}$$

$$\Rightarrow\ \langle\hat{A}\rangle \equiv \sum_{i,j} \rho_{ij} A_{ij} \equiv \mathrm{Tr}\Big(\hat{A}\hat{\rho}(t)\Big) \tag{5}$$

From equation (4) we can see that the density operator can be written as a matrix in the same way as other operators. Therefore, it is often referred to as the *density matrix* (Blum 2012. p. 39). In the following, we will mostly refer to operators as operators, unless we explicitly want to treat them as matrices.

For a mixed state consisting of n state vectors $|\psi_n(t)\rangle$[3], i.e., we do not know (fully) the (initial) state of the system, but we know that the system is in state $|\psi_n(t)\rangle$[4] with probability W_n, which is assumed to be time-independent because it is due to our lack of knowledge of the initial state of the system. The expectation value of an operator $\hat{A}$ for the mixed state, is thus given by the arithmetic mean of the expectation values $\langle \hat{A} \rangle_n$ of $\hat{A}$ for state $|\psi_n(t)\rangle$ (Blum 2012, p. 38):

$$\langle \hat{A} \rangle = \sum_n W_n \langle \hat{A} \rangle_n = \sum_{n,i,j} W_n c_{n,i}(t)^* c_{n,j}(t) A_{ij} \tag{6}$$

If we generalize our definition of $\hat{\rho}(t)$ to mixed states in the following way (Blum 2012, p. 39):

$$\begin{aligned} \hat{\rho}(t) &\equiv \sum_n W_n |\psi_n(t)\rangle \langle \psi_n(t)| & (7) \\ &= \sum_{n,i,j} W_n c_{n,i}(t)^* c_{n,j}(t) |\phi_i\rangle \langle \phi_j| \text{ (by: (1))} & (8) \\ \Rightarrow \rho_{ij}(t) &\equiv \langle \phi_i | \hat{\rho}(t) | \phi_j \rangle \\ &= \sum_n W_n c_{n,i}(t)^* c_{n,j}(t), & (9) \end{aligned}$$

we once again get for $\langle \hat{A} \rangle$ (Blum 2012, p. 40):

$$\langle \hat{A} \rangle \equiv \mathrm{Tr}\Big(\hat{A}\hat{\rho}(t)\Big).$$

Properties of $\hat{\rho}$ and measures of the purity of a state

Firstly, from equation (7) it follows for the eigenvalues λ_n of $\hat{\rho}$:

$$\begin{aligned} \hat{\rho} |\psi_n(t)\rangle = W_n |\psi_n(t)\rangle \equiv \lambda_n |\psi_n(t)\rangle \\ \Rightarrow \lambda_n = W_n; \end{aligned} \tag{10}$$

that is, the eigenvalues of $\hat{\rho}$ are the probability of being in the pure state $|\psi_n(t)\rangle$, which is an eigenstate of $\hat{\rho}$ for all n. Thus,

$$\begin{cases} 0 \leq \lambda_n \leq 1, \lambda_n \in \mathbb{R} \\ \sum_i \lambda_n = 1, \end{cases} \tag{11}$$

[3] $|\psi_n(t)\rangle$ are not necessarily mutually orthogonal.

[4] We assume that all $|\psi_n(t)\rangle$ can be expressed in w.r.t. the same basis $\{|\phi_i\rangle\}$

which in turn implies that $\hat{\rho}$ is Hermitian, i.e., $\hat{\rho} = \hat{\rho}^\dagger$. Furthermore, each element of $\hat{\rho}$, ρ_{ij}, is given by equation (9). This implies that

$$\begin{aligned}\rho_{ii} &\equiv \langle\phi_i|\,\hat{\rho}\,|\phi_i\rangle \\ &= \sum_n W_n |c_{n,i}(t)|^2 \\ &= |c_{i,\,\mathrm{tot}}|^2,\end{aligned} \tag{12}$$

which is equal to the probability P_i of being in state $|\phi_i\rangle$ for the case of a mixed state. This implies that (Blum 2012, p. 39f.):

$$\begin{cases} 0 \le \rho_{ii} \le 1 \\ \mathrm{Tr}(\hat{\rho}) = \sum_i P_i = 1 \end{cases} \tag{13}$$

Note that when only measuring the populations of individual energy states (and not coherences, i.e. superpositions between different states, i.e. via their transition dipole moments, see *Sec. 2.3*), one cannot distinguish between pure and mixed states. For example, there is no difference between knowing with 100% certainty that the system is in the superposition of states $|\phi_0\rangle$ and $|\phi_1\rangle$, $|\psi\rangle = 1/\sqrt{2}(|\phi_0\rangle + |\phi_1\rangle)$ and knowing that the system is either purely in state $|\psi_a\rangle = |\phi_0\rangle$ or state $|\psi_b\rangle = |\phi_1\rangle$ with 50% probability each, the diagonal elements of $\hat{\rho}$ would be the same in both cases. (Hamm 2005, p. 4) Moreover, the following properties of the density matrix can be shown (Blum 2012, p. 41f.; Hamm and Zanni 2011, p. 52):

$$\begin{cases} \mathrm{Tr}(\hat{\rho}) = 1, \text{ for pure states} \\ \mathrm{Tr}(\hat{\rho}) < 1, \text{ for mixed states} \end{cases} \tag{14}$$

Time evolution of the density operator

The time evolution of the density operator can be derived from the Schrödinger equation (SE) (Hamm 2005, p. 2; Blum 2012, p. 47-51). This derivation can be found in *Sec. 7.1.1* in the Appendix. The results is the Liouville-von Neumann equation (LvNE):

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\rho} = -\frac{i}{\hbar}[\hat{H}, \hat{\rho}], \tag{LvNE}$$

where the square brackets in the last line denote the *commutator* of $\hat{H}$ and $\hat{\rho}$.

2.2 Dephasing, Lindblad

This section leans heavily on Schlosshauer 2007, pp. 153-168. The (SE) and (LvNE) are unitary, thus "a state stays a state" (Wacker 2024, personal communication). But we want to (phenomenologically) describe processes where a pure state "loses coherence" and becomes a statistical mixture, without having to calculate the time evolution of the (large) density matrix of the environment. Thus, we need a non-unitary process, i.e. non-unitary terms added to the (LvNE). This approach is called *Master equation* approach, and the equations used are called *Master equations*. Since we are pursuing a phenomenological approach, we can in principle add anything. For the sake of reasonable computational times however, one commonly only considers Master equations that are first-order differential equations w.r.t. time, and that are furthermore local in time,

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\rho}_{\mathcal{S}}(t) = -\frac{i}{\hbar}[\hat{H}(t) + \hat{H}_C(t), \hat{\rho}_{\mathcal{S}}(t)] + \hat{\mathcal{L}}(\hat{\rho}_{\mathcal{S}}(t)),$$

that is, the superoperator $\hat{\mathcal{L}}$ should only depend on $\hat{\rho}_{\mathcal{S}}(t)$, the density matrix of the system at time t, and not on $\hat{\rho}_{\mathcal{S}}$ evaluated at other times. Here, $\hat{H}_C(t)$ is the Hamiltonian due to perturbation of the system by the environment. This effect is referred to as *Lamb-shift*. Additionally, for many systems the following two approximations can be made to simplify the Master equation: The Born approximation assumes firstly that the coupling between system and environment is weak s.t. they can be treated separately. Secondly, it assumes that the system is much smaller than the environment s.t. the environment is not changed significantly by the systems behaviour and the system, much like the reservoir used to derive the canonical ensemble in statistical mechanics. The Markov approximation assumes that if the environment has been altered by the system, these coupling-induced effects will decay rapidly and the environment will return to its initial state. This is referred as the environment having *no memory.* This gives the Born-Markov master equation, the general form of which is not discussed further here. That is because we will exclusively use a special case of this equation, namely, the Lindbladian. This class of master equations is designed to not yield unphysical density matrices, which is a problem with other classes of phenomenological master equations. More specifically, the Lindbladian is defined to satisfy the following conditions:

$$\begin{cases} \lambda_n(\hat{\rho}_{\mathcal{S}}(t)) \geq 0 \\ \dfrac{\mathrm{d}}{\mathrm{d}t}\operatorname{Tr}(\hat{\rho}_{\mathcal{S}}(t)) = 0 \end{cases} \forall t, n.$$

That is, we demand that our master equation preserves the trace of the density matrix as well as its positivity. This makes sense, since the eigenvalues λ_n of $\hat{\rho}$ are equivalent to the probabilities W_n of being in the pure state $|\psi_n(t)\rangle$ (see equation (10)) and should therefore always be non-negative. Also, the diagonal elements of $\hat{\rho}$ correspond to the probabilities of being in the state $|\phi_i\rangle$, thus the trace should remain unchanged (and equal to 1). By applying these requirements additionally to the assumptions of the Born-Markov approximation, we obtain the Lindblad form/ Lindbladian/ Gorini–Kossakowski–Sudarshan–Lindblad (GKSL) equation):

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\rho}_{\mathcal{S}}(t) = -\frac{i}{\hbar}[\hat{H}(t) + \hat{H}_C(t), \hat{\rho}_{\mathcal{S}}(t)] + \sum_{\alpha,\beta}\Gamma_{\alpha\beta}\left(\hat{J}_\alpha\hat{\rho}_{\mathcal{S}}(t)\hat{J}_\beta^\dagger - \frac{1}{2}\hat{\rho}_{\mathcal{S}}(t)\hat{J}_\beta^\dagger\hat{J}_\alpha - \frac{1}{2}\hat{J}_\beta^\dagger\hat{J}_\alpha\hat{\rho}_{\mathcal{S}}(t)\right).$$

This equation can be further simplified by diagonalizing it:

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\rho}_{\mathcal{S}}(t) = -\frac{i}{\hbar}[\hat{H}(t) + \hat{H}_C(t), \hat{\rho}_{\mathcal{S}}(t)] + \sum_{i}\Gamma_{i}\left(\hat{J}_i\hat{\rho}_{\mathcal{S}}(t)\hat{J}_i^\dagger - \frac{1}{2}\hat{\rho}_{\mathcal{S}}(t)\hat{J}_i^\dagger\hat{J}_i - \frac{1}{2}\hat{J}_i^\dagger\hat{J}_i\hat{\rho}_{\mathcal{S}}(t)\right),$$

where $\hat{J}_i$ are referred to as *jump operators*. They correspond to certain non-unitary processes, such as spontaneous emission and dephasing. Γ_i corresponds to the rate of the process associated with the jump operator $\hat{J}_i$.

2.3 Semi-classical approach & transition dipole operator

When a quantum mechanical system is subjected to an external electric field, this macroscopic external field is commonly classically. This is known as a *semi-classical* approach or model (see Greenberger et al. 2009, p. 699f.). Although being a simplification, it is often sufficient to the interaction of light with electronic systems, which is in turn the basis of spectroscopy. Since we are assuming that the each electron can only occupy one of two possible energy levels, we can conveniently write the total Hamiltonian in the basis of these two energy eigenstates $|\phi_g\rangle$ and $|\phi_e\rangle$ of the system Hamiltonian (see equation (7) in the Appendix):

$$\hat{H}(t)\,|\psi_{\text{2-level}}\rangle = \begin{pmatrix} 0 & F(t) \\ F^*(t) & \Delta \end{pmatrix} \cdot \begin{pmatrix} c_g(t) \\ c_e(t) \end{pmatrix} = \begin{pmatrix} 0 & \vec{\mu}_{eg}\vec{E}(t) \\ \vec{\mu}_{eg}^*\vec{E}(t) & \Delta \end{pmatrix} \cdot \begin{pmatrix} c_g(t) \\ c_e(t) \end{pmatrix} \tag{15}$$

where $\Delta = \varepsilon_e - \varepsilon_g$ is the energy difference between the excited state and the ground state when the system is subject to the system (unperturbed) Hamiltonian only, and $F(t) = \vec{\mu}_{eg}\vec{E}(t)$ is the energy perturbation due to the external electric field. Note that here we have set the energy of the ground state to zero to simplify the computations. The above expression introduces $\vec{\mu}$, the transition dipole operator, that couples the transitions between states to the external electric field. It is superficially analogous with the classical dipole moment, for instance, it has the same dimensions (charge times length), but it is a complex vector operator that contains phase factor from the initial and final state. In an basis set of states, it can be written as a matrix with its elements being defined as

(IUPAC 2014, p. 1558):

$$\vec{\mu}_{if} \equiv \langle\phi_f|\,\hat{\vec{\mu}}\,|\phi_i\rangle \equiv \langle\phi_f|\sum_n q_n \cdot \hat{\vec{r}}_n\,|\phi_i\rangle\,, \tag{16}$$

where $|\phi_i\rangle$ and $|\phi_f\rangle$ are the initial and final states, respectively. The sum over n runs over all charges in the system and sums up the product of their charge q_n and their position. Intuitively, one might think of the above expression as the expectation value of the electric dipole moment of the transition state or the superposition between the initial and final state, or as the expectation value of the dipole moment of the overlap of the wavefunctions of the initial and final state.

2.4 Coherent multidimensional spectroscopy (CMDS)

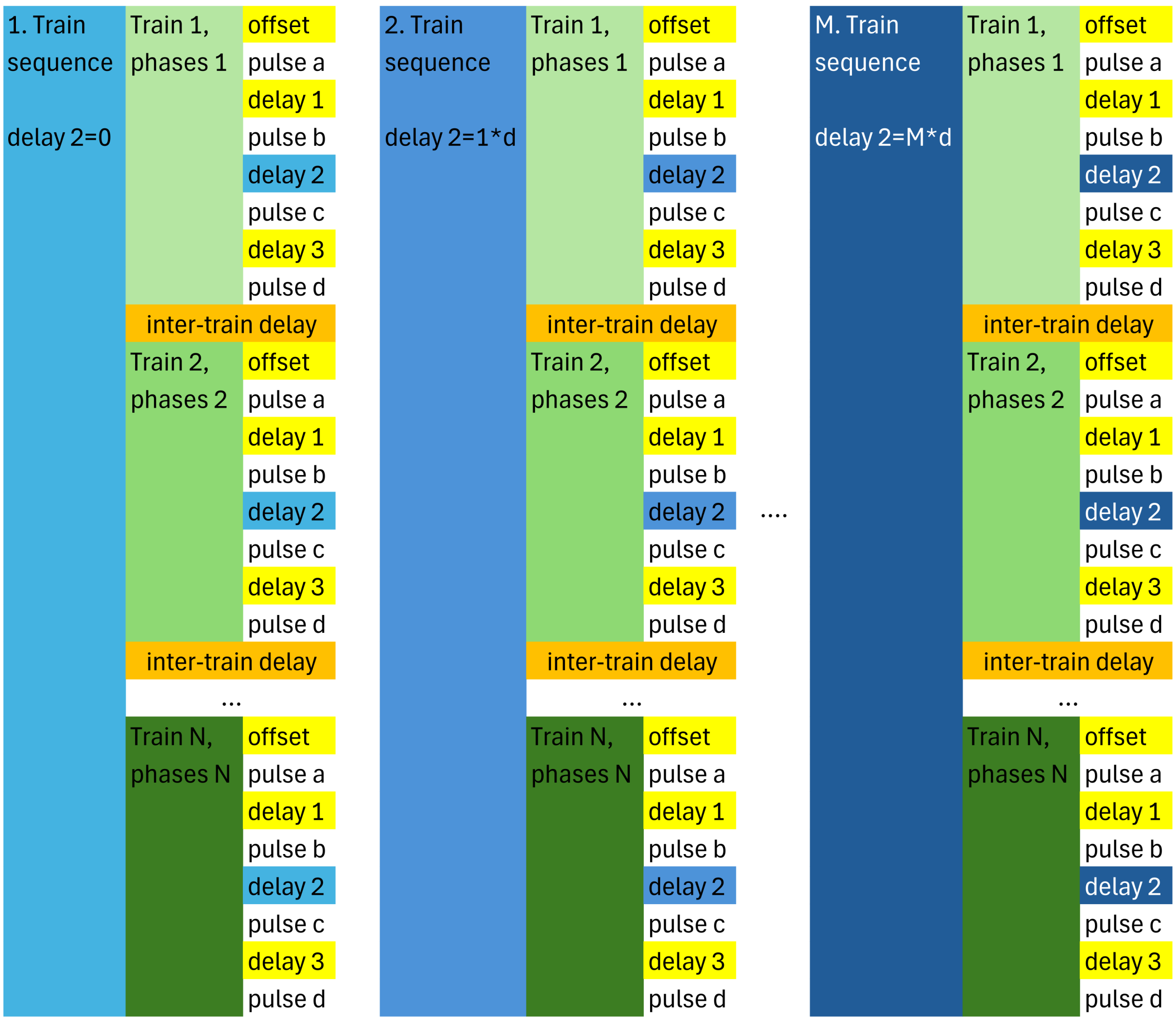


Figure 3: Diagram of the course of the spectroscopic measurement with CMDS/ 2DES. In this example, only delay 2 is varied for simplicity. In the simulations descried later on, only pulses b and c were used, and delay 2 is referred to as τ. The d in delay2=1*d corresponds to the step size in which τ is varied. (Own diagram, made with MS Excel)

In this section, the basic underlying concepts behind Coherent multidimensional spectroscopy (CMDS) are introduced. The goal of these methods is to measure decoherence times and thereby identify substances as well as molecular structures.

