POLITECNICO
DI TORINO

Université
de Paris

Master double degree course in
Nanotechnologies for ICTs and Quantum Devices

Master Degree Thesis

Quantum toolbox for cavity quantum electrodynamics

Supervisors
Prof. Carlo Ricciardi
Prof. Maria Luisa Della Rocca

Candidate
Fabrizio Berritta

Internship Tutor
Center for Nanoscience and Nanotechnology (C2N)
Dr. Loïc Lanco

June 2020

I hereby declare that the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Fabrizio Berritta

Paris, 8 of July 2020

Abstract

This Thesis describes a custom-made cavity quantum electrodynamics (QED) toolbox for a quantum dot (QD) emitter in an optical micropillar. The toolbox has been developed for MATLAB® and it allows using either a full cavity-QED model or an effective adiabatic Hamiltonian to work only with the QD subspace. The toolbox simulates output intensities, first- and second-order correlations, and flux spectral densities both in continuous and pulsed wave regime. The results show that the adiabatic model reduces the computational cost in comparison to the full model, and allows performing accurate quantum optics simulations in the weak coupling regime between the QD and the cavity. For the approximation to yield satisfactory results, the cavity must decay at a faster timescale than the other subsystems, including the QD dynamics and the incoming field: the Rabi frequency of the QD must be much slower than the cavity damping rate, whereas, for the incoming field, its evolution must be slow compared to the photon lifetime in the cavity. This work also finds applications in the more general context of excited dipoles in 1-D photonic crystal waveguides and nanocavities, and it can be generalized to more complex and realistic systems. This includes the description of anisotropic neutral quantum dots, described by 3-level systems, or charged quantum dots with a spin degree of freedom, modeled by 4-level systems, taking into account the polarization degree of freedom for the cavity and input/output fields.

Acknowledgements

I would like to express my uttermost gratitude to my supervisor Dr. Loïc Lanco of the C2N laboratory, for his persistent guidance and help. I acknowledge the great effort that he has given to my internship. His relentless, yet thoughtful, perseverance has contributed greatly to making my thesis more accurate, comprehensible, and thorough.
I am very thankful to the Ph.D. students Clément Millet and Elham Mehdi for being always available to answer my questions and for having introduced me to the laboratory. Indeed, I would also like to thank all the members of the Solid-State Quantum Optics group for all the interesting and fruitful discussions that have kept me company during a worldwide lockdown.

The people that I am going to mention should stop reading right now. They could easily choose something more exciting for them to read or to do. Then, I would love to hear about it. However if you are one of them, unfortunately, you have decided to continue, not listening to my warning.

I thank the members of *L'Associazione*, with whom I share my dreams, literally and figuratively. I recognize their unconditional support during these last five years, drown into a carousel of emotions and experiences that have greatly outlined my way of thinking.

I would like to mention Eleonora Cappuccio. Although she has not contributed directly to this work, she has played a major role in all my life decisions that have led to this manuscript.

I am grateful to Gabriele Caliri and Lorenzo Giannì for their presence and all the valuable discussions, no matter the subject and the distance.

It goes without saying, I thank my Family.

Contents

List of Figures

Chapter 1

Introduction

Quantum technology is based on the development and control of systems governed by the laws of quantum physics, while conventional technology can be understood in the context of classical mechanics. The first practical reason that drives quantum technology is the miniaturization of devices down to the nanometer scale [1], where Planck's constant becomes comparable with the action scales. The second reason, more fundamental, is that quantum mechanics may improve the performances of the classical picture. In the *first quantum revolution*, quantum mechanics was used to model and understand something that already existed. It was possible to explain the different behavior of metals and semiconductors, but not to design and build "artificial atoms". In the *second quantum revolution*, we are not passive observers anymore. It is possible to generate quantum states of matter and energy that would not likely exist elsewhere.

1.1 Photons and Quantum dots

Photons are of particular interest being the smallest units of energy of the quantized electromagnetic field and it is possible to manipulate them separately. They propagate at the speed of light, hence they are ideal as *flying qubits* for long-range information transfer. The fundamental difference, compared to a classical bit of information, is that a *qubit* can be prepared in a coherent superposition of states. Photons are promising also to interconnect different physical systems. The nodes of the quantum network are interrelated by quantum channels and at each node, the information is processed [2]. To exploit the quantum mechanical properties of photons, bright, efficient, and integrable quantum light sources are needed, as well as efficient detectors and two-photon gates. Superconducting nanowires detectors already show efficiencies close to unity in a wide spectral range and high spatial resolution [3]. Previously single-photon based technologies were based on heralded single-photon techniques, based on probabilistic generation in non-linear crystals. However, one problem of these devices is the source brightness scaling linearly with the multi-photon pairs probability. Possible solutions include neutral atoms and ions in traps, solid-state emitters such as quantum dots and defects and molecules, without excluding hybrid systems to optimize wavelength and bandwidth [4]. Still, currently available sources have intrinsically limited efficiencies of a few percents.

When light of a certain frequency is directed onto a semiconductor, an electron-hole pair may be generated. The latter is bound by the Coulomb interaction and it is called an exciton. Moreover, if the exciton is confined in a QD, then the exciton acts like an artificial atom with complex spectra. It is possible to obtain bound excitons also by injecting electrons and holes into a quantum dot. QD excitons act as a bridge between quantum electronics and the world of quantum optics, paving the way to quantum optoelectronics. Much attention in this field has been focused on the development of a single-photon source that generates pulses of light, with each pulse containing one and only one photon, starting from almost twenty years ago [4]. Besides, the spin of a single charge carrier in the QD can be used as a stationary qubit and can be optically manipulated [5]. Finally, QDs are compatible with the III-V semiconductor nanofabrication techniques. More specifically, it is possible to couple QDs to a variety of photonic structures, such as waveguides, microlenses, microcavities, etc. for large scale applications [4].

1.2 Internship Presentation & State of The Art

This Master's thesis has been realized in the group of Prof. Pascale Senellart at Centre for Nanoscience and Nanotechnology (C2N/CNRS) under the supervision of Dr. Loïc Lanco. The group activity is focused on the development of quantum devices based on single QDs in cavity quantum electrodynamics (CQED) systems. The research team developed a new technology in 2008, called in-situ optical lithography [6], which allows for the deterministic and scalable fabrication of CQED devices by a single QD coupled to a micropillar cavity. Through this technique, it is possible to work both in the weak and strong coupling regimes [6; 7]. Afterward, an ultrabright source of entangled photon pairs was fabricated in 2010 [8]. In 2013, bright source QD-micropillar devices were realized, with indistinguishability ranging from 70% to 90% [9]. In 2016 a brightness 20 times brighter than any source of equal quality [10] was obtained with near-unity indistinguishability. Nowadays, a key challenge is the source operation wavelength variability [11] because of inhomogeneities during the growth process across the wafer. Another quest is controlling the temporal profile of a single photon wavepacket, since it requires control of the fine-structure splitting and the axes' orientation between the cavity and the neutral QD.

My internship was expected to be experimental, however, because of the Covid-19 lockdown, I worked on the modeling and simulation of CQED open systems. For this reason, I have chosen here to enter more in the details of quantum system simulations rather than the devices themselves. The choice of the quantum dot in the micropillar will be discussed more thoroughly in the next chapter, in comparison with other quantum emitter systems. In quantum optics, it is often required to simulate the physical properties of a system coupled to a reservoir. The system of interest has in general much fewer degrees of freedom compared to the environment, e.g. phonons, photons, nuclear spins, etc. Quantum dissipative systems in general require numerical integrations to be solved, but they are still a delicate task. For an arbitrary system, there exist *path-integral* expressions [12] for density matrix evolution, but they suffer from instability at long times. In the Schrödinger picture approach, a solution is to integrate the *Master equation* for the density matrix [13] or by

using *quantum Monte Carlo* simulations [14].
If a system is described by an n dimensional Hilbert space, the number of matrix elements is n^2 for Schrödinger equation and n^4 for the density matrix equations. This polynomial computational memory is further reduced by considering that many of the previous coefficients are zero, hence making viable the integration of systems of $10^2 : 10^3$ equations numerically by using sparse matrices, within a few hours on an average personal computer. Moreover, when CQED simulations require few photons and atoms, it is possible to truncate the Fock space basis for the light field modes with satisfactory results. Even so, sometimes it may be required to increase the dimension of the Fock space to better reproduce higher photon number experiments, resulting in much longer simulation times.
A possible way to overcome this problem is offered in quantum systems composed of several interacting subsystems [15]. One may be interested in the dynamics of one subsystem, getting rid of the others by making some approximations. This operation is called *adiabatic elimination* and it offers an accurate model as long as the selected subsystem has much slower dissipation rates than the others, to which it is weakly coupled. In the specific case of a cavity interacting with an electromagnetic field, the total Hilbert space is given by the tensor product of the spaces of the two individual systems, increasing the number of variables. This is where *adiabatic elimination* may become a useful tool to simplify the problem numerically or to get a better physical understanding of some phenomenon, or both. Indeed, by adiabatic elimination one works only with the QD sub-space, taking the cavity sub-space into account by an effective Hamiltonian.
The aim of my internship was the implementation of adiabatic elimination of the light modes Fock space in high-loss cavities for several quantum optics simulations, concerning mainly the simplest model of a QD as ground and excited state. It was also expected to investigate the validity and differences of the adiabatic elimination relating to the "full model" and literature. Moreover, my work was to be included in a new quantum toolbox to simulate CQED multi-level systems.

1.3 Outline of the Thesis

In Chapter 2 the QD is introduced as a simple ground and excited state inside a micropillar, first as a closed quantum system by recalling the Jaynes-Cummings Hamiltonian. In the following, to treat the cavity QED as an open quantum system, the input-output formalism and Lindblad Master equation are reviewed. In Chapter 3 the custom made cavity-QED Quantum Toolbox is presented, showing some simulations regarding output intensities, first- and second-order correlations, and time-frequency analysis. Some results are related to previous ones in literature for referring theory to past experiments. The adiabatic elimination is presented in Chapter 4, where the key approximations from the full model are discussed, followed by possible criteria for its domain of validity. Finally, generalizations of this work are suggested, with applications on more complex CQED models.

Chapter 2

Quantum Dots in Micropillar Cavities

This chapter aims to develop a quantum mechanical description of a neutral quantum dot (QD) in a micropillar cavity. The most important approximations will be outlined, at the end of which the Hamiltonian will fully describe the closed quantum system evolution. Dissipation and decoherence will be briefly mentioned in the following section. They will be addressed more thoroughly at the end of this chapter, focused on the system coupling with the environment.

Why Cavity-QED with Quantum Dots? A single-photon (SP) source is necessary to perform quantum communication. It will be shown why optical microcavities are studied nowadays for the realization of a SP source. For that purpose, it is important to briefly review the main properties that an ideal SP source should satisfy. They are presented in figure 2.1, where each slot corresponds to a time bin.
The first row (a) represents a source that emits identical single-photon wavepackets at each excitation pulse. The following rows represent deviations from the perfect situation when the source lacks one or more properties that label the corresponding row. For instance, in (b), when an emitted photon is lacking given an excitation, then the brightness is lower than 100%, lowering the overall efficiency. Another important property is the SP purity (c), which is related to the probability of having a second photon in the emitted wavepacket. This is a flaw for instance in quantum key distribution since secure communications would require only a stream of single photons to avoid photon-number splitting attack [16]. The purity is measured by a Hanbury Brown and Twiss (HBT) experiment, which evaluates the second-order correlation function of a given source. For a pure SP source $g^{(2)}(0) = 0$, whereas when $g^{(2)}(0) > 0$ there is a non-zero probability to have at least a second emitted photon after the excitation pulse.
As for classical fields, the coherence of a SP source describes the light phase stability and in (d) it is represented by a phase jump in the wavepacket. The total coherence time T_2 is given by

$$\frac{1}{T_2} = \frac{1}{2\,T_1} + \frac{1}{T_2^*}, \tag{2.1}$$

where T_1 is the transition radiative lifetime and T_2^* is the pure dephasing time, i.e. a loss of coherence due to the interaction with the environment without recombination. Equivalently, equation (2.1) can be written for the total dephasing rate γ as

$$\gamma = \frac{\gamma_{sp}}{2} + \gamma^*, \tag{2.2}$$

being γ_{sp} the damping rate of the population due to spontaneous emission in the external environment and γ^* the damping rate of the phase amplitude.
When the rate of pure dephasing is low enough in equation (2.1), then the so-called Fourier transform-limit is obtained:

$$\frac{T_2}{2\,T_1} \to 1 \quad . \tag{2.3}$$

This limit is a necessary condition for photons indistinguishability (e), which occurs

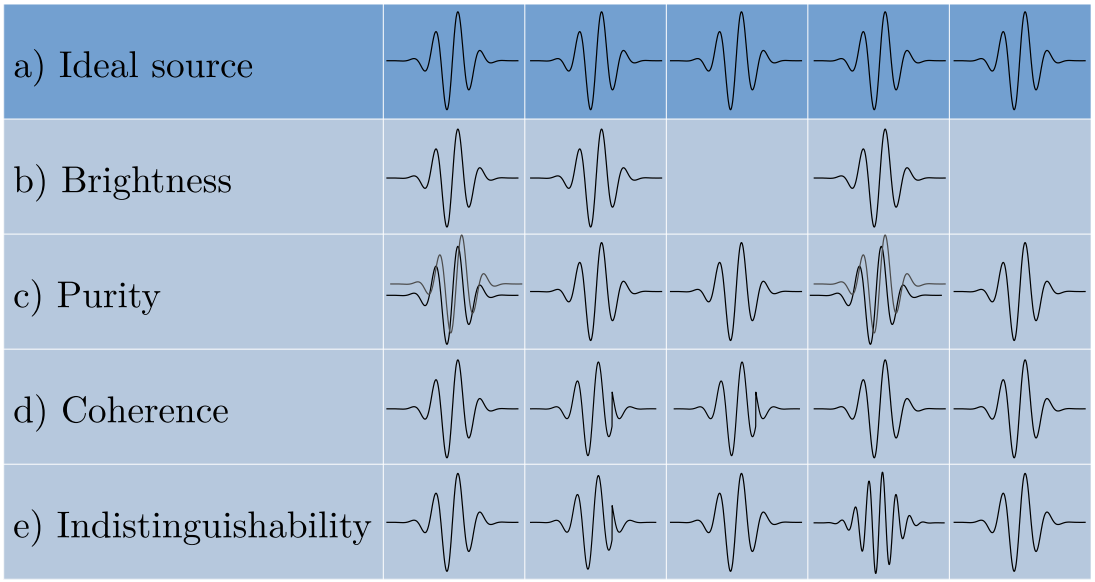


Figure 2.1: (a) An ideal source with identical, single photons generated per excitation pulse. (b) When the brightness is lower than unity, sometimes no photon emission occurs. (c) As a second identical photon is present in the wavepacket, the purity is degraded. (d) Coherence loss for phase instability due to $T_2 \leq 2\,T_1$. (e) Indistinguishability $V < 1$ because of decoherence and spectral wandering.

when two photons have the same spectral bandwidth, pulse width, polarization, carrier frequency, and mode profile. In such a case, when they impinge at the same time at the two ports of a beamsplitter (BS), quantum mechanics predicts they both will leave the BS from the same output port. This phenomenon was first measured by Hong-Ou-Mandel. Spectral wandering is another process contributing to distinguishability because of the environmental fluctuations. It is characterized by a shift of the photon wavelength, spoiling indistinguishability which is important for low-error quantum computation.
The main limitation for brightness is the refractive index mismatch between the QD layer

and air ($n \approx 3.5$ for GaAs), confining most of the light inside the semiconductor. It is possible to improve the extraction efficiency by inserting the QD between two Distributed Bragg Reflectors (DBRs), realizing a 3-D microcavity to redirect spontaneous emission toward the surface. The brightness can be defined as [4]

$$\text{brightness} = p \times \eta, \tag{2.4}$$

where η is the extraction efficiency and, in a two-level system, p is the occupation probability for the excited state

$$\eta = \eta_{\text{out}} \times \beta = \eta_{\text{out}} \times \frac{\Gamma_{\text{m}}}{\Gamma_{\text{m}} + \gamma_{\text{sp}}}. \tag{2.5}$$

Here η_{out} is the output extraction efficiency and the *spontaneous emission coupling factor* β measures the fraction of photons emitted into the cavity mode Γ_{m}, considering also the other radiative modes γ_{sp}. The *increase* of spontaneous emission in the mode of interest can be achieved through the Purcell effect, in the domain of cavity quantum electrodynamics (CQED) in the weak coupling regime [17]. This is employed for instance in micropillars, whose first lens brightness nowadays is about 14% with 91% indistinguishability [11]. Because of the Purcell-enhanced emission in the cavity, the inverse of the QD lifetime is now given by $1/T_1 = \Gamma_{\text{tot}} \equiv \Gamma_{\text{m}} + \gamma_{\text{sp}}$, with Γ_{tot} the total emission rate taking into account both emissions into and outside the cavity mode. It also improves the ratio $T_2/(2T_1)$, since the coherence time T_2 is now given by

$$\frac{1}{T_2} = \frac{\Gamma_{\text{tot}}}{2} + \gamma^*. \tag{2.6}$$

A simple schematic of the micropillar used at C2N is shown in figure 2.2. The InGaAs is at

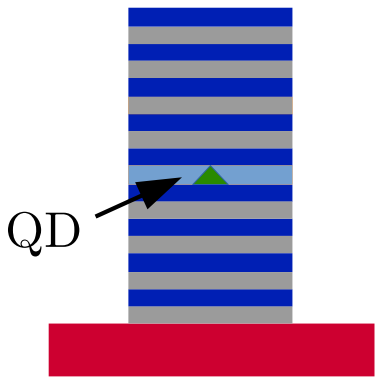


Figure 2.2: In green, the QD sandwiched between two Bragg mirrors of the micropillar, standing on the substrate.

the center of a cavity with a cylindrical shape. The GaAs spacer is sandwiched between the top and bottom AlGaAs/GaAs DBRs. The latter confine light along the vertical direction, while total internal reflection guarantees lateral confinement.

2.1 The Quantum Dot and Pseudo-spin Systems

Quantum dots are structures confining charge carriers in regions of space over the nanometer scale. In general direct-bandgap semiconductors are used for optical applications to have direct transitions in reciprocal space. The aim here is to show why a QD can be seen as an artificial atom, interacting with an electromagnetic field in the dipole approximation. The analysis will be restricted to two energy levels, to be described in terms of Pauli spin algebra.
In self-assembled QDs, quantum confinement generates localized states, having shells in the conduction and valence band with discrete energy levels. These quantization effects become relevant when the confinement dimensions are comparable with De Broglie wavelength of the charge carrier. At low temperatures, λ_{DB} is of the order of $10 \div 100\,\text{nm}$ [18]. For the QD fabricated at C2N/CNRS, a Transmission Electron Microscopy (TEM) image and the energy levels are shown in figure 2.3. The InGaAs QD is separated by the GaAs bulk by an InAs wetting layer, originated by the growth process. The energy levels have been computed in a non-interacting charge picture; electrons and holes fill the shells according to the Pauli exclusion principle.
The fundamental state is when the valence band is filled and the conduction band is empty. Among the possible excited states, the neutral exciton will be of interest, with an electron in the conduction band and a hole in the valence band. For the CQED simulation toolbox, without loss of generality, it is assumed that the interband transition between the two lowest energy levels is dipole allowed.

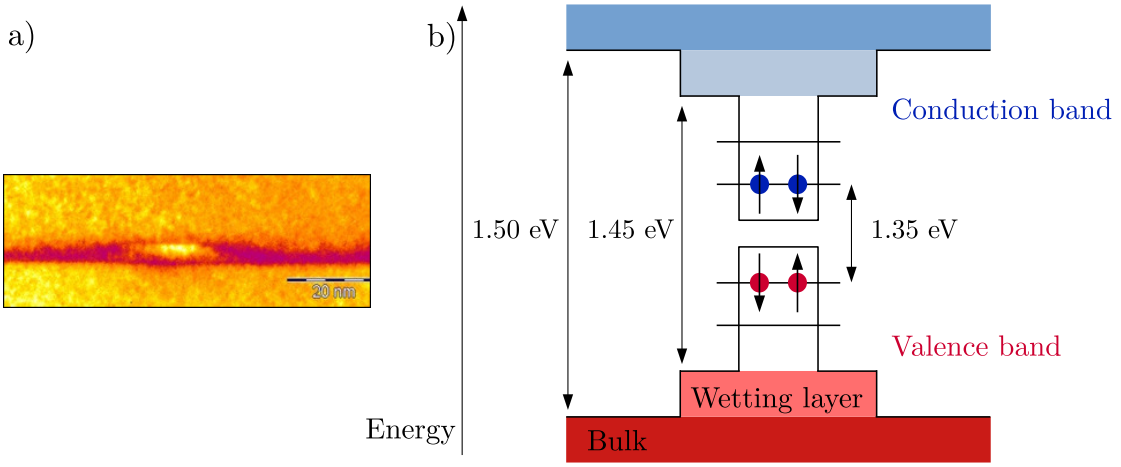


Figure 2.3: **(a)** Transmission Electron Microscope image of a single InGaAs QD embedded into GaAs. Image measured by A. Lemaitre and C. Gomez at C2N. **(b)** Schematics of energy levels of non-interacting charge in the InGaAs QD, including the InAs wetting material and GaAs bulk. The lowest states in energy are filled according to the Pauli exclusion principle.

For the quantum mechanical description, the unexcited state of the QD is the state $|g\rangle$, whereas the exciton is $|e\rangle$, according to Dirac notation. At this point it is favorable to

define the so-called *atomic transition operators* [16]

$$\hat{\sigma}_+ \equiv |e\rangle\langle g|\,, \quad \hat{\sigma}_- \equiv |g\rangle\langle e| = \hat{\sigma}_+^\dagger, \tag{2.7}$$

being $\hat{\sigma}_+^\dagger$ the hermitian conjugate of $\hat{\sigma}_+$. The *projection operator* in the excited state is

$$|e\rangle\langle e| = \hat{\sigma}_+\hat{\sigma}_-. \tag{2.8}$$

The zero energy is referred to the ground state $|g\rangle$, thus the free atomic Hamiltonian is

$$\hat{H}_\mathrm{d} = (E_\mathrm{e} - E_\mathrm{g})\hat{\sigma}_+\hat{\sigma}_- = \hbar\omega_\mathrm{d}\hat{\sigma}_+\hat{\sigma}_-, \tag{2.9}$$

Throughout this manuscript, "quantum dot" and "two-level atom" will be used interchangeably, since only the two-level system will be addressed.

2.2 The Micropillar as a Cavity-QED

In this section a model of the micropillar will be shown based on common CQED parameters, that will be reviewed shortly based on [17]. The same arguments can be applied to different micro-optical cavities, to be possibly implemented in the Cavity-QED Quantum Toolbox. It is assumed that the atom in the cavity is a two-level system with fixed energies and that it can radiatively decay, emitting a photon in the cavity, or absorb one from the latter. The parameters that describe the interaction between the atom, that will represent our QD, and the cavity field (see figure 2.4, left) are:

- the photon decay rate of the cavity (damping rate) κ, given by $\kappa = \omega/Q$, where ω is the angular frequency and Q is the quality factor. Higher quality factors will reduce the cavity losses.
- the coupling strength g between the atom and the electromagnetic field;
- the dephasing rate γ, taking into account only the interaction with the external environment, i.e. emission outside the cavity mode and pure dephasing : $\gamma = \gamma_\mathrm{sp}/2+\gamma^*$.