In the following, a quick overview of the experimental procedure is given (see Hedse et al. 2023 and Fresch et al. 2023 for more details): A long sequence of laser pulse trains, with 3-4, short, pulses[5] per pulse train, is sent onto a sample; followed by inter-train delay time that is long relative to the inter-pulse delays within the train and that allows for a significant amount of spontaneous emission to occur. The base frequencies (wavelengths) of the light are constant and lie within a relatively broad spectrum since the pulse width - measured as full width at half maximum (FWHM) - is on the order of $100 fs$ ($= 10^{-13}$ s). The spectrum should contain the resonance frequencies of the transitions of interest of substances (potentially) present in the sample.

The inter-pulse delays within the pulse train are different between different pairs of consecutive pulses, and are held constant between different pulse trains within the same sequence of pulse trains. They are varied between different sequences of pulse trains and therefore, by plotting the integrated rate of spontaneous emission, which is proportional to the integrated excited state population, against the inter-pulse delay, the decay times of the coherences (superpositions) can be determined.

Additionally, the amplitude of each pulse is modulated with a different frequency, s.t. the Fourier transform (FT) can be used to identify if the rate of spontaneous emission is proportional to the intensity - and therefore the modulation frequency - of one or more of the pulses. In the latter case, the frequency of the signal from spontaneous emission would be the sum and/or difference of the frequencies of the pulses it is proportional to. *Fig. 3* tries to illustrate the measurement procedure. Note that Hedse et al. only simulated two pulses per pulse train, since that is sufficient to demonstrate the pulse overlap artefacts.

3 Methods

3.1 General research methodology

Since the size of the *QDT* and the *QuTiP* code was to large, the approach in this thesis was high-level testing of the codes, in particular trying to get insight into the actual influence of the different parameters of each toolbox, in particular for *QDT*, since Kenneweg et al. claim that the user should be able to perform meaningful simulations ”without requiring [...] an explicit knowledge of the underlying math”. Since there are many potentially relevant parameters, this is a multifactorial optimization process, which was approached

[5]only two in this simulations since we only want to prove the point that pulse overlap causes artefacts

iteratively. The results in the next section will therefore be presented together with the conclusions drawn from the results and the adaptations that were performed and led to a new set of simulation results.

3.2 Setting up the the simulations with *QDT*

In the following, we will compare the general simulation approaches of the two codes. A table with the symbols used in this section can be found on page VII. Both codes simulate two phase matched, ultra-short laser pulses with a Gaussian envelope and the modulation phase φ_k. Thus, the electric field $\vec{E}_k(t)$ of the kth pulse is given by:

$$\vec{E}_k(t) = \vec{E}_{max} \cdot \exp\left(-\frac{(t-t_k)^2}{2\sigma^2}\right) \cdot \cos(\omega(t-t_k)+\varphi_k), \tag{17}$$

where $\vec{E}_{max}$ is the amplitude of the electric field of the laser pulses, t_k is the time when peak of the kth pulse occurs, ω is the angular frequency of the laser light, φ_k is the modulation phase of the kth pulse, and σ is a pulse width parameter of the Gaussian pulses that is analogous to the standard derivation. The difference between Hedse et al. 2023 and *QDT* is that Hedse et al. use modulation frequencies ϕ_k to modulate the amplitude of the laser beam and to obtain the modulation phases φ_k:

$$\begin{aligned} \varphi_k &= 2\pi \cdot n_t \Delta t_t \phi_k. \\ \Rightarrow \ \Delta\varphi &= \varphi_1 - \varphi_2 = 2\pi \cdot n_t \Delta t_t (\phi_1 - \phi_2), \end{aligned} \tag{18}$$

where n_t is the number of the current pulse train in the pulse train sequence, and Δt_t is the time difference between two pulse trains. In contrast, *QDT*, the phase shift $\Delta\varphi$ between the pulses is obtained by explicit phase-cycling (see Tan 2008 for more details) instead, which is implemented in the `CMDS` class of *QDT*. Phase-cycling systematically runs through all possible combinations of phases of the two pulses, which means that it is more efficient at probing pulse phases than phase cycling, but it only works if the desired pulse phases are divisors of 360° or 2π, respectively. This is shown in *Fig. 4*.

	$\phi_1 \cdot \Delta t_t=$ 90°	$\phi_2 \cdot \Delta t_t=$ 120°		$N_1=$ 4	$N_2=$ 3		$N_1=$ 1	$N_2=$ 12	
	mod. frequencies			**phase cycling 1**			**phase cycling 2**		
n_t	φ_1	φ_2	$\Delta\varphi$	φ_1	φ_2	$\Delta\varphi$	φ_1	φ_2	$\Delta\varphi$
0	0°	0°	0°	0°	0°	0°	0°	0°	0°
1	90°	120°	30°	0°	120°	120°	0°	30°	30°
2	180°	240°	60°	0°	240°	240°	0°	60°	60°
3	270°	0°	90°	90°	0°	270°	0°	90°	90°
4	0°	120°	120°	90°	120°	30°	0°	120°	120°
5	90°	240°	150°	90°	240°	150°	0°	150°	150°
6	180°	0°	180°	180°	0°	180°	0°	180°	180°
7	270°	120°	210°	180°	120°	300°	0°	210°	210°
8	0°	240°	240°	180°	240°	60°	0°	240°	240°
9	90°	0°	270°	270°	0°	90°	0°	270°	270°
10	180°	120°	300°	270°	120°	210°	0°	300°	300°
11	270°	240°	330°	270°	240°	330°	0°	330°	330°

Figure 4: Illustration of different strategies for probing inter-pulse phase differences.

Since we are looking for a signal component with whose phase is the difference between the phases of the first and second pulse, it is sufficient to only vary the phase of second pulse, φ_2, which also speeds up computations considerably. This is illustrated in the *phase cycling 2* scheme in above figure for 12 phases. Mathematically, this means that

$$\Delta\varphi = 2\pi \cdot \frac{n_t}{N_2}, \tag{19}$$

where N_2 is the number of different phases φ_2 of the second pulse to be probed. In the following, for testing purposes, only 6 phases were used in the simulations, though, in order to reduce run time and to get a first impression of the curve shape. Later the *python* code was re-run with only 6 phases as well due to limitations of available computational power and to enable a better ground for comparison. Also, later on, the number of phases was increased to test the influence of the number of phases (see *Sec. 4.8*)

parameter	value	
`offset`$= t_1 - t_0$	0	fs[6]
Pulse delay step size	5	fs
Total delay steps used	41	
`gamma0`	1.0	
N_1	1	
N_2	6	
N_t (per τ)	6	
`runTmax`	800	fs
`runTstep`	0.5	fs

Table 1: Values of the parameters used in the initial simulations, that where different from the values used in Hedse et al., or that were not specified in the paper. Names typeset in `typewriter font` correspond to variable names in the *MATLAB* script written for the simulations with *QDT*. The parameters have the following significance: `gamma0` specifies that *QDT* should performing the calculations in the lab frame, a opposed to performing them in the rotating frame. `runTmax` is is the time up to which the system is to be simulated, that is, the density matrix to be propagated for each simulation run (pulse pair/train, see *Fig. 3*). `runTstep` is the time step of that simulation run, i.e. the step size of the numerical integration

Apart from using phase-cycling, the parameters from TABLE 1 from Hedse et al. 2023 were generally used to be able to compare the results with FIG. 4 in Hedse et al. 2023. That means, a two-level system with an energy difference $\Delta = 1.46$ eV between the ground state $|\phi_g\rangle$ and the excited state $|\phi_e\rangle$. However, the amount of time-steps that were used to probe the inter-pulse delay $\tau = t_2 - t_1$ in the range [0, 200] femtoseconds (fs) was reduced from the 334 used in Hedse et al. to 41, because the resolution in τ is not so important for first comparison of the curve shape of the amplitude vs tau plot. Lastly, since not specified in Hedse et al., the `offset`$= t_1 - t_0$ of the first pulse, that is, the time difference between the start of the simulation and the maximum of the Gaussian-shaped laser pulse, was initially set to 0, but later changed to 200 fs to match with the value used in Hedse et al. (see *Sec. 4.2*). *Tab. 1* shows the values of the parameters used in the initial simulations, that where different from the values used in Hedse et al., or that were not specified in Hedse et al. 2023.

4 Results & Discussion

4.1 Initial results

The result from this simulation with *QDT* in *MATLAB* is shown in *Fig. 5a*. It shows one signal for each of the 6 phases. Each signal is periodic w.r.t. τ with the same periodicity, but shifted w.r.t. one another. The shift is proportional to $\Delta\varphi$. The envelope of the

[6]This was later changed to 200 fs to match the *QuTiP*-based script.

signals didn't match FIG. 4 in Hedse et al. 2023 (which is reproduced in *Fig. 2* in the Introduction). In the following, we will describe the conclusions drawn from this initial result and the changes made to the simulation code/ parameters.

4.2 `offset`

The envelopes in *Fig. 5c* and *d* correspond to the lowest two coefficients of the discrete Fourier transform (DFT) of the raw signal w.r.t. $\Delta\varphi$. The procedure of isolating these envelopes, similar to how it was done in Hedse et al. 2023, will be discussed in *Sec. 4.5*. Firstly, regarding the shape of the envelope, it seems that one needs to use `offset=200 fs`, which is the value used in Hedse et al. 2023. Otherwise, the left tail of the first, and - crucially - at low values of τ also the second, Gaussian pulse are cut off, i.e., occur before the start of the simulation window. This means that the sum of areas under the curve of electric field of the two pulses that is actually fired onto the system increases with τ for low τ, which most likely explains the positive slope of the signal's envelope observed for low τ. From now on, `offset=200 fs` was always used, which is the value used by Hedse et al. 2023.

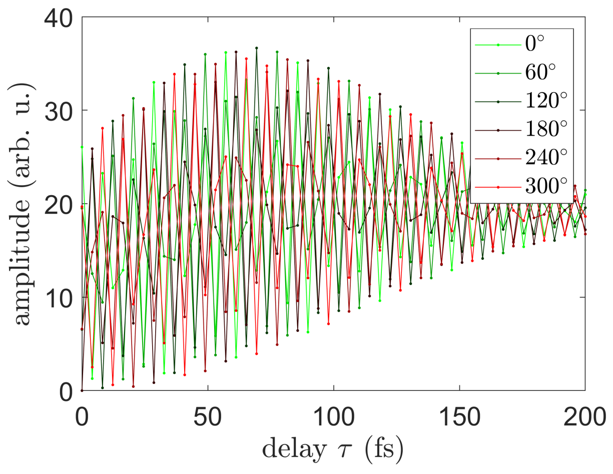
4.3 Oscillations of the signal

Moreover, when the resolution in τ was increased to better resolve the oscillations w.r.t. τ[7], it turned out that the oscillations actually had a shorter period. At a resolution of 0.02 fs, it was finally possible to resolve the oscillations. This is illustrated in *Fig. 14* in the Appendix. The periodicity was calculated as illustrated in *Fig. 15* in the Appendix. The result is 2.8289 fs. This is within 0.13% error margin of the value of:

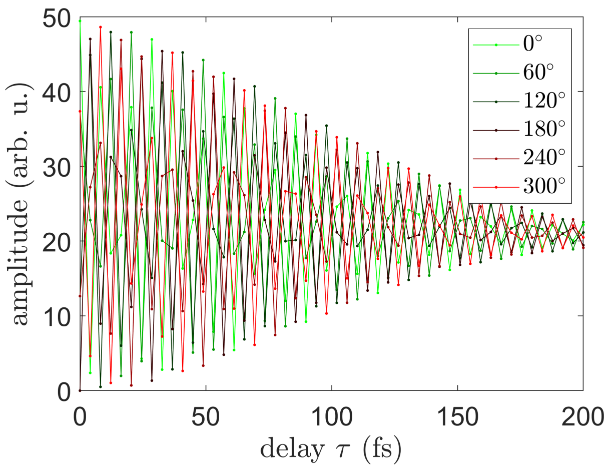
$$\frac{h}{\Delta} = \frac{h}{1.46 \text{ eV}} = \frac{h}{1.46 \cdot 1.6022 \cdot 10^{-19} \text{ J}} = 2.8326 \text{ fs},$$

where Δ is the energy difference between the ground state and the excited state of the two-level system.

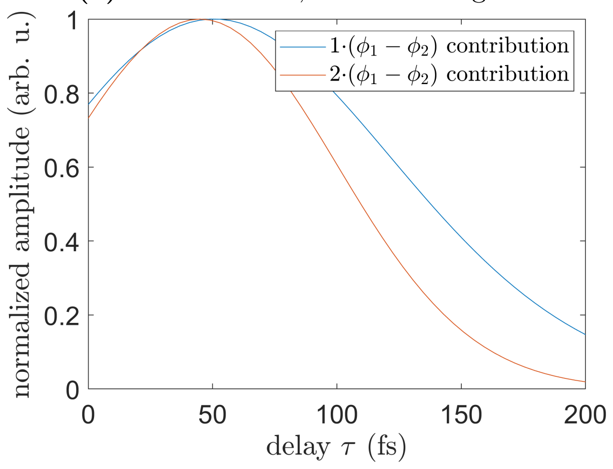
[7]It is only shown for $\Delta\varphi = 0$, but the periodicity is the same for other values of $\Delta\varphi$



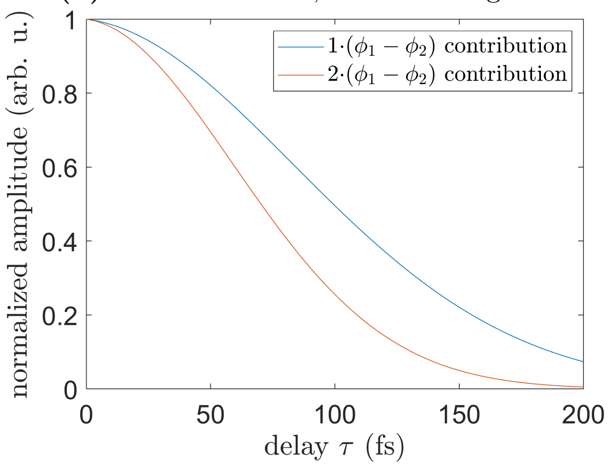
(a) `offset=0 fs`, individual signals



(b) `offset=200 fs`, individual signals



(c) `offset=0 fs`, $\phi_1 - \phi_2$ DFT components



(d) `offset=200 fs`, $\phi_1 - \phi_2$ DFT components

Figure 5: Comparing no `offset` (sub-figures a and c), and an `offset` of 200 fs (sub-figures b and d). The legends in sub-figures a and b display the value of $\Delta\varphi$.

4.4 Explaining the signal's periodicity w.r.t. τ from the (LvNE)

To explain this periodicity, we are going to make some simplifications and then solve the (LvNE). The full derivation is given in the Appendix (*Sec. 7.1.3*). The relevant Hamiltonian $\hat{H}(t)$ is given by (see equation (15)):

$$\hat{H}(t) = \begin{pmatrix} 0 & F(t) \\ F^*(t) & \Delta \end{pmatrix},$$

where $F(t)$ is the time-dependent perturbation due to the external electric field. Solving the (LvNE) for this Hamiltonian yields the following expression for $\rho_{22}(t)$:

$$\rho_{22}(t) = \frac{2}{\hbar^2} \int_{t_0}^{t} \mathrm{d}t' \, \mathrm{Im}\left\{ iF(t') \int_{t_0}^{t'} \mathrm{d}t'' \, e^{+i\frac{\Delta}{\hbar}\cdot(t''-t')} \cdot F^*(t'') \right\}$$

Now, we need to plug in $F(t)$. Combining equations (15) and (17) yields:

$$F(t) = \sum_{k=1}^{2} \vec{E}_{max}\vec{\mu}_{eg} \cdot \exp\left(-\frac{(t-t_k)^2}{2\sigma^2}\right) \cdot \cos(\omega(t-t_k) + \varphi_k);$$

that is, the Gaussian pulses simulated with QDT and the code from Hedse et al. However, in the following derivation, we are going to approximate them by Dirac delta functions, since we are only interested in a more qualitative understanding of the signal's oscillations:

$$F(t) = \sum_{k=1}^{2} \vec{E}_{max}\vec{\mu}_{eg} \cdot \sigma\sqrt{2\pi}\delta(t-t_k) \cdot \cos(\omega(t-t_k) + \varphi_k).$$

This yields the following expression for $\rho_{22}(t)$:

$$\begin{aligned}\rho_{22}(t) &= \frac{\text{Amp}}{2} \cdot \left[\cos^2(\varphi_1) + \cos^2(\varphi_2) + 2\,\text{Im}\left\{i\cos(\varphi_1)\cos(\varphi_2) \cdot e^{-i\frac{\Delta}{\hbar}\cdot(t_2-t_1)}\right\}\right] \\ &= \frac{\text{Amp}}{2} \cdot \left[\cos^2(\varphi_1) + \cos^2(\varphi_2) + 2\cos(\varphi_1)\cos(\varphi_2)\cos\left(\frac{\Delta}{\hbar}\tau\right)\right],\end{aligned}$$

with:

$$\text{Amp} = \frac{4\pi\sigma^2}{\hbar^2}\left|\vec{E}_{max}\vec{\mu}_{eg}\right|^2.$$

Thus, the angular frequency ω_s of the signal w.r.t. τ is equal to $\frac{\Delta}{\hbar}$. This model therefore - at least qualitatively - explains the oscillations of the signal w.r.t. τ with the periodicity

$$T = \frac{2\pi}{\omega_s} = \frac{2\pi}{\Delta/\hbar} = \frac{h}{\Delta}.$$

Furthermore, the signal is proportional to $\cos(1 \cdot \Delta\varphi)$. Note that this $1\cdot(\phi_1 - \phi_2)$ contribution is thus not a pulse-overlap artefact, because we modelled it under the approximation that the pulses are delta functions, that is, the pulse width was approximated to be 0. Additionally, there is a contribution proportional to $\cos(2 \cdot \Delta\varphi)$, which is the pulse overlap artefact (see FIG. 2 in Hedse et al. 2023). However, the phase shift of the curves in *Fig. 15a* to one another is proportional to $1 \cdot \Delta\varphi$. This makes sense, because the un-normalised $1 \cdot (\phi_1 - \phi_2)$ signal component is on the order of 100 times greater than the $2 \cdot (\phi_1 - \phi_2)$ component. However, in the following, both contributions are always plotted normalised, to be able to compare their τ-dependence. Moreover, also the $\cos^2(\varphi_k)$ components are four times weaker than the $2\cos(\varphi_1)\cos(\varphi_2)\cos\left(\frac{\Delta}{\hbar}\tau\right)$, and also only φ_2 was varied in QDT, which is probably why the periodic phase shift observed between the signals of different φ only corresponded to the $\cos(\varphi_1)$ term.

4.5 Isolating the envelope of the signal

We now want to finally isolate the envelope of the $m \cdot \Delta\varphi$, $m \in \{1, 2\}$ signal component, which is equivalent to $m(\phi_1 - \phi_2)$ signal component in Hedse et al. 2023, since $\Delta\varphi$ is proportional to $(\phi_1 - \phi_2)$ for the modulation frequency approach employed there. (see (18)). Since $\Delta\varphi$ is obtained differently in *QDT* (See (19)), we will perform a discrete Fourier transform (DFT) w.r.t. to $\Delta\varphi$, in order for the approach and its result to be valid for datasets acquired from either simulation.

Previously, we have seen that the raw signal has components periodic w.r.t. τ and $m\Delta\varphi$. Mathematically,

$$\rho_{22} \propto a \cdot b, \text{ s.t.: } \begin{cases} a(\tau + h/\Delta) = a(\tau) \\ b(m\Delta\varphi + 2\pi) = b(m\Delta\varphi) \end{cases}$$

Furthermore, the envelopes/ amplitudes of the signal decrease with increasing τ comparatively slowly. This τ-dependence is due to pulse overlap artefacts and spontaneous emission (see Hedse et al. 2023), both of which are not taken into account by the model presented in the previous section. Since the time scale over which the envelope varies is significantly bigger than the period of approximately 3 fs, we can assume the two τ-dependences to be separable. We can thus write ρ_{22} as a multivariable function in the following way:

$$\rho_{22}\left(\tau, \frac{\Delta}{\hbar}\tau \pm m\Delta\varphi\right),$$

where the first argument corresponds to the time-dependence of the envelope only, and the second argument to the τ-dependence due to the $\cos\left(\frac{\Delta}{\hbar}\tau\right)$ term as well as the $\cos(m\Delta\varphi)$ dependence. The $\pm$ comes from the fact that $\cos(\alpha)\cos(\beta) = [\cos(\alpha + \beta) + \cos(\alpha - \beta)]/2$. Here, we have not considered the $\varphi_1 + \varphi_2$ dependence, because we are not interested in it. The DFT was performed as follows to isolate the Fourier coefficients $A_{m\Delta\varphi}$ of the $m \cdot \Delta\varphi$ frequency component:

$$|A_{m\Delta\varphi}| = \left| \frac{1}{N_t} \sum_{n_t=0}^{N_t-1} \rho_{22}\left(\tau, \frac{\Delta}{\hbar}\tau \pm 2\pi m \frac{n_t}{N_t}\right) \cdot \exp\left(2\pi i \cdot m \frac{n_t}{N_t}\right) \right|$$

Introducing the substitution

$$\Theta_m^{\pm} = \frac{\Delta}{\hbar} \pm 2\pi m \frac{n_t}{N_t}$$

gives:

$$\begin{aligned}|A_{m\Delta\varphi}| &= \left|\frac{1}{N_t}\sum_{n_t=0}^{N_t-1}\rho_{22}(\tau,\Theta_m^\pm)\cdot\exp\left(i\cdot\left[\pm\Theta_m^\pm \mp \frac{\Delta}{\hbar}\tau\right]\right)\right| \\ &= \left|\exp\left(\mp i\frac{\Delta}{\hbar}\tau\right)\left[\frac{1}{N_t}\sum_{n_t=0}^{N_t-1}\rho_{22}(\tau,\Theta_m^\pm)\cdot\exp\left(\pm i\Theta_m^\pm\right)\right]\right| \\ &\equiv \underbrace{\left|\exp\left(\mp i\frac{\Delta}{\hbar}\tau\right)\right|}_{=1}\left|A_{\Theta_m^\pm}(\tau)\right|,\end{aligned}$$

which is the Fourier coefficient of ρ_{22} w.r.t. $\Theta_m^\pm$ and thus not dependent on that variable. This means that by taking the absolute value of the first and second coefficients of the discrete Fourier transform (DFT) of the raw signal w.r.t. $\Delta\varphi$, respectively, our signal processed in this way only retains the τ-dependence of the slowly varying envelope. This is what we want since it is equivalent to the processing done in Hedse et al. 2023. This processing was first conducted with self-written code and then using a function from *QDT* to calculate $\frac{1}{N_t}\exp\left(2\pi i\cdot m\frac{n_t}{N_t}\right)$ automatically.

4.6 Comparison of intermediate results

After this improvements, the intermediate results obtained with *QDT* were compared with the results from *python* code. The parameters, used in the simulation can be found in *Tab. 2*. The isolated envelopes of the signals have a very similar shape, as it can be seen in *Fig. 6*. However, they diverge from each other for large values of τ. Thus, further inspection of the codes was undertaken.

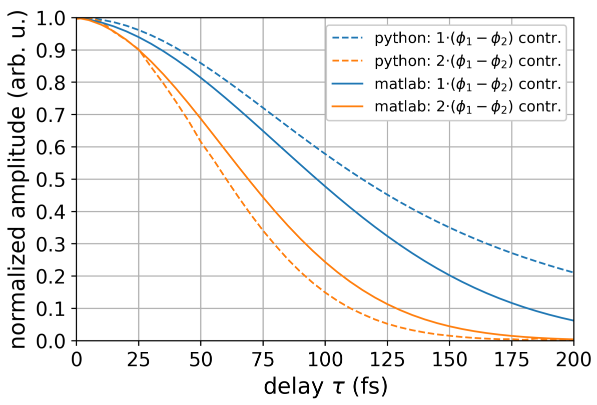


Figure 6: Comparing intermediate results obtained with *QDT* with results from *python* code. Parameters: See *Tab. 2*.

parameter	***QDT***	***python***
N_1	1	-
N_2	6	-
ϕ_1 [MHz]	-	0
ϕ_2 [MHz]	-	50/3
Δt_t [ns]	-	10
$\Delta\phi$ [rad]	$n_t 2\pi/6$	
N_t	6	
integration window*	fixed	variable
pulse width [fs]**	≈ 141	100
`runTstep` [fs]	0.5	≈ 0.1

Table 2: Parameters used in the simulations for *Fig. 6*
*"fixed"$\equiv [0, 800]$ fs, "variable"$\equiv [\texttt{offset} + \tau + 2 \cdot \text{FWHM} + 50 \text{ fs}]$.
**FWHM of electric field amplitude

4.7 Integration time

The next factor investigated and compared between the two codes was the integration time of the excited state population ($|e\rangle \langle e|$). It turned out that Hedse et al. use a dynamic integration window of 50 fs starting after the second pulse has subsided, in particular the integration interval is defined as follows:

$$[\texttt{offset} + \tau + 2 \cdot \text{FWHM} + 50 \text{ fs}].$$

In contrast, the default return function of the `CMDS` class of *QDT* uses a static integration window that in fact runs over the whole simulation time; that is, independent of the value of τ, the integration interval in *QDT* is defined as:

$$[0, 800] \text{ fs}.$$

Since the *python* code was easier to change, for the purpose of comparison of the codes, the *python* codes integration window was changed by Andreas Wacker to match the one of *QDT*. (see *Tab. 3*). The simulation results from the changed *python* code are significantly altered, but they still do not match with the results from *QDT*, as can be seen in *Fig. 7*.

parameter	***python***		***QDT***
integration window*	variable	fixed	fixed
pulse width [fs]**	100	100	≈ 141
`runTstep` [fs]	≈ 0.1	≈ 0.1	0.5

Table 3: Parameters used in the simulations for *Fig. 7*
*"fixed"$\equiv [0, 800]$ fs, "variable"$\equiv [\texttt{offset} + \tau + 2 \cdot \text{FWHM} + 50 \text{ fs}]$.
**FWHM of electric field amplitude

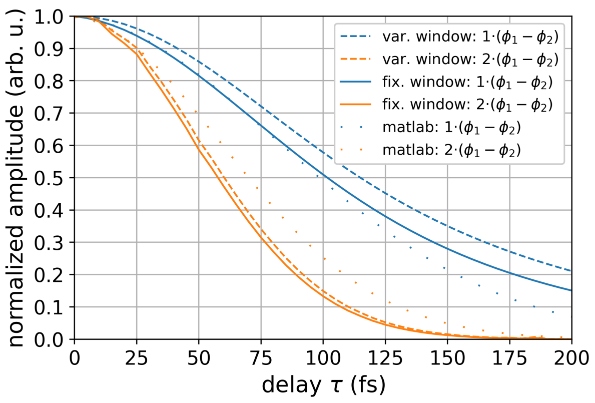


Figure 7: Comparing the results from the *python* code for two different integration windows with the results from *QDT*. The parameters that were changed compared to *Tab. 2* are listed in *Tab. 3*. "fix."≡ fixed, "var."≡ variable.

4.8 Number of phases probed

Another difference between the implementations is the number of phase differences calculated. In Hedse et al., the maximal phase difference is equal to

$$2\pi(\phi_1 - \phi_2) \cdot \Delta t_t \cdot N_t = 2\pi \cdot 500\ \text{kHz} \cdot 14\ \text{ns} \cdot 5000 = 35 \cdot 2\pi,$$

which means effectively, a thousand different phase differences were probed, with the increment of the phase difference between subsequent pulse trains being $14\pi/1000$, and each value of the phase difference was probed by 5 pulse trains, which appears to be redundant. The effect of increasing the number of phases from 6 to 1000, which is equivalent to what was simulated in Hedse et al., was studied in *Fig. 8*.

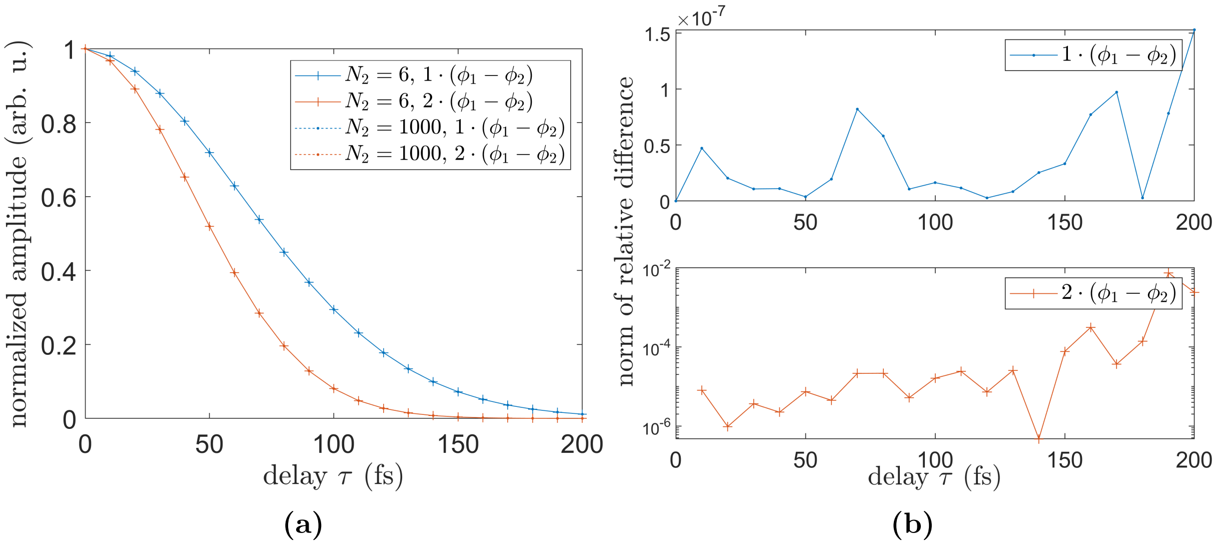


Figure 8: The effect of increasing N_2. a shows the overlays of the $1 \cdot (\phi_1 - \phi_2)$ and the $2 \cdot (\phi_1 - \phi_2)$ contributions for $N_2 = 6$ and $N_2 = 1000$. b shows the relative difference between the $N_2 = 6$ and $N_2 = 1000$.

Apart from an outlier for $\tau = 190$ fs, the relative difference was below 10^{-4} for the $2 \cdot (\phi_1 - \phi_2)$ contribution, and even lower for the $1 \cdot (\phi_1 - \phi_2)$ contribution. Also, changing the *python* code from 1000 to 6 phase differences, did not alter the results of the *python* code significantly. Therefore, the number of phases is largely irrelevant for the shape of DFT component's curves, and most certainly does not explain the differences observed. Also, it was tried to vary the phases of both pulses, but this also did not alter the results. This is what we expect, since we are isolating the signal components with the multiple of the phase difference between the pulses, so it makes sense that only the pulse difference matters. In the following, other parameters were varied whilst keeping the number of phases differences at 6 in both codes to speed up the computations and to make it easier to compare the raw results. Also, only φ_2 was varied in both codes since we are only looking at a signal contribution with the difference frequency.

4.9 Pulse width

The next discrepancy found was in the conversion of the FWHM into the sigma parameter for the Gaussian envelope of the laser pulses:

$$|\vec{E}|(t) = \sum_{k=1}^{2} |\vec{E}_0| \cdot \exp\left(-\frac{(t-t_k)^2}{2\sigma^2}\right) \cdot \cos\left(\omega(t-t_k) + \frac{2\pi \cdot n_t}{N_k}\right),$$

$$\text{with } \sigma = \frac{\text{FWHM}}{2\sqrt{2\ln(2)}}, \text{ and thus:}$$

$$|\vec{E}|(t) = \sum_{k=1}^{2} |\vec{E}_0| \cdot \exp\left(-4\ln(2)\frac{(t-t_k)^2}{\text{FWHM}^2}\right) \cdot \cos\left(\omega(t-t_k) + \frac{2\pi \cdot n_t}{N_k}\right).$$

However, the *QDT* code uses $\exp\left(-2\ln(2)\frac{(t-t_k)^2}{\text{FWHM}^2}\right)$, possibly because they define the FWHM as the FWHM of the laser intensity, which is the square of the amplitude:

```
function exF = inner(t)
    exF = amp*exp(-2*log(2)*((t-tdsum(1))/(tp)).^2) .*
    exp(-1i*(we*(t-gamma*tdsum(1)) + phsum(1))...
    + amp*exp(-2*log(2)*((t-tdsum(2))/(tp)).^2) .*
    exp(-1i*(we*(t-gamma*tdsum(2)) + phsum(2)));
    exF = real(exF);
end
```

Listing 1: Code for the electric field strength over time for a two-pulse sequence of Gaussian pulses.

Thus, to have the same pulse width as the *python* code, the pulse width parameter in the *MATLAB* code was set to $(100 \text{ fs})/\sqrt{2}$. This changed the curves significantly, but still the differences between the results remained (see *Fig. 9* and *Tab. 4*).