Before commenting more on the previous parameters, for the following discussion, it is important to distinguish between two possible regimes of the atom interacting with the cavity. In the so-called *strong coupling* limit $g \gg (\kappa, \gamma)$ the coupling strength is much greater than the highest value between the cavity decay rate κ and the non-resonant decay rate γ. In this regime, the atom strongly interacts with the cavity, and a *reversible* process occurs. Indeed, the losses are so low that the emitted photon from the atom instead of escaping from the cavity, it is readily absorbed by the atom. Viceversa is the *weak coupling* limit, $g \ll (\kappa, \gamma)$: it describes an irreversible process, as the emitted photon is promptly lost and no re-absorption can occur.
As mentioned before, in this manuscript the weak coupling regime will be of interest because of the Purcell effect, to increase spontaneous emission in the mode of interest. Before describing how this phenomenon affects the micropillar, it is crucial to outline the coupling strength g. Differently from the case when a two-level atom interacts with a resonant light field originating from an external source, here there is no external source that gives the

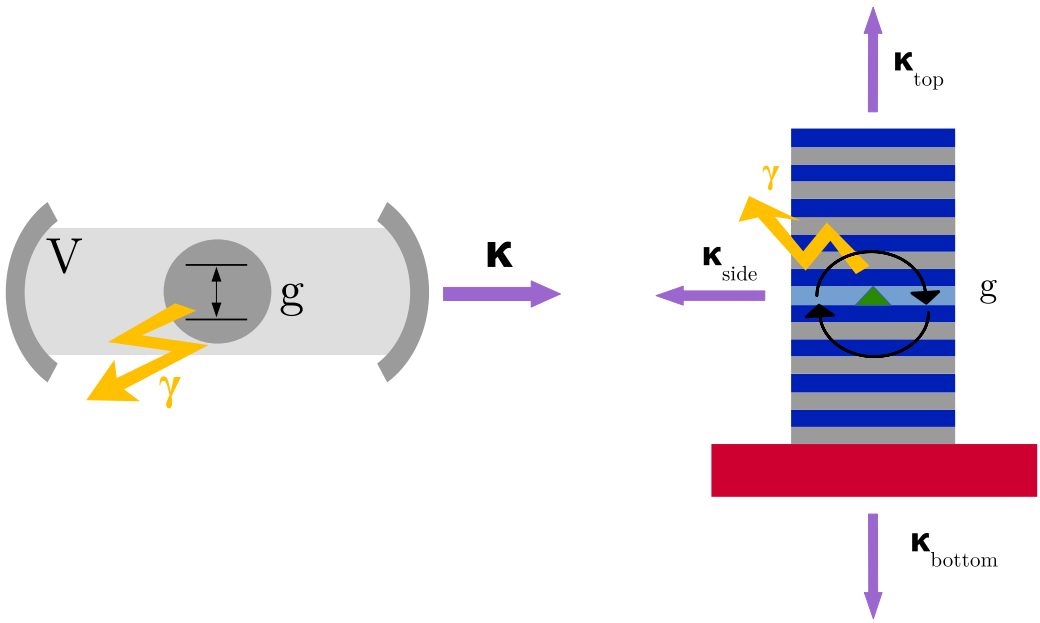


Figure 2.4: (Left) A two-level atom is in a resonant cavity. The cavity parameters are g, κ, γ, and V. They are respectively the atom-cavity coupling, the photon decay rate from the cavity, the dephasing rate, and the modal volume. (Right) Micropillar with QD and corresponding cavity parameters.

field strength.
In the weak coupling regime, since the effect of the cavity is relatively small, the atom-cavity interaction can be modeled by perturbation theory. Distinctively from the free space case, the emission rate in the cavity can be tuned since the cavity changes the photon number density of states that appears in Fermi's golden rule. By considering an atom coupled resonantly to a single mode of a high-Q cavity, it is found that the ratio between the cavity emission rate Γ_{m} and the one in a homogeneous medium γ_{sp}, is

$$F_{\mathrm{P}} \equiv \frac{\Gamma_{\mathrm{m}}}{\gamma_{sp}} = \frac{4g^2}{\kappa\gamma_{\mathrm{sp}}}. \tag{2.10}$$

In the previous equation, the Purcell factor F_{P} has been defined and it emerges that high-quality factor and low mode volume are required to get a strong emission rate in the cavity mode. By introducing the Purcell factor in equation (2.5), one gets

$$\beta = \frac{F_{\mathrm{P}}}{F_{\mathrm{P}} + 1}. \tag{2.11}$$

So as $F_{\mathrm{P}} \gg 1$, the coupling factor β approaches unity, increasing the brightness of the optical cavity.

The micropillar cavity In the micropillar (figure 2.4, right) g describes the time scales at which energy can be coherently exchanged between the cavity field and the exciton. The

dephasing rate is $\gamma = \gamma_{\text{sp}}/2 + \gamma^*$. It models the rate at which the QD leaks information in the external (i.e. not including the cavity) environment. The total cavity damping rate is $\kappa = \kappa_{\text{top}} + \kappa_{\text{side}} + \kappa_{\text{bottom}}$, being κ_{top} and κ_{bottom} accounting for the photons escaping the cavity from the top and bottom mirror, respectively, while κ_{side} trough the sidewalls of the cavity.
Three important parameters describe the performances of a QD-micropillar device [19] and they will be used in the Quantum Toolbox:

- the *cooperativity* $C = g^2/\kappa\gamma$, which quantifies the coherent processes component over the incoherent one;
- the top-mirror *output coupling efficiency* $\eta_{\text{top}} = \kappa_{\text{top}}/\kappa$, identifying the fraction of photons escaping the cavity through the top mirror and to be collected;
- the *input coupling efficiency* η_{in}, which is the probability for an incoming photon to be coupled to the cavity mode.

2.3 Jaynes–Cummings Model

Here the Jaynes-Cummings Hamiltonian in the dressed states formalism is reviewed to model the interaction of a two-level atom with the single mode of an electromagnetic field. Experimentally it is indeed possible to have the optical cavities supporting only a single mode, as in the version of the micropillar of figure 2.2. The consequence of introducing an interaction term with the field is that in general $|e\rangle$ an $|g\rangle$ will not be eigenstates anymore of the Hamiltonian, leading to the so-called Rabi oscillations [16].
In the *dipole approximation* [16] the interaction Hamiltonian with a quantized field $\hat{\mathbf{E}} \propto (\hat{a} - \hat{a}^\dagger)$ is:

$$\hat{H}_{\text{int}} \approx -\hat{\mathbf{d}} \cdot \hat{\mathbf{E}}, \tag{2.12}$$

being $\hat{\mathbf{d}}$ the dipole moment of the quantum emitter. The free-field Hamiltonian, to be added to the free-atom one and the interaction term, is

$$\hat{H}_{\text{cav}} = \hbar\omega_{\text{c}}\hat{a}^\dagger\hat{a}, \tag{2.13}$$

having neglected the zero-point energy. By making the *rotating wave approximation* (RWA), the Jaynes-Cummings Hamiltonian [16] is

$$\begin{aligned}\hat{H}_{\text{JC}} &\approx \hat{H}_{\text{d}} + \hat{H}_{\text{cav}} + \hat{H}_{\text{int}} \\ &= \hbar\omega_{\text{d}}\,\hat{\sigma}_+\hat{\sigma}_- + \hbar\omega_{\text{c}}\,\hat{a}^\dagger\hat{a} - \mathrm{i}\hbar g\left(\hat{\sigma}_-\hat{a}^\dagger - \hat{\sigma}_+\hat{a}\right),\end{aligned} \tag{2.14}$$

The interaction term $\hat{H}_{\text{int}}$ causes only transitions of the kind

$$|e\rangle\,|n\rangle \longleftrightarrow |g\rangle\,|n+1\rangle\,, \tag{2.15}$$

between two product states, where the second kets n are labeled by the number of photons in the mode. These states are also called "bare" states of the Jaynes-Cummings model. By

letting $n \geq 0$ and expressing $\hat{H}_{\mathrm{JC}}$ in the basis $|\Psi_{1n}\rangle = |e\rangle\,|n\rangle$ and $|\Psi_{2n}\rangle = |g\rangle\,|n+1\rangle$, the eigenvalues are

$$E_{\pm}(n) = \hbar\omega_{\mathrm{c}}\left(n + \frac{1}{2}\right) + \frac{\hbar\omega_{\mathrm{d}}}{2} \pm \frac{\hbar}{2}\Omega_n(\Delta), \tag{2.16}$$

where

$$\Omega_n(\Delta) = \left[\Delta^2 + 4g^2(n+1)\right]^{1/2} \quad (\Delta = \omega_{\mathrm{d}} - \omega_{\mathrm{c}}) \tag{2.17}$$

is the Rabi frequency comprising half the detuning. An example at resonance, i.e. $\omega_{\mathrm{c}} = \omega_{\mathrm{d}} \equiv \omega_0$, is shown in figure 2.5. In (a) the detuning is null and the atom is decoupled

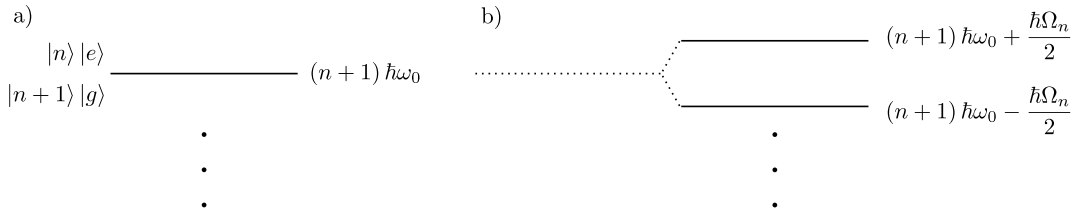


Figure 2.5: (a) Degenerate energy levels for an uncoupled atom-field system at resonance, $\omega_{\mathrm{c}} = \omega_{\mathrm{d}} \equiv \omega_0$. (b) Level splitting due to the atom-field interaction, Ω_{n} is the Rabi frequency.

from the field, i.e. $g = 0$. The energy levels are degenerate. The degeneracy is lifted by introducing coupling ($g > 0$), splitting the original two degenerate eigenstates into two new ones separated by the energy $\hbar\Omega_{\mathrm{n}}$. This phenomenon is known also as dynamic or AC Stark effect. The perturbed eigenstates $|n, \pm\rangle$ are called the "dressed states" of the Jaynes-Cummings model.

2.4 Open Quantum Systems

Quantum states that are described by state vectors are called *pure* states. States that cannot be described by state vectors are called *mixed* states. Both are described in terms of the density operator $\hat{\rho}$ in the following way:

$$\hat{\rho} \equiv \sum_i p_i\ |\psi_i\rangle\langle\psi_i|\,, \tag{2.18}$$

where the sum is over a statistical ensemble such that p_i is the probability of the system of being in the state $|\psi_i\rangle$, given the normalized state $\langle\psi_i|\psi_i\rangle = 1$. A density matrix $\hat{\rho}$ has unit trace, i.e. $\mathrm{tr}\,(\hat{\rho}) = 1$, and it is positive. Moreover, if and only if the state is pure, $\mathrm{tr}\,(\hat{\rho}^2) = 1$. Similarly for $\mathrm{tr}\,(\hat{\rho}^2) < 1$, the system is a mixed state. For a general quantum state, the average of some operator $\hat{A}$ is given by

$$\left\langle \hat{A} \right\rangle = \mathrm{tr}\left(\hat{\rho}\hat{A}\right) = \sum_i p_i\,\langle\psi_i|A|\psi_i\rangle\,. \tag{2.19}$$

In the absence of dissipative interactions and in the case of no explicitly time dependent interaction, the density operator evolves following the unitary transformation, called the *Von Neumann equation* [16]:

$$\frac{\mathrm{d}\hat{\rho}}{\mathrm{d}t} = \frac{\mathrm{i}}{\hbar}\left[\hat{\rho}, \hat{H}\right], \tag{2.20}$$

whereas the *Heisenberg equation of motion* is given by

$$\frac{\mathrm{d}\hat{A}^{(H)}}{\mathrm{d}t} = -\frac{\mathrm{i}}{\hbar}\Big[\hat{A}^{(H)}, \hat{H}\Big]. \tag{2.21}$$

2.4.1 Master equation

The density matrix formalism is useful for the description of incoherent and dissipative processes that occur mainly when the system is coupled to a reservoir, figure 2.6. The aim is to describe the dynamics of the system of interest with the reservoir entering only as parameters, reviewing the approximations stated in [20]. The total Hamiltonian of the reservoir and the system is given by

$$\hat{H} = \hat{H}_S + \hat{H}_R + \hat{H}_{SR}, \tag{2.22}$$

where $\hat{H}_S$ and $\hat{H}_R$ are Hamiltonians for the undamped system (S) and the reservoir (R), respectively, and $\hat{H}_{SR}$ is the interaction Hamiltonian. The density operator $\hat{\rho}(t)$ of the

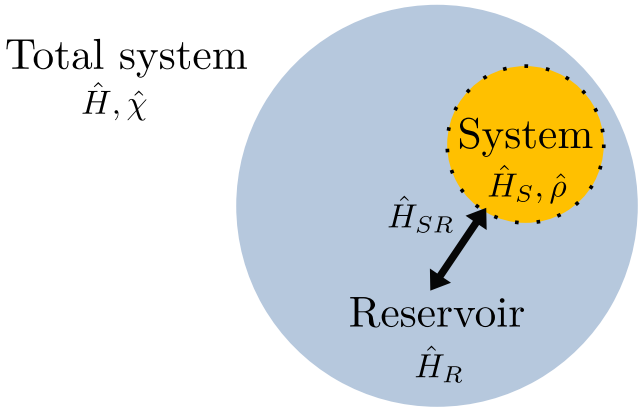


Figure 2.6: Total system with Hamiltonian $\hat{H}$ and density operator $\hat{\chi}$ divided into the system of interest, "System", and the Reservoir.

system is given by tracing the density matrix $\hat{\chi}(t)$ of the composite system $S \otimes R$ over all possible states of the reservoir, i.e.

$$\hat{\rho}(t) \equiv \mathrm{tr}_R\left[\hat{\chi}(t)\right]. \tag{2.23}$$

It is assumed that the interaction is turned on at $t = 0$, with no correlation between S and R at this initial time. Then it is possible to factorize the density operator of the composite system as

$$\hat{\chi}(0) = \hat{\rho} \otimes \hat{R}_0, \tag{2.24}$$

being R_0 the initial reservoir density operator. In general, at later times the correlations between S and R will couple the system and the reservoir. By assuming that the coupling is very weak and that R is a very large system weakly affected by S, it is legit to write

$$\hat{\chi}(t) = \hat{\rho}(t) \otimes \hat{R}_0 + \mathcal{O}\Big(\hat{H}_{SR}\Big). \tag{2.25}$$

Notice that the previous expression is still exact, based on the previous assumptions. The first Born approximation consists of neglecting terms higher than the second order in H_{SR}. Still, the evolution of $\hat{\rho}(t)$ depends on its history [20]. The second major approximation is the Markov approximation, which is based on the existence of two very different time scales: a slow time scale for the system dynamics, compared to the decay of the reservoir correlation functions.
Under these approximations, in the special case where there is a single initial state $|i\rangle$ and final state $|f\rangle$, the incoherent process are described in terms of collapse operators $\hat{C}_{if}$ with rate γ_{if}:

$$\hat{C}_{if} = \sqrt{\gamma_{if}}\,|f\rangle\langle i|\,. \tag{2.26}$$

For each collapse operator, a superoperator called Lindbladian $\hat{\mathcal{L}}_{if}$ is defined such that

$$\hat{\mathcal{L}}_{if}(\hat{\rho}) \equiv \hat{C}_{if}\hat{\rho}\hat{C}_{if}^{\dagger} - \frac{1}{2}\Big\{\hat{C}_{if}^{\dagger}\hat{C}_{if}, \hat{\rho}\Big\}, \tag{2.27}$$

being the last term the anticommutator defined as $\Big\{\hat{A}, \hat{B}\Big\} \equiv \hat{A}\hat{B} + \hat{B}\hat{A}$. The density matrix evolution in the RWA is given by the *Lindblad master equation*:

$$\frac{\mathrm{d}\hat{\rho}}{\mathrm{d}t} = \frac{\mathrm{i}}{\hbar}\Big[\hat{\rho}, \hat{H}\Big] + \sum_{if}\hat{\mathcal{L}}_{if}(\hat{\rho}), \tag{2.28}$$

where the coherent processes are described by the first term on the RHS and the Lindbladian operators account for the ensemble of the possible incoherent processes.

2.4.2 Input-output formalism

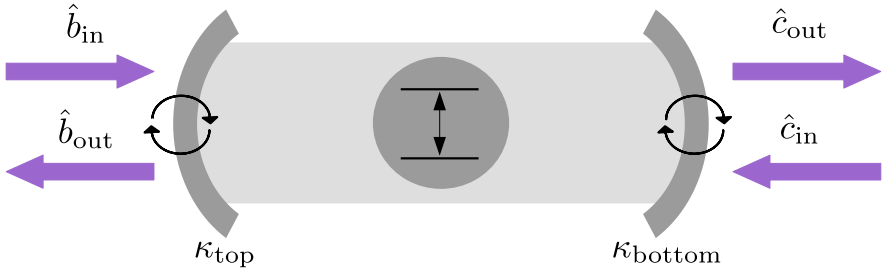


Figure 2.7: Input-output field formalism with a 1-D atom.

In the previous section the master-equation has been introduced to calculate the physical properties of a damped system, in this thesis the cavity with an artificial atom. In this section the aim is to treat explicitly the external field as the reservoir, in order to determine the effect of the internal and external cavity dynamics. In the input-output formalism the properties of the field exiting the system can be determined based upon information on the dynamics of the atom-cavity and the input field. The following results have been derived assuming the RWA and system-bath interactions that are linear in the bath operators. Moreover, the coupling constant is assumed to be a constant independent of frequency [21].

In the ideal case of a one-dimensional cavity as in figure 2.7, the first port (top-mirror) is associated to the operators $\hat{b}_{\text{in}}$ and $\hat{b}_{\text{out}}$, whereas $\hat{c}_{\text{in}}$ and $\hat{c}_{\text{out}}$ for the second port (bottom-mirror). In the 1D-atom limit, where the dominant atomic radiative interactions are via the cavity mode, the side losses are much lower than the ones from the mirrors ($\kappa_{\text{side}} \ll \kappa_{\text{top}} + \kappa_{\text{bottom}}$) and the cooperativity ($C \gg 1$) [22].

Given an input or output operator $\hat{s}$, the expected value $\langle \hat{s}^\dagger \hat{s} \rangle$ is the associated photon flux defined as *number of photons per unit time*. Thus the unit of each operator is $\text{s}^{-1/2}$. Given the single confined mode of annihilation operator $\hat{a}$, the continuity relations of the fields are:

$$\hat{b}_{\text{out}} = \hat{b}_{\text{in}} + \sqrt{\kappa_{\text{top}}}\,\hat{a}, \tag{2.29a}$$
$$\hat{c}_{\text{out}} = \hat{c}_{\text{in}} + \sqrt{\kappa_{\text{bottom}}}\,\hat{a}, \tag{2.29b}$$

reflecting the interference between the incoming field and the one escaping from the cavity. The interaction Hamiltonian between the cavity and the external electromagnetic field is given by:

$$\hat{H}_{\text{pump}} = -\text{i}\hbar\sqrt{\kappa_{\text{top}}}\left(\hat{b}_{\text{in}}\hat{a}^\dagger - \hat{b}_{\text{in}}^\dagger\hat{a}\right). \tag{2.30}$$

A useful simplification is the specific case of a coherent beam at resonant excitation. In such case the input fields are fully described by the mean input field amplitudes, namely $b_{\text{in}} = \left\langle \hat{b}_{\text{in}} \right\rangle$ and $c_{\text{in}} = \langle \hat{c}_{\text{in}} \rangle$. This approximation has simplified the computational costs of the numerical results of the `CQED Quantum Toolbox`, to be addressed in the next chapter. It is anticipated here that the numerical simulations are based on an input field b_{in} solely from the top-mirror. Therefore, $c_{\text{in}} = 0$ and in this limit equations (2.29, 2.30) become

$$\hat{b}_{\text{out}} = b_{\text{in}}\hat{I} + \sqrt{\kappa_{\text{top}}}\hat{a}, \tag{2.31a}$$
$$\hat{c}_{\text{out}} = \sqrt{\kappa_{\text{bottom}}}\hat{a}, \tag{2.31b}$$

and

$$\hat{H}_{\text{pump}} = -\text{i}\hbar\sqrt{\kappa_{\text{top}}}\left(b_{\text{in}}\hat{a}^\dagger - b_{\text{in}}^*\hat{a}\right), \tag{2.32}$$

where the $\hat{I}$ on the right-hand side of equation (2.31a) must be understood as the identity operator. These relations will be exploited to evaluate the properties of the output fields both in the continuous wave (CW) or pulsed wave (PW) laser regime.

2.4.3 The "full" model

At this point it is possible to model the time evolution of the exciton in the micropillar, starting from the Lindblad Master equation (2.28). It is thus necessary to define the Hamiltonian of interest and the collapse operators. In the following the rotating frame of reference will be considered, centered at the laser frequency ω. The Hamiltonian is given by

$$\hat{H} = \hat{H}_{\text{d}} + \hat{H}_{\text{cav}} + \hat{H}_{\text{int}} + \hat{H}_{\text{pump}}, \tag{2.33}$$

where

$$\hat{H}_{\mathrm{d}} = (\omega_{\mathrm{d}} - \omega)\,\hat{\sigma}_{+}\hat{\sigma}_{-}, \tag{2.34a}$$

$$\hat{H}_{\mathrm{cav}} = (\omega_{\mathrm{c}} - \omega)\,\hat{a}^{\dagger}\hat{a}, \tag{2.34b}$$

$$\hat{H}_{\mathrm{int}} = \mathrm{i}g\left(\hat{\sigma}_{+}\hat{a} - \hat{\sigma}_{-}\hat{a}^{\dagger}\right), \tag{2.34c}$$

$$\hat{H}_{\mathrm{pump}} = -\mathrm{i}\sqrt{\kappa_{\mathrm{top}}}\left(b_{\mathrm{in}}\hat{a}^{\dagger} - b_{\mathrm{in}}^{*}\hat{a}\right), \tag{2.34d}$$

where the $\hbar$ constant has been dropped. At last, the collapse operators are required to describe the dissipative processes. The *cavity damping* $\hat{C}_{\mathrm{cav}} = \sqrt{\kappa}\,\hat{a}$, with κ the total damping rate, models the cavity optical losses. The QD *spontaneous emission in the leaky modes* is given by $\hat{C}_{\mathrm{QD}} = \sqrt{\gamma_{\mathrm{sp}}}\,\hat{\sigma}_{-}$, being γ_{sp} the spontaneous emission rate. Last, the QD pure dephasing is described by $\hat{C}_{\mathrm{deph}} = \sqrt{2\gamma^{*}}\,\hat{\sigma}_{+}\hat{\sigma}_{-}$, where the population is preserved during the dephasing. From equation (2.27), the overall Lindbladian becomes then:

$$\hat{\mathcal{L}} = \hat{\mathcal{L}}_{\mathrm{cav}} + \hat{\mathcal{L}}_{\mathrm{QD}} + \hat{\mathcal{L}}_{\mathrm{deph}}, \tag{2.35}$$

whose labels reflect the recently defined collapse operators.

Conclusions

This chapter has presented the main properties that single-photon quantum emitters should satisfy and the potential of QDs in microcavities. The micropillar solution, in the weak coupling regime, is known to guarantee simultaneously high brightness and degree of indistinguishability. After having introduced a pseudo-spin formalism to model the two-level system of an exciton in the QD, the main parameters characterizing an optical cavity have been briefly reviewed and translated to the micropillar. For the following simulations, the Jaynes-Cummings Hamiltonian has been introduced to model the coupling between the exciton and the micropillar as closed quantum systems. At last, to describe the system dissipation, an open quantum system description based on the Master Equation has been presented with its approximations. In the following chapter, the `CQED Quantum Toolbox` will be presented with the implemented quantum optics simulations.

Chapter 3

Cavity-QED Quantum Toolbox

Given a quantum emitter in an optical cavity interacting with an external field, one may be interested in some physical properties such as output intensities, first- and second-order correlations, flux spectral densities, etc. In this manuscript the `Quantum Optics Toolbox` [23] has been exploited to numerically solve the Master Equation on `MATLAB`®. A custom made `Cavity QED Quantum Toolbox` has been developed to efficiently organize different simulations for two-level systems coupled to a microcavity mode. For instance, even though field spectrum processing under stationary resonant excitation and second-order correlations require different approaches to be simulated, they both share the same device, i.e. the QD in the micropillar, and they are modeled by the same Hilbert space and Master Equation formalism.
The scripts for the two-level system are contained in Appendix C and, based on figure 3.1, they are divided in the following sections:

Main programs It contains the main scripts, for both continuous wave (CW) and pulsed wave (PW) regime. They incorporate parameters commonly set by an experimentalist, the field input-output operators and the core of the code to let the reader understand which quantities are actually computed and how they are defined.

CQED device parameters The parameters of the device are imported from a file `Init 2level device parameters`. It includes the atom-cavity coupling, the photon decay rate from the micropillar, the spontaneous emission rate, the pure dephasing rate, the resonance frequencies etc., which characterize a given QD-cavity device.

CQED subprograms The subroutine `Init 2level Hilbert space and operators` defines the Hilbert space and some of the operators introduced in the previous chapter. Mainly two other kinds of subroutines are present. The ones that start by `Init...` contain the initialization of some additional variables and they allow to preallocate memory. The `plot...` scripts generate plots based on the choice set by the user in the associated main script.

Each program contained in the main folder can be executed by choosing the *full model* presented at the end of the previous chapter, or by selecting the *adiabatic model*, where the cavity mode Fock space is not explicitly considered (see Chapter 4) by proper redefinition of some of the full model operators. Before introducing the several quantum optics simulations,

it is instructive to briefly introduce the "matrix representation" of Lindblad Master equation, required for the numerical integration.

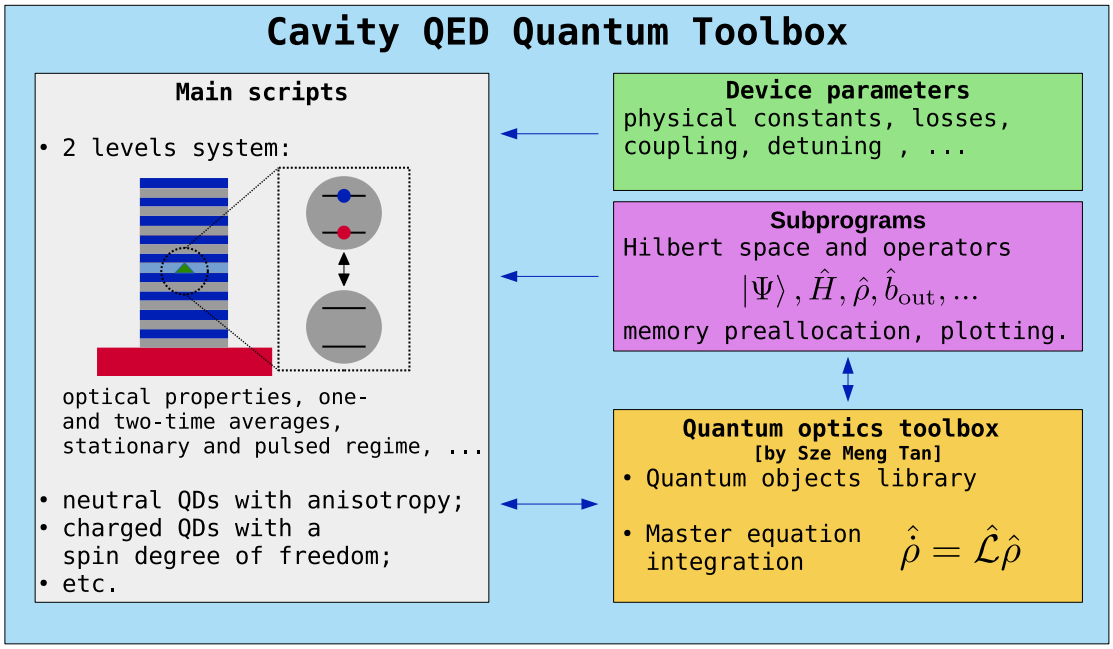


Figure 3.1: Custom `Cavity QED toolbox`. Here $A \leftarrow B$ means that B sends data to A.