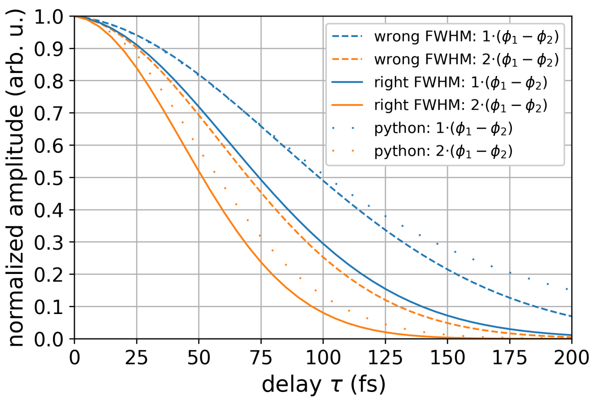


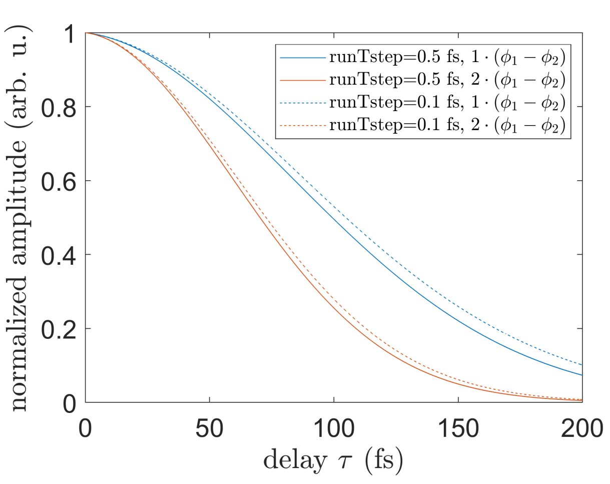
Figure 9: Comparing the results from *QDT* before and after correcting the pulse width to match the results from the *python* code. The results from the *python* code are also shown for comparison. The parameters that were changed compared to *Tab. 2* are listed in *Tab. 4*. "fix."≡ fixed, "var."≡ variable.

parameter	***QDT***		***python***
integration window*	fixed	fixed	fixed
pulse width [fs]**	≈ 141	100	100
`runTstep` [fs]	0.5	0.5	≈ 0.1

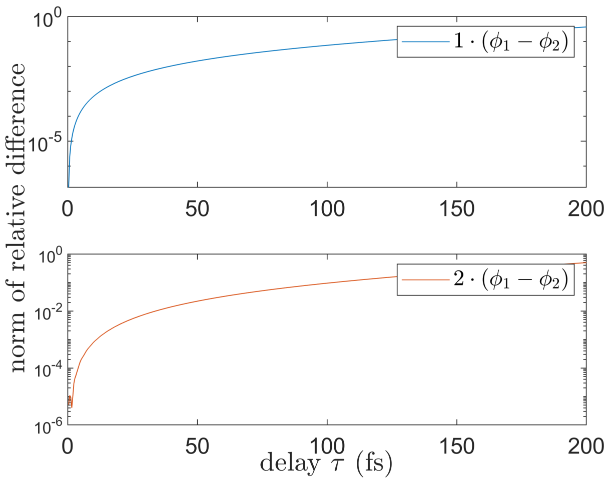
Table 4: Parameters used in the simulations for *Fig. 9*
*"fixed"≡ $[0, 800]$ fs, "variable"≡ $[\texttt{offset} + \tau + 2 \cdot \text{FWHM} + 50\text{ fs}]$.
**FWHM of electric field amplitude

4.10 Step size of the numerical integration

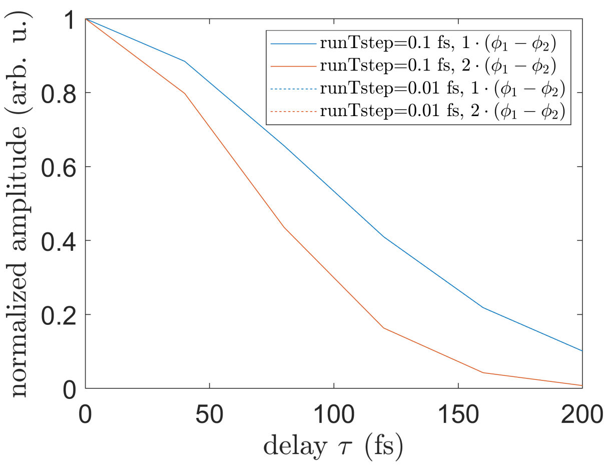
Another parameter investigated was the step size of the numerical integration `runTstep`. Hedse et al. use a step size of 0.012 fs. As can be seen in *Fig. 10* (sub-figure a), decreasing `runTstep` from 0.5 fs to 0.1 shifts the curves of both the $1 \cdot (\phi_1 - \phi_2)$ and the $2 \cdot (\phi_1 - \phi_2)$ contribution slightly upwards, but not anywhere close to the results from Hedse et al. Further decreasing `runTstep` from 0.1 to 0.01 fs did not alter the results any further (sub-figure c), thus it seems that this resolution is sufficient and also that `runTstep` is not responsible for the discrepancies.



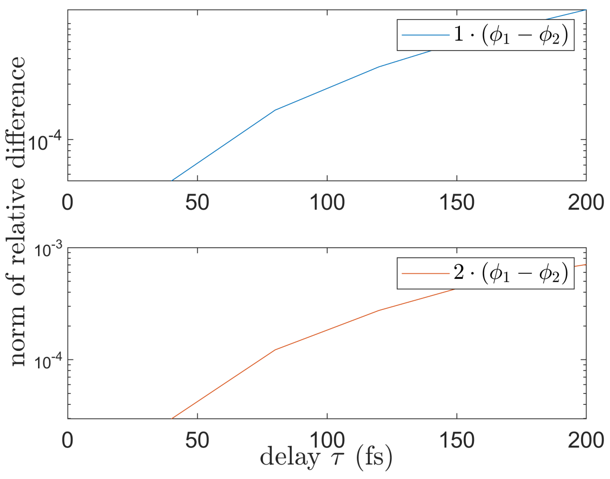
(a) Overlay of `runTstep`=0.5 and 0.1 fs.



(b) Relative difference between `runTstep`=0.5 and 0.1 fs.



(c) Overlay of `runTstep`=0.1 and 0.01 fs.



(d) Relative difference between `runTstep`=0.1 and 0.01 fs.

Figure 10: Comparison of results for different values of `runTstep`. Other parameters are the same as listed in *Tab. 2* for *QDT*.

Fig. 16 in the Appendix shows the electric field of the second pulse at different values of `runTstep`, which shows that `runTstep`=0.1 fs is sufficient to resolve the oscillations of the electric field with the laser frequency of 1.45 eV and therefore further strengthening the claim that this resolution is sufficient. Therefore, `runTstep`=0.1 fs was used for all further simulations.

4.11 Unit conversion

A last point of scrutiny was the unit conversions functionalities provided by *QDT*. Since the code uses atomic units, the toolbox offers extensive unit conversion factors so that the user can easily multiply parameter given in SI units by the provided conversion factors to convert them into atomic units, and the reverse for the simulation results. However, the conversion factor from fs to atomic time units is 41.49 instead of 41.34, which means an error of 0.37%. This does not affect the results significantly, but it was replaced by a manually set conversion factor because it might lead to imprecisions at high values of τ:

$$200 \text{ fs} \cdot 0.37 = 0.74 \text{ fs},$$

which might be significant w.r.t. a periodicity of 2.85 fs. Motivated by this discovery, the other unit conversion factor used, the unit conversion factors between electronvolt and atomic units of angular frequency were checked, but they are correct to high precision.

4.12 Comparison of final results

Fig. 11 shows the overlay of curves of the results obtained with *QDT* and with the *QuTiP*-based *python* code from the paper from Hedse et al., after both codes were changed to have matching parameters (see *Tab. 5*). In particular, the *python* code was changed in that ϕ_1 was set to 0 and ϕ_2 was set so that $\Delta\varphi = n_t \frac{2\pi}{N_2}$.

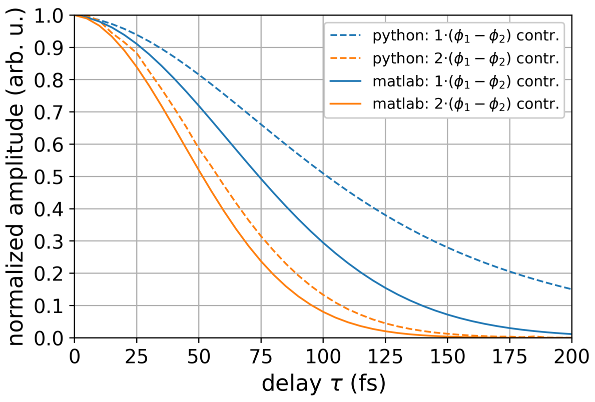


Figure 11: Comparing the final results from *QDT* with final results from *python* code, after trying to have matching parameters. The parameters that were matched are listed in *Tab. 5* below.

parameter	***QDT*** ***python***
integration window*	fixed
pulse width [fs]**	100
`runTstep` [fs]	$800/2^{13} \approx 0.1$

Table 5: Parameters used in the simulations for *Fig. 11*
*"fixed"≡ $[0, 800]$ fs, "variable"≡ $[\texttt{offset} + \tau + 2 \cdot \text{FWHM} + 50 \text{ fs}]$.
**FWHM of electric field amplitude

As it can be seen from the figure, the results still do not match.

4.13 Curve overlays & differing phases

To gain additional insight, $\rho_{22}(\tau, \Delta\varphi)$ was plotted against τ for fixed values of $\Delta\varphi$ (*Fig. 12*) and against $\Delta\varphi$ for fixed values of τ (*Fig. 13*). The parameters of *Tab. 5* were used.

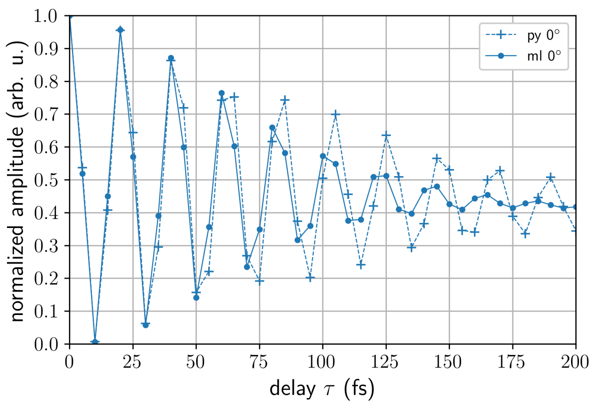
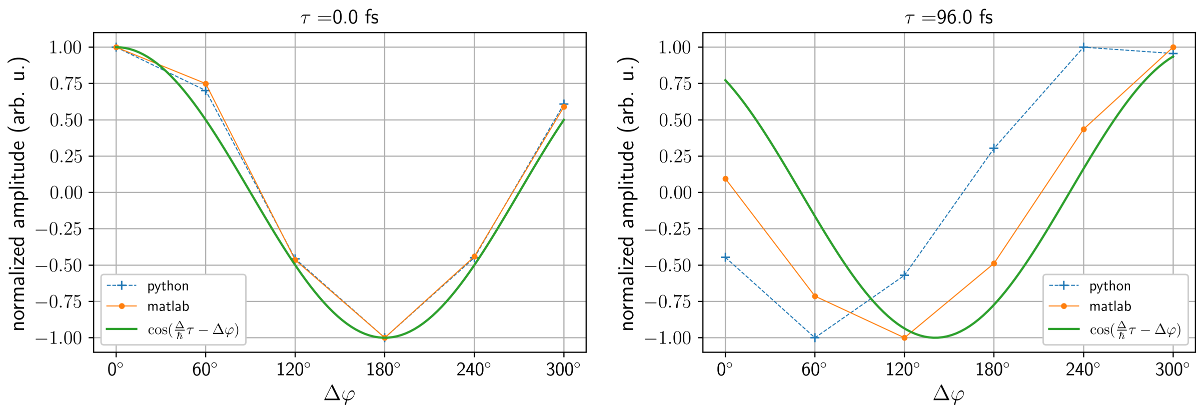


Figure 12: Plot of $\rho_{22}(\tau, \Delta\varphi)$ against τ for $\Delta\varphi = 0°$. Other parameters: See *Tab. 5*.



(a) $\tau = 0$ fs.

(b) $\tau = 96$ fs.

Figure 13: Plot of $\rho_{22}(\tau, \Delta\varphi)$ against $\Delta\varphi$ for fixed values of τ. Other parameters: See *Tab. 5*.

It can be seen, firstly that the envelopes diverge w.r.t. each other. Secondly, the curves for $\rho_{22}(\tau, \Delta\varphi)$ from both simulations start off in phase for small τ, but seem to get out of phase for larger values of τ. No explanation could be found to account for this behaviour.

5 Conclusions & Outlook

It was not possible to reproduce the results from Hedse et al. with the *QDT* code, in particular it was only possible to reproduce FIG. 4 in Hedse et al. 2023 qualitatively, but not quantitatively. Also after changing the code used in Hedse et al., in order for the simulation to resemble more closely the simulation performed by the *QDT* code, the curves produced by the two code bases have significant quantitative discrepancies between them, for which no explanation has been found to date, even though several smaller discrepancies could be resolved, with the differences in the definition of the integration time and the definition of the FWHM having the largest influence on the results. However, changing the FWHM parameter in the *QDT*-based simulation to match the *QuTiP*-based codes definition, made the results obtained with *QDT* more dissimilar from the results obtained with the *python* code from Hedse et al., suggesting that there are other factors causing the differences.

A possible angle for further study could be to investigate why the raw signals from both simulations start off in phase for small τ, but seem to get out of phase for larger values of τ.

6 References

Adams, Robert A. *Calculus: A Complete Course.* 8. ed. Toronto: Pearson Education, 2014. ISBN: 978-0-321-78107-9 978-1-4479-5892-5 978-0-273-74290-6.

BIPM, (Bureau international des poids et mesures). “Conférence Générale Des Poids et Mesures - Comptes Rendus de La 26e Réunion de La CGPM”. In: *Comptes Rendus de La 26e Réunion de La Conférence Générale Des Poids et Mesures (Novembre 2018).* Conférence Générale Des Poids et Mesures. Nov. 16, 2018.

Blum, Karl. *Density Matrix Theory and Applications.* Vol. 64. Springer Series on Atomic, Optical, and Plasma Physics. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. ISBN: 978-3-642-20560-6. DOI: 10.1007/978-3-642-20561-3.

Brown, Peter N., George D. Byrne, and Alan C. Hindmarsh. “VODE: A Variable-Coefficient ODE Solver”. In: *SIAM Journal on Scientific and Statistical Computing* 10.5 (Sept. 1989), pp. 1038–1051. DOI: 10.1137/0910062.

Curtiss, C. F. and J. O. Hirschfelder. “Integration of Stiff Equations”. In: *Proceedings of the National Academy of Sciences of the United States of America* 38.3 (Mar. 1952), pp. 235–243. DOI: 10.1073/pnas.38.3.235. pmid: 16589085.

Envelope (Waves) - Wikipedia. In: *Wikipedia.* Jan. 5, 2024.

Fresch, Elisa et al. “Two-Dimensional Electronic Spectroscopy”. In: *Nature Reviews Methods Primers* 3.1 (Nov. 9, 2023), pp. 1–16. DOI: 10.1038/s43586-023-00267-2.

Gear, C. W. “The Numerical Integration of Ordinary Differential Equations”. In: *Mathematics of Computation* 21.98 (1967), pp. 146–156. DOI: 10.1090/S0025-5718-1967-0225494-5.

Greenberger, Daniel, Klaus Hentschel, and Friedel Weinert, eds. *Compendium of Quantum Physics.* Berlin, Heidelberg: Springer, 2009. ISBN: 978-3-540-70622-9. DOI: 10.1007/978-3-540-70626-7.

Gupta, V. P., ed. *Molecular and Laser Spectroscopy. Volume 3.* Amsterdam Oxford Cambridge, MA: Elsevier, 2022. ISBN: 978-0-323-91249-5.

Hamm, Peter. *Principles of Nonlinear Optical Spectroscopy: A Practical Approach - or: Mukamel for Dummies.* Aug. 26, 2005. URL: https://www.chem.uci.edu/~dmitryf/manuals/Fundamentals/Mukamel%20for%20dummies.pdf (visited on 03/01/2024).

Hamm, Peter and Martin T. Zanni. *Concepts and Methods of 2d Infrared Spectroscopy.* Cambridge ; New York: Cambridge University Pres, 2011. ISBN: 978-1-107-00005-6.

Hedse, Albin et al. “Pulse Overlap Artifacts and Double Quantum Coherence Spectroscopy”. In: *The Journal of Chemical Physics* 158.14 (Apr. 12, 2023), p. 141104. DOI: 10.1063/5.0146148.

IUPAC. *Compendium of Chemical Terminology - IUPAC Gold Book.* International Union of Pure and Applied Chemistry, Apr. 24, 2014.

Johansson, J. R., P. D. Nation, and Franco Nori. “QuTiP 2: A Python Framework for the Dynamics of Open Quantum Systems”. In: *Computer Physics Communications* 184.4 (Apr. 1, 2013), pp. 1234–1240. DOI: 10.1016/j.cpc.2012.11.019.

– “QuTiP: An Open-Source Python Framework for the Dynamics of Open Quantum Systems”. In: *Computer Physics Communications* 183.8 (Aug. 1, 2012), pp. 1760–1772. DOI: 10.1016/j.cpc.2012.02.021.

Kenneweg, Tristan et al. “QDT — A Matlab Toolbox for the Simulation of Coupled Quantum Systems and Coherent Multidimensional Spectroscopy”. In: *Computer Physics Communications* 296 (Mar. 1, 2024), p. 109031. DOI: 10.1016/j.cpc.2023.109031.