The Fock-Liouville Hilbert space Some linear combinations of density matrices are still density matrices as long as they have unit trace and positivity. A Hilbert space of density matrices can be defined converting the matrices into vectors of the so called *Fock-Liouville space* (FLS). The corresponding scalar product between $\hat{\phi}$ and $\hat{\rho}$ is defined as $\langle\langle\phi|\rho\rangle\rangle \equiv \mathrm{tr}\left(\hat{\phi}^\dagger\hat{\rho}\right)$. The Liouville superoperator $\hat{\mathcal{L}}$ of equation (2.27) is now an operator $\tilde{\mathcal{L}}$ acting on the FLS. For instance, it is possible to define for a two-level system:

$$|\rho\rangle\rangle \equiv \begin{pmatrix}\rho_{00}\\ \rho_{01}\\ \rho_{10}\\ \rho_{11}\end{pmatrix}, \tag{3.1}$$

where ρ_{ij} is an element of the density matrix in the chosen representation basis. At last, the time evolution of the system corresponds to a system of first order differential equations $\frac{\mathrm{d}|\rho\rangle\rangle}{\mathrm{d}t} = \tilde{\mathcal{L}}|\rho\rangle\rangle$, to be numerically solved.

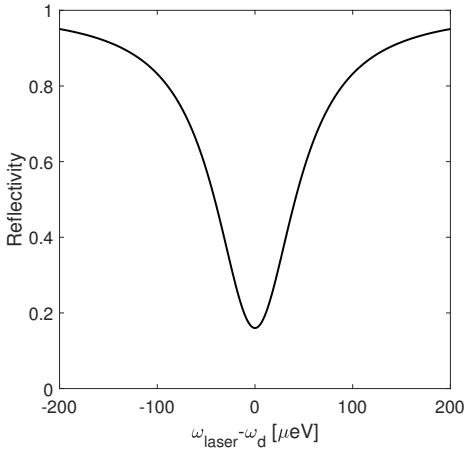
3.1 Nonlinear Optics with CQED

In this section the spectral response of an atom-cavity device will be addressed based on the parameters that have been introduced in Chapter 2. Before showing the simulation results of

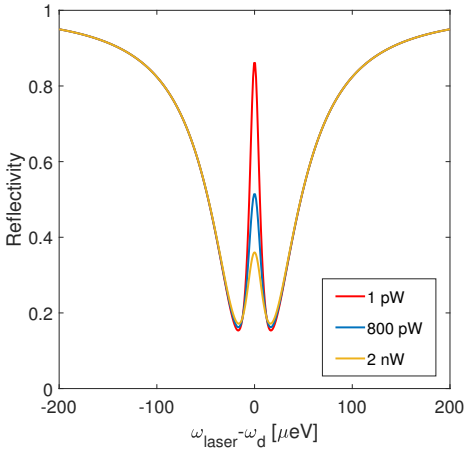
the reflection coefficient as a function of the laser energy ω, some features of the reflectivity can be predicted based on the analytical formula [19] obtained in the semi-classical approach, valid only for low power and $\gamma^* = 0$. It is found that for the reflection coefficient

$$r(\omega) \equiv \left\langle \hat{b}_{\text{out}} \right\rangle / b_{\text{in}} = 1 - \frac{\kappa_{\text{top}}}{\frac{\kappa}{2} - \text{i}\left(\omega - \omega_{\text{c}} - \frac{g^2}{(\omega - \omega_{\text{d}} + \text{i}\gamma)}\right)}, \tag{3.2}$$

being ω the energy of the excitation laser, ω_{c} and ω_{d} the energy of the cavity and the quantum dot resonance, respectively. As in the previous chapter, g is the coupling strength, γ is the total dephasing rate and κ the cavity damping rate. The reflectivity in the simulation is given by $R \equiv \left\langle \hat{b}^{\dagger}_{\text{out}} \hat{b}_{\text{out}} \right\rangle / |b_{\text{in}}|^2$ and for the uncoupled cavity ($g = 0$), a Lorentzian dip in the reflectivity spectrum, at the cavity frequency ω_c, is described from equation (3.2). The following curves have been obtained by simulating at resonance, $\omega_{\text{d}} = \omega_{\text{c}}$, with null pure dephasing, $\gamma_{\text{sp}} = 0.6\,\mu\text{eV}$, $\gamma^* = 0$, $\kappa = 100\,\mu\text{eV}$ and extraction efficiency of the top mirror $\eta_{\text{top}} = \frac{\kappa_{\text{top}}}{\kappa} = 0.7$. The reflectivity spectrum of an uncoupled cavity ($g = 0$) is the expected



(a) Simulated reflectivity as a function of the laser frequency impinging on an empty cavity, input power $P_{\text{in}} = 10\,\text{pW}$.

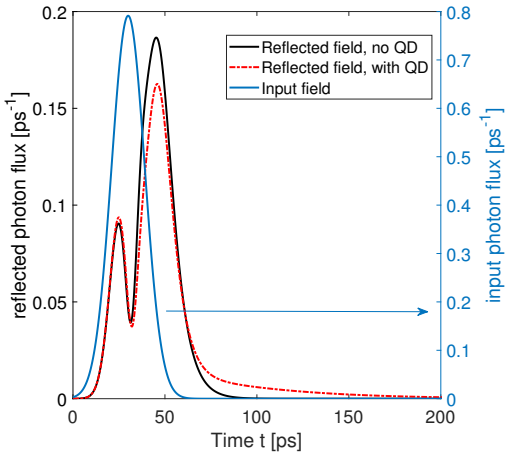


(b) Simulated reflectivity of the cavity coupled to the QD, $g = 17\,\mu\text{eV}$, for different input powers P_{in}.

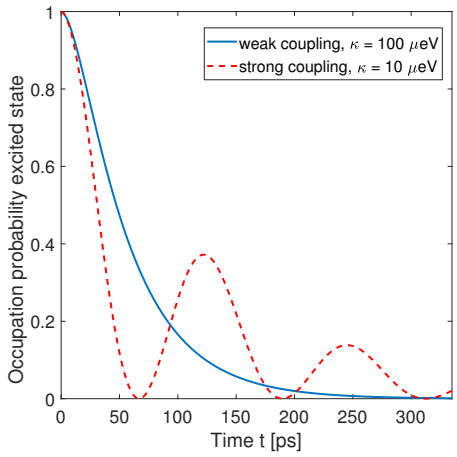
Figure 3.2: Reflectivity spectra in continuous wave excitation.

Lorentzian curve centered at the cavity resonance, see figure 3.2a. The two-level atom strongly modifies the reflectivity with a peak because of light being resonantly scattered. In the particular case of weak coupling regime, setting $g = 17\,\mu\text{eV}$ in figure 3.2b, the light intensity tunes the light-atom coupling because when the transition is saturated, the atom has equal probability to be in the ground ot excited state, and it becomes transparent to the optical field since it has the same probability to experience absorption or stimulated emission. This phenomenon has been verified experimentally, for instance in figure B.1 in Appendix B. As the contribution of the emission in the mode of interest decreases, also the fraction of photons emitted outside the mode reduces with increasing input power, as shown in figure B.2.

In *pulsed wave* (PW) regime one can analyze the temporal evolution of a reflected wavepacket. In the following simulation, it has been considered an input Gaussian pulse of 15 ps Full Width Half Maximum (FWHM), longer than the cavity lifetime of approximately 7 ps. The coherent laser pulse has a fixed center frequency, temporal width and average number of photons.
In figure 3.3a it is shown the response of the micropillar with and without the QD. In the case of an empty cavity, the first peak is the light being directly reflected from the top mirror. It is followed by partially-destructive interference with the light escaping from the cavity through the same top mirror, according to equation (2.31a). The second peak is mostly due to the light which exits the cavity at the end of the excitation pulse, so that no interference occurs and the decay time corresponds to the cavity damping rate.
As the exciton is coupled to the cavity mode, the second peak height is reduced because of light absorbed by the artificial atom. Moreover, the reflected photon flux tail decays at a lower rate than before because of the exciton spontaneous emission and the light slowly being emitted by the QD in the cavity mode. The last process occurs through Purcell-enhanced single-photon emission, with a typical decay time Γ_{tot}, and the light is extracted from the cavity mode by the top-mirror.
By using the same program, it is possible to describe a non-resonant excitation experiment



(a) Reflected field as a function of the time evolution. The input field has 15 ps FWHM and the average photon number is $N_{\text{P}} = 10$.



(b) Occupation probability as a function of time for the QD in the initial excited state, in the weak and strong coupling regime.

Figure 3.3: Results from pulsed-wave excitation script.

where the exciton state $|e\rangle$ is populated at time zero. To simulate such an effect, one can simply change the initial state and use a very small value for the pulse average photon number, to ensure that the output fields are almost entirely induced by the initial excitation. The result is shown in figure 3.3b. For the blue curve, the high value of the cavity damping rate $\kappa = 100\,\mu\text{eV}$ determines that the QD-cavity system is in the weak coupling regime, being the merit factor $S \equiv 4g/\kappa = 0.68 < 1$ [24]. Vice-versa, for the dashed red curve, the lower damping rate $\kappa = 10\,\mu\text{eV}$ implies that the system is in the strong-coupling regime. In this figure, the Rabi oscillations are visible, with damping mainly caused by spontaneous

emission outside the cavity mode and intracavity photon losses. If not stated otherwise, the default simulation parameters of this chapter will be those defined in this section, in the weak coupling regime, i.e. $g = 17\,\mu\text{eV}$, $\gamma_{\text{sp}} = 0.6\,\mu\text{eV}$ and $\kappa = 100\,\mu\text{eV}$.

3.2 First-order Coherence and Spectrum

In this section the quantum-mechanical first-order coherence functions will be briefly reviewed, highlighting how they evaluate the coherent and incoherent contributions to the reflected photon flux. In the CW regime the first-order temporal coherence has been obtained by the *quantum regression theorem* [25], which greatly simplifies the task of computing two-time correlation functions as one-time averages. The incoherent flux component has been Fourier transformed to obtain the incoherent spectral density of the field. On the other hand, in PW regime the Wigner Distribution Function of the first-order correlation has been evaluated, yielding information on both the photon flux and energy spectral densities. In the dipole interaction, mentioned in Chapter 2, using the Heisenberg picture, the general first-order correlation function [16] for a scalar field is defined as:

$$G^{(1)}\left(t_1, t_2\right) = \text{tr}\left(\hat{\rho}\hat{E}^{(-)}\left(t_2\right)\hat{E}^{(+)}\left(t_1\right)\right), \tag{3.3}$$

with normal ordering and where, summing over the modes k of frequency ω_{k}, electric constant ε_0 and cavity volume V,

$$\hat{E}^{(+)}(t_i) = \text{i}\sum_k \left(\frac{\hbar\omega_k}{\varepsilon_0 V}\right)^{1/2} \hat{a}_k(t_i). \tag{3.4}$$

The normalized first-order quantum coherence function is

$$g^{(1)}\left(t_1, t_2\right) = \frac{G^{(1)}\left(t_1, t_2\right)}{\left[G^{(1)}\left(t_1, t_1\right) G^{(1)}\left(t_2, t_2\right)\right]^{1/2}}, \tag{3.5}$$

such that three possible degrees of coherence are defined:

$$\left|g^{(1)}\left(t_1, t_2\right)\right| = 1 \quad \text{complete coherence,} \tag{3.6a}$$

$$0 < \left|g^{(1)}\left(t_1, t_2\right)\right| < 1 \quad \text{partial coherence,} \tag{3.6b}$$

$$\left|g^{(1)}\left(t_1, t_2\right)\right| = 0 \quad \text{incoherent.} \tag{3.6c}$$

Such a quantity plays a major role in any experiment using an interference between two time-delayed components of an optical field, as in a Michelson or Mach-Zender setup. Notice that complete coherence is obtained when the expectation value of the numerator can be factorized, i.e. $\left\langle \hat{E}^{(-)}\left(t_2\right)\hat{E}^{(+)}\left(t_1\right)\right\rangle \equiv \left\langle \hat{E}^{(-)}\left(t_2\right)\right\rangle \left\langle \hat{E}^{(+)}\left(t_1\right)\right\rangle$, where the right hand side is the coherent contribution.

3.2.1 Continuous wave regime

For the following discussion, it is assumed a CW excitation. Being at stationary regime, one may set a general reference time $t_1 = 0$, and define the delay $\tau \equiv t_2 - t_1$ so that equation (3.5) reduces to

$$g^{(1)}(\tau) = \frac{\left\langle \hat{E}^{(-)}(t+\tau)\hat{E}^{(+)}(t) \right\rangle}{\left\langle \hat{E}^{(-)}(t)\hat{E}^{(+)}(t) \right\rangle}. \tag{3.7}$$

The details on the normalized-first order coherence computation by the *quantum regression*

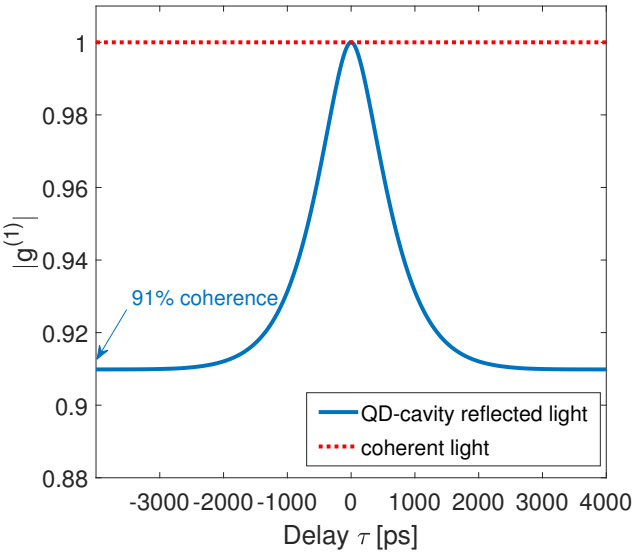


Figure 3.4: First order coherence of the reflected light in continuous wave regime, compared to a coherent field. The input power is $P_{\text{in}} = 10\,\text{pW}$ and $\kappa = 400\,\mu\text{eV}$. The reflected field is partially coherent.

theorem are discussed in Appendix A. As before, the reflected flux is given by $\left\langle \hat{b}^{\dagger}_{\text{out}}(t)\hat{b}_{\text{out}}(t) \right\rangle$, in Heisenberg picture. Being here in the CW regime, the flux takes a constant value $\Phi_{\text{R}} = \left\langle \hat{b}^{\dagger}_{\text{out}}\hat{b}_{\text{out}} \right\rangle$ in the stationary state. Given that $\hat{b}_{\text{out}}$ is a fluctuating operator, due to the random nature of spontaneous emission, it should become uncorrelated with itself at very distant times. Mathematically speaking,

$$\lim_{\tau\to\infty} \left\langle \hat{b}^{\dagger}_{\text{out}}(t+\tau)\hat{b}_{\text{out}}(t) \right\rangle = \left\langle \hat{b}^{\dagger}_{\text{out}} \right\rangle \left\langle \hat{b}_{\text{out}} \right\rangle = \Phi_{\text{R,coh}}, \tag{3.8}$$

where $\Phi_{\text{R,coh}}$ is the coherent photon flux reflected by the micropillar with the QD inside. By comparing equation (3.8) with the definition of the normalized first-order correlation function (3.7), it follows that $\lim_{\tau\to\infty} g^{(1)}(\tau) = \Phi_{\text{R,coh}}/\Phi_{\text{R}}$, being the total reflected flux the sum of the coherent and incoherent components, $\Phi_{\text{R}} = \Phi_{\text{R,coh}} + \Phi_{\text{R,incoh}}$. An example is shown in figure 3.4, where the $g^{(1)}(\tau)$ of a coherent field is compared to the reflected light

from the optical micropillar. At zero delay the normalized-first order coherence has unit modulus as expected from the definition. For positive infinite delay, and negative, being $g^{(1)}(-\tau) = g^{(1)}(\tau)^*$, the coherent percentage $\Phi_{\mathrm{R,coh}}/\Phi_{\mathrm{R}}$ of the reflected light is recovered, in this example about 91 %.

Resonance fluorescence spectrum and Mollow triplets In this paragraph the radiation due to an isolated atom driven by a monochromatic field will be addressed. The optical spectrum of a CW optical field is characterized by its spectral density of flux, $S(\omega)$, whose integral over the whole spectrum returns the total photon flux. Given a field operator, e.g. the reflected field $\hat{b}_{\mathrm{out}}$, the Wiener-Khinchin theorem [25] states that its spectrum is the Fourier transform of the related correlation function $G^{(1)}(\tau)$. In the rotating frame at the laser frequency ω_{s}, the spectrum of the reflected field is

$$S_{\mathrm{R}}(\omega) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \left\langle \hat{b}^{\dagger}_{\mathrm{out}}(\tau)\hat{b}_{\mathrm{out}}(0)\right\rangle \mathrm{e}^{\mathrm{i}(\omega-\omega_{\mathrm{s}})\tau}\mathrm{d}\tau, \tag{3.9}$$

normalized such that its integral over the whole spectrum gives the total reflected flux:

$$\int_{-\infty}^{+\infty} S_{\mathrm{R}}(\omega)\mathrm{d}\omega = \left\langle \hat{b}^{\dagger}_{\mathrm{out}}(0)\hat{b}_{\mathrm{out}}(0)\right\rangle = \Phi_{\mathrm{R}}. \tag{3.10}$$

As stated before, the quantity $\left\langle \hat{b}^{\dagger}_{\mathrm{out}}(\tau)\,\hat{b}_{\mathrm{out}}(0)\right\rangle$ is the sum of the coherent component $\left\langle \hat{b}^{\dagger}_{\mathrm{out}}(0)\right\rangle \left\langle \hat{b}_{\mathrm{out}}(0)\right\rangle$ and a varying contribution that tends towards zero for larger delays. Correspondingly, the total spectrum $S_{\mathrm{R}}(\omega)$ will be the sum of two contributions: $S_{\mathrm{R}}(\omega) = S_{\mathrm{R,coh}}(\omega) + S_{\mathrm{R,incoh}}(\omega)$. The coherent contribution has a DC value which, when Fourier transformed over time, leads to a monochromatic spectrum centered at the laser frequency, described by $S_{\mathrm{R,coh}}(\omega) = \left\langle \hat{b}^{\dagger}_{\mathrm{out}}\right\rangle \left\langle \hat{b}_{\mathrm{out}}\right\rangle \times \delta(\omega-\omega_{\mathrm{s}})$. On the other hand, the varying contribution leads, after Fourier Transform, to a continuous spectrum $S_{\mathrm{R,incoh}}(\omega)$ which is denoted as the spectrum of the incoherent component of the photon flux. Such a component is called incoherent since, not being monochromatic, it cannot interfere with the incoming laser. In the weak-coupling regime this incoherent spectrum has the shape of a Lorentzian at low power, but of a Mollow triplet at high power, see figure 3.5. Experimental curves from literature are shown in Appendix B, figure B.3.
The laser frequency is $\omega_{\mathrm{s}} = \omega_{\mathrm{d}}$, so it is at resonance with the QD driven at the Rabi frequency Ω. At the low power or weak field limit, $\Omega \ll \gamma_{\mathrm{sp}}$ and the Rabi frequency of the driving field is much smaller than the spontaneous emission rate. The atom behaves as an over-damped quantum harmonic oscillator and the spectrum is a Lorentzian. Yet, when the Rabi frequency becomes comparable to the exciton linewidth in the strong excitation limit $\Omega \gg \gamma_{\mathrm{sp}}$, the energy level splitting described in section 2.3 occurs. This generates two side-bands at $\omega_s \pm \Omega$ in addition to the transition at ω_{s}, resulting in the Mollow triplet.

3.2.2 Pulsed wave regime

Constant CW regime can be used to design a two-level system single photon source. However, one would be interested in deterministic sources that can be triggered on-demand. This is in general done by an optical pulse, which requires a quantum dynamical treatment of

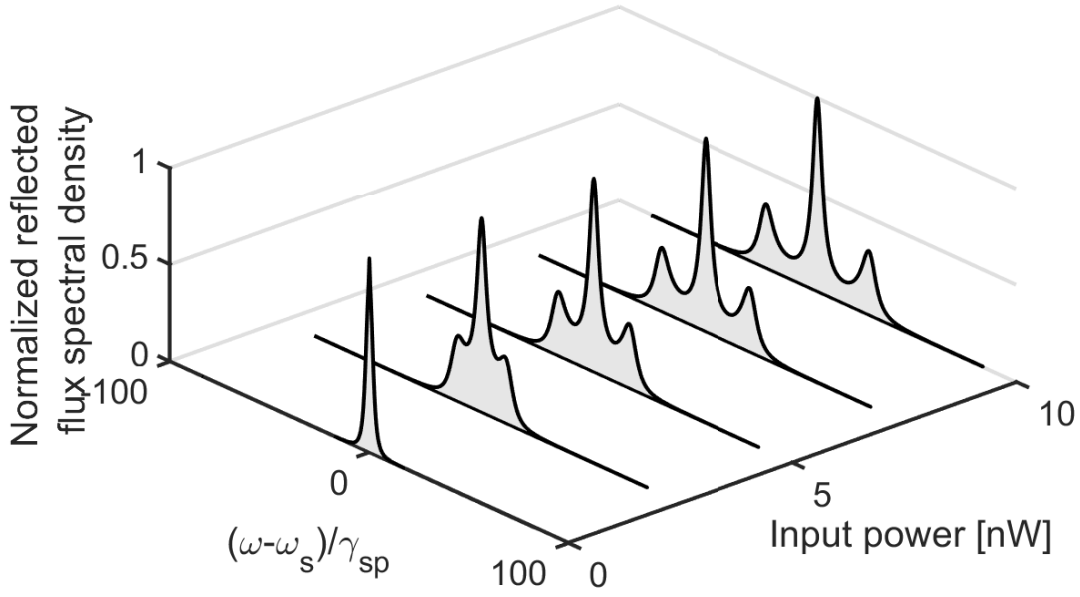


Figure 3.5: Simulated spectral density of the reflected flux in the rotating frame, $S_{\mathrm{R}}(\omega)$, for increasing input power. The Lorentzian profile turns into the Mollow triplet at higher powers. Image for section 3.2.1.

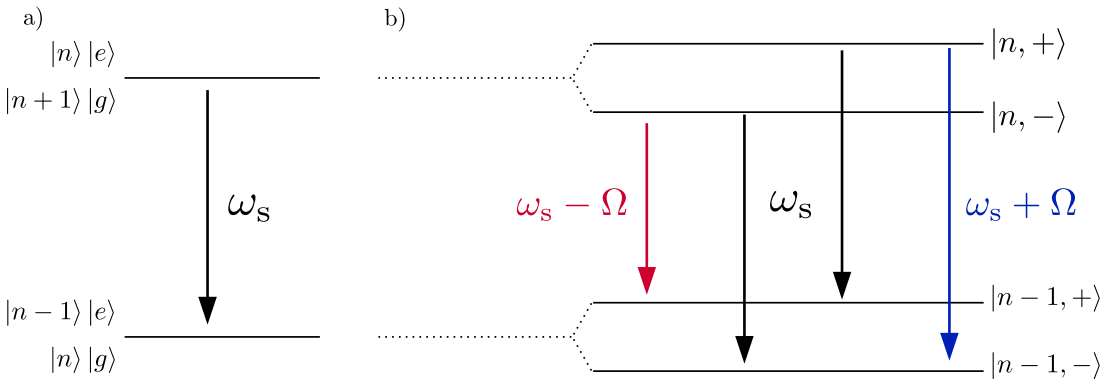


Figure 3.6: (a) Two-level system resonant with the cavity field ω_{s}, weak field and weak coupling regime. (b) Degeneracy splitting due to strong field regime. The four dressed states add two side-bands transitions at $\omega_s \pm \Omega$, in addition to the transition at ω_{s}, leading to the Mollow triplet. Image for section 3.2.1.

resonance fluorescence. In pulsed regime two time evolution must be numerically computed to obtain, e.g. for the reflected field

$$G^{(1)}(t_1, t_2) = \left\langle \hat{b}^\dagger_{\text{out}}(t_2)\hat{b}_{\text{out}}(t_1) \right\rangle , \tag{3.11}$$

which is the first-order correlation evaluated at two different times, t_1 and t_2.
The aim has been to compute the energy spectral density and to check, as in all the simulations, the proper normalization of the spectrum. Assuming a transform-limited[1] Gaussian pulse shape, the electric field can be represented in the time domain and the rotating frame as

$$\mathcal{E}(t) = \exp\left[-\frac{t^2}{2\sigma_t^2}\right], \tag{3.12}$$

where σ_t characterizes the standard deviation of the pulse.
A common tool for time-frequency analysis is the Wigner distribution function (WDF), which is used as transform in time-frequency analysis. It provides the highest possible temporal and frequency resolutions, mathematically limited by the uncertainty principle in quantum wave theory. Information on the single photon wavepacket can be obtained by calculating the associated Wigner-Ville function (WVF, also known as chronocyclic Wigner distribution), which is the quantum analogue of the WDF, the latter being the name chosen in this text. In the rotating frame centered at the average pulse frequency ω_p, its expression [27] for a field mode of annihilation operator $\hat{s}$ is:

$$W(t,\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \left\langle s^\dagger\left(t+\frac{\tau}{2}\right) s\left(t-\frac{\tau}{2}\right)\right\rangle \mathrm{e}^{-\mathrm{i}(\omega-\omega_\text{p})\tau}\,\mathrm{d}\tau\,. \tag{3.13}$$

Indeed, it is the Fourier transform of the electric field correlations as a function of time. In the following only the reflected field will be shown. Due to the projection property of the WDF function, the following holds:

$$\int_{-\infty}^{+\infty} W(t,\omega)\,\mathrm{d}\omega = \left\langle \hat{b}^\dagger_{\text{out}}(t)\hat{b}_{\text{out}}(t)\right\rangle \equiv \Phi(t), \tag{3.14a}$$

$$\int_{-\infty}^{+\infty} W(t,\omega)\,\mathrm{d}t = S(\omega). \tag{3.14b}$$

Thus by integrating the WDF over the frequency domain, the original flux is obtained because of the correct normalization of the distribution. Whereas integrating the WDF over time gives the total spectrum of the optical field, see equation (3.14b). Last, by integrating in time and frequency, the total number of photons is recovered. In figure 3.7 the reflected total flux $\Phi(t)$ is shown as a function of the simulation time. The flux has been evaluated first, as in section 3.1, by $\left\langle \hat{b}^\dagger_{\text{out}}(t)\hat{b}_{\text{out}}(t)\right\rangle$. Whereas, the second curve has been computed by integrating the WDF over the frequency domain. Being superimposed to the previous one, it has proven the WDF correct normalization according to equation (3.14a). The input Gaussian pulse has 15 ps FWHM with average photon number equal to one. The first peak

[1] A transform-limited pulse is one that has minimal phase variation over its spectrum and has a minimal time-bandwidth product [26].

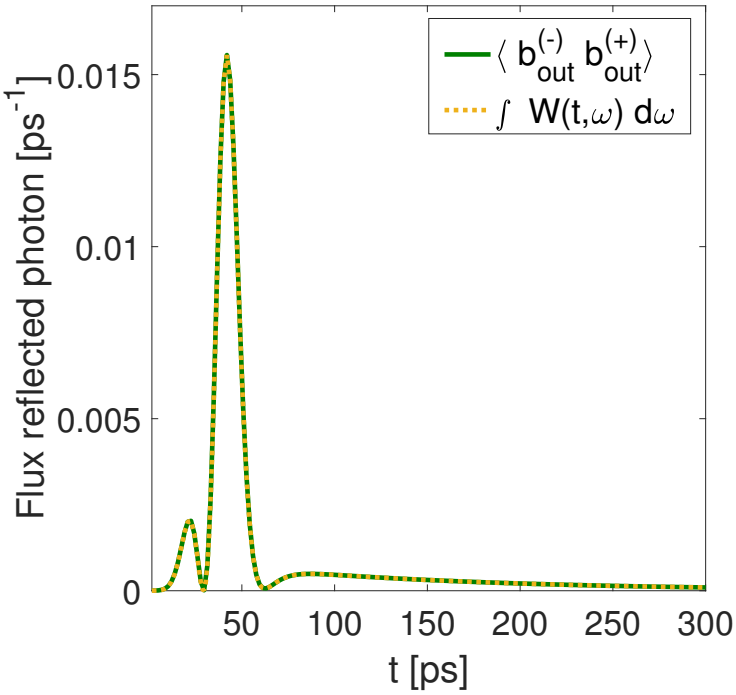


Figure 3.7: Reflected photon flux computed by the output operator and by the Wigner Distribution Function. Differently from figure 3.3a, $\kappa = 200\,\mu\text{eV}$, $N_\text{p} = 1$.

is due to the reflected field from the top mirror, which interferes destructively with the field reflected from the bottom mirror. The decaying tail corresponds to the the quantum emitter emission in the cavity.

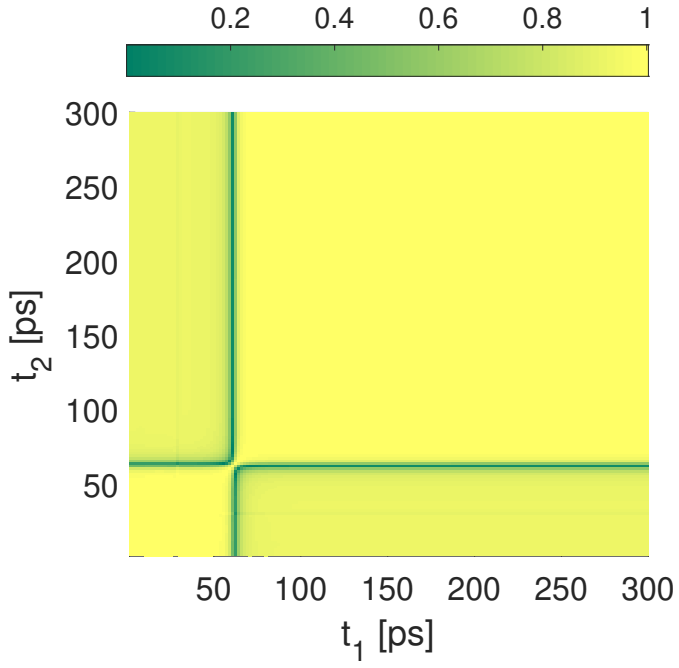


Figure 3.8: Absolute value of the normalized-first order coherence $\left|g^{(1)}(t_1, t_2)\right|$ of the reflected field in pulsed wave regime.

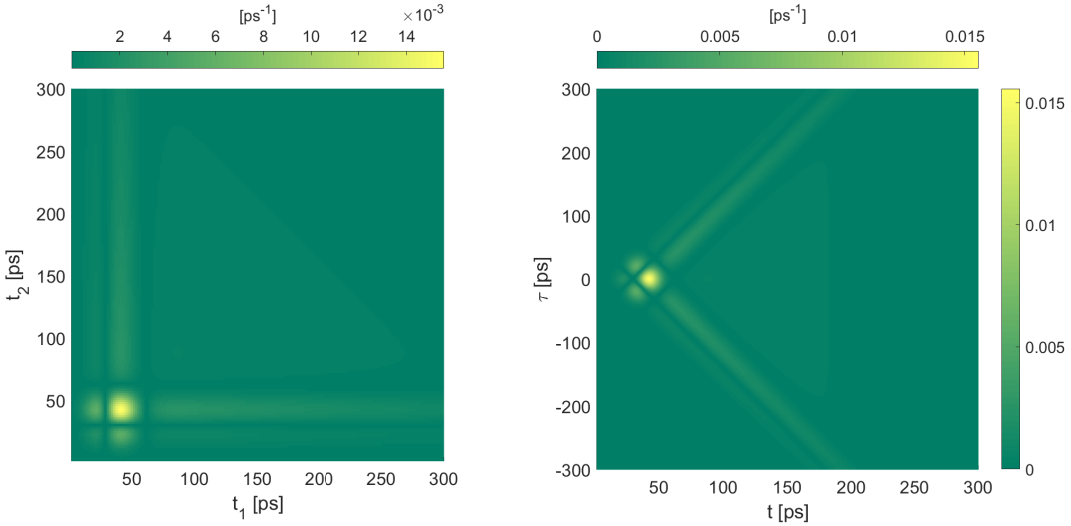
Computing the first-order correlation function Being in pulsed regime, two time evolution were required for simulating $g^{(1)}(t_1, t_2)$, shown in figure 3.8. The mathematical details are contained in Appendix A. There is a transition time around 60 ps before which most of the light is induced by the empty-cavity response, and after which most of the light is induced by the slowly-decaying QD signal. The figure can be analyzed according to different time sets:

- Along the diagonal $t_1 = t_2$, the normalized first order correlation is unity by definition. In the yellow region where both t_1 and t_2 are before the transition time, it regards coherence of the empty-cavity reflection with itself, which comes from the laser input: perfect relative coherence is obtained, being $\left|g^{(1)}(t_1, t_2)\right| \approx 1$. For the yellow region where both t_1 and t_2 are after the transition time, it regards coherence of the QD-emitted signal with itself, which is also near-unity due to the negligible pure dephasing chosen in the simulation.

- When t_1 is before the transition time and t_2 after, or the reverse, $\left|g^{(1)}(t_1, t_2)\right|$ measures the relative coherence of the QD-signal with the directly-reflected laser, showing only partial coherence. This is in accordance with the partial coherence also observed in

figure 3.4 in the CW regime. In general, the QD-emitted light is not fully coherent with the incoming laser.

- The zero coherence lines presumably arise from the specific transition moment where there should be no reflected light at all, due to perfect interference, if the QD signal were entirely coherent. This would happen in the limit of very low power, in addition to no pure dephasing. Since there is some amount of signal at this transition moment, this signal only arises from the incoherent part, and this explains why there is absolutely no coherence at that time.



(a) $\left|G^{(1)}(t_1, t_2)\right|$. (b) Interpolated $\left|G^{(1)}(t, \tau)\right|$.

Figure 3.9: First-order correlation function of the reflected field in pulsed wave excitation.

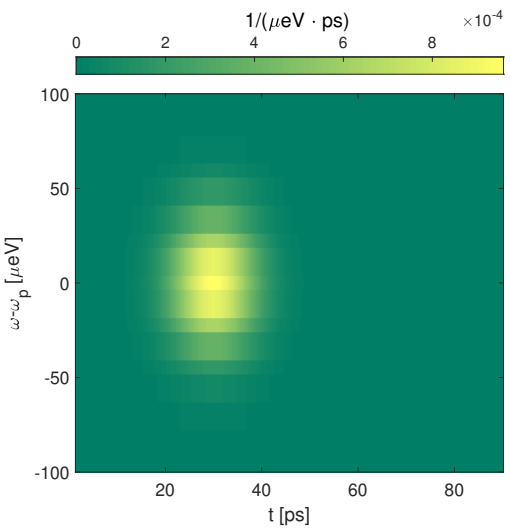
Computing the Wigner Distribution Function In equation (3.13) the WDF is described in terms of time t and delay τ. The latter are related to t_1, t_2 according to the following mapping:

$$\begin{cases} t & = (t_1 + t_2)/2 \\ \tau & = t_2 - t_1. \end{cases}$$

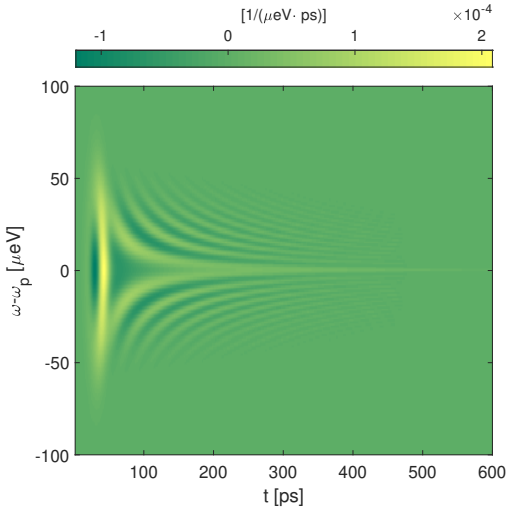
The first-order correlation function $G^{(1)}(t_1, t_2)$ is shown in figure 3.9a, and the corresponding interpolated function $G^{(1)}(t, \tau)$ is shown in 3.9b, where the highest correlation occurs nearby the maximum peak at about 50 ps of the reflected flux in figure 3.7. Comments on the interpolation procedure, and on the computation of the Wigner Distribution Function using the Fast Fourier Transform (FFT) are discussed in Appendix A. Indeed, a new WDF algorithm has been developed independently, because the associated MATLAB® routine (available from R2018b) is designed for single-time series and not correlation functions.

The WDF function, real because of the time-reversal symmetry of the first-order correlation functions, is presented in figure 3.10b. As a function of time, the spectral content is centered at the average pulse frequency. The negative values assumed by the WDF are a consequence

of the so called *Heisenberg–Gabor limit*, which states that one cannot simultaneously sharply localize a signal in both time ad frequency domain. Thus, an averaging along a time or frequency window is necessary to deduce a physical and positive quantity. The WDF of a Gaussian is still a bell-shaped surface which represents the pulse in the time-frequency phase space, see figure 3.10a. In the simulated WDF with the QD-cavity, the effect of the optical cavity and the QD spontaneous emission can be appreciated. The incident Gaussian pulse is divided into a reflected pulse in phase space (weak yellow) and interference fringes because of the many emission sources: the light directly reflected without entering, the light extracted after entering the cavity (but without having interacted with the QD), and finally, the light emitted due to QD decay. In figure 3.11 it is shown the spectrum $S(\omega)$



(a) The WDF of a Gaussian is still a bell-shaped surface. Result obtained from the reflected field by setting $\eta_{\text{top}} \approx 0$.



(b) WDF of the reflected field interacting with the QD-cavity. Interference fringes are due to the multiple light sources present, as explained in the text.

Figure 3.10: Wigner Distribution Functions in time-frequency axes given an input Gaussian pulse.

obtained by integrating along time the WDF. The total spectrum is given by the sum of a coherent and incoherent contribution, with the quantum emitter responsible for the latter.

3.3 Second-order Coherence

The first-order coherence correlation functions do not provide information on the photon statistics, which is contained in the second-order quantum correlation function (at fixed position)

$$g^{(2)}(t,\tau) = \frac{G^{(2)}(t,\tau)}{G^{(1)}(t,t)G^{(1)}(t+\tau,t+\tau)}, \tag{3.15}$$

where

$$G^{(2)}(t,\tau) = \text{tr}\Big(\hat{\rho}\hat{E}^{(-)}(t)\hat{E}^{(-)}(t+\tau)\hat{E}^{(+)}(t+\tau)\hat{E}^{(+)}(t)\Big). \tag{3.16}$$

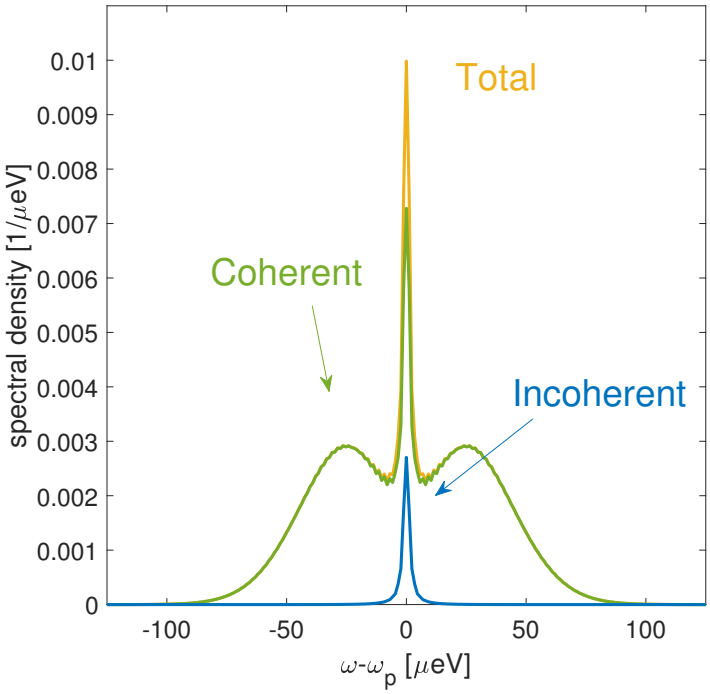
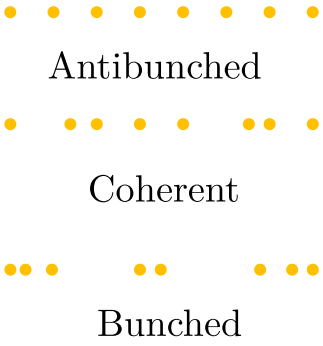
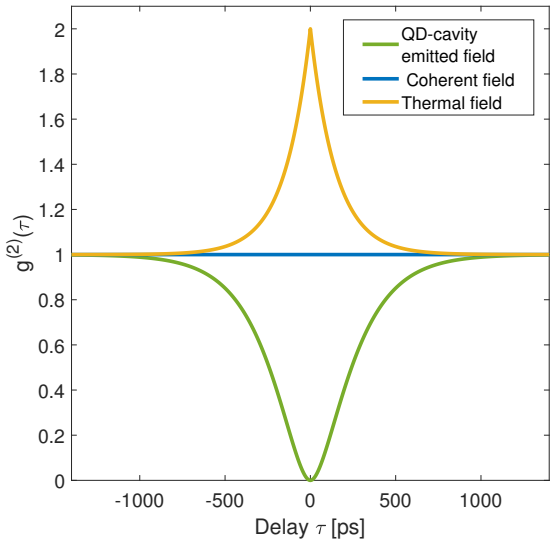


Figure 3.11: Reflected flux spectral density obtained from the Wigner Distribution Function by equation (3.14b). Image discussed in section 3.2.2.

A quantum field is said to be second order coherent if $g^{(2)}(t, t+\tau) = 1$. Equation (3.15) is interpreted as the conditional probability to detect a photon at time $t+\tau$ knowing that a first photon has already been detected at time t, divided by the unconditioned probability. For a coherent multimode state, it can be shown that $g^{(2)}(\tau) = 1$ and thus the photon stream obeys a Poisson distribution. In the case of $g^{(2)}(0) < g^{(2)}(\tau)$, called photon antibunching, photons tend to impinge evenly spaced in time and it represents a non-classical state, figure 3.12a. In figure 3.12b the simulated $g^{(2)}(\tau)$ of the quantum dot spontaneous emission in the cavity is compared with a multimode coherent field and thermal state. Being $g^{(2)}(0) \approx 0$, this represents a non-classical light source as mentioned at the beginning of Chapter 2. To reduce numerical errors in the simulation, it has been



(a) Comparison of the photon streams for antibunched light, coherent light, and bunched light. For the case of coherent light, the Poissonian photon statistics correspond to random time intervals between the photons.



(b) The $g^{(2)}(\tau)$ of the QD emitted field is compared with a thermal (bunched) and a multimode coherent field. $\kappa = 200\,\mu\text{eV}$ and $P_{\text{in}} = 10\,\text{pW}$.

Figure 3.12: Photon statistics and simulated second-order correlation for the QD spontaneously emitted field.

decided to evaluate the second-order quantum coherence $g^{(2)}(t_2, t_1)$ in terms of a conditional density matrix. The field operator for spontaneous emission is given by $\hat{e} = \sqrt{\gamma_{\text{sp}}}\hat{\sigma}_-$, which projects the excited state $|e\rangle$ into the ground state $|g\rangle$. To physically interpret the results and to get a better numerical convergence, it is computed the system's density matrix just after a detection event at time t. This new *conditional density matrix* $\hat{\rho}'(t_1)$ is given by

$$\hat{\rho}'(t_1) = \frac{\hat{e}\,\hat{\rho}(t_1)\hat{e}^\dagger}{\text{tr}(\hat{\rho}(t_1)\,\hat{e}^\dagger\,\hat{e})}. \tag{3.17}$$

This is a valid density matrix with unit trace, verified by taking the trace in the previous equation and by exploiting the cyclic permutation property. From this density matrix at time $t_1^{(+)}$, right after the first detection at t_1, one deduces the conditional density

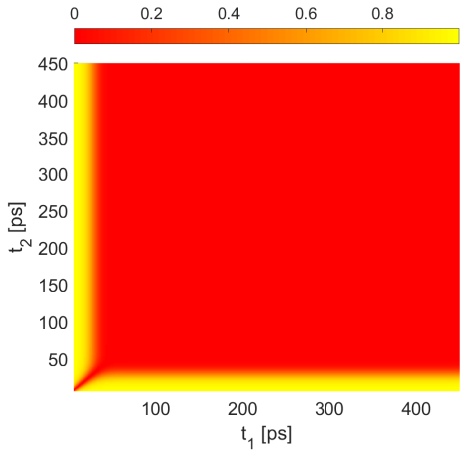
matrix at time t_2, leading to a density matrix $\hat{\rho}(t_2 \mid t_1) \equiv \hat{\rho}(t_2, \text{conditioned to a click at } t_1)$. Therefore, the quantity

$$\mathrm{tr}\Big(\hat{e}^\dagger \hat{e} \hat{\rho}(t_2 \mid t_1)\Big) = \frac{\langle \hat{e}^\dagger(t_1)\hat{e}^\dagger(t_2)\hat{e}(t_2)\hat{e}(t_1)\rangle}{\langle \hat{e}^\dagger(t_1)\hat{e}(t_1)\rangle}. \tag{3.18}$$

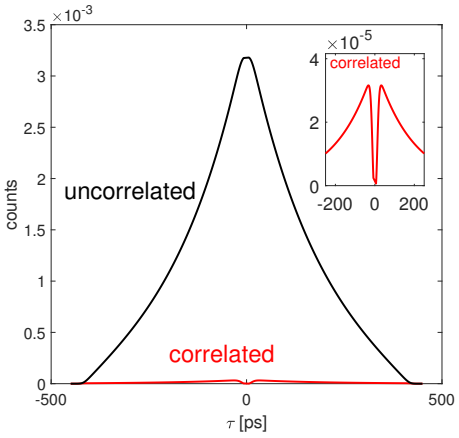
Last, the second term in the denominator of equation (3.15) is given by

$$\Big\langle \hat{e}^\dagger(t_2)\hat{e}(t_2)\Big\rangle = \mathrm{tr}\Big(\hat{e}^\dagger \hat{e}\, \hat{\rho}(t_2)\Big). \tag{3.19}$$

So it is found that the normalized correlation function $g^{(2)}(t_1, t_2)$ is indeed the ratio between two quantities: the photon flux at time t_2, conditioned by a previous photon detection event at time t_1, and the photon flux at time t_2, unconditioned. This is the equivalent experimental definition of the second-order correlation function.
In the CW regime, the second-order quantum coherence function is only a function of the delay, $g^{(2)}(\tau)$, which simplifies the simulation of figure (3.12b) to just one-time evolution.
On the other hand, for PW excitation, the $g^{(2)}(t_1, t_2)$ for the QD emitted light is shown in



(a) $g^{(2)}(t_1, t_2)$ of the quantum dot emitted light.



(b) Correlated and uncorrelated coincidences of the quantum dot emitted light, $\kappa = 400\,\mu\text{eV}$. Inset is the correlated count, centered at zero delay.

Figure 3.13: Second order correlation function, correlated and uncorrelated coincidences in pulsed wave excitation.

figure 3.13a. Along the diagonal, so for zero-delay time, $g^{(2)}(t, t) = 0$ as expected: the QD cannot emit a second photon immediately after having emitted a first one, since it has been projected to the ground state. After the pulse arrival time, there is a "dead time" before the laser field drives the electron back into the excited state at which moment another photon may be emitted (yellow region). For longer times, the pulse is mostly extinguished and the conditional probability is null (red region).
Note that in the $g^{(2)}(t_1, t_2)$ the denominator $\langle \hat{e}^\dagger(t_1)\hat{e}(t_1)\rangle \langle \hat{e}^\dagger(t_2)\hat{e}(t_2)\rangle$ represents the rate of *uncorrelated coincidences* corresponding to the expected coincidence rate for photons from

independent sources (or, as usually done in the laboratory, by independent excitation pulses from the same source). In comparison, the numerator $\langle \hat{e}^\dagger(t_1)\hat{e}^\dagger(t_2)\hat{e}(t_2)\hat{e}(t_1)\rangle$ corresponds to the rate of correlated coincidences, that is the expected coincidences rate for photons from the same pulse.
In a HBT experiment in pulsed regime, $g^{(2)}(\tau)$ is obtained by an histogram integrating all the coincidences corresponding to a given delay $\tau = t_2 - t_1$. Here the normalization is chosen such that the area of the $g^{(2)}(\tau)$ peak is unity for uncorrelated coincidences, figure B.4a in Appendix B. The correlated coincidences, figure B.4b, lead to a zero-delay peak corresponding to photons emitted during the same pulse. Both the uncorrelated and correlated coincidences are shown as a function of the delay τ in figure 3.13b, whose experimental counterpart is represented in Appendix B, figure B.5. The purity of the source is linked to the ratio between the correlated and uncorrelated areas. Here the area for the correlated coincidences is almost negligible (inset in figure 3.13b), i.e. negligible probability to have a second photon emitted during the same pulse compared to the correlated case. Thus, this result confirms the very good single-photon purity of the simulated quantum emitter.