Kong, Qingkai, Timmy Siauw, and Alexandre M. Bayen. *Python Programming and Numerical Methods: A Guide for Engineers and Scientists*. London: Elsevier, Academic Press, 2021. ISBN: 978-0-12-819550-5 978-0-12-819549-9. DOI: 10.1016/C2018-0-04165-1.

Minhaeng Cho, ed. *Coherent Multidimensional Spectroscopy*. Vol. 226. Springer Series in Optical Sciences. Singapore: Springer, 2019. ISBN: 9789811397523. DOI: 10.1007/978-981-13-9753-0.

Mueller, Stefan et al. “Rapid Multiple-Quantum Three-Dimensional Fluorescence Spectroscopy Disentangles Quantum Pathways”. In: *Nature Communications* 10.1 (Oct. 18, 2019), p. 4735. DOI: 10.1038/s41467-019-12602-x.

Mukamel, Shaul. *Principles of Nonlinear Optical Spectroscopy*. Oxford Series in Optical and Imaging Sciences 6. New York: Oxford Univ. Press, 1995. ISBN: 978-0-19-509278-3 978-0-19-513291-5.

Neumann, Johann Von. “Eine Spektraltheorie Für Allgemeine Operatoren Eines Unitären Raumes. Erhard Schmidt Zum 75. Geburtstag in Verehrung Gewidmet”. In: *Mathematische Nachrichten* 4.1-6 (1950), pp. 258–281. DOI: 10.1002/mana.3210040124.

Press, William H., ed. *Numerical Recipes: The Art of Scientific Computing*. 3. ed. Cambridge: Cambridge University Press, 2007. ISBN: 978-0-521-88068-8.

Qutip.Mesolve — QuTiP 4.0 Documentation. 2011. URL: https://qutip.org/docs/4.0.2/modules/qutip/mesolve.html (visited on 05/08/2024).

Rand, Stephen C. *Lectures on Light: Nonlinear and Quantum Optics Using the Density Matrix*. Oxford University Press, June 9, 2016. ISBN: 978-0-19-181783-0. DOI: 10.1093/acprof:oso/9780198757450.001.0001.

Schlosshauer, Maximilian. *Decoherence and the Quantum-To-Classical Transition*. Frontiers Collection. Berlin, Heidelberg: Springer, 2007. ISBN: 978-3-540-35773-5. DOI: 10.1007/978-3-540-35775-9.

Scipy.Integrate.Ode — SciPy v1.13.0 Manual. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.ode.html (visited on 05/08/2024).

Tan, Howe-Siang. “Theory and Phase-Cycling Scheme Selection Principles of Collinear Phase Coherent Multi-Dimensional Optical Spectroscopy”. In: *The Journal of Chemical Physics* 129.12 (Sept. 22, 2008), p. 124501. DOI: 10.1063/1.2978381.

Weisstein, Eric W. *Matrix Norm.* URL: https://mathworld.wolfram.com/ (visited on 03/26/2024).

Wong, Hiu Yung. *Introduction to Quantum Computing: From a Layperson to a Programmer in 30 Steps.* Cham: Springer International Publishing, 2022. ISBN: 978-3-030-98339-0. DOI: 10.1007/978-3-030-98339-0.

7 Appendix

7.1 Additional derivations

7.1.1 Derivation of the (LvNE)

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\hat{\rho} &= \frac{\mathrm{d}}{\mathrm{d}t}\sum_n W_n \left|\psi_n(t)\right\rangle \left\langle\psi_n(t)\right| \\
&= \sum_n W_n \left[\left(\frac{\mathrm{d}}{\mathrm{d}t}\left|\psi_n(t)\right\rangle\right)\left\langle\psi_n(t)\right| + \left|\psi_n(t)\right\rangle\left(\frac{\mathrm{d}}{\mathrm{d}t}\left\langle\psi_n(t)\right|\right)\right] \\
&= \sum_n W_n \left[\left(-\frac{i}{\hbar}\hat{H}\left|\psi_n(t)\right\rangle\right)\left\langle\psi_n(t)\right| + \left|\psi_n(t)\right\rangle\left(-\frac{i}{\hbar}\hat{H}\left|\psi_n(t)\right\rangle\right)^{\dagger}\right] \quad \text{(by (SE))} \\
&= \sum_n W_n \left[\left(-\frac{i}{\hbar}\hat{H}\left|\psi_n(t)\right\rangle\right)\left\langle\psi_n(t)\right| + \left|\psi_n(t)\right\rangle\left(\frac{i}{\hbar}\left\langle\psi_n(t)\right|\hat{H}\right)\right] \\
&= -\frac{i}{\hbar}\left(\hat{H}\sum_n W_n \left|\psi_n(t)\right\rangle\left\langle\psi_n(t)\right| - \sum_n W_n \left|\psi_n(t)\right\rangle\left\langle\psi_n(t)\right|\hat{H}\right) \\
&= -\frac{i}{\hbar}[\hat{H},\hat{\rho}] \qquad \text{(LvNE)}
\end{aligned}$$

7.1.2 Interaction picture & perturbative expansion

The goal of this section is to derive an expression for the time evolution of an otherwise isolated quantum system that is subjected to a weak (optical) interaction that can be treated perturbatively. The derivations are largely taken from Hamm 2005, pp. 12f., 16-19; and Hamm and Zanni 2011, pp. 54-57.
We begin by splitting the full Hamiltonian $\hat{H}(t)$ into the time-independent *system Hamiltonian* $\hat{H}_0$, which describes the total energy of the system in the absence of a perturbation, and the Hamiltonian of the perturbation $\hat{H}'(t)$, that contains the potential(s) due to the interaction:

$$\hat{H}(t) = \hat{H}_0 + \hat{H}'(t), \tag{20}$$

s.t. $\|\hat{H}'(t)\| \ll \|\hat{H}_0\| \, \forall t$[8]. It is convenient to write $\hat{H}(t)$ in the eigenstate basis of $\hat{H}_0$. Working in this space means that the matrix representation of $\hat{H}_0$ will be diagonal but those of $\hat{H}'(t)$ and $\hat{H}(t)$ not. It is those off-diagonal elements of $\hat{H}'(t)$ and $\hat{H}(t)$ that couple between the eigenstates. Next, let's look at the time-evolution of the system subject to

[8]There are several definitions for the norm of a matrix (Weisstein 2024). However, since we are only interested in stating that the weak perturbation is weak, we will leave it at this qualitative (and not so rigorous) statement.

only $\hat{H}_0$:

$$\begin{aligned} \frac{\mathrm{d}\,|\psi(t)\rangle}{\mathrm{d}t} &= -\frac{i}{\hbar}\hat{H}_0\,|\psi(t)\rangle \\ \Leftrightarrow\ \frac{\frac{\mathrm{d}}{\mathrm{d}t}\,|\psi(t)\rangle}{|\psi(t)\rangle} &= -\frac{i}{\hbar}\hat{H}_0 \\ \Leftrightarrow\ \int_{t_0}^{t}\mathrm{d}t\,\frac{\mathrm{d}}{\mathrm{d}t}(\ln|\psi(t)\rangle) &= \int_{t_0}^{t}\mathrm{d}t\left(-\frac{i}{\hbar}\right)\hat{H}_0 \\ \Leftrightarrow\ \ln\left(\frac{|\psi(t)\rangle}{|\psi(t_0)\rangle}\right) &= -\frac{i}{\hbar}(t-t_0)\hat{H}_0 \\ \Leftrightarrow\ |\psi(t)\rangle &= |\psi(t_0)\rangle\cdot e^{-\frac{i}{\hbar}\hat{H}_0(t-t_0)} \\ \Leftrightarrow\ |\psi(t)\rangle &\equiv \hat{U}_0(t,t_0)\,|\psi(t_0)\rangle\,, \end{aligned} \tag{21}$$

with

$$\hat{U}_0(t_2,t_1) = e^{-\frac{i}{\hbar}\hat{H}_0(t_2-t_1)} \tag{22}$$

and using the Taylor to define the exponential of an operator:

$$e^{\hat{A}} = \sum_{k=0}^{\infty}\frac{\hat{A}^k}{k!}. \tag{23}$$

Also, $\hat{U}_0$ has the following properties that follow directly from its definition (22):

$$\begin{cases} \hat{U}_0(t_1,t_1) = 1 \\ \hat{U}_0(t_3,t_1) = \hat{U}_0(t_3,t_2)\hat{U}_0(t_2,t_1) \\ \hat{U}_0(t_1,t_2) = \hat{U}_0^{-1}(t_2,t_1) \\ \qquad\qquad\ \ = \hat{U}_0^{\dagger}(t_2,t_1) \quad \text{(unitarity)} \end{cases} \tag{24}$$

Moreover, from (23) it follows that $\hat{U}_0(t,t_0)$ commutes with $\hat{H}_0$, since $\hat{H}_0$ commutes with itself:

$$\hat{U}_0(t,t_0)\hat{H}_0 = \hat{H}_0\hat{U}_0(t,t_0), \tag{25}$$

which we will use later. Lastly, for the time-derivative of $\hat{U}_0(t,t_0)$, we obtain:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\hat{U}_0(t,t_0) &\equiv \frac{\mathrm{d}}{\mathrm{d}t}\left(e^{-\frac{i}{\hbar}\hat{H}_0(t-t_0)}\right)\\ &= -\frac{i}{\hbar}\hat{H}_0 e^{-\frac{i}{\hbar}\hat{H}_0(t-t_0)}\\ &\equiv -\frac{i}{\hbar}\hat{H}_0\hat{U}_0(t,t_0)\\ &= -\frac{i}{\hbar}\hat{U}_0(t,t_0)\hat{H}_0\end{aligned} \tag{26}$$

Now, we use these results to perform a change of variable that eliminates the time-dependence of $|\psi(t)\rangle$ which is due to the system Hamiltonian $\hat{H}_0$:

$$|\psi_I(t)\rangle \equiv \hat{U}_0^{-1}(t,t_0)\,|\psi(t)\rangle \tag{27}$$

From this definition it follows that $|\psi_I(t)\rangle$ depends only on $\hat{H}'(t)$, i.e., if $\hat{H}'(t) = 0$, $|\psi_I(t)\rangle = |\psi(t_0)\rangle$. Also, note that

$$|\psi_I(t_0)\rangle = |\psi(t_0)\rangle\,. \tag{28}$$

Now we can show explicitly that the time-dependence of $|\psi_I(t)\rangle$ is only due to $\hat{H}'(t)$ by inserting the definition of $|\psi_I(t)\rangle$ (27) into the (SE):

$$\begin{aligned}-\frac{i}{\hbar}\hat{H}(t)\left(\hat{U}_0(t,t_0)\,|\psi_I(t)\rangle\right) &= \frac{\mathrm{d}}{\mathrm{d}t}\left(\hat{U}_0(t,t_0)\,|\psi_I(t)\rangle\right)\\ &= \left(\frac{\mathrm{d}}{\mathrm{d}t}\hat{U}_0(t,t_0)\right)|\psi_I(t)\rangle + \hat{U}_0(t,t_0)\left(\frac{\mathrm{d}}{\mathrm{d}t}\,|\psi_I(t)\rangle\right)\\ &= -\frac{i}{\hbar}\hat{H}_0\hat{U}_0(t,t_0)\,|\psi_I(t)\rangle + \hat{U}_0(t,t_0)\frac{\mathrm{d}}{\mathrm{d}t}\,|\psi_I(t)\rangle\\ \Leftrightarrow\ \hat{U}_0(t,t_0)\frac{\mathrm{d}}{\mathrm{d}t}\,|\psi_I(t)\rangle &= -\frac{i}{\hbar}(\hat{H}(t)-\hat{H}_0)\hat{U}_0(t,t_0)\,|\psi_I(t)\rangle\\ &\equiv -\frac{i}{\hbar}\hat{H}'(t)\hat{U}_0(t,t_0)\,|\psi_I(t)\rangle\\ \Leftrightarrow\ \frac{\mathrm{d}}{\mathrm{d}t}\,|\psi_I(t)\rangle &= -\frac{i}{\hbar}\hat{U}_0^\dagger(t,t_0)\hat{H}'(t)\hat{U}_0(t,t_0)\,|\psi_I(t)\rangle\\ \Leftrightarrow\ \frac{\mathrm{d}}{\mathrm{d}t}\,|\psi_I(t)\rangle &\equiv -\frac{i}{\hbar}\hat{H}_I'(t)\,|\psi_I(t)\rangle\,,\end{aligned} \tag{SE$_I$}$$

with

$$\hat{H}_I'(t) \equiv \hat{U}_0^\dagger(t,t_0)\hat{H}'(t)\hat{U}_0(t,t_0). \tag{29}$$

Furthermore, we want to apply the same change of variable to the Liouville-von Neumann equation (LvNE). Firstly, we transform the density operator $\hat{\rho}$ from the Schrödinger picture to the Interaction picture:

$$\begin{aligned}
\hat{\rho}_I(t) &\equiv \sum_n W_n \left|\psi_{I,n}(t)\right\rangle \left\langle\psi_{I,n}(t)\right| \\
&= \sum_n W_n \hat{U}_0^\dagger(t,t_0) \left|\psi_n(t)\right\rangle \left\langle\psi_n(t)\right| \hat{U}_0(t,t_0) \\
&= \hat{U}_0^\dagger(t,t_0) \left[\sum_n W_n \left|\psi_n(t)\right\rangle \left\langle\psi_n(t)\right|\right] \hat{U}_0(t,t_0) \\
&\equiv \hat{U}_0^\dagger(t,t_0)\hat{\rho}\hat{U}_0(t,t_0)
\end{aligned} \tag{30}$$

When inserting this into the (LvNE), we get for the left-hand side (LHS):

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\left(\hat{U}_0(t,t_0)\hat{\rho}_I(t)\hat{U}_0^\dagger(t,t_0)\right) &= \left(\frac{\mathrm{d}}{\mathrm{d}t}\hat{U}_0(t,t_0)\right)\hat{\rho}_I(t)\hat{U}_0^\dagger(t,t_0) + \hat{U}_0(t,t_0)\left(\frac{\mathrm{d}}{\mathrm{d}t}\hat{\rho}_I(t)\right)\hat{U}_0^\dagger(t,t_0) \\
&+ \hat{U}_0(t,t_0)\hat{\rho}_I(t)\left(\frac{\mathrm{d}}{\mathrm{d}t}\hat{U}_0^\dagger(t,t_0)\right) \\
&= -\frac{i}{\hbar}\hat{H}_0\hat{U}_0(t,t_0)\hat{\rho}_I(t)\hat{U}_0^\dagger(t,t_0) + \hat{U}_0(t,t_0)\left(\frac{\mathrm{d}}{\mathrm{d}t}\hat{\rho}_I(t)\right)\hat{U}_0^\dagger(t,t_0) \\
&+ \frac{i}{\hbar}\hat{U}_0(t,t_0)\hat{\rho}_I(t)\hat{U}_0^\dagger(t,t_0)\hat{H}_0 \\
&= \hat{U}_0(t,t_0)\left(\frac{\mathrm{d}}{\mathrm{d}t}\hat{\rho}_I(t)\right)\hat{U}_0^\dagger(t,t_0) \\
&- \frac{i}{\hbar}\left[\hat{H}_0, \hat{U}_0(t,t_0)\hat{\rho}_I(t)\hat{U}_0^\dagger(t,t_0)\right],
\end{aligned}$$

and for the right-hand side (RHS):

$$-\frac{i}{\hbar}\left[\hat{H}(t), \hat{U}_0(t,t_0)\hat{\rho}_I(t)\hat{U}_0^\dagger(t,t_0)\right], \tag{31}$$

and thus we can write:

$$\begin{aligned}
\hat{U}_0(t,t_0)\left(\frac{\mathrm{d}}{\mathrm{d}t}\hat{\rho}_I(t)\right)\hat{U}_0^\dagger(t,t_0) &= -\frac{i}{\hbar}\left(\left[\hat{H}(t), \hat{U}_0(t,t_0)\hat{\rho}_I(t)\hat{U}_0^\dagger(t,t_0)\right]\right. \\
&\quad \left. - \left[\hat{H}_0, \hat{U}_0(t,t_0)\hat{\rho}_I(t)\hat{U}_0^\dagger(t,t_0)\right]\right) \\
&= -\frac{i}{\hbar}\left[\hat{H}(t) - \hat{H}_0, \hat{U}_0(t,t_0)\hat{\rho}_I(t)\hat{U}_0^\dagger(t,t_0)\right] \\
&= -\frac{i}{\hbar}\left[\hat{H}'(t), \hat{U}_0(t,t_0)\hat{\rho}_I(t)\hat{U}_0^\dagger(t,t_0)\right] \\
\Leftrightarrow \quad \left(\frac{\mathrm{d}}{\mathrm{d}t}\hat{\rho}_I(t)\right) &= -\frac{i}{\hbar}\left[\hat{U}_0^\dagger(t,t_0)\hat{H}'(t)\hat{U}_0(t,t_0), \hat{\rho}_I(t)\right] \\
&= -\frac{i}{\hbar}\left[\hat{H}'_I(t), \hat{\rho}_I(t)\right]
\end{aligned} \tag{LvNE$_I$}$$