Conclusions

In this chapter the custom Cavity QED Quantum Toolbox structure has been presented, showing some quantum optics simulations results. It has been addressed the quantum emitter optical non-linearity which modifies the reflectivity spectra, showing the 2-level atom saturation at higher input power. From the spectrum of the first-order correlation functions in the continuous-wave regime, the Mollow triplet phenomenon has been reproduced as the driving Rabi frequency becomes much larger than the exciton spontaneous emission rate. To investigate the first-order coherence function in the pulsed regime, a custom algorithm for evaluating the Wigner Distribution Function has been presented with the associated results. Last, the single-photon emission property of the artificial atom has been proved by the second-order correlation function, both in continuous and pulsed wave excitation. In the next chapter, the adiabatic elimination of the cavity mode Fock space is discussed, highlighting its advantages for the computational cost and some results, in comparison with the “full” model used in this Chapter.

Chapter 4

Adiabatic Elimination

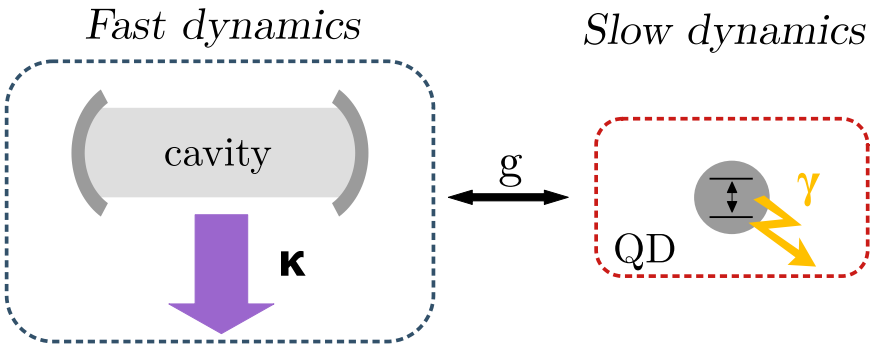


Figure 4.1: A fast sub-system, the cavity, is coupled to the slow sub-system, the neutral QD. The cavity acts as a perturbation and it is adiabatically eliminated.

Adiabatic elimination is an approximation that produces an effective Hamiltonian for the sub-space of interest and it is valid when the cavity damping rate κ is much higher than any other evolution rate: $\kappa \gg g$, γ_{sp}, γ^*. This is the so-called *bad cavity* regime and these conditions are always fulfilled for the simulations in this Chapter. As presented in figure 4.1, the cavity is subjected to high losses and its dynamics are much faster than the QD ones. If not stated otherwise, the QD-Cavity system has:

- $\kappa \approx 400\,\mu\text{eV}$, leading to a photon lifetime in the cavity of about 1.6 ps. The decay time comes from the Lorentzian cavity spectrum, whose Fourier transform is an exponential decay. The decay rate is $\kappa/(2\hbar)$ for the field amplitude, whereas $\kappa/\hbar$ for the intensity, i.e. a photon lifetime $\hbar/\kappa$. Since $\hbar = 6.582 \cdot 10^{-4}\,\text{eV} \cdot \text{ps}$, it explains why κ is denoted alternatively as a cavity linewidth or, by considering $\hbar \equiv 1$ units, as a decay rate.

- $g \approx 17\,\mu\text{eV}$, corresponding to 120 ps timescale. Notice that g is related to the period of the vacuum Rabi oscillations illustrated in figure 3.3b. From equation (2.17), the Rabi angular frequency is $2g/\hbar$, with $2g$ the vacuum Rabi splitting between the two first excited eigenstates of the Jaynes-Cummings model.

- $\gamma_{\text{sp}} \approx 0.6\,\mu\text{eV}$, about 1 ns of spontaneous emission time in other modes than the cavity mode, with negligible pure dephasing.

In such a case, the photons emitted by the QD in the cavity mode are not stored in the cavity: they quickly escape it. From the QD point of view, there is no difference between the spontaneous emission in the other modes, at the rate γ_{sp}, and the emission in the cavity mode followed by photon escape, at the rate Γ_{m}. It is reminded that the ratio between the latter and the former defines the Purcell factor, equation (2.10). Two important parameters for the rate of emitted photons in the cavity mode (Γ_{m}) are:

- the normalized QD-Cavity detuning:

$$\Delta_{\text{QD-C}} \equiv \frac{2(\omega_{\text{d}} - \omega_{\text{c}})}{\kappa}. \tag{4.1}$$

- the Purcell-enhanced emission rate at zero detuning:

$$\Gamma_0 = \frac{4g^2}{\kappa}. \tag{4.2}$$

One can show that, with these notations and for a 2-level system in a cavity, the following Lorentzian dependence with the detuning is obtained for the Purcell-enhanced emission rate in the cavity mode, Γ_{m}:

$$\Gamma_{\text{m}} = \frac{\Gamma_0}{1 + \Delta^2_{\text{QD-C}}}. \tag{4.3}$$

With the above given values, at zero detuning it is found that $\Gamma_{\text{m}} = \Gamma_0 = 2.9\,\mu\text{eV}$, thus $\Gamma_{\text{tot}} = \Gamma_{\text{m}} + \gamma_{\text{sp}} = 3.5\,\mu\text{eV}$. This value is also much lower than κ and corresponds to a total emission time of 180 ps. The coupling factor, proportional to the brightness, is $\beta = 2.9\,\mu\text{eV}/3.5\,\mu\text{eV} \approx 83\%$ at zero detuning.
Before discussing how the adiabatic model is obtained, in the following table the computation time of the full model is compared with the adiabatic version. As the dimension of the cavity mode Fock space increases, which is required for higher input powers, also the simulations take more time. On the contrary, for the adiabatic model the Fock space has been neglected and taken into account by an effective Hamiltonian. In the following table simulation times for a CW and PW program are shown to illustrate the computational cost reduction of the adiabatic approximation. The average times have been computed over ten simulations for each program, without plotting.

Program	Full model, $N = \dim(\mathcal{H}_{\text{cav}})$			Adiabatic model
	$N = 10$ (s)	$N = 15$ (s)	$N = 20.$ (s)	(s)
Reflectivity spectrum in CW	6.9 ± 0.3	11 ± 2	18 ± 1	4.4 ± 0.3
Photon flux evolution and QD occupation in PW	9.6 ± 0.1	16.6 ± 0.5	33 ± 1	1.13 ± 0.05

4.1 Approximations from the "Full Model"

In the general case, the exact calculation starts from the standard CQED Hamiltonian for a CQED system coherently excited by an input field b_{in}, presented in section 2.4. The full Hamiltonian acts both on the QD and the cavity subspaces: the goal of adiabatic elimination is to work *only* within the QD Hilbert space.
In the *fast*, or *bad*, cavity regime, the point is that the variables associated to the cavity mode have no memory of the past. At a given time t they simply adapt to the values of the other important quantities at that time, in particular $b_{\text{in}}(t)$ and $\langle\hat{\sigma}_-\rangle(t)$. Following a similar derivation from [28, sec. 13.2.1], one can use the Heisenberg's representation for operators $\hat{a}(t)$, $\hat{b}_{in}(t)$, $\hat{\sigma}_-(t)$ and write in the rotating frame, as Loïc Lanco has done:

$$\begin{aligned}\dot{\hat{a}}(t) = &\left[-\frac{\kappa}{2} - \mathrm{i}\,(\omega_{\text{c}} - \omega)\right]\hat{a}(t) - g\,\hat{\sigma}_-(t) - \sqrt{\kappa_{\text{top}}}\,\hat{b}_{\text{in}}(t) \\ &+ \textit{noise operator} \text{ averaging to zero.}\end{aligned} \tag{4.4}$$

Finally, integrating this equation gives:

$$\begin{aligned}\hat{a}(t) = &\,\hat{a}(0)\exp\left\{-\left[\frac{\kappa}{2} + \mathrm{i}\,(\omega_c - \omega)\right]t\right\} \\ &- g\int_0^t dt'\,\hat{\sigma}_-(t-t')\exp\left\{-\left[\frac{\kappa}{2} + \mathrm{i}\,(\omega_{\text{c}} - \omega)\right]t'\right\} \\ &- \sqrt{\kappa_{\text{top}}}\int_0^t dt'\,\hat{b}_{\text{in}}(t-t')\exp\left\{-\left[\frac{\kappa}{2} + \mathrm{i}\,(\omega_{\text{c}} - \omega)\right]t'\right\},\end{aligned} \tag{4.5}$$

where:

- the first (red) term quickly disappears after a fast transient regime;
- the other (blue) terms are almost a Dirac $\delta(t')$ function, up to a coefficient, since κ implies fast decays. Thus, the past times $t - t'$ play no role except for very small values of t'.

Since one can focus on times t' close to zero, at first order

$$\hat{b}_{\text{in}}(t-t') \approx \hat{b}_{\text{in}}(t) \tag{4.6a}$$

$$\hat{\sigma}_-(t-t') \approx \hat{\sigma}_-(t)e^{\mathrm{i}(\omega_{\text{d}}-\omega)t'}, \tag{4.6b}$$

where the exponential arises since the QD is detuned from the laser frequency. The simplified integrals over t' lead to

$$\int_0^\infty \mathrm{d}t'\exp\left\{-\left[\frac{\kappa}{2} + \mathrm{i}\,(\omega_{\text{c}} - \omega_{\text{d}})\right]t'\right\} = \frac{1}{\frac{\kappa}{2} + \mathrm{i}\,(\omega_{\text{c}} - \omega_{\text{d}})} = \frac{1}{\frac{\kappa}{2}\,(1 - \mathrm{i}\Delta_{\text{QD-C}})}, \tag{4.7a}$$

$$\int_0^\infty \mathrm{d}t'\exp\left\{-\left[\frac{\kappa}{2} + \mathrm{i}\,(\omega_{\text{c}} - \omega)\right]t'\right\} = \frac{1}{\frac{\kappa}{2} + \mathrm{i}\,(\omega_{\text{c}} - \omega)} = \frac{1}{\frac{\kappa}{2}\,(1 - \mathrm{i}\Delta)}, \tag{4.7b}$$

where

$$\Delta \equiv \frac{2(\omega - \omega_{\text{c}})}{\kappa}. \tag{4.8}$$

Disregarding the noise term, one obtains:

$$\hat{a}(t) = -\frac{g\,\hat{\sigma}_-(t)}{\frac{\kappa}{2}\,(1 - \mathrm{i}\Delta_{\text{QD-C}})} - \frac{\sqrt{\kappa_{\text{top}}}\,\hat{b}_{\text{in}}(t)}{\frac{\kappa}{2}\,(1 - \mathrm{i}\Delta)}. \tag{4.9}$$

This also gives the correct equation when one comes back to the Schrödinger's representation:

$$\hat{a} = -\frac{g\,\hat{\sigma}_-}{\frac{\kappa}{2}\,(1 - \mathrm{i}\Delta_{\text{QD-C}})} - \frac{\sqrt{\kappa_{\text{top}}}\,\hat{b}_{\text{in}}}{\frac{\kappa}{2}\,(1 - \mathrm{i}\Delta)}. \tag{4.10}$$

It is recalled that when the input field arises from a coherent (laser) light source, one can replace $\hat{b}_{\text{in}}$ by $b_{\text{in}} \cdot \hat{I}$. Thus, given this last assumption, the quantum fluctuations of the annihilation operator are due to the quantum fluctuations of the QD lowering operator. Note that the adiabatic elimination corresponds to a completely different point of view since now the operator $\hat{a}$ does not act on the cavity subspace, that is not considered, but is viewed only through its effect on the QD subspace, as given by equation (4.10).

4.2 Adiabatic Hamiltonian and Output Operators

In the previous section the annihilation operator has been expressed in terms of the lowering operator and the coherent input field. Equation (4.10) is significant since it states that the cavity field adapts instantaneously to its two sources: the input field and the Purcell-enhanced emission from the QD into the cavity mode. The cavity forgets any previous value it had. From this equation a new input-output equation is deduced

$$\hat{b}_{\text{out}} = \hat{b}_{\text{in}} + \sqrt{\kappa_{\text{top}}}\,\hat{a} \tag{4.11}$$

$$\downarrow$$

$$\hat{b}_{\text{out}} = b_{\text{in}} \left(1 - \frac{2\,\eta_{\text{top}}}{1 - \mathrm{i}\Delta}\right) \hat{I} - \frac{\sqrt{\Gamma_0\,\eta_{\text{top}}}}{1 - \mathrm{i}\Delta_{\text{QD-C}}}\,\hat{\sigma}_-, \tag{4.12}$$

being $\hat{I}$ the identity operator. Note that equation (4.12) has a direct and fundamental interpretation, where the reflected optical field is directly viewed as the sum of two contributions. The first one, proportional to b_{in}, is the optical field reflected by the empty cavity alone, as would be obtained in the absence of QD. The second term, proportional to $\hat{\sigma}_-$, is the optical field induced by a photon emitted in the cavity mode by the QD, and then quickly extracted out from the cavity through the top mirror. Also note that the operators $\hat{I}$ and $\hat{\sigma}_-$ in equation (4.12) are directly related to the effect that the detection of a reflected photon can have on the QD subspace. If the detected photon comes from the empty-cavity reflection, it does not change the QD state (operator $\hat{I}$) since this photon was not emitted by the QD. If the detected photon comes from the QD emission, however, this means that the QD has just decayed to the ground state (operator $\hat{\sigma}_-$). The superposition of these two fields in equation (4.12) is also a direct illustration that they can interfere, as already discussed in the interpretation of some of the results obtained in Chapter 3.
At last, one can limit the Hamiltonian to the part acting on the QD subspace, using the above equation for $\hat{a}$, to obtain a simplified, adiabatic one

$$\hat{H}_{\text{ad}} = (\omega_{\text{eff}} - \omega)\;\hat{\sigma}_+\hat{\sigma}_- - \mathrm{i}\sqrt{\Gamma_0\,\eta_{\text{top}}} \left(\frac{b_{\text{in}}}{1 - \mathrm{i}\Delta}\,\hat{\sigma}_+ - \frac{b_{\text{in}}^*}{1 + \mathrm{i}\Delta}\,\hat{\sigma}_-\right), \tag{4.13}$$

with

$$\omega_{\text{eff}} = \omega_{\text{d}} + \frac{\Gamma_0\,\Delta_{\text{QD-C}}}{2\left(1+\Delta^2_{\text{QD-C}}\right)}. \tag{4.14}$$

The difference $\omega_{\text{eff}} - \omega_d$ is a cavity-induced frequency shift [28]. It is proportional to Γ_{m} and to the detuning $\Delta_{\text{QD-C}}$. The second term is analogous to the Rabi Hamiltonian for an atom in free space, with the proper amplitude and phase factor describing how the input field excites the QD.
Finally, replacing $\hat{a}$ in the optical Bloch equations an obtaining the time derivatives $\langle\dot{\hat{\sigma}}_-\rangle$, $\langle\dot{\hat{\sigma}}_z\rangle$, it is found that they are consistent with the adiabatic Hamiltonian providing that one modifies the collapse operator for the quantum dot spontaneous emission in the leaky modes

$$\hat{C}_{\text{QD}} = \sqrt{\gamma_{\text{sp}}}\,\hat{\sigma}_- \longrightarrow \hat{C}_{\text{QD}} = \sqrt{\Gamma_{\text{tot}}}\,\hat{\sigma}_- \quad . \tag{4.15}$$

As before, $\Gamma_{\text{tot}} = \Gamma_{\text{m}} + \gamma_{\text{sp}}$, so it takes into account QD decay both due to Purcell-enhanced emission via the cavity mode and emission in the leaky modes. In practice, this means that regarding the QD subspace all the dynamics is described by the Hamiltonian in equation (4.13) and the collapse operator in equation (4.15), provided that the adiabatic approximation holds.

4.3 Comparing Adiabatic and Full Model

During this internship, all the programs of the cavity-QED quantum toolbox have been developed with two available options: "full model" or "adiabatic model". It has been verified each time that the higher the cavity damping rate κ, the quicker the cavity adapts to its environment (i.e. the QD and the external field), and the better the two models coincide. However, the inequality $\kappa \gg g$, γ_{sp}, γ^* should *not* be considered as a sufficient condition for the adiabatic elimination to be valid. What is important is the possibility to approximate as in (4.6), when t' is of the order of the photon lifetime in the cavity.
Out of conciseness, it has been chosen to only focus in the following two examples, highlighting when these approximations may or may not be valid.

4.3.1 Mollow triplets

In the CW regime, the constant input field obviously verifies $b_{in}(t-t') = b_{in}(t)$, but the approximation $\hat{\sigma}_-(t-t') \approx \hat{\sigma}_-(t)e^{\text{i}(\omega_{\text{d}}-\omega)t'}$ can fail at high input power. This could lead to discrepancies in the predictions for the first-order coherence $g^{(1)}(\tau)$, and thus also for the predicted spectral densities. In section 3.2.1 it has been shown that under resonance fluorescence, for higher input powers the degeneracy splitting of the uncoupled system eigenstates results in the so-called Mollow triplets spectrum. In this specific simulation, adiabatic elimination is very promising because higher input powers require a larger dimension N of the cavity truncated Fock space. Indeed, by adiabatic elimination, an element of the Fock-Liouville space previously defined has always four components instead of $4 \times N^2$.
In CW regime, given equation (2.17) and assuming zero detuning, all Fock states contribute with the same Rabi frequency $\Omega_{\text{R}} \approx 2g\sqrt{\langle\hat{n}\rangle}$, where $\hat{n} = \hat{a}^\dagger\hat{a}$ is the photon number operator

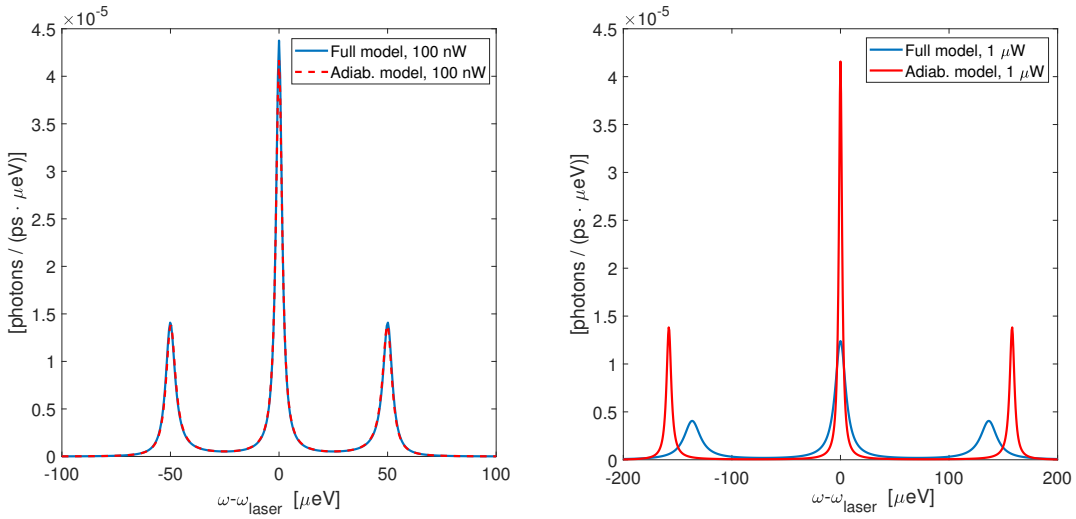
for the optical mode of interest. By definition, this Rabi frequency is the rate at which the QD state oscillates in the Bloch sphere [16], which is also the rate at which the QD lowering operator is evolving. This means that, for the approximation $\hat{\sigma}_-(t-t') \approx \hat{\sigma}_-(t)e^{\mathrm{i}(\omega_\mathrm{d}-\omega)t'}$ to be valid when t' is of the order of the cavity photon lifetime, it is required that $\Omega_\mathrm{R} \ll \kappa$ for the effective Hamiltonian to be accurate enough. In the semi-classical approximation, which is a very good assumption for higher optical input powers, the average photon number from equation (4.10) is given by

$$n \equiv \langle \hat{n} \rangle = \frac{4\eta_\mathrm{top}}{\kappa} b_\mathrm{in}^2. \tag{4.16}$$

Since the aim is to obtain an order of magnitude estimate for the adiabatic approximation to be valid, from here on the semi-classical approximation will be taken for granted. Substituting equation (4.16) into the previous inequality regarding the Rabi frequency, and considering that by definition $b_\mathrm{in} = \sqrt{\eta_\mathrm{top} P_\mathrm{in} / (\hbar\omega_\mathrm{c})}$, it is found that:

$$P_\mathrm{in} \ll \frac{\hbar\omega_\mathrm{c}\kappa^2}{8\eta_\mathrm{top}^2\, g}. \tag{4.17}$$

Given the usual parameters of this chapter, the input power should be much lower than few



(a) $P_\mathrm{in} = 100\,\mathrm{nW}$. (b) $P_\mathrm{in} = 1\,\mu\mathrm{W}$.

Figure 4.2: Comparing Mollow triplets spectra of the spontaneously emitted field, obtained with adiabatic and full model, for different input powers.

micro-watts. In figure 4.2a the spontaneous emitted spectrum under resonance fluorescence is shown for 100 nW input power, in figure 4.2b at 1 µW. In the former case the adiabatic model curve is superimposed onto the one obtained by the full model. In the second case, in accordance to the newly found rule of thumb, the effective Hamiltonian fails, as the side-peaks distance becomes comparable with the cavity damping rate.

A related phenomenon that the adiabatic model is not able to reproduce is when the QD frequency is detuned from the cavity, the expected asymmetry of the Mollow triplet arises,

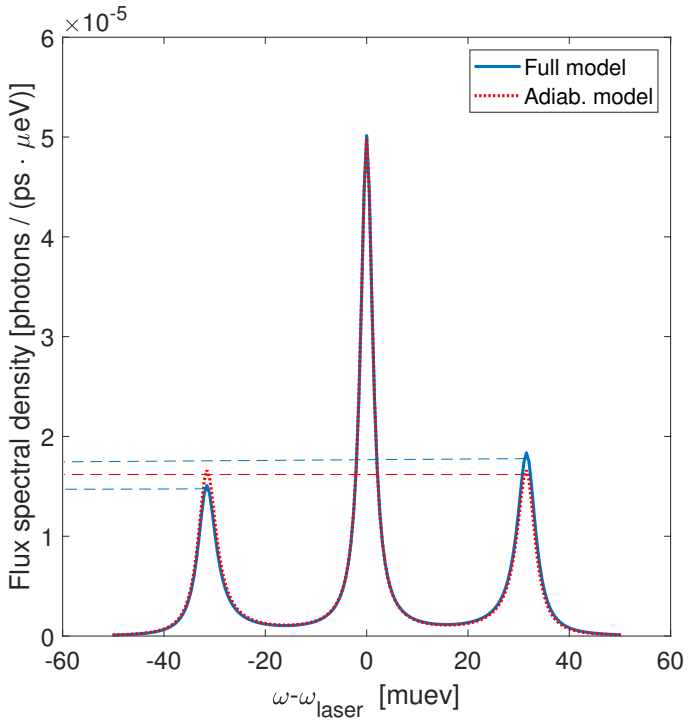


Figure 4.3: Mollow triplets of the emitted field spectrum, for $P_{\text{in}} = 50\,\text{nW}$, detuning between the cavity and the QD, $\omega_{\text{d}} - \omega_{\text{c}} = 100\,\mu\text{eV}$ and laser at resonance with the QD. The expected asymmetry in the full model is not recovered in the adiabatic model.

figure 4.3. This is because one side peak emits more in the mode than the other side peak. Indeed, in this case, one side peak has a significantly different normalized detuning with the cavity, compared to the other side-peak that is significantly closer or further away from the cavity mode resonance. This leads to an asymmetry in the values of Γ_m (Purcell-enchanced emission rate in the cavity mode), and thus an asymmetry in the β factor[1] for the two side peaks. The adiabatic model is intrinsically unable to predict such asymmetry since it considers only one value of Γ_m, and thus one value of β, evaluated at the laser frequency and not at the frequency of the side peaks. Still, at lower power, the adiabatic model retrieves its validity since all the peaks, the central one and its two side-peaks, span a small frequency range compared to the cavity damping rate. In this limit, the side-peaks approximately have the same value of the normalized detuning, hence equal Γ_m and β.

4.3.2 Reflectivity in pulsed wave regime

In the following, the optical responses to two incoming Gaussian pulses are compared. The pulses impinge towards the optical micropillar at different delays and with different $FWHM$, keeping constant the average photon number in each pulse. In pulsed wave regime also the approximation $\hat{b}_{in}(t-t') \approx \hat{b}_{in}(t)$ should be considered for the adiabatic model domain of validity.
For the time derivative in the pulsed regime, the important inequality is formally $\mathrm{d}b_{in}/\mathrm{d}t \ll \kappa/2 \times b_{in}$. Indeed, the question is whether $b_{in}(t-t')$ is close to $b_{in}(t)$ in a time-scale t' such that $\kappa/2 \times t' \approx 1$. And on such a time scale one can say that

$$b_{in}(t-t') \approx b_{in}(t) - \frac{\mathrm{d}b_{in}}{\mathrm{d}t}\frac{2}{\kappa}, \tag{4.18}$$

that reduces to $b_{in}(t)$ given the inequality mentioned above. In this case, for a Gaussian pulse, the maximal value of $b_{in}(t)$ time derivative is of the order of the amplitude of b_{in}, divided by the pulse $FWHM$. The condition becomes

$$\frac{\kappa}{2} \times FWHM \gg 1, \tag{4.19}$$

for a Gaussian pulse, where the photon number does not appear because both the signal and its derivative are proportional to it. In the simulation of figure 4.4 the coupling constant g has been set to zero to not have any QD lowering operator contribution. In the case of the short FWHM of 5 ps the effective Hamiltonian is not able to reproduce the reflected flux of the full model because of the too-short time scale for the system to adapt to the Gaussian pulse. Indeed, in the effective Hamiltonian approximation only one peak is obtained instead of two, and almost half the maximum peak value. By substituting the parameters in equation (4.19), it is found that $FWHM \gg 1.3$ ps for the adiabatic model to be valid. The short pulse does not satisfy the inequality, having a $FWHM$ only about four times longer. While the long pulse is better recovered, there is still some visible delay induced by the fact that in the adiabatic model the cavity instantaneously reflects the incoming field, while in

[1] Fraction of photons emitted in the cavity mode, compared to the fraction emitted outside the mode, introduced in Chapter 2.

the full model it does store it for a short time before letting it be reflected. It is important to outline that the inequality (4.19) has been obtained for a Gaussian pulse. For instance, the cavity would not be able to respond to a square pulse adiabatically.