As Hamm 2005 (p.15f.) points out, the interaction picture can be interpreted as a representation of quantum mechanics that lies in between the Schrödinger picture and the Heisenberg picture. Whereas in the Schrödinger picture, all the time dependence lies in the states, and in the Heisenberg picture, all the time dependence lies in the operators, in the interaction picture, the time-dependence due to the system Hamiltonian $\hat{H}_0$ is moved into the operators, so that the states' time dependence is only due to the perturbative Hamiltonian $[\hat{H}'(t)]$. In other words, "[t]he interaction picture adopts the Schrödinger picture for the small perturbation $\hat{H}'(t)$, while it uses the Heisenberg picture for the larger system Hamiltonian $[\hat{H}_0]$." (Hamm 2005, p. 15)

Equations (SE$_I$) and (LvNE$_I$) are useful, since in the following, we want to perform a perturbative expansion of the equations of motion of $|\psi(t)\rangle$ and $\hat{\rho}$. One can perform the same expansion with the (SE) and the (LvNE) in their native, Schrödinger-picture form (see Hamm 2005, p. 12f.), but, as Hamm points out, convergence it not guaranteed since the expansion would be in powers of the total Hamiltonian. It is therefore preferable to expand in powers of the much smaller $\hat{H}'(t)$ (Hamm 2005, p. 17). Hence, we will be expanding equations (SE$_I$) and the (LvNE$_I$) in their interaction picture form instead. We start with the (SE$_I$) where we integrate both sides:

$$\begin{aligned}
\int_{t_0}^{t} \mathrm{d}\tau \, \frac{\mathrm{d}}{\mathrm{d}\tau} |\psi_I(\tau)\rangle &= |\psi_I(t_0)\rangle - \frac{i}{\hbar} \int_{t_0}^{t} \mathrm{d}\tau \, \hat{H}_I'(\tau) \, |\psi_I(\tau)\rangle \\
\Leftrightarrow \; |\psi_I(t)\rangle &= |\psi_I(t_0)\rangle - \frac{i}{\hbar} \int_{t_0}^{t} \mathrm{d}\tau \, \hat{H}_I'(\tau) \, |\psi_I(\tau)\rangle
\end{aligned} \tag{32}$$

We assume $\hat{H}_I'(\tau)$ to be known $\forall \tau$, but we do not know anything more about $|\psi_I(\tau)\rangle$ then we know about $|\psi_I(t)\rangle$. Instead, we reinsert equation (32) into itself to obtain $|\psi_I(\tau)\rangle$, which yields the following expression for $|\psi_I(t)\rangle$:

$$\begin{aligned}
|\psi_I(t)\rangle &= |\psi_I(t_0)\rangle - \frac{i}{\hbar} \int_{t_0}^{t} \mathrm{d}\tau_1 \, \hat{H}_I'(\tau_1) \, |\psi_I(\tau_1)\rangle \\
&= |\psi_I(t_0)\rangle - \frac{i}{\hbar} \int_{t_0}^{t} \mathrm{d}\tau_1 \, \hat{H}_I'(\tau_1) \left[|\psi_I(t_0)\rangle - \frac{i}{\hbar} \int_{t_0}^{\tau_1} \mathrm{d}\tau_2 \, \hat{H}_I'(\tau_2) \, |\psi_I(\tau_2)\rangle \right] \\
&= |\psi_I(t_0)\rangle - \frac{i}{\hbar} \int_{t_0}^{t} \mathrm{d}\tau_1 \, \hat{H}_I'(\tau_1) \, |\psi_I(t_0)\rangle \\
&+ \left(-\frac{i}{\hbar}\right)^2 \int_{t_0}^{t} \mathrm{d}\tau_1 \int_{t_0}^{\tau_1} \mathrm{d}\tau_2 \, \hat{H}_I'(\tau_1) \hat{H}_I'(\tau_2) \, |\psi_I(\tau_2)\rangle
\end{aligned} \tag{33}$$

This can be iterated to obtain the following expansion:

$$\begin{aligned}|\psi_I(t)\rangle &= |\psi_I(t_0)\rangle \\ &\quad + \sum_{n=1}^{N}\left(-\frac{i}{\hbar}\right)^n \int_{t_0}^{t} \mathrm{d}\tau_1 \int_{t_0}^{\tau_1} \mathrm{d}\tau_2 \;\cdots \int_{t_0}^{\tau_{N-1}} \mathrm{d}\tau_N \, \hat{H}_I'(\tau_1)\hat{H}_I'(\tau_2)\dots\hat{H}_I'(\tau_N)\,|\psi_I(\tau_N)\rangle\end{aligned} \tag{34}$$

For $N \to \infty$ and $\tau_N \to t_0$ we can therefore write that $|\psi_I(t)\rangle$ is equal to:

$$\left[1 + \sum_{n=1}^{N\to\infty}\left(-\frac{i}{\hbar}\right)^n \int_{t_0}^{t} \mathrm{d}\tau_1 \int_{t_0}^{\tau_1} \mathrm{d}\tau_2 \;\cdots \int_{t_0}^{\tau_{N-1}} \mathrm{d}\tau_N \, \hat{H}_I'(\tau_1)\hat{H}_I'(\tau_2)\dots\hat{H}_I'(\tau_N)\right]|\psi_I(t_0)\rangle \tag{35}$$

If we transform back to the Schrödinger picture, using the relations (27), (28) and (29), we obtain that:

$$\begin{aligned}|\psi(t)\rangle = \Bigg[&\hat{U}_0(t,t_0) + \sum_{n=1}^{N\to\infty}\left(-\frac{i}{\hbar}\right)^n \int_{t_0}^{t} \mathrm{d}\tau_1 \int_{t_0}^{\tau_1} \mathrm{d}\tau_2 \;\cdots \int_{t_0}^{\tau_{N-1}} \mathrm{d}\tau_N \\ &\hat{U}_0(t,\tau_1)\hat{H}'(\tau_1)\hat{U}_0(\tau_1,\tau_2)\hat{H}'(\tau_2)\hat{U}_0(\tau_2,\tau_3)\dots\hat{U}_0(\tau_{N-1},\tau_N)\hat{H}'(\tau_N)\hat{U}_0(\tau_N,t_0)\Bigg]|\psi(t_0)\rangle\end{aligned} \tag{36}$$

Here we have used the properties of $\hat{U}_0$ from equation (22) to replace $\hat{U}_0(\tau_n,t_0)\hat{U}_0^\dagger(\tau_{n+1},t_0)$ by $\hat{U}_0(\tau_n,\tau_{n+1})$:

$$\begin{aligned}\hat{U}_0(\tau_n,t_0)\hat{U}_0^\dagger(\tau_{n+1},t_0) &= \hat{U}_0(\tau_n,t_0)\hat{U}_0(t_0,\tau_{n+1}) \\ &= \hat{U}_0(\tau_n,\tau_{n+1})\end{aligned}$$

If we have an experimental situation where the perturbation Hamiltonian $\hat{H}'(t)$, of for example an external electric field, can be reasonably assumed to be the finite sum of N delta-functions or narrow Gaussians (with amplitudes A_n):

$$\begin{aligned}\hat{H}'(t) &= \sum_{n=1}^{N} A_n \cdot \cos(t-\tau_n)\cdot \exp\left(-\frac{(t-\tau_n)^2}{2\sigma^2}\right) \\ &\overset{\sigma\to 0}{\approx} \sum_{n=1}^{N} A_n \cdot \cos(t-\tau_n)\cdot \sigma\sqrt{2\pi}\delta(t-\tau_n),\end{aligned} \tag{37}$$

we can describe the time evolution of the system with a finite number of terms in our expansion (36). We can then physically interpret the expansion in the following way: The system propagates freely, that is only under the influence of the system Hamiltonian, until time $t = \tau_1$, the time of the first pulse of $\hat{H}'$, i.e., the first point in time where $\hat{H}'(t)$ is non-zero. At time $t = \tau_1$, the system is assumed to interact instantaneously with the perturbation $\hat{H}'(\tau_1)$, potentially changing the populations of some levels, and thereafter, the system continues to evolve unperturbed - freely - under the influence of $\hat{H}_0$ only, until time $t = \tau_2$, and so on. (Hamm 2005, pp. 12f., 16-19; Hamm and Zanni 2011, pp. 54-57).

This can be a useful alternative to numerical integration, especially when using ultrafast spectroscopic techniques, since we use sequences of ultrafast light pulses, and thus the electric field applied to the system can be written as in equation (37) (Gupta 2022, p. 17). Moreover, this expansion can be visualized in a useful manner using so-called *single-sided Feynman diagrams*.

The perturbative expansion of $\hat{\rho}$ is done along the same lines as the expansion of $|\psi(a)\rangle$ nd yields the following expression (Hamm 2005, p. 18):

$$\begin{aligned}\rho(t) &= \hat{U}_0(t,t_0)\rho(t_0)\hat{U}_0^\dagger(t,t_0)\\ &+ \sum_{n=1}^{N\to\infty}\left(-\frac{i}{\hbar}\right)^n \int_{t_0}^{t} \mathrm{d}\tau_1 \int_{t_0}^{\tau_1} \mathrm{d}\tau_2 \cdots \int_{t_0}^{\tau_{N-1}} \mathrm{d}\tau_N \\ &\hat{U}_0(t,t_0)\,[\hat{H}_I'(\tau_1),[\hat{H}_I'(\tau_2),\ldots,[\hat{H}_I'(\tau_n,\hat{\rho}(t_0))]\ldots]]\,\hat{U}_0^\dagger(t,t_0)\end{aligned} \tag{38}$$

This expansion can be illustrated with so-called *double sided Feynman diagrams*.

7.1.3 Derivation of the signal's periodicity

To explain the periodicity of the signal w.r.t. τ, we are going to make some simplifications and then solve the (LvNE). The Hamiltonian $\hat{H}(t)$ is given by:

$$\hat{H}(t) = \begin{pmatrix} 0 & F(t) \\ F^*(t) & \Delta \end{pmatrix},$$

where $F(t)$ is the time-dependent perturbation due to the external electric field. We start by writing out the commutator explicitly:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\hat{\rho} &= -\frac{i}{\hbar}[\hat{H}(t), \hat{\rho}] \\
\Leftrightarrow\ \frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} \rho_{11}(t) & \rho_{12}(t) \\ \rho_{21}(t) & \rho_{22}(t) \end{pmatrix} &= -\frac{i}{\hbar}\begin{pmatrix} 0 & F(t) \\ F^*(t) & \Delta \end{pmatrix} \cdot \begin{pmatrix} \rho_{11}(t) & \rho_{12}(t) \\ \rho_{21}(t) & \rho_{22}(t) \end{pmatrix} \\
&\quad + \frac{i}{\hbar}\begin{pmatrix} \rho_{11}(t) & \rho_{12}(t) \\ \rho_{21}(t) & \rho_{22}(t) \end{pmatrix} \cdot \begin{pmatrix} 0 & F(t) \\ F^*(t) & \Delta \end{pmatrix} \\
\Leftrightarrow\ \begin{pmatrix} \dot{\rho}_{11}(t) \\ \dot{\rho}_{12}(t) \\ \dot{\rho}_{21}(t) \\ \dot{\rho}_{22}(t) \end{pmatrix} &= -\frac{i}{\hbar}\begin{pmatrix} F(t)\rho_{21}(t) & -\rho_{12}(t)F^*(t) \\ F(t)\rho_{22}(t) & -\rho_{11}(t)F(t) - \rho_{12}(t)\Delta \\ F^*(t)\rho_{11}(t) + \Delta\rho_{21}(t) & -\rho_{22}(t)F^*(t) \\ F^*(t)\rho_{12}(t) + \Delta\rho_{22}(t) & -\rho_{21}(t)F(t) - \rho_{22}(t)\Delta \end{pmatrix}
\end{aligned}$$

$$\Leftrightarrow \begin{cases} \hbar\dot{\rho}_{11}(t) = +2\,\mathrm{Im}\{F(t)\rho_{21}(t)\} \\ \hbar\dot{\rho}_{12}(t) = +iF(t)(\rho_{11}(t) - \rho_{22}(t)) + i\Delta\rho_{12}(t) \\ \hbar\dot{\rho}_{21}(t) = -iF^*(t)(\rho_{11}(t) - \rho_{22}(t)) - i\Delta\rho_{21}(t) \\ \hbar\dot{\rho}_{22}(t) = -2\,\mathrm{Im}\{F(t)\rho_{21}(t)\} \end{cases}$$

We only care about the last two lines, since the signal is proportional to $\rho_{22}(t)$ and $\dot{\rho}_{22}(t)$ only depends on $\rho_{21}(t)$. We proceed by solving the differential equation for $\rho_{21}(t)$, which is a first-order ODE linear w.r.t. y. Equations of this form, that is

$$\dot{f}(t) = -p \cdot f(t) + q(t)$$

have the following general solution (see Adams 2014, p. 450):

$$f(t) = e^{-p\cdot t}\left[f(t_0) + \int_{t_0}^{t} \mathrm{d}t'\, e^{+p\cdot t'} \cdot q(t')\right].$$

We can bring the differential equation for $\rho_{21}(t)$ into this form:

$$\underbrace{\dot{\rho}_{21}(t)}_{\dot{f}(t)} = -\underbrace{i\frac{\Delta}{\hbar}}_{p}\underbrace{\rho_{21}(t)}_{f(t)} + \underbrace{\left(-\frac{i}{\hbar}F^*(t)(\rho_{11}(t) - \rho_{22}(t))\right)}_{q(t)},$$

and it thus has the solution:

$$\rho_{21}(t) = e^{-i\frac{\Delta}{\hbar}\cdot t}\left[\rho_{21}(t_0) - \frac{i}{\hbar}\int_{t_0}^{t} \mathrm{d}t'\, e^{+i\frac{\Delta}{\hbar}\cdot t'} \cdot F^*(t')(\rho_{11}(t') - \rho_{22}(t'))\right].$$

Next, we integrate both sides in the differential equation for $\rho_{22}(t)$, which gives:

$$\rho_{22}(t) = \rho_{22}(t_0) - \frac{2}{\hbar}\int_{t_0}^{t} \mathrm{d}t' \, \mathrm{Im}\{F(t')\rho_{21}(t')\}$$

To simplify the calculations, we will make some assumptions, in particular about the initial conditions: We assume that all electrons are in the ground state when hit by the laser pulse, thus $\rho_{11}(t_0) = 1$ and $\rho_{22}(t_0) = \rho_{21}(t_0) = 0$, and that the excited-state population $\rho_{22}(t)$ remains small compared to $\rho_{11}(t)$ throughout the course of the experiment, thus $\rho_{11}(t) - \rho_{22}(t) \approx 1$. The expressions for $\rho_{21}(t))$ and $\rho_{22}(t)$ then simplify to:

$$\begin{cases} \rho_{21}(t) = -\dfrac{i}{\hbar}\displaystyle\int_{t_0}^{t} \mathrm{d}t' \, e^{+i\frac{\Delta}{\hbar}(t'-t)} \cdot F^*(t') \\ \rho_{22}(t) = -\dfrac{2}{\hbar}\displaystyle\int_{t_0}^{t} \mathrm{d}t' \, \mathrm{Im}\{F(t')\rho_{21}(t')\} \end{cases}$$

Now we are ready to insert the first equation into the second one, giving:

$$\rho_{22}(t) = \frac{2}{\hbar^2}\int_{t_0}^{t} \mathrm{d}t' \, \mathrm{Im}\left\{ iF(t') \int_{t_0}^{t'} \mathrm{d}t'' \, e^{+i\frac{\Delta}{\hbar}\cdot(t''-t')} \cdot F^*(t'') \right\}$$

Now, we need to insert $F(t)$. Combining equations (15) and (17) yields:

$$F(t) = \sum_{k=1}^{2} \vec{E}_{max}\vec{\mu}_{eg} \cdot \exp\left(-\frac{(t-t_k)^2}{2\sigma^2}\right) \cdot \cos(\omega(t-t_k)+\varphi_k).$$

that is, the Gaussian pulses simulated with *QDT* and the code from Hedse et al. However, in the following derivation, we are going to approximate them by Dirac delta functions, since we are only interested in a more qualitative understanding of the signal's oscillations:

$$F(t) = \sum_{k=1}^{2} \vec{E}_{max}\vec{\mu}_{eg} \cdot \sigma\sqrt{2\pi}\delta(t-t_k) \cdot \cos(\omega(t-t_k)+\varphi_k).$$

We thus obtain for $\rho_{22}(t)$:

$$\rho_{22}(t) = \overbrace{\frac{4\pi\sigma^2}{\hbar^2}\left|\vec{E}_{max}\vec{\mu}_{eg}\right|^2}^{\text{Amp}} \int_{t_0}^{t} \mathrm{d}t' \, \mathrm{Im}\left\{ i\sum_{k=1}^{2} \overbrace{\cos(\omega(t'-t_k)+\varphi_k) \cdot e^{-i\frac{\Delta}{\hbar}\cdot t'}}^{g_k(t')} \delta(t'-t_k) \right.$$
$$\left. \int_{t_0}^{t'} \mathrm{d}t'' \sum_{k=1}^{2} \underbrace{\cos(\omega(t''-t_k)+\varphi_k) \cdot e^{+i\frac{\Delta}{\hbar}\cdot t''}}_{g_k^*(t'')} \delta(t''-t_k) \right\}$$