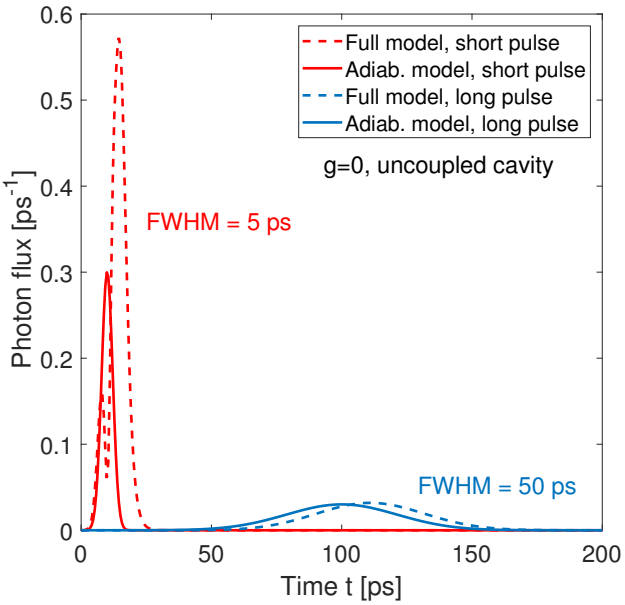


Figure 4.4: Adiabatic and full model comparison of the reflected photon flux in pulsed wave regime, average photon number $N_{\mathrm{p}} = 10$, for different FWHM of the two pulses.

Conclusions

In this chapter, an effective Hamiltonian has been introduced, based on the adiabatic elimination of the cavity mode Hilbert space. This approximation simplifies the computational cost of most of the quantum optics phenomena addressed in the previous chapter, in the weak coupling regime. The quantum fluctuations of the annihilation operator, in the semiclassical approximation for the input field, are related only to the lowering operator. It has been shown that in CW excitation for resonance fluorescence, one condition for the adiabatic elimination validity is that the driving Rabi frequency should be much lower than the cavity damping rate. Moreover, it has been shown that the effective Hamiltonian is not able to reproduce the Mollow triplets asymmetry. The second condition, in PW regime, is that the cavity should be able to follow the input signal time variations.

Chapter 5

Conclusions

In this work, a new custom Cavity QED Quantum Toolbox for a two-level system has been developed to simulate common quantum optics problems, such as non-linear optical properties, first- and second-order correlations, and flux spectral densities for reflected, transmitted, diffracted and emitted fields. While it has been previously established that the Master equation integration based on a pre-existing quantum optics toolbox [23] provides accurate results, given a proper fitting of the experimental parameters, phenomena that require high photon number can be computationally expensive. In this direction, an effective adiabatic Hamiltonian has been implemented to adiabatically eliminate the cavity mode Fock space, and to deal only with the quantum-dot subspace.

The custom cavity QED quantum toolbox took inspiration from pre-existing codes, written by Loïc Lanco, that have been fully revisited. The overall aim has been to find an equilibrium between easiness for future readers and efficiency. An exception is the case of spectral analysis, where the balance has been tipped strongly in favor of efficiency. Indeed, while the fast Fourier transform is much more efficient than the standard definition of the discrete Fourier transform, $\mathcal{O}(N \log N)$ versus $\mathcal{O}(N^2)$, it has required more attention to obtain the proper normalization and phase, in the CW regime as well as in the custom-made Wigner Distribution Function for time-frequency signal analysis.
The adiabatic approximation has shown to be promising in the weak coupling regime, which is the working condition for the deterministic and scalable CQED built at C2N, employed as single-photon source [11] and as an interface between the spin of a single charge and the polarization of a single photon [29]. In the continuous-wave regime, its domain of validity has resulted to be strictly related to the cavity damping rate, responsible for the cavity time scale dynamics. An important parameter has been proven to be the laser input power, that sets the Rabi frequency of the QD sub-system dynamics, which should be much slower than the cavity one for the adiabatic approximation to hold. More studies are required in pulsed-wave excitation for the input pulse shape, because of the limited cavity response time that would prevent the cavity from adapting in time to sudden signal changes.

Among prospects, the most natural one is the generalization to less simple QD systems introducing a charge, spin, and photons polarization, already begun by Clément Millet, Elham Mehdi, and Nathan Coste in the C2N team. Starting from the programs developed

in this work, other physical phenomena such as Mollow triplets in pulsed-wave regime [30] and the quantum Zeno effect [31] could be addressed. Even though the adiabatic elimination has been introduced for an optical micropillar, it would be possible to generalize it in the larger context of dipoles in 1-D photonic crystal waveguides and nanocavities [32]. For future developments, it would be important to translate the scripts from MATLAB® into the open-source `Quantum Toolbox in Python` (QuTiP) package [33], which is used by a larger community and contains several already-implemented routines. Still, however attractive also the idea of having already developed routines in QuTiP, the lower-level programming of this thesis provides useful insights by detailing how each physical quantity is computed, explicitly linking theoretical and experimental results.

Appendix A

Correlations in Time Domain

A.1 First-order Coherence and Quantum Regression Theorem

In section 3.2 the field operator first-order correlation function (3.3) has been introduced to evaluate the frequency dependence of the scattered radiation. The *quantum regression theorem* simplifies the problem of evaluating a two-time correlation function by Lindbladian evolution of a fictitious density matrix. Dropping the hat symbol, here $A(t)$ and $B(t)$ are two system operators (i.e. they do not act on the reservoir coupled to the system) and the system density matrix satisfies the Lindbland master equation $\partial_t \rho(t) = \mathcal{L}\rho(t)$. In the Heisenberg representation, the quantum regression theorem states that in the long time limit [25]

$$\lim_{t\to\infty} \langle A(t)B(t+\tau)\rangle = \mathrm{tr}\left(B\,\Lambda(\tau)\right), \tag{A.1}$$

where $\Lambda(t+\tau, t)$ acts as a sort of "fictitious" density matrix with respect to the B operator. As the system density matrix evolves in time, the newly introduced operator evolves along the delay τ according to

$$\frac{\mathrm{d}\Lambda}{\mathrm{d}t} = \mathcal{L}\Lambda(\tau), \tag{A.2}$$

with initial condition

$$\Lambda(0) = \rho(t\to\infty)\,A. \tag{A.3}$$

Here $\rho(t\to\infty)$ is the density matrix which is obtained in stationary regime, where it is possible to simplify the two-time correlation function as a single-time average. The quantum regression theorem can be applied for the pulsed wave case, but a two-time evolution is then required.

Two-time evolution for pulsed-wave excitation To calculate the first correlation function from equation (3.5), in Heisenberg representation, a general two-time representation is $\langle A(t_1)B(t_2)\rangle$, where for instance $A(t) = U^\dagger(t,0)A\,U(t,0)$. Here $U(t,0)$ is the unitary time-evolution operator from 0 to t. By exploiting the *composition* and *inversion properties* of

the evolution operator

$$U(t,t')U(t',t'') = U(t,t'') \tag{A.4}$$
$$U^{\dagger}(t,t') = U(t',t), \tag{A.5}$$

it is found, by calling ρ the density matrix of the system

$$\langle A(t_1)B(t_2)\rangle = \mathrm{tr}(A(t_1)B(t_2)\,\rho) \tag{A.6}$$
$$= \mathrm{tr}\Big(B\,U(t_2,t_1)\rho(t_1)A\,U^{\dagger}(t_2,t_1)\Big) \tag{A.7}$$

where $\rho(t_1) = U(t_1{,}0)\rho\,U(t_1{,}0)^{\dagger}$. Therefore, first the density matrix evolves from time 0 to t_1. Then the density matrix is multiplied on the right by the operator A, obtaining a "fictitious" density matrix that is let evolve from t_1 to t_2. At last, the latter is used to calculate the expectation value of the operator B.

A.2 Custom Wigner Distribution Function Script

As mentioned in section 3.2, to compute the Wigner Distribution Function of a (e.g. reflected) field it is necessary to map $G^{(1)}(t_1,t_2)$ into $G^{(1)}(t,\tau)$ according to

$$\begin{cases} t &= (t_1+t_2)/2 \\ \tau &= t_2 - t_1. \end{cases}$$

In the MATLAB® script, $G^{(1)}(t_1,t_2)$ is saved as a matrix $G^{(1)}(t1\_index,\, t2\_index)$ with $t1\_index, t2\_index = 1,\ldots,N$, whereas $G^{(1)}(t,\tau)$ in terms of $G^{(1)}(time\_index,\, tau\_index)$, being $time\_index,\, tau\_index = 1,\ldots,2N-1$. The coordinate systems are shown in figure A.1. It follows that

$$tau\_index = t2\_index - t1\_index + N, \tag{A.8}$$
$$time\_index = t1\_index + t2\_index - 1, \tag{A.9}$$

where $N \times N$ is the dimension of the square matrix. It is then found that

$$t1\_index = (time\_index - tau\_index + 1 + N)/2 \tag{A.10}$$
$$t2\_index = (time\_index + tau\_index + 1 - N)/2. \tag{A.11}$$

From the previous system it is clear that $G^{(1)}$ must be interpolated at half-integer values of the indices, by averaging with the values of $G^{(1)}(t1\_index, t2\_index)$ obtained at the closest integer indices.

Fast Fourier transform implementation The fast Fourier transform (FFT) algorithm computes the discrete Fourier transform of a finite sequence $\{x_n\}$, converting in this thesis the signal from time domain to frequency domain. It reduces the computational cost from

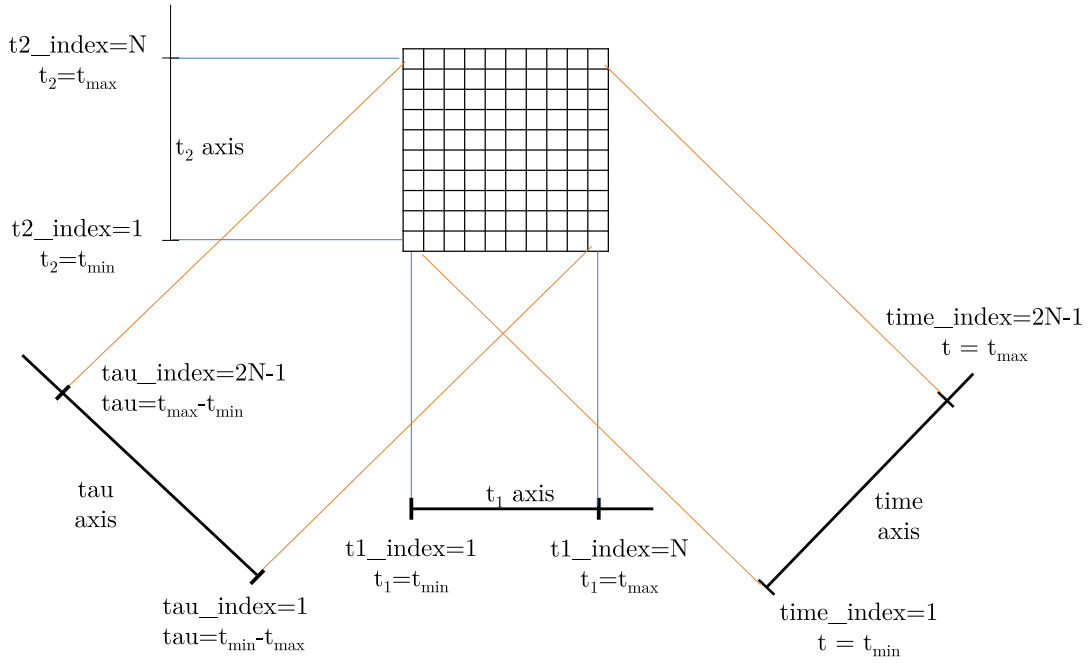


Figure A.1: Discrete first order correlation function $G^{(1)}(t1\_index, t2\_index)$ in the blue coordinate system. In orange as $G^{(1)}(time\_index, tau\_index)$.

$\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$, where N is the number of elements of the signal, according to the DFT definition

$$X_k \equiv \mathcal{F}\left(\{x_n\}\right)_k = \sum_{n=0}^{N-1} x_n e^{-\mathrm{i}2\pi kn/N} \qquad k = 0, \ldots, N-1 \tag{A.12}$$

with $x_0, \ldots, x_{N-1}$ complex numbers. In the following, the phase shifting and normalization criteria will be illustrated.

Phase shift A first-order correlation function is always anti symmetric in the sense that

$$G^{(1)}(-\tau) = G^{(1)}(\tau)^*. \tag{A.13}$$

This anti symmetry ensures that the spectral density of the optical field, being the Fourier Transform of the first-order correlation function, is a real quantity. Moreover, the integral of the spectral density recovers the photon flux (3.10), i.e. a real physical quantity. In the pulsed-wave regime a similar anti-symmetry relation holds:

$$G^{(1)}(t_1, t_2) = G^{(1)}(t_2, t_1)^*, \tag{A.14}$$

which ensures that the Wigner-Ville function, also denoted Wigner Distribution Function and abbreviated by WDF in this manuscript, is a real quantity - though it can assume negative values.
However, directly applying the FFT algorithm to the correlation functions does not lead to real values, as it is now explained. In the simulations each signal is discrete and stored with positive indices. For instance given a signal x_n defined over $n = -N, -N+1, \ldots, N-1, N$, it is treated by the FFT algorithm as it were defined over $n = 1, 2, \ldots, 2N+1$. This determines a phase shifted output, which is in general not a problem since one is commonly interested in the absolute values of the Fourier transform (as in a circuit transfer function). Still, it must be corrected if one wants to check the normalization or have the correct WDF. The right phase is recovered by exploiting the *shift theorem* of the DFT, that is

$$\mathcal{F}\left(\{x_{n-m}\}\right)_k = X_k \cdot e^{-\frac{2\pi \mathrm{i}}{N} km}, \tag{A.15}$$

where in this case m is the number of elements associated to negative delays in the first-order correlation function, or the Wigner Distribution Function at each time bin *time_index*. Thus, a circular shift of the input x_n corresponds to multiplying the output X_k by a linear phase.

Normalization Apart from the phase shift, also proper normalization of the FFT result has to be ensured. Parseval's theorem for the DFT states that

$$\sum_{n=0}^{N-1} |x_n|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X_k|^2. \tag{A.16}$$

The discrepancy occurs when, due to the FFT algorithm optimization, the original signal x_n is padded with zeros so that its length is a power of two. Because of the N dependency in the last equation, Parseval's theorem does not hold anymore for the original signal. To recover it, it can be shown that the output of the FFT must be divided by the sampling frequency. More information can be found in the commented code.

Appendix B

Supplementary Figures

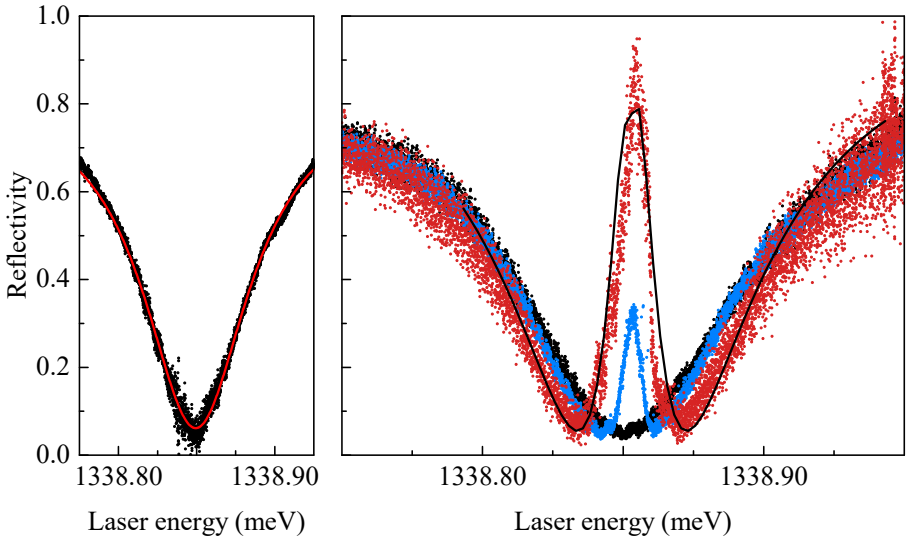


Figure B.1: Experimental (symbols) and simulated (lines) reflectivity as a function of the incident laser energy of (left) an empty cavity, (right) a QD with increasing input power. The black line refers to the lowest power. Image from [19].

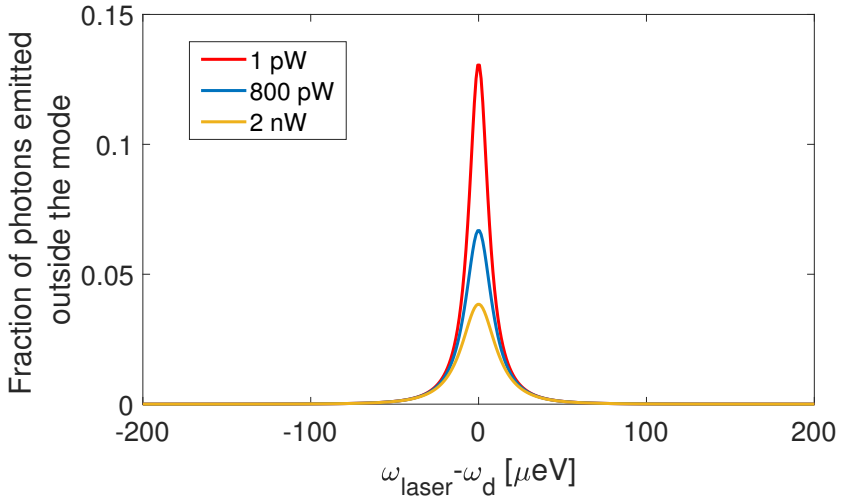


Figure B.2: Fraction of photons emitted outside the mode. Simulation parameters in section 3.1.

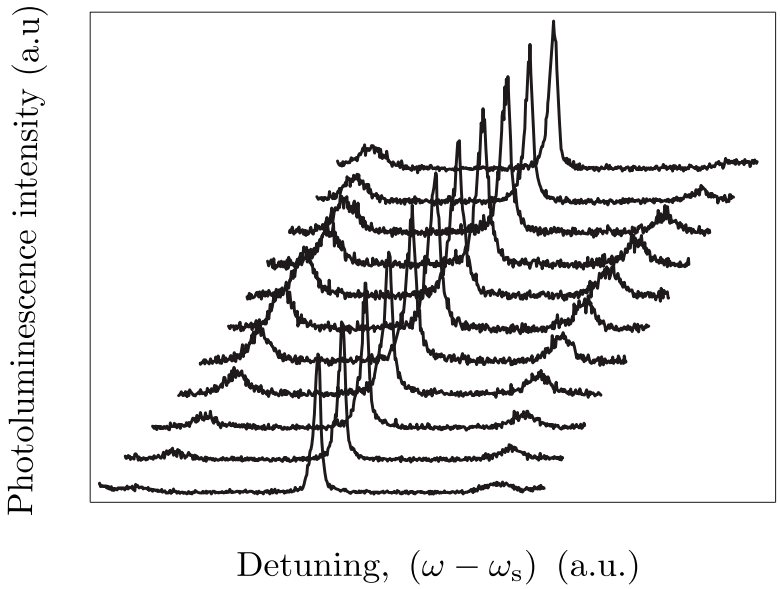
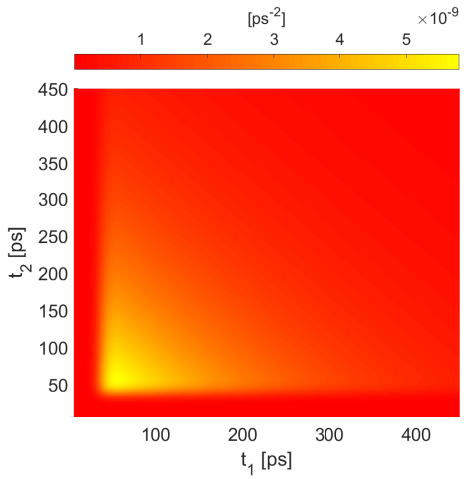
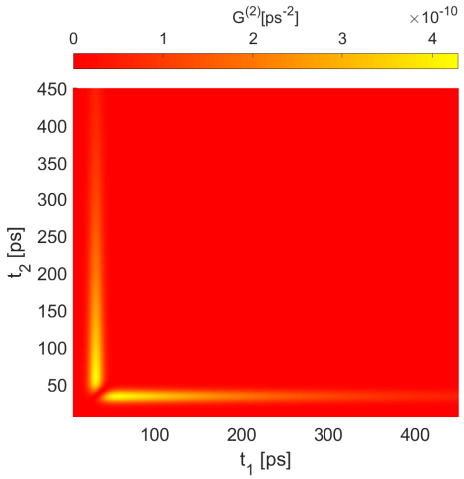


Figure B.3: Laser detuning dependent resonance fluorescence spectra at increasing power, showing Mollow triplets as the power increases. Image from [34].



(a) Uncorrelated coincidences rate of the emitted field. It is the denominator of equation (3.15), but expressed in terms of (t_1, t_2) instead of $t \equiv t_1$ and delay $\tau \equiv t_2 - t_1$.



(b) Correlated coincidences $G^{(2)}$ of the emitted field. It is the numerator of equation (3.15), but expressed in terms of (t_1, t_2) instead of $t \equiv t_1$ and delay $\tau \equiv t_2 - t_1$.

Figure B.4: Correlated and uncorrelated coincidences of the emitted field in pulsed wave excitation, simulation parameters in section 3.3.

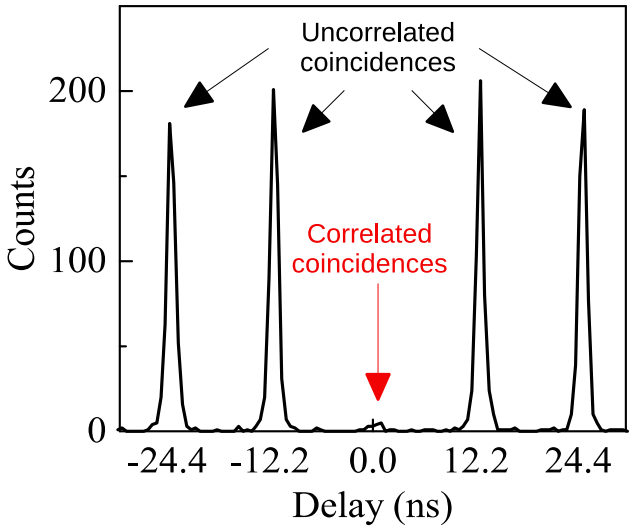


Figure B.5: Experimental second order correlation of the emission from a QD in a micropillar. Image from [19].

Appendix C

MATLAB® Scripts

C.1 Main scripts

C.1.1 Reflectivity spectrum in CW

```
clear
clc
%close all
%%%%%%%      Common basis to every programs based "two levels" %%%%%%%%%%
% Important note: these paths must be modified if needed
addpath(genpath('..\QotoolboxV015'));
addpath(genpath('..\CQED subprograms'));
addpath(genpath('..\CQED device parameters'))

% In addition, for the mesolve function to operate the executable files
% (.exe) and batch files (.bat) contained in  '[...] \QotoolboxV015\bin'
% have to be copied to  a folder that is on the Windows system path, in the
% main hard drive where Windows is installed. This can be for example in:
% 'C:\Program Files\Matlab\R2014a\bin'.

% Warning: for the adiabatic version to converge, the tolerance in
% mesolve.m function must be reduced compared to the defaut values. For
% example:
% ode2file('ode_input.dat',L,rho0,t_list,struct('reltol',7e-8,'abstol',8e
    -8));

%% %%%%%%%%%% CW - spectra under stationary resonant excitation %%%%%%%%%%
% This script indexed "CW" considers a fixed incoming power and a variable
% angular frequency for the laser. One calculates the spectral response
% associated to the various fields (reflected, transmitted,
% diffracted/lost, and spontaneously emitted outside the cavity mode). One
% finally verifies the conservation of the total photon flux

```

```
28 % Choice of full model 'F' or adiabatic 'A'
29 model = 'F';
30
31 %%% Experimental conditions
32 detuning_QD_C_muev = 0; %Detuning between the QD and cavity frequencies, in
       mueV
33 eta_in = 1; % Injection efficiency for the incoming photons (depends on
      experimentally-achieved spatial coupling)
34 P_in_CW_pW=10;% Incoming continuous-wave power in pW
35
36 %Parameters for the calculation of spectra
37 min_detuning_muev=-100; %minimal detuning (left part of the spectrum), in
      muev
38 max_detuning_muev=100; %maximal detuning (right part of the spectrum), in
      muev
39 nb_points_spectrum=400;
40
41 Init2levelDeviceParametersOngoingTest;
42 Init2levelHilbertSpaceAndOperators;
43 InitLists2levelCWscanLaserFrequency;
44
45 %Incoming power
46 P_in_CW = P_in_CW_pW*1e-12;% Incoming power in W %%
47 b_in_CW = sqrt(eta_in*P_in_CW*1e-24/(hbar*omega_c)); % square root of the
      photon number per unit time, in ps-1/2
48 total_flux_injected_photons=abs(b_in_CW)^2; % total flux of incoming
      photons taking into account  eta_in (so only the photons coupled to the
       cavity mode), in ps-1
49
50 % tic % NB: "tic" is used as a "start" time for the measurement of the
51 % computing time between "tic" and "toc"
52
53 %%%%%%%%%%%%%  Start the calculation of spectra %%%%%%%%%%%%
54 for omega_index=1:nb_points_spectrum   %Loop for frequency scan
55
56     omega_laser=omega_laser_list(omega_index);  %Current value of
          omega_laser, in rad/ps
57
58     switch model
59         case 'A' % Adiabatic case
60
61             Delta = 2*(omega_laser-omega_c)/kappa; %normalized laser
                detuning appearing in Eq.12 of the pdf notes
62
63             %%%%%%%%% Definition of the adiabatic-model Hamiltonian (which
                depends on omega_laser)
```

```
64              H_CW = (omega_eff-omega_laser)*sigma_dag*sigma...
65                  - 1i*sqrt(Gamma_0*eta_top)*(b_in_CW*sigma_dag/(1-1i*Delta)-
                        b_in_CW'*sigma/(1+1i*Delta)); % Adiabatic Hamiltonian
66
67              %%% For the redefinition of the operator "a" acting in the QD
                    space
68              % (Eq. 10 of the pdf notes
69              a = -2*g*sigma/(kappa*(1-1i*Delta_QDC))-2*sqrt(kappa_top)*
                    b_in_CW*Id/(kappa*(1-1i*Delta)); %annihilation operator a
                    in adiabatic approximation
70
71              % UNUSED HERE: Ansatz for the annihilation operator, obtained
                    by taking the time derivative of "a" equal to 0
72              % (OK for CW but not for PR (pulsed regime) programs)
73              % a = -2*g*sigma/(kappa*(1-1i*Delta))-2*sqrt(kappa_top)*b_in_CW
                    *Id/(kappa*(1-1i*Delta));
74
75          case 'F'  % Full model
76
77              %%%%%%%%% Definition of the full-model Hamiltonian (which
                    depends on omega_laser)
78              H_CW = (omega_d-omega_laser)*sigma_dag*sigma...
79                   + (omega_c-omega_laser)*a_dag*a...
80                   + 1i*g*(sigma_dag*a-a_dag*sigma)...
81                   - 1i*sqrt(kappa_top)*b_in_CW*(a_dag-a);
82
83      end % end of the "switch model"
84
85      %Superoperator associated to the coherent processes (Hamiltonian)
86      L_coh = -1i * (spre(H_CW) - spost(H_CW));
87
88      %%%%%%%%% Calculation of the Liouvillian superoperator
89      Liouvillian  = L_coh + L_incoh; % Total Liouvillian superoperator
            including both coherent processes (Hamiltonian) and incoherent
            processes (dissipative jumps)
90
91      %%%%%%%%% Calculation of the density matrix corresponding to the
            stationary state
92      rhoss_CW = steady(Liouvillian);
93
94      %%%%%%%%% Definition (or re-definition) of the output operators
95      if (model=='A' ||  omega_index==1)  %
96          % In the full model, the output flux operators are not
97          % frequency-dependent and thus need to be defined only the first
98          % time In the adiabatic model the value of "a" is
99          % frequency-dependent and the output operators have to be
```

```
100         % redefined for each frequency
101
102         b_out = b_in_CW*Id + sqrt(kappa_top)*a; % definition of the
                operator b_out, i.e. the output operator for the reflected
                light, in ps^(-1/2)
103         c_out = sqrt(kappa_bottom)*a; % definition of the operator c_out, i
                .e. the output operator for the transmitted light, in ps^(-1/2)
104         d_out = sqrt(kappa_loss)*a; % definition of the operator d_out, i.e
                . the output operator for the diffracted/lost light, in ps
                ^(-1/2)
105         % e_out = sqrt(gamma_sp)*sigma; % definition of the operator e_out,
                 i.e. the output operator for the light spontaneously emitted
                outside the cavity mode, in ps^(-1/2)
106
107         % NB: in the adiabatic model the operators could also have been
                written directly as:
108         % b_out = b_in_CW*Id*(1-2*eta_top/(1-1i*Delta))-sqrt(Gamma_0*
                eta_top)*sigma/(1-1i*Delta_QDC); %output flux operator,  (eq
                .12)
109         % c_out = -2*g*sqrt(kappa_bottom)*sigma/(kappa*(1-1i*Delta_QDC))-2*
                sqrt(kappa_top*kappa_bottom)*b_in_CW*Id/(kappa*(1-1i*Delta));
110         % d_out = -2*g*sqrt(kappa_loss)*sigma/(kappa*(1-1i*Delta_QDC))-2*
                sqrt(kappa_top*kappa_loss)*b_in_CW*Id/(kappa*(1-1i*Delta));
111         % Such formulas are obtained by directly replacing the value of "a"
112         % from the adiabatic model, and are thus equivalent to the above,
113         % more general definitions In addition, e_out is independent on the
114         % experimental conditions and thus defined in the subprogram
115         % "Init_2level_Hilbert_space_and operators.m". It is given here for
116         % information and clarity purposes only
117
118     end
119
120     %%%%%%  Calculation of useful expectation values %%%%%%%%%%%%
121
122     % Calculation of the total photon flux as a function of omega_laser
123     total_flux_reflected_photons_vs_omega(omega_index) = expect(b_out'*
            b_out,rhoss_CW); % Total reflected flux  = <b_out_dag b_out>, in ps
            ^(-1)
124     total_flux_transmitted_photons_vs_omega(omega_index) = expect(c_out'*
            c_out,rhoss_CW); % Total transmitted flux = <c_out_dag c_out>, in
            ps^(-1)
125     total_flux_diffracted_photons_vs_omega(omega_index) = expect(d_out'*
            d_out,rhoss_CW);% Total diffracted/lost flux = <d_out_dag d_out>,
            in ps^(-1)
```

```
126     total_flux_emitted_photons_vs_omega(omega_index) = expect(e_out'*e_out,
            rhoss_CW);% Total flux of spontaneously-emitted photons outside the
             mode = <e_out_dag e_out>, in ps^(-1)
127
128     occupation_excited_state_vs_omega(omega_index) = expect(sigma_dag*sigma
            ,rhoss_CW); % Occupation probability for the excited state |e>
129     occupation_ground_state_vs_omega(omega_index) = expect(sigma*sigma_dag,
            rhoss_CW); % Occupation probability for the ground state |g>
130 end % end of the frequency scan
131
132 % toc
133 % NB: "toc" is used as a "stop" time for the measurement of the computing
        time between "tic" and "toc"
134
135 % Normalization: coefficients of reflectivity, transmission, diffracted/
        lost part, and spontaneously-emitted part
136 R_vs_omega = total_flux_reflected_photons_vs_omega /
        total_flux_injected_photons;
137 T_vs_omega = total_flux_transmitted_photons_vs_omega /
        total_flux_injected_photons;
138 D_vs_omega = total_flux_diffracted_photons_vs_omega /
        total_flux_injected_photons;
139 E_vs_omega = total_flux_emitted_photons_vs_omega /
        total_flux_injected_photons;
140
141 %% %%%%%%%% Plots %%%%%%%%
142 % For plot selection:
143 % - Reflected photons : 'R'
144 % - Transmitted + diffracted/lost photons : 'T'
145 % - Photons emitted outside the mode : 'E'
146 % - Occupation probabilities : 'O'
147
148 % plot_choice = ['T'];
149 plot_choice = ['T';'R';'E';'O'];
150 Plot2levelCWvsLaserFrequency;
```

C.1.2 Photon flux evolution and QD occupation in PW

```
1 clear
2 clc
3 %close all
4 %% Important note: these paths must be modified if needed
5 addpath(genpath('..\QotoolboxV015'));
6 addpath(genpath('..\CQED subprograms'));
7 addpath(genpath('..\CQED device parameters'))
8 savepath
```

```
9
10 % In addition, for the mesolve function to operate the executable files
11 % (.exe) and batch files (.bat) contained in  '[...] \QotoolboxV015\bin'
12 % have to be copied to  a folder that is on the Windows system path, in the
13 % main hard drive where Windows is installed. This can be for example in:
14 % 'C:\Program Files\Matlab\R2014a\bin'.
15
16 % Warning: for the adiabatic version to converge, the tolerance in
17 % mesolve.m function must be reduced compared to the defaut values. For
18 % example:
19 % ode2file('ode_input.dat',L,rho0,t_list,struct('reltol',7e-8,'abstol',8e
       -8));
20 %% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
21 %%%%%%%%%%%%%%%%%%%%%%%%% Pulsed regime %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
22 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
23 % This section indexed "PR" computes the time evolution of the photon flux
24 % for various fields, and of the exciton occupation, in response to a
25 % coherent laser pulse with a fixed center frequency, temporal width, and
26 % average number of photons. One finally verifies that the total photon
27 % flux has been conserved after the simulation.
28 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
29
30 %% Choice of full model 'F' or adiabatic model 'A'
31 model = 'F';
32
33 %% Experimental conditions
34 detuning_QD_C_muev = 10; %Detuning between the QD and cavity frequencies,
       in mueV
35 detuning_pulse_QD_muev = 0; %Detuning between the pulse central frequency
       and the QD frequency, in mueV
36 eta_in = 1; % Injection efficiency for the incoming photons (depends on
       experimentally-achieved spatial coupling)
37 Nb_photons = 10;  % Average number of incoming photons in a pulse. This
       quantity should be multiplied by eta_in to know the number of incoming
       photons actually coupled to the optical mode
38 FWHM = 15; %in ps, full width at half-maximum of the incoming Gaussian
       pulse intensity (unit: ps since angular frequencies are in rad/ps)
39
40 %%
41 %Initialization of parameters, operators, arrays, etc...
42 Init2levelDeviceParametersOngoingTest;
43 Init2levelHilbertSpaceAndOperators;
44 InitLists2levelPRvsTime;
45
46 % Definition of the input field in ps^(-1/2), in the form of a fseries (
       necessary for integrating the master equation)
```

```
47 Standard_deviation_b_in_PR = FWHM/(2*sqrt(log(2))); %Deduced from the
       properties of a Gaussian function
48 b_in_fn = fn('gauss',t_delay,Standard_deviation_b_in_PR) * sqrt( eta_in*
       Nb_photons / ( sqrt(pi) * Standard_deviation_b_in_PR ) ); % square root
        of the incoming photon number per time unit, in ps^-1/2
49 b_in_vs_time = fsval(b_in_fn,t_list); %scalar array representing b_in vs
       time
50
51 % Initial density matrix before the pulse has started
52 switch model
53     case 'F' %Full model
54         psi0 = tensor(Vacuum_state,g_ket); % Initial state: tensorial
               product of photonic vacuum and QD ground state
55         rho0 = psi0*psi0'; % Density matrix corresponding to the initial
               pure state
56     case 'A' % Adiabatic model
57         rho0 = g_ket*g_ket'; % Density matrix corresponding to the initial
               pure state
58 end
59
60 %%% UNUSED HERE: to describe a non-resonant excitation experiment where the
61 % exciton state |e> is populated at time zero, one can simply change psi0
62 % and use a very small value for N_in, like 0.0001, to ensure that the
63 % output fields are almost entirely induced by the initial excitation. In
64 % such a case we use (e.g. for the full model):
65 % psi0=tensor(Vacuum_state,e_ket); % Initial state: tensorial product of
       photonic vacuum and QD excited state
66 % rho0=psi0*psi0'; % Density matrix corresponding to the initial pure state
67
68 %% %%%%%%%% System Hamiltonian and time-dependent operators  %%%%%%%%%%%
69 %
70 % The system Hamiltonian is time-dependent due to the function b_in_fn
71 % describing the input field b_in(t).
72
73 % In addition, in the case of adiabatic elimination of the cavity mode an
74 % effective operator $a$ is defined, acting on the QD subspace, based on
75 % the formula for adiabatic elimination (Eq. 10 of the pdf notes). Since
76 % this formula depends on b_in(t), we define a time-dependent quantity
77 % "a_vs_time", which is an array containing, for each time of t_list, the
78 % corresponding operator "a". In the full model case, to simplify the
79 % following calculations, we define the same quantity a_vs_time, yet this
80 % time this array contains the same operator (annihilation operator "a"
81 % acting on the cavity subspace), replicated for all times of t_list.
82
83
84 switch model
```

```
85          case 'A' % Adiabatic model
86
87              Delta = 2*(omega_pulse-omega_c)/kappa; %normalized laser
                    detuning appearing in Eq.12 of the pdf notes
88
89              %%%%%%%%% Definition of the adiabatic-model Hamiltonian (which
                    depends on omega_pulse)
90              H_PR = (omega_eff-omega_pulse)*sigma_dag*sigma - 1i*sqrt(
                    Gamma_0*eta_top)*...
91                      ((1-1i*Delta)^(-1)*b_in_fn*sigma_dag-(1+1i*Delta)^(-1)*
                            b_in_fn'*sigma); % Hamiltonian (eq.13)
92
93              %%% For the redefinition of the operator "a" acting in the QD
94              %%% space vs time (Eq. 10 of the pdf notes)
95              a_vs_time = -2*(kappa*(1-1i*Delta_QDC))^(-1)*g*sigma*Id_vs_time
                    -2*sqrt(kappa_top)*(kappa*(1-1i*Delta))^(-1)*fsval(b_in_fn,
                    t_list).*Id_vs_time; %annihilation operator a in adiabatic
                    approximation (eq.10), as fseries
96
97          case 'F'  % Full model
98
99              %%%%%%%%% Definition of the full-model Hamiltonian (which
                    depends on omega_pulse)
100             H_PR = (omega_d-omega_pulse)*sigma_dag*sigma...
101                  + (omega_c-omega_pulse)*a_dag*a...
102                  + 1i*g*(sigma_dag*a-a_dag*sigma)...
103                  - 1i*sqrt(kappa_top)*b_in_fn*(a_dag-a);
104             %definition of a_vs_time, even though a is costant in the full
105             %model, to reduce the number of "switch" in the following code
106              a_vs_time = a*Id_vs_time;
107 end % end of the "switch model"
108
109 %Superoperator associated with the coherent processes (Hamiltonian)
110 L_coh = -1i * (spre(H_PR) - spost(H_PR));
111
112 %%%%%%%%% Calculation of the Liouvillian superoperator
113 Liouvillian  = L_coh + L_incoh; % Total Liouvillian superoperator including
        both coherent processes (Hamiltonian) and incoherent processes (
    dissipative jumps)
114
115 %%%%%%%%%% Numerical Integration of the Master Equation %%%%%%%%
116 % Computation of the density matrix vs time with t_list, i.e. between t_min
117 % and t_max, requiring a large enough time resolution.
118 rho_vs_time = mesolve(Liouvillian,rho0,t_list); % second evolution of the
    system
119
```

```
120
121 %%%%%%%%%%%%%%%%%% Definition of the output operators %%%%%%%%%%%%%%%%%%
122 % These are general formulas for both the adiabatic and full model,
        depending on "a_vs_time"
123 b_out_vs_time = b_in_vs_time*Id_vs_time + sqrt(kappa_top)*a_vs_time; %
        definition of the operator b_out, i.e. the output operator for the
        reflected light, in ps^(-1/2)
124 c_out_vs_time = sqrt(kappa_bottom)*a_vs_time; % definition of the operator
        c_out_vs_time, i.e. the output operator for the transmitted light, in
        ps^(-1/2)
125 d_out_vs_time = sqrt(kappa_loss)*a_vs_time; % definition of the operator
        d_out_vs_time, i.e. the output operator for the diffracted/lost light,
        in ps^(-1/2)
126 % e_out = sqrt(gamma_sp)*sigma %  output operator for the light
        spontaneously emitted outside the cavity mode, in ps^(-1/2), already
        defined in Init_2level_Hilbert_space_and_operators.m
127 a_dag_vs_time = a_vs_time';  % definition of describing a_dag(t)
128
129 %Calculation of the state population
130 expect_sigma_dag_sigma_vs_time = expect(sigma_dag*sigma,rho_vs_time); %
        List describing <sigma_dag sigma>(t), i.e. the excited state population
131 expect_sigma_sigma_dag_vs_time = expect(sigma*sigma_dag,rho_vs_time); %
        List describing <sigma sigma_dag>(t), i.e. the ground state population
132
133 % Calculation of the total photon flux as a function of time
134 flux_injected_photons_vs_time = b_in_vs_time.^2; % total flux of injected
        photons taking into account  eta_in (so only the photons coupled to the
         cavity mode), in ps(-1)
135 flux_reflected_photons_vs_time = real(expect(b_out_vs_time'*b_out_vs_time,
        rho_vs_time));%flux in ps^(-1)
136 flux_transmitted_photons_vs_time = real(expect(c_out_vs_time'*c_out_vs_time
        ,rho_vs_time));%flux in ps^(-1)
137 flux_diffracted_photons_vs_time = real(expect(d_out_vs_time'*d_out_vs_time,
        rho_vs_time)); %flux in ps^(-1)
138 flux_emitted_photons_vs_time = real(expect(e_out'*e_out,rho_vs_time));%flux
         in ps^(-1)
139 %% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
140 %%%%%%%%%%%%%%%%%%%%%%%%    Plots     %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
141 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
142
143 % For plot selection:
144 % - photon fluxes vs time vs delay  : 'F'
145 % - occupation probabilities vs time: 'O'
146
147 plot_choice = ['F';'O'];
148 Plot2levelPRvsTime;
```