Above equation is simplified by introducing the substitutions

$$\begin{cases} \text{Amp} = \dfrac{4\pi\sigma^2}{\hbar^2} \left|\vec{E}_{max}\vec{\mu}_{eg}\right|^2 \\ g_k(t) = \cos(\omega(t-t_k)+\varphi_k) \cdot e^{-i\frac{\Delta}{\hbar}\cdot t} \end{cases}$$

We now evaluate the integral to get the time evolution of ρ_{22}[9]:

$$\begin{aligned} \rho_{22}(t) = \text{Amp} \cdot \text{Im} \Bigg\{ & i \int_{t_0}^{t} \mathrm{d}t' \left[g_1(t')\delta(t'-t_1) + g_2(t')\delta(t'-t_2)\right] \\ & \int_{t_0}^{t'} \mathrm{d}t'' \left[g_1^*(t'')\delta(t''-t_1) + g_2^*(t'')\delta(t''-t_2)\right] \Bigg\} \\ = \text{Amp} \cdot \text{Im} \Bigg\{ & i \int_{t_0}^{(t_1+t_2)/2} \mathrm{d}t' \left[g_1(t')\delta(t'-t_1) + g_2(t')\delta(t'-t_2)\right] \\ & \int_{t_0}^{t'} \mathrm{d}t'' \left[g_1^*(t'')\delta(t''-t_1) + g_2^*(t'')\delta(t''-t_2)\right] \\ & + i \int_{(t_1+t_2)/2}^{t} \mathrm{d}t' \left[g_1(t')\delta(t'-t_1) + g_2(t')\delta(t'-t_2)\right] \\ & \int_{t_0}^{t'} \mathrm{d}t'' \left[g_1^*(t'')\delta(t''-t_1) + g_2^*(t'')\delta(t''-t_2)\right] \Bigg\} \\ = \text{Amp} \cdot \text{Im} \Bigg\{ & i\, g_1(t_1)\frac{g_1^*(t_1)}{2} + i g_2(t_2)\left[g_1^*(t_1) + \frac{g_2^*(t_2)}{2}\right] \Bigg\} \\ = \frac{\text{Amp}}{2} & \cdot \text{Im}\left\{ i \left[|g_1(t_1)|^2 + |g_2(t_2)|^2 + 2g_1^*(t_1)g_2(t_2)\right]\right\} \\ = \frac{\text{Amp}}{2} & \cdot \left[|g_1(t_1)|^2 + |g_2(t_2)|^2 + 2\,\text{Im}\{i g_1^*(t_1)g_2(t_2)\}\right] \end{aligned}$$

Since

$$g_k(t_k) = \cos(\varphi_k) \cdot e^{-\frac{\Delta}{\hbar}\cdot t_k},$$

[9]Here, we follow the convention that $\int_a^b \mathrm{d}t'\, \delta(t'-a)g(t') = g(a)/2$. Note that this integral could alternatively be solved using Heaviside step functions.

when reinserting the expressions for $g_k(t)$, we get that:

$$\begin{aligned}
\rho_{22}(t) &= \frac{\text{Amp}}{2} \cdot \left[\cos^2(\varphi_1) + \cos^2(\varphi_2) + 2\,\text{Im}\left\{i\cos(\varphi_1)\cos(\varphi_2) \cdot e^{-i\frac{\Delta}{\hbar}\cdot(t_2-t_1)}\right\}\right] \\
&= \frac{\text{Amp}}{2} \cdot \left[\cos^2(\varphi_1) + \cos^2(\varphi_2)\right. \\
&\left. +2\,\text{Im}\left\{\cos(\varphi_1)\cos(\varphi_2) \cdot \left(i\cos\left(\frac{\Delta}{\hbar}\tau\right) + i(-i)\sin\left(\frac{\Delta}{\hbar}\tau\right)\right)\right\}\right] \\
&= \frac{\text{Amp}}{2} \cdot \left[\cos^2(\varphi_1) + \cos^2(\varphi_2) + 2\cos(\varphi_1)\cos(\varphi_2)\cos\left(\frac{\Delta}{\hbar}\tau\right)\right]
\end{aligned}$$

Thus, the angular frequency ω_s of the signal w.r.t. τ is equal to $\frac{\Delta}{\hbar}$.

7.1.4 Numerical integration methods

As previously mentioned, both codes employ explicit propagation of the modelled systems' density matrices. In other words, they perform numerical integration of the Lindblad master equation, which is an ordinary differential equation (ODE). ODEs are equations of the form:

$$\dot{f}(t) = G(f(t), t)$$

Thus a method for numerical integration is required, the choice of which might significantly influence run time and accuracy. In the following, a very short introduction to the numerical integration methods utilized by the two codes. For comparison, Euler's method is also quickly mentioned. Euler's method is the simplest method fo solving ODEs. It also has the lowest accuracy for a given step size. It approximates the solution linearly with the first order term of the Taylor expansion of $f(t)$:

$$\begin{aligned}
f(t+\Delta t) &\approx f(t) + \Delta t \cdot \dot{f}(t) \\
&= f(t) + \Delta t \cdot G(f(t), t),
\end{aligned}$$

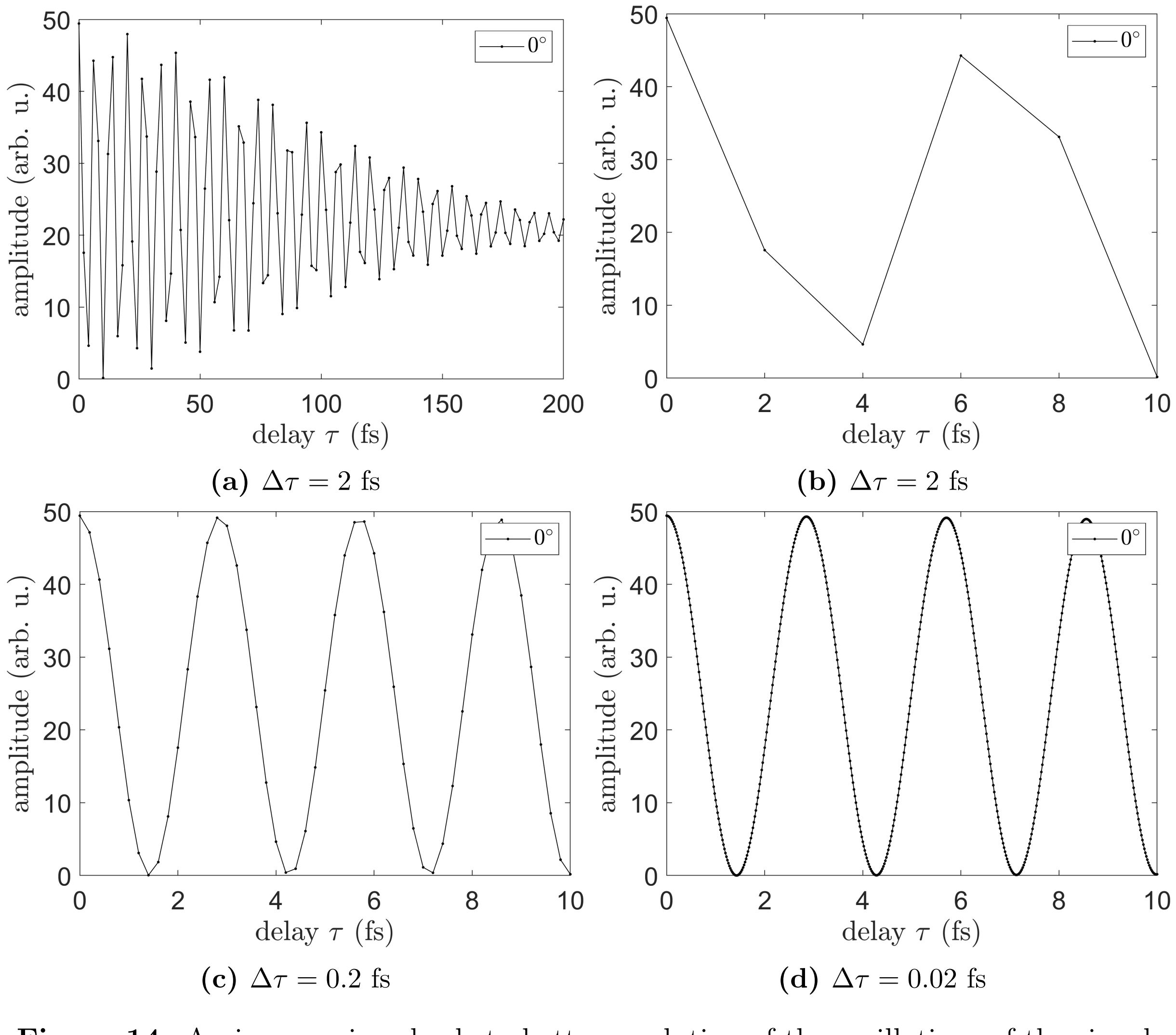
where Δt is the step size of the numerical integration. (Kong et al. 2021, p. 375 f.)

Runge-Kutta methods use more terms in the Taylor expansion of $f(t)$, and approximates the necessary partial derivatives of $G(f(t), t)$ by evaluating $G(f(t), t)$ at points in the interval $[t, t+\Delta t]$, that is $G(f(t)+\beta, t+\alpha)$, by reverse application of multivariable Taylor expansions. For many problems, this method offers more accuracy for he same amount of computational effort. (Kong et al. 2021, p. 383f.; Press 2007, p. 908f.) *QDT* uses fourth-order Runge-Kutta. In contrast, `qutip.mesolve()` uses *VODE* by default[10][11].

[10] *Qutip.Mesolve — QuTiP 4.0 Documentation* 2011.
[11] *Scipy.Integrate.Ode — SciPy v1.13.0 Manual* 2024.

” *VODE* is a[n] initial value ODE solver for stiff and nonstiff systems. It uses variable coefficient Adams-Moulton and Backward Differentiation Formula (BDF) methods [...].” (Brown et al. 1989). Both of these methods are linear multistep methods, which means that taking into account results from more than one previous point. (Kong et al. 2021, p. 389). Here, a stiff ODE is an ODE whose solution varies slowly, but will diverge easily and that therefore requires a comparatively small step size to ensure convergence of the solution. (Curtiss and Hirschfelder 1952; Kong et al. 2021, p. 389) *VODE* uses BDF since it is more robust at solving stiff ODEs. (Curtiss and Hirschfelder 1952, Gear 1967).

7.2 Additional figures



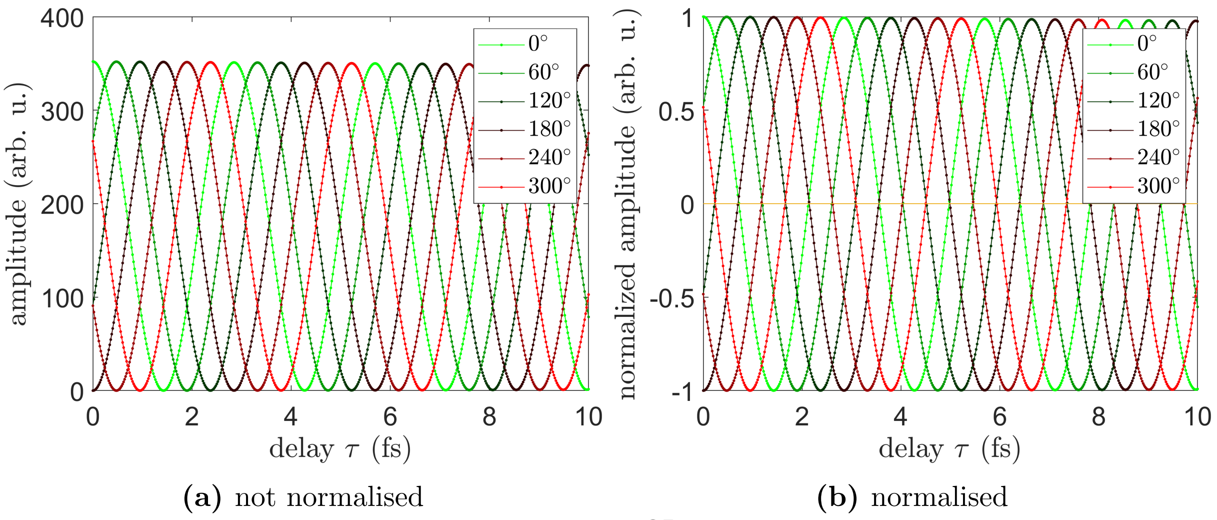
(a) $\Delta\tau = 2$ fs

(b) $\Delta\tau = 2$ fs

(c) $\Delta\tau = 0.2$ fs

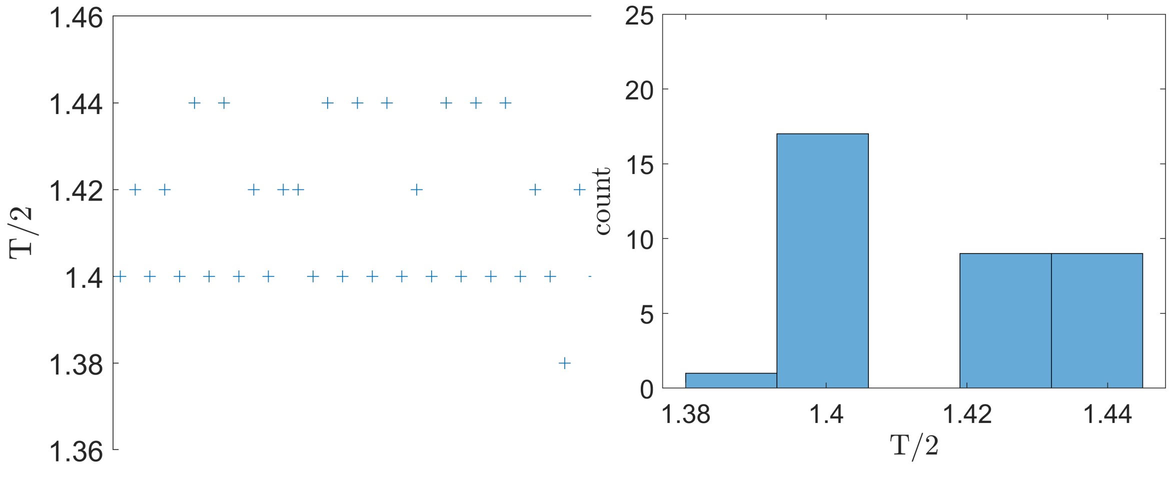
(d) $\Delta\tau = 0.02$ fs

Figure 14: An increase in τ leads to better resolution of the oscillations of the signal.



(a) not normalised

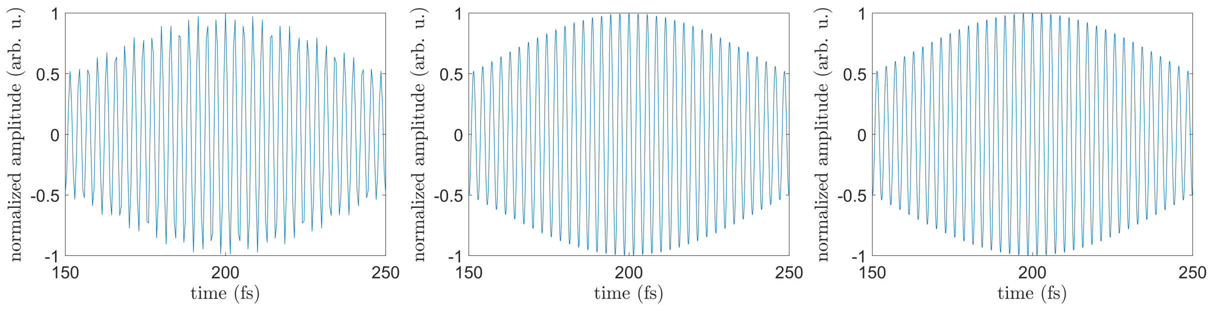
(b) normalised



(c) dot plot of values for T/2

(d) histogram of values for T/2

Figure 15: Process for calculating the periodicity of the signal: normalize the signal, and then measure the distance between two points were the signal is (almost) equal to zero. This distance corresponds to $T/2$. The legends in sub-figures a and b display the value of $\Delta\varphi$. Deviating parameters of the simulation: $\Delta\tau = 0.02$ fs, `runTstep` $= 0.1$ fs, `offset` $= 200$ fs, 6 phases



(a) `runTstep`=0.5 fs **(b)** `runTstep`=0.1 fs **(c)** `runTstep`=0.01 fs

Figure 16: Electric field of the second pulse for different values of `runTstep`

7.3 Code

7.3.1 *MATLAB* script for running the simulations

```matlab
clear; clc;
s = System;
c = CMDS(s);
Nilsfstoau=100/2.4188843265857;

parl     = true;
runTmax =  800; % fs
runTstep=  800/(2^13+1);%fs
Offset   =  200; % fs (from Andreas)
Tmax     =  200; % fs

Tstep    =   40+1;
pcmax1   =    1;
pcmax2   =    6;
gamma0   =    1.0; % 0.0 = rot. frame; 1.0 = lab frame
w0       =    1.45*s.evtoau;% laser frequency
amp      = 2*10^-3*s.evtoau;% E-field amplitude
t_pulse =  100/sqrt(2)*Nilsfstoau; % REAL pulse FWHM!!!

gaps = [1.46, 2.9-1.46]*s.evtoau;
a= Nlevel(gaps(1));
s.addEntity(a, 'two-level-sys');
s.setTmax(runTmax*Nilsfstoau);
s.setTimestep(runTstep*Nilsfstoau);
s.lso = true; %pre-calculate the lindblad superoperator
s.addDissipation('two-level-sys', 1.0*10^6*Nilsfstoau, [1,0]);
s.addDecoherence('two-level-sys', 1.0*10^2*Nilsfstoau, [1,0]);
s.addDecoherence('two-level-sys', 1.0*10^2*Nilsfstoau, [0,1]);

c.setEfieldparameter(amp,t_pulse,w0,gamma0);
c.setdelayMaxs([Offset, Tmax]*Nilsfstoau);
c.setdelaySteps([Tstep]);
c.setReturnFunction(returnExcited)
c.setContributions([1, -1]);
c.setPcScheme([pcmax1, pcmax2]);
c.parallel = parl; % enable parallel computing
c.generate_CMDS('two-level-sys');
test_res=c.CMDSdata;

timestamp=datetime();
timestamp.Format='yyMMdd''_''HHmmss';
timestr=string(timestamp);

namestr='mlres';
```

```
disp(namestr+timestr);

tau_axis=c.delayAxes{1}.*s.autofs;
save(strcat('res',     namestr,timestr,'.dat'),'test_res','-ascii')
save(strcat('Tmax_fs',namestr,timestr,'.dat'),'Tmax',     '-ascii')
save(strcat('runTmax_fs', namestr,timestr,'.dat'),'runTmax', '-ascii')
save(strcat('runTstep_fs',namestr,timestr,'.dat'),'runTstep','-ascii')
save(strcat('tau_axis_fs',namestr,timestr,'.dat'),'tau_axis','-ascii')
```

7.3.2 *MATLAB* script for DFT and plotting

```
%% load & pre-process data
clear;clc;
load workspaceMIN.mat;

%%%%%% Load Files %%%%%%
% namestrs=["mlres240513_193229"]; % intermediate curve overlay
% namestrs=["mlres240513_184751"]; % final curve overlay

%%%%%% Plot Parameters %%%%%%
axis_font_size=17;
legend_font_size=15;
filetype='.png';
fftstr='_fft';
xaxisname ='delay $\tau$ (fs)';
yaxisname1='amplitude (arb. u.)';
yaxisname2='normalized amplitude (arb. u.)';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for ii=1:length(namestrs)
    namestr=namestrs(ii);
    test_res=load(strcat('res',namestr,'.dat'));
    transp_test_res=transpose(test_res);
    Tmax=load(strcat('Tmax_fs',namestr,'.dat'));
    xaxis=load(strcat('tau_axis_fs',namestr,'.dat'));
    % xaxis= c.delayAxes{1}*s.autofs;%linspace(0,Tmax,dim(2));

    dim=size(test_res);
    disp(dim);
    pcmax = dim(1);

    %%% plot single phase-shift contribution(s)

    from=1;
    to  =pcmax;

    phases  = linspace(0,360,pcmax+1);
```

```matlab
    phases  = phases(from:to);

    tplot=plot(xaxis, test_res(from:to,:), '.-');
    xlabel(xaxisname, Interpreter='latex');
    ylabel(yaxisname1,Interpreter='latex');
    fig= gcf;
    ax = gca;
    ax.XAxis.FontSize = axis_font_size;
    ax.YAxis.FontSize = axis_font_size;

    %%% lagend and colours
    legmax = max(phases);
    legstr1 = [];
    ii=1;
    for jjj = phases
        legstr1=[legstr1;strcat('$',string(jjj),'^\circ$')];
        tplot(ii).Color=[max(2*jjj/legmax-1,0),max(1-2*jjj/legmax,0),0];
        ii=ii+1;
    end
    l=legend(legstr1);
    set(l,'FontSize',legend_font_size, 'Interpreter', 'latex');

    exportgraphics(fig,strcat(namestr,filetype),'Resolution',400);

    %%% plot phase-weighted avg
    freqmult=1:2;
    empti=zeros(size(transp_test_res,1),1);
    summ=[];
    legstr2=[];
    ii=1;
    jj=1;

    for fm = freqmult
        summ=[summ,empti];
        ii=1;
        [~, tweights]=c.calculatePcWeights(fm*[1, -1], [1, dim(1)]);

        for iii = transp_test_res
            summ(:,jj)=summ(:,jj)+iii*tweights(ii);
            ii=ii+1;
        end
        summ(:,jj)=summ(:,jj)/max(summ(:,jj));
        legstr2=[legstr2;string(fm)+"$\cdot(\phi_1-\phi_2)$ contribution
  "];
        jj=jj+1;
    end

    tplot2=plot(xaxis, abs(summ));
    xlabel(xaxisname, Interpreter='latex');
    ylabel(yaxisname2,Interpreter='latex');
    fig= gcf;
    ax = gca;
    ax.XAxis.FontSize = axis_font_size;
    ax.YAxis.FontSize = axis_font_size;

    l=legend(legstr2);
    set(l,'FontSize',legend_font_size, 'Interpreter', 'latex');

    exportgraphics(fig,strcat(namestr,fftstr,filetype),'Resolution',400)
   ;

    %%% save data to csv
    timestr="tau [fs]";
    export1=[[timestr,phases];[transpose(xaxis),transp_test_res]];
    export2=[[timestr,transpose(legstr2)];[transpose(xaxis),abs(summ)]];
    writematrix(export1,'matlab_res_'+namestr+'.csv','Delimiter','tab');
    writematrix(export2,'matlab_res_'+namestr+'_fft.csv','Delimiter','
   tab');
    disp('matlab_res_'+namestr+'_fft.csv');
end
```

7.3.3 *python* script for running the simulations

```
import numpy as np
import qutip as q

q.settings.num_cpus = 1

HBARINV = 1519.267  # 1/hbar, ps^{-1} eV^{-1}
HBAR = 1 / HBARINV  # hbar, ps eV

# matrix of zeros except for a one at (i,j)
def matrix_ij(n, i, j):
    m = np.zeros((n, n))
    m[i, j] = 1
    return m

# string for time dependent interaction hamiltonian in cython
   implementation
def get_string(displace, tau, omega, phi, delay):
    return f"exp(-4*log(2)*((t-{displace})/{tau})**2)*cos({omega}*(t-{
   displace})+2*{np.pi}*{phi}*m*{delay})"

def get_stringRotWave(displace, tau, omega, phi, delay,imag):
    return f"exp(-4*log(2)*((t-{displace})/{tau})**2)*exp({imag}*({omega
    }*(t-{displace})+2*{np.pi}*{phi}*m*{delay}))/2"

# integrate relevant part of lindblad evolution and print to CLI
def do_output(m, window, output, n_levels):
    qobj = sum(output.states[i] for i in window) / len(window)
    z = np.diag(qobj.__array__().real)
    # print(f"{m}\t" + "\t".join(str(zi) for zi in z))
    return z[-1]

def double_quantum_coherence_simulation(
    cfmu=0.002,  # charge*field*dipole moment, eV
    phi1=5.14e-5,  # Linear frequency of pulse 1, THz
    phi2=5.19e-5,  # Linear frequency of pulse 2, THz
    pulse_width=0.1,  # pulse width, ps
    centre_of_first_pulse=0.2,  # Centre of first pulse, ps
    delay_to_second_pulse=0,  # delay centre of second pulse from first,
    ps
    omega_ev=1.45,  # angular freq. of light, eV
    dephasing_times=[
    ((0, 1), 0.1),
    ((1, 0), 0.1),
    ],  # (g0,g1,..),td: Dephasing time td with weights g0,g1,g_{n_level
    -1}, ps
    relaxation_times=[(0, 1, 1000)],  # i,j,tr: Relaxation time tr from
    j to i, ps
    start_time=0,  # Initial time, ps
    end_time=0.8,  # Final time, ps
    integral_window=0.05,  # integral window for data collection, ps
    sum_all=False, # Result is sum from start to end, otherwise
    integral_window after last pulse
    n_steps=65536,  # Time step during simulation, = 2^16 from start to
    end
    train_delay=14000,  # Distance between pulsetrains, ps
    n_pulses=5000,  # no. of pulse trains
    bare_energies=[0.0, 1.46],  # state energies, eV
    rot_wave=False # apply rotating wave approximation
):

    n_levels = len(bare_energies)
    omega = omega_ev * HBARINV  # eV --> rad/ps.

    pulse_times = np.array([centre_of_first_pulse, delay_to_second_pulse
    ]).cumsum()
    # hamiltonian array to mesolve
    H0 = q.Qobj(HBARINV * np.diag(bare_energies))
    if rot_wave:
```

```python
        Hup = HBARINV * cfmu * q.Qobj(np.eye(n_levels, k=1))
        Hdown=HBARINV * cfmu * q.Qobj(np.eye(n_levels, k=-1))
        H = [
            H0,
            [Hup, get_stringRotWave(pulse_times[0], pulse_width, omega, phi1, train_delay,1j)],
            [Hup, get_stringRotWave(pulse_times[1], pulse_width, omega, phi2, train_delay,1j)],
            [Hdown, get_stringRotWave(pulse_times[0], pulse_width, omega, phi1, train_delay,-1j)],
            [Hdown, get_stringRotWave(pulse_times[1], pulse_width, omega, phi2, train_delay,-1j)],
        ]
    else:
        transitions = q.Qobj(np.eye(n_levels, k=1) + np.eye(n_levels, k=-1))
        Hi = HBARINV * cfmu * transitions
        H = [
            H0,
            [Hi, get_string(pulse_times[0], pulse_width, omega, phi1, train_delay)],
            [Hi, get_string(pulse_times[1], pulse_width, omega, phi2, train_delay)],
        ]

    # Lindbladian dephasing is implemented as the operator
    # L_l = \sum_d g_{di}|i><i|
    # with strength \Gamma_d >= 0. (g_{di} are real dimensionless scalars)
    # This translates to Bloch-Redfield dephasing in the form of off-diagonal decay rates
    # \dot{\rho_{mn}} =1/2 \sum_d \Gamma_d (g_{dm}-g_{dn})^2
    # (For further discussion, see section 2.3 of my thesis.)
    # The argument weights is an array (g0, g1, ..., g{n_levels-1}) of dephasing weights
    if any(len(weights) != n_levels for weights, _ in dephasing_times):
        raise ValueError(
            "All dephasing weight arrays must have same length as n_levels!"
    )
    dephasing_operators = [
        np.sqrt(1 / td) * q.Qobj(np.diag(weights)) for weights, td in dephasing_times
    ]

    relaxation_operators = [
        np.sqrt(1 / tr) * q.Qobj(matrix_ij(n_levels, i, j))
        for i, j, tr in relaxation_times
    ]
    collapse_operators = dephasing_operators + relaxation_operators

    # time (neglect steps after integration window)
    base_times = np.linspace(start_time, end_time, n_steps + 1)
    if sum_all:
        times= base_times
        window = np.where(times > 0)[0]
    else:
        times = base_times[base_times < (pulse_times[1] + 2 * pulse_width + integral_window)]
        window = np.where(times > (pulse_times[1] + 2 * pulse_width))[0]

    # initial state
    rho0 = q.Qobj(matrix_ij(n_levels, 0, 0))

    # zeroth iteration
    output = q.mesolve(H, rho0, times, collapse_operators, [], args={"m": 0})
    ress=[do_output(0, window, output, n_levels)]
    # reuse qutip information about hamiltonian
    opts = q.Options(rhs_reuse=True)

    # loop over different trains
    for m in np.arange(1, n_pulses):
        output = q.mesolve(
            H, rho0, times, collapse_operators, [], args={"m": m}, options=opts
        )
        ress.append(do_output(m, window, output, n_levels))

    f = open("info.txt", "a")
    f.write(str(round(delay_to_second_pulse*10000))+"  "+str(n_levels)
        +"  "+str(pulse_width)+"  "+str(phi1)+"  "+str(phi2)
        +"  "+str(train_delay)+"  "+str(n_pulses)+"\n")
    f.close()
    return ress
```

```
from dqc2_multlevel import double_quantum_coherence_simulation
from datetime import datetime
import numpy as np

n_puls=6
phi_1=0
phi_2=10**-4/6
tr_delay=10000
```

```python

Tmin=  0
Tmax=  0.2
Tstep=40+1

nsteps=8192
Intall=False

Date = str(datetime.now())
Date_Str = '_'+Date[5:7]+Date[8:10]+'_'+Date[11:13]+Date[14:16]+Date
    [17:19]
name="pyres_Nils"+Date_Str+".txt"
del Date, Date_Str

f = open(name, "a")
f.write("tau [ps]\t")
for phase in range(0,n_puls):
    f.write(str(int(phase*phi_2*tr_delay*360))+"\t")
f.write("\n")
f.close()

for tau in np.linspace(Tmin, Tmax, Tstep):
    tau=round(tau,8)
    print(str(tau)+' \t/'+str(Tmax)+' ps')
    ress=double_quantum_coherence_simulation(
        centre_of_first_pulse=0.2,# ps
        delay_to_second_pulse=tau,  # ps
        pulse_width=0.1,  # ps
        phi1=phi_1, # Linear frequency of pulse 1, THz
        phi2=phi_2, # Linear frequency of pulse 2, THz
        n_pulses=n_puls, # no. of pulse trains
        train_delay=tr_delay, # Distance between pulsetrains, ps
        n_steps=nsteps,  # Time step during simulation, = 2^13
        end_time=0.8, # ps
        sum_all=Intall,
        rot_wave=False
    )

    f = open(name, "a")
    f.write(str(tau)+"\t")
    for res in ress:
        f.write(str(res)+"\t")
    f.write("\n")
    f.close()
```

7.3.4 *python* script for DFT and plotting

```python
#%% imports
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from datetime import datetime

lw=1.2
pstr=r"python: "
mstr=r"matlab: "
fstr=r"$\cdot(\phi_1-\phi_2)$ contr."

#%  Read and plot python results
pFile = pd.read_csv("pyres_Nils_0513_184447.txt",
sep = "\t",      # Here we specify what values are seperated by. \t
    means we seperate by the "tab" button
header = 0)      # This skips the first n+1 rows of the file

pdata = pFile.to_numpy()
px=pdata[:,0]
px=px*1000

pphases=pdata[:,1:-1]
N_t=len(pphases[0])

del pFile, pdata

#% DFT
fm_list=[]
for fm in range(1,3):
    summ_list=[]
    for pphase in pphases:
        summ=0
        for n_t,el in enumerate(pphase):
            summ=summ+el*np.exp(fm*1j*2*np.pi/N_t*n_t)
        summ_list.append(summ/N_t)
    fm_list.append(summ_list)
del summ, summ_list, n_t, el, pphase, fm

abs_fm_list=np.abs(fm_list)
pf1=abs_fm_list[0]/max(abs_fm_list[0])
pf2=abs_fm_list[1]/max(abs_fm_list[1])

#% plot python results

plt.plot(px, pf1, '--', label=pstr+"1"+fstr, linewidth=lw, color='tab:
    blue')
```

```python
plt.plot(px, pf2, '--', label=pstr+"2"+fstr, linewidth=lw, color='tab:orange')

#%  Read and plot matlab results
mFile = pd.read_csv("matlab_res_mlres240513_195045_fft.csv",
sep = "\t",       # Here we specify what values are seperated by. \t means we seperate by the "tab" button
header = 0)       # This skips the first n+1 rows of the file

mdata = mFile.to_numpy()
mx,mf1,mf2=mdata.transpose()
del mFile, mdata
plt.plot(mx,mf1, '-',  label=mstr+"1"+fstr, linewidth=lw, color='tab:blue')
plt.plot(mx,mf2, '-',  label=mstr+"2"+fstr, linewidth=lw, color='tab:orange')

plt.legend(framealpha=1)
plt.xlim(0,200)
plt.ylim(0,  1)
plt.grid()
plt.xticks(fontsize=13)
plt.yticks(fontsize=13, ticks=np.arange(0, 1.1, 0.1))
plt.xlabel(r"delay $\tau$ (fs)", fontsize=15)
plt.ylabel(r"normalized amplitude (arb. u.)", fontsize=15)


#% Save plot to png file
Date = str(datetime.now())
Date_Str = '_'+Date[5:7]+Date[8:10]+'_'+Date[11:13]+Date[14:16]+Date[17:19]
del Date
plt.savefig("curve_overlay"+Date_Str+".png", dpi=400, bbox_inches='tight')
```