C.1.3 First-order correlation and spectral densities in CW

```
clear
clc
close all

%% Important note: these paths must be modified if needed
addpath(genpath('..\QotoolboxV015'));
addpath(genpath('..\CQED subprograms'));
addpath(genpath('..\CQED device parameters'))
savepath

% In addition, for the mesolve function to operate the executable files
% (.exe) and batch files (.bat) contained in  '[...] \QotoolboxV015\bin'
% have to be copied to  a folder that is on the Windows system path, in the
% main hard drive where Windows is installed. This can be for example in:
% 'C:\Program Files\Matlab\R2014a\bin'.

% Warning: for the adiabatic version to converge, the tolerance in
% mesolve.m function must be reduced compared to the defaut values. For
% example:
% ode2file('ode_input.dat',L,rho0,t_list,struct('reltol',7e-8,'abstol',8e-8));

%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%% g1SDCW : g1(tau) and spectral densities in CW %%%%%%%%%%%%%%%%%%%%%%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This section called "g1SDCW" evaluates the first order temporal coherence
% g(1) as a function of the delay tau, for the various fields. It indicates
% the fractions of the coherent and incoherent contributions to the photon
% flux for each field. It also computes the Fourier transform of the
% incoherent contribution to g(1)(tau) as a function of omega, which gives
% the spectral density of flux for the incoherent part of the optical
% field. It verifies that the integral of the spectral densities
% corresponds to the photon flux (incoherent part only).
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Choice of full model 'F' or adiabatic model 'A'
model = 'F';
```

```
% Warning: in the full model 'F', the size of the Fock state must be large
% enough to ensure that the last Fock state is negligibly occupied.
% Artifacts can otherwise arise, especially when increasing the incoming
% power.

%%% Experimental conditions
detuning_QD_C_muev = 0; %Detuning between the QD and cavity frequencies, in
     mueV
detuning_laser_QD_muev = 0; %Detuning between the pulse central frequency
    and the QD frequency, in mueV
eta_in = 1; % Injection efficiency for the incoming photons (depends on
    experimentally-achieved spatial coupling)
P_in_CW_pW = 10;% Incoming continuous-wave power in pW

% Parameters for the evaluation of the temporal evolution
%
% NB1: the maximum delay "tau_max" will also dictate the frequency
% resolution of the spectra, given by the angular frequency step
% "omega_step". The corresponding angular frequency lists are defined in
% the "Init_lists_..." subprogram (see also below details on the
% calculation and Fast Fourier Transform (FFT) algorithm)
%
% NB2: one should be careful that tau_max is large enough to include a good
% approximation of "infinite delays" (check that the g1(tau) function has
% had enough time to truly converge), while keeping a number of points
% large enough to ensure a good temporal resolution. This is especially
% important for high input powers where artifacts can appear.

tau_max = 4000; %maximum positive delay in ps
nb_points_delay = 2^13 + 1; % Number of points in the list of positive
    delays (tau_list).
% --> This must be of the form 2^N+1 for FFT optimization. For example:
    2^13+1=8193

% Parameter defining the observed spectral window, in mueV, to avoid
% plotting and calculating spectra over an inadequately large angular
% frequency ranges. NB: should not exceed the size of the full FFT
% spectrum, which depends on the temporal time step and thus on tau_max and
% nb_points_delay.

width_spectral_window_muev = 300; % width of the spectral window to be
    displayed, centered on omega_laser

%%%%%%%%%% Plots %%%%%%%%
% For plot selection:
% - g1(tau) : 'G'
```

```
77 % - Spectral densities of flux : 'S'
78 %
79 % plot_choice = ['G'];
80 plot_choice = ['G';'S'];
81
82 %%
83 % Initialization of parameters, operators, arrays, etc...
84 Init2levelDeviceParametersOngoingTest;
85 Init2levelHilbertSpaceAndOperators;
86 InitLists2levelG1SDCWvsDelayAndFrequency;
87
88 % Incoming power
89 P_in_CW = P_in_CW_pW*1e-12;% Incoming power of the CW laser, in W %%
90 b_in_CW = sqrt(eta_in*P_in_CW*1e-24/(hbar*omega_c)); % square root of the
       photon number per unit time, in ps-1/2
91 flux_injected_photons = abs(b_in_CW)^2; % total flux of injected photons,
       taking into account eta_in (so only the photons coupled to the cavity
       mode), in ps-1
92
93 switch model
94     case 'A' % Adiabatic elimination model
95
96         Delta = 2*(omega_laser-omega_c)/kappa; % Normalized laser-cavity
               detuning
97
98         % Hamiltonian including the QD operators, the cavity effect being
               included by adiabatic elimination
99         Hamiltonian_CW = (omega_eff-omega_laser)*sigma_dag*sigma...
100                        - 1i*sqrt(Gamma_0*eta_top)*(b_in_CW*sigma_dag/(1-1i*
                               Delta)-b_in_CW'*sigma/(1+1i*Delta)); %
                               Hamiltonian
101
102        % Definition of the operator "a" acting in the QD subspace,
103        % following adiabatic elimination (Eq. 10 of the pdf notes)
104        a = -2*g*sigma/(kappa*(1-1i*Delta_QDC))-2*sqrt(kappa_top)*b_in_CW*
               Id/(kappa*(1-1i*Delta)); %annihilation operator a in adiabatic
               approximation
105
106        % UNUSED HERE: Ansatz for the annihilation operator, obtained by
               taking the time derivative of "a" equal to 0
107        % (OK for CW but not for PR (pulsed regime) programs)
108        % a = -2*g*sigma/(kappa*(1-1i*Delta))-2*sqrt(kappa_top)*b_in_CW*Id
               /(kappa*(1-1i*Delta));
109
110    case 'F'  % Full model
111
```

```
112         % Hamiltonian including both QD and cavity operators
113         Hamiltonian_CW = (omega_d-omega_laser)*sigma_dag*sigma...
114                        + (omega_c-omega_laser)*a_dag*a...
115                        + 1i*g*(sigma_dag*a-a_dag*sigma)...
116                        - 1i*sqrt(kappa_top)*b_in_CW*(a_dag-a);
117
118 end % end of the "switch model"
119
120 %Superoperator associated with the coherent processes (Hamiltonian)
121 L_coh = -1i * (spre(Hamiltonian_CW) - spost(Hamiltonian_CW));
122
123 %%%%%%%%% Calculation of the Liouvillian superoperator
124 Liouvillian  = L_coh + L_incoh; % Total Liouvillian superoperator including
        both coherent processes (Hamiltonian) and incoherent processes (
    dissipative jumps)
125
126 %%%%%%%%% Calculation of the density matrix corresponding to the stationary
        state
127 density_matrix_stationary_state = steady(Liouvillian);
128
129
130 %%%%%%%%%%%%%%%%%% Definition of the output operators %%%%%%%%%%%%%%%%%%
131 % These are general formulas for both the adiabatic and full model
132 b_out = b_in_CW*Id + sqrt(kappa_top)*a; % definition of the operateur b_out
    , i.e. the output operator for the reflected light, in ps^(-1/2)
133 c_out = sqrt(kappa_bottom)*a; % definition of the operateur c_out, i.e. the
        output operator for the transmitted light, in ps^(-1/2)
134 d_out = sqrt(kappa_loss)*a; % definition of the operateur d_out, i.e. the
    output operator for the diffracted/lost light, in ps^(-1/2)
135 % e_out = sqrt(gamma_sp)*sigma; % definition of the operateur e_out, i.e.
    the output operator for the light spontaneously emitted outside the
    cavity mode, in ps^(-1/2)
136
137 % NB: in the adiabatic model the operators could also have been written
138 % directly as a contribution from the empty cavity (term proportionnal to
139 % the identity operator "Id") and a contribution describing QD emission
140 % (term proportionnal to the decay operator "sigma"):
141 % b_out = b_in_CW*Id*(1-2*eta_top/(1-1i*Delta))-sqrt(Gamma_0*eta_top)*sigma
    /(1-1i*Delta_QDC); %output flux operator,  (eq.12)
142 % c_out = -2*g*sqrt(kappa_bottom)*sigma/(kappa*(1-1i*Delta_QDC))-2*sqrt(
    kappa_top*kappa_bottom)*b_in_CW*Id/(kappa*(1-1i*Delta));
143 % d_out = -2*g*sqrt(kappa_loss)*sigma/(kappa*(1-1i*Delta_QDC))-2*sqrt(
    kappa_top*kappa_loss)*b_in_CW*Id/(kappa*(1-1i*Delta)); %annihilation
    operator a in adiabatic approximation
144 %
145 % Such formulas are obtained by directly replacing the value of "a" from
```

```
146 % the adiabatic model, and are thus equivalent to the above, more general
147 % definitions. In addition, e_out is independent on the experimental
148 % conditions and thus defined in the subprogram
149 % "Init_2level_Hilbert_space_and operators.m". It is given here for
150 % information and clarity purposes
151
152
153 % Total photon flux for the various fields, i.e. expectation values of the
        form
154 % < b'b >, including both coherent and incoherent contributions to the
        optical flux
155 flux_reflected_photons=real(expect(b_out'*b_out,
        density_matrix_stationary_state));  %flux in ps^(-1)
156 flux_transmitted_photons=real(expect(c_out'*c_out,
        density_matrix_stationary_state)); %flux in ps^(-1)
157 flux_diffracted_photons=real(expect(d_out'*d_out,
        density_matrix_stationary_state)); %flux in ps^(-1)
158 flux_emitted_photons=real(expect(e_out'*e_out,
        density_matrix_stationary_state)); %flux in ps^(-1)
159
160 % Coherent part of the photon flux, i.e. expectation values of the form <
161 % b' > < b >, corresponding to the fraction of light that has a
162 % well-defined amplitude and phase (characterized by the complex field
163 % amplitude < b > ), with respect to the incoming monochromatic laser.
164 flux_reflected_photons_laser_coherent = abs(expect(b_out,
        density_matrix_stationary_state))^2; %flux in ps^(-1)
165 flux_transmitted_photons_laser_coherent = abs(expect(c_out,
        density_matrix_stationary_state))^2;%flux in ps^(-1)
166 flux_diffracted_photons_laser_coherent = abs(expect(d_out,
        density_matrix_stationary_state))^2;%flux in ps^(-1)
167 flux_emitted_photons_laser_coherent = abs(expect(e_out,
        density_matrix_stationary_state))^2;%flux in ps^(-1)
168
169 % Incoherent part of the photon flux, corresponding to the fraction of the
170 % optical flux which has no well-defined phase with respect to the incoming
171 % laser, and thus cannot interfere with it.
172 flux_reflected_photons_incoh = flux_reflected_photons-
        flux_reflected_photons_laser_coherent;%flux in ps^(-1)
173 flux_transmitted_photons_incoh = flux_transmitted_photons-
        flux_transmitted_photons_laser_coherent;%flux in ps^(-1)
174 flux_diffracted_photons_incoh = flux_diffracted_photons-
        flux_diffracted_photons_laser_coherent;%flux in ps^(-1)
175 flux_emitted_photons_incoh = flux_emitted_photons-
        flux_emitted_photons_laser_coherent;%flux in ps^(-1)
176
177 %% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
178 %%%%%%%%%%%%%%%%%%%%%%%%%%%%  Evaluation of g1(tau) %%%%%%%%%%%%%%%%%%%%%%%%
179 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
180 %
181 % For a given output field operator b, the two-time first-order
182 % auto-correlation function, g1(t1,t2), also called the "degree of
183 % first-order coherence", is defined by:
184 %      g1(t1,t2)= < b'(t2) b(t1) > / sqrt[ <b'b>(t2) <b'b>(t1) ]
185 % Such a quantity plays a major role in any experiment using an
186 % interference between two time-delayed components of an optical field (as
187 % in a Michelson or Mach-Zender setup ensuring the interference of a short
188 % and long path). It also plays a huge role in the calculation of optical
189 % spectra (as in photoluminescence or resonance fluorescence experiments),
190 % due to the Wiener-Khinchine theorem which directly links the optical
191 % spectra with the Fourier Transform of the first-order autocorrelation
192 % function.
193 %
194 % To calculate such a quantity, one needs to use the "Quantum Regression
195 % Theorem". This theorem allows making the link between the Heisenberg
196 % representation of a two-time correlation function < A(t2) B(t1) >, where
197 % operators A and B are time-dependant, and the Schrodinger approach that
198 % we have to use here  (where the density matrix varies). The "recipe" to
199 % deduce < A(t2)B(t1) > consists in:
200 %     - Letting the density matrix evolve from time 0 to t1
201 %     - Replacing rho(t1) by a fictitious density matrix B*rho(t1)
202 %     - Computing the evolution of this fictitious density matrix between
203 %     times t1 and t2
204 %- Calculating the expectation value of operator A using this fictitious
        density matrix
205 %
206 % Note that even if B*rho(t1) is not a real/valid density matrix (it's not
207 % even Hermitian), we can at least make it of the order of unity, to ensure
208 % an optimal numerical convergence (especially important in the pulsed
209 % regime where for example the operator b_out is extremely small at the
210 % beginning of the pulse). Looking at the definition of g1(t1,t2), we see
211 % that this is readily obtained by taking the operator B as b/sqrt( <b'b>),
212 % with the consequence that operator A has to be taken equal to b'/sqrt(
213 % <b'b> ).
214 %
215 % NB1: To read more about the Quantum Regression Theorem, and the validity
216 % of the "recipe": -->
217 % http://atomoptics-nas.uoregon.edu/~dsteck/teaching/quantum-optics/quantum
        -optics-notes.pdf
218 %     (see in particular Sec. 5.7.3, page 199)
219 % --> "Quantum Noise" by Gardiner & Zoller
220 %     (see in particular Eq. 5.2.11 and an alternative formulation in Sec.
221 %     5.2.3)
```

```
% ---> "Statistical Methods in Quantum Optics 1" by H. J. Carmichael
%     (see in particular Sec. 1.5 up to equations 1.97 and 1. 98)

% NB2: The approach used in the "g2CW_vs_delay" and "g2PR_vs_t1_t2"
% programs, to calculate second-order autocorrelation functions, is more
% focused on the physical/experimental definition of these quantities. It
% also makes use of real/normalized/valid density matrices, contrary to the
% fictitious density matrices used below. But the theory behind is also
% entirely linked to the use of the Quantum Regression Theorem.

% NB3: In the stationary regime, under CW excitation, one can take any
% initial time as time 0, and consider the density matrix of the stationary
% state at this time. Also, the flux <b'b> does not depend on time, hence
% the degree of coherence only depends on the delay tau through:
%      g1(tau)= < b'(tau) b(0) > / <b'b>(0)
% From this formula one can directly see that at zero delay g1(0) has to be
% equal to unity, indeed:
%      g1(0)= < b'b > / < b'b > = 1
% In addition, for very long delays one can guess that the field at delay
% tau=infty is completely uncorrelated from the field at delay zero, and we
% find that g1(infty) has to be equal to the coherent fraction of the
% optical field, corresponding to the fraction of light that has a
% well-defined amplitude and phase with respect to the incoming
% monochromatic laser. Indeed:
%      g1(infty) = < b' > < b > / < b'b >
% where we used < b'(infty) b(0) > = < b' (infty) > < b(0) >  (uncorrelated
% fields). Conversely, the difference between g1(0) and g(infty) gives the
% incoherent fraction of the optical field, i.e the fraction of intensity
% with no well-defined phase with respect to the incoming laser.


% Evaluation of g1(tau) for positive delays only, starting from the
% stationary-state density matrix at time 0.

tic %Start timer to evaluate the computation time
g_1_reflected_vs_tau = expect(b_out'/sqrt(flux_reflected_photons),mesolve(
    Liouvillian,b_out/sqrt(flux_reflected_photons)*
    density_matrix_stationary_state,tau_list)); %Equivalent to <
    b_out_normalized_dag(t) b_out_normalized(t+tau)> for t--> infinity
g_1_transmitted_vs_tau = expect(c_out'/sqrt(flux_transmitted_photons),
    mesolve(Liouvillian,c_out/sqrt(flux_transmitted_photons)*
    density_matrix_stationary_state,tau_list)); %Equivalent to <
    c_out_normalized_dag(t) c_out_normalized(t+tau)> for t--> infinity
```

```
259 g_1_emitted_vs_tau = expect(e_out'/sqrt(flux_emitted_photons),mesolve(
        Liouvillian,e_out/sqrt(flux_emitted_photons)*
        density_matrix_stationary_state,tau_list)); %Equivalent to <
        e_out_normalized_dag(t) e_out_normalized(t+tau)> for t--> infinity
260 toc % Stop timer
261
262 % Evaluation of g1(tau) for both negative and positive delays (
        full_tau_list).
263 full_g_1_reflected_vs_tau = [fliplr(conj (g_1_reflected_vs_tau(2:end-1)))
        g_1_reflected_vs_tau ];
264 full_g_1_transmitted_vs_tau = [fliplr(conj (g_1_transmitted_vs_tau(2:end-1)
        )) g_1_transmitted_vs_tau ];
265 full_g_1_emitted_vs_tau = [fliplr(conj (g_1_emitted_vs_tau(2:end-1)))
        g_1_emitted_vs_tau ];
266 % NB: It is ensured that the total number of points is a power of 2, as
        required for
267 % the calculation of spectra using the Fast Fourier Transform algorithm
268
269
270 %% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
271 %%%%%%%%%%%%%%%%%%%%%%%% Spectral densities of flux %%%%%%%%%%%%%%%%%%%%%%
272 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
273
274 % The optical spectrum of a CW optical field is characterized by its
275 % "spectral density of flux", S(omega), whose integral  over the whole
276 % spectrum gives the total photon flux. Computing such a spectrum requires
277 % using the "Wiener-Khinchine theorem", which states that the spectral
278 % density of flux, for a field described by operator "b", is the Fourier
279 % Transform of the non-normalized autocorrelation function vs delay :
280 %    S(\omega) = (1/2pi)* \int < b'(\tau) b(0) > exp(- i \omega \tau) d\tau
281 % The normalization of this quantity implies that indeed the integral of
282 % S(omega) is the total flux:
283 %       \int S(omega) d\omega = < b'(0) b(0) > = total flux
284 %
285 % As we saw above, however, the quantity < b'(tau) b > is the sum of two
286 % contributions:
287 %   - A contribution  < b' > < b >, corresponding to the coherent part of
288 %   the flux.
289 %     For monochromatic (i.e. infinitely coherent) CW laser excitation,
290 %     this contribution never decays even at infinite delay (constant value
291 %     < b' > < b >).
292 %   - Complementary, a contribution < b'(tau)b > - < b' > < b >, induced
293 %   by the incoherent
294 %     part of the flux. This contribution tends towards zero for large
295 %     delays.
296 %
```

```
% Thus, after Fourier transforming < b'(tau) b > to obtain the spectrum:
%   - The first constant contribution leads to a delta function centered on
%   omega_laser,
%     with an area given by < b' > < b >: this  is the coherent part of the
%     optical field, which is monochromatic and oscillating at the same
%     frequency as the laser (with a phase and amplitude given by the
%     complex number < b > ).
%   - The decaying contribution leads to a continuous spectrum, which can
%   not interfere
%     with the incoming laser since it is not monochromatic: this is the
%     spectrum of the incoherent part of the optical field, whose total
%     area is the incoherent flux <b'b> - <b'><b>. Typically, in the
%     weak-coupling regime this incoherent spectrum has the shape of a
%     single Lorentzian peak at low power, but of a Mollow triplet at high
%     power. This is this incoherent part of the spectrum that is
%     calculated here, by Fourier transforming < b'(tau) b > - < b' >< b >
%     as a function of the delay tau.
%
% NB1: To know more about Wiener-Khinchin theorem and the computation of
% optical spectra:
% ---> http://atomoptics-nas.uoregon.edu/~dsteck/teaching/quantum-optics/
    quantum-optics-notes.pdf
%     (see in particular Chap. 2 for Wiener-Khinchin theorem, and Sec. 5.7
%     for resonance fluorescence)
% ---> "The Quantum Theory of Light" by Rodney Loudon
%     (see in particular Chap. 3 for classical optics, and Chap. 8 for
%     resonance fluorescence spectra)
% ---> "Statistical Methods in Quantum Optics 1" by Howard Carmichael
%     (see in particular Sec. Z.3.4)
%
% NB2: We choose to define a spectral density in terms of the photon energy
% (in mueV), instead of frequency (in ps^-1) or angular frequency (in
% rad/ps). As a flux is measured here in ps^-1 (number of photons per unit
% time), the dimension of our spectral densities has to be in ps^-1 / mueV.
% Indeed each spectral contribution to the total flux is obtained through
% multiplying the spectral density of flux (in ps^-1 / mueV) by the photon
% energy step (in mueV).
%
% NB3: The default technique used here is the Fast Fourier Transform (FFT)
% algorithm, which can provide accurate spectra in a very fast way provided
% we use a number of points that is a power of 2 (same number of points in
% the time and frequency domain). The spectral width of the FFT spectrum
% (here denoted as "FFT_sampling_frequency") and its resolution (here
% related to the step in angular frequency, "omega_step") are fixed by the
% width and resolution used in in the time domain (related to "tau_max" and
% "tau_step" characterizing the "full_tau_list"). In practice, we are
```

```
342 % interested only in a spectral window of width
343 % "width_spectral_window_muev", and the list of angular frequencies omega
344 % in "omega_list" (that we use to plot quantities "versus omega" and
345 % calculate integrals) is just a subset of the full FFT spectrum, centered
346 % on omega_laser. Also note that the raw FFT spectra have to be adequately
347 % transformed into real physical quantities:
348 %    a) Proper normalization of the FFT result has to be ensured, so that
349 %    the integral of the
350 %       spectral density indeed corresponds to the optical flux in ps^-1
351 %       (or more precisely the incoherent component of the optical flux, as
352 %       discussed above).
353 %    b) One needs to ensure that the optical spectra are centered on the
354 %    laser frequency/photon
355 %       energy, to take into account the  fact that we worked in the
356 %       rotating frame at this laser frequency. Using the "fftshift"
357 %       function just ensures that the first half of the FFT spectrum
358 %       corresponds to negative frequencies, and the second half to
359 %       positive frequencies. In addition, we ensure that the lists of
360 %       angular frequencies ("omega_list_full_spectrum" for the complete
361 %       list in rad/ps, "omega_list" for the selected spectral window in
362 %       rad/ps, and "omega_list_eV" for the selected energy window in eV)
363 %       are defined to be centered. on the laser frequency/photon energy.
364 %    c) The list of delays in "full_tau_list" includes negative and
    positive
365 %    delays, with the
366 %       zero delay in the middle of the list. But the FFT algorithm
367 %       considers that this is the first point of the list which is the
368 %       time zero, which leads to a frequency-dependent phase shift along
369 %       the full FFT spectrum. This phase shift has to be compensated (the
370 %       spectral density of flux is a real quantity), which is done through
371 %       a multiplication by "phase_shift_compensation_full_FFT_spectrum".
372 % --> All the angular frequency lists and quantities related to the
373 % calculation of the FFT spectra
374 %     are defined in the "Init_lists_..." subprogram, for clarity.
375
376
377 % Computing the spectral densities of flux over the full frequency spectrum
378 spectral_density_flux_reflected_photons_incoh_full_spectrum = 1/(2*pi)*
    fftshift(fft(full_g_1_reflected_vs_tau*flux_reflected_photons-
    flux_reflected_photons_laser_coherent,nb_points_full_spectrum)./
    phase_shift_compensation_vs_omega_full_spectrum)/FFT_sampling_frequency
     * (ev/hbar*1e-18); % spectral density in ps^-1 / muev
```

```
379 spectral_density_flux_transmitted_photons_incoh_full_spectrum = 1/(2*pi)*
        fftshift(fft(full_g_1_transmitted_vs_tau*flux_transmitted_photons-
        flux_transmitted_photons_laser_coherent,nb_points_full_spectrum)./
        phase_shift_compensation_vs_omega_full_spectrum)/FFT_sampling_frequency
         * (ev/hbar*1e-18); % spectral density in ps^-1 / muev
380 spectral_density_flux_emitted_photons_incoh_full_spectrum = 1/(2*pi)*
        fftshift(fft(full_g_1_emitted_vs_tau*flux_emitted_photons-
        flux_emitted_photons_laser_coherent,nb_points_full_spectrum)./
        phase_shift_compensation_vs_omega_full_spectrum)/FFT_sampling_frequency
         * (ev/hbar*1e-18); % spectral density in ps^-1 / mueV
381
382 % Spectral densities of flux as a function of omega, limited to the
383 % selected spectra window (range of angular frequencies defined by "omega
384 % list")
385 spectral_density_flux_reflected_photons_incoh_vs_omega =
        spectral_density_flux_reflected_photons_incoh_full_spectrum(
        index_min_zoomed_spectrum:index_max_zoomed_spectrum); % spectrum
        restricted to the selected spectral window, in ps^-1 / mueV
386 spectral_density_flux_transmitted_photons_incoh_vs_omega =
        spectral_density_flux_transmitted_photons_incoh_full_spectrum(
        index_min_zoomed_spectrum:index_max_zoomed_spectrum); % spectrum
        restricted to the selected spectral window, ps^-1 / mueV
387 spectral_density_flux_emitted_photons_incoh_vs_omega =
        spectral_density_flux_emitted_photons_incoh_full_spectrum(
        index_min_zoomed_spectrum:index_max_zoomed_spectrum);% spectrum
        restriced to the selected spectral window, in ps^-1 / mueV
388
389
390 %%
391 % %%% Alternative method: Fourier transform by explicit calculation of the
392 % Fourier integrals % This method is very inefficient in terms of computing
393 % time, yet is crucial to allow verifying % the results obtained with the
394 % FFT algorithm: it must give the same result as the FFT method.
395 %
396 % tic  % Start timer to evaluate the computation
397 %
398 % for omega_index=1:nb_points_spectrum
399 %     spectral_density_flux_reflected_photons_incoh_vs_omega(omega_index) =
         1/(2*pi)* sum ( (full_g_1_reflected_vs_tau*flux_reflected_photons-
        flux_reflected_photons_laser_coherent) .* exp(-1i* (omega_list(
        omega_index) - omega_laser) .*full_tau_list )) * tau_step * (ev/hbar*1e
        -18) ; % spectral density in ps^-1 / mueV
```

```
%     spectral_density_flux_transmitted_photons_incoh_vs_omega(omega_index)
      = 1/(2*pi)* sum ( (full_g_1_transmitted_vs_tau*
    flux_transmitted_photons-flux_transmitted_photons_laser_coherent) .*
    exp(-1i* (omega_list(omega_index) - omega_laser) .*full_tau_list )) *
    tau_step * (ev/hbar*1e-18); % spectral density in ps^-1 / mueV
%     spectral_density_flux_emitted_photons_incoh_vs_omega(omega_index) =
    1/(2*pi)* sum ( (full_g_1_emitted_vs_tau*flux_emitted_photons-
    flux_emitted_photons_laser_coherent) .* exp(-1i* (omega_list(
    omega_index) - omega_laser) .*full_tau_list )) * tau_step * (ev/hbar*1e
    -18); % spectral density in ps^-1 / mueV
% end
%
% toc   % Stop timer


%% %%%%%%%%% Plots %%%%%%%%

Plot2LevelG1SDCWvsDelayAndFrequency;
```

C.1.4 First-order correlation and Wigner Distribution Function in PW

```
clear
clc
close all

%% Important note: these paths must be modified if needed
addpath(genpath('..\QotoolboxV015'));
addpath(genpath('..\CQED subprograms'));
addpath(genpath('..\CQED device parameters'))
savepath

% In addition, for the mesolve function to operate the executable files
% (.exe) and batch files (.bat) contained in  '[...] \QotoolboxV015\bin'
% have to be copied to  a folder that is on the Windows system path, in the
% main hard  drive where Windows is installed. This can be for example in:
% 'C:\Program Files\Matlab\R2014a\bin'.

% Warning: for the adiabatic version to converge, the tolerance in
% mesolve.m function must be reduced compared to the defaut values. For
% example:
% ode2file('ode_input.dat',L,rho0,t_list,struct('reltol',7e-8,'abstol',8e
    -8));


%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%  g1SDPR : g1(t1,t2) and spectral densities in PR  %%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This section labeled "g1SDPR" studies the resonant resonance fluorescence
%spectra in pulsed regime, by prior evaluation of the normalized
%correlation function g1(t1,t2). By setting t=(t1+t2)/2 and tau = t2-t1, an
%interpolated (unnormalized) G1(t,tau) is computed from G1(t1,t2). The
%Fourier Transform is evaluated over tau to find the associated
%Wigner-Ville-Distribution WDF(t,omega). To check its normalization,
%WDF(t,omega) is integrated over omega and compared with the photon fluxes.
%WDF is integrated over t for determining the spectral energy density
%ESD(omega), which is what is measured by the spectrometer. It is verified
%that the area of this spectrum gives the total number of photons. At last,
%the spectra of the total and the coherent component of the fluxes are
%compared for the reflected, transmitted, diffracted and emitted fields.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Choice of full model 'F' or adiabatic model 'A'
model = 'A';

%% Experimental conditions (to be edited)
detuning_QD_C_muev = 0; %Detuning between the QD and cavity frequencies, in
     mueV
detuning_pulse_QD_muev = 0; %Detuning between the pulse central frequency
    and the QD frequency, in mueV
eta_in = 1; % Injection efficiency for the incoming photons (depends on
    experimentally-achieved spatial coupling)
Nb_photons_pulse = 1;  % Average number of incoming photons in a pulse.
    This quantity should be multiplied by eta_in to know the number of
    incoming photons actually coupled to the optical mode
FWHM_pulse = 15; %in ps, full width at half-maximum of the incoming
    Gaussian pulse intensity (unit: ps since angular frequencies are in rad
    /ps)
% Parameters for the evaluation of the temporal evolution
%
% NB1: the maximum delay "t_max_ps" will also dictate the frequency
% resolution of the spectra, given by the angular frequency step
% "omega_step". The corresponding angular frequency lists are defined in
% the "Init_lists_..." subprogram (see also below details on the
% calculation and Fast Fourier Transform (FFT) algorithm)
%
% NB2: one should be careful that "t_max_ps" is large enough to include a
% good approximation of "infinite delays" (check that the g1(tau) function
% has had enough time to truly converge), while keeping a number of points
% large enough to ensure a good temporal resolution. This is especially
% important for high input powers where artifacts can appear.
```

```
62 t_delay = 2*FWHM_pulse; % Time at which the pulse is maximally intense, so
       that the computation starts when the pulse has not arrived yet
63 t_max_ps = (t_delay + 4*FWHM_pulse)*5; % Final time where we stop the
       computation and plots of time evolutions
64 nb_points_time = 2^7; % Time resolution/Number of iterations / <100000
       otherwise the integrating the master equation gets difficult (odesolve)
       )
65 % --> This should be such that 4*nb_points_time-3 is less than a power of 2
66 % if one wants to reduce the amount of zero padding, the latter being
67 % required for FFT optimization.
68 t_min = 0.1*FWHM_pulse; % Initial time considered for the computations and
       plots of time evolutions
69 width_spectral_window_muev = 300; % width of the spectral window to be
       displayed, centered on omega_pulse
70 %% Initialization of parameters, operators, arrays, etc...
71 Init2levelDeviceParametersOngoingTest;
72 Init2levelHilbertSpaceAndOperators;
73 InitMaps2levelG1wdfPR;
74 %%
75 % Definition of the input field in ps^(-1/2), in the form of a fseries (
       necessary for integrating the master equation)
76 Standard_deviation_b_in_PR = FWHM_pulse/(2*sqrt(log(2))); %Deduced from the
        properties of a Gaussian function
77 b_in_fn = fn('gauss',t_delay,Standard_deviation_b_in_PR) * sqrt( eta_in*
       Nb_photons_pulse / ( sqrt(pi) * Standard_deviation_b_in_PR ) ); %
       square root of the incoming photon number per time unit, in ps-1/2
78 b_in_vs_time = fsval(b_in_fn,t_list); %scalar array representing b_in vs
       time
79
80 % Initial density matrix before the pulse has started
81 switch model
82     case 'F' %Full model
83         psi0 = tensor(Vacuum_state,g_ket); % Initial state: tensorial
               product of photonic vacuum and QD ground state
84         rho0 = psi0*psi0'; % Density matrix corresponding to the initial
               pure state
85     case 'A' % Adiabatic model
86         rho0 = g_ket*g_ket'; % Density matrix corresponding to the initial
               pure state
87 end
88
89 %% %%%%%%%% System Hamiltonian and time-dependent operators  %%%%%%%%%%%
90 %
91 % The system Hamiltonian is time-dependent due to the function b_in_fn
92 % describing the input field b_in(t).
93
```

```
94  % In addition, in the case of adiabatic elimination of the cavity mode an
       effective
95  % operator "a" is defined, acting on the QD subspace, based on the formula
       for
96  % adiabatic elimination. Since this formula depends
97  % on b_in(t), we define a time-dependent quantity "a_vs_time", which is an
98  % array containing, for each time of t_list, the corresponding operator
99  % "a".
100 % In the full model case, to simplify the following calculations, we define
       the
101 % same quantity a_vs_time, yet this time this array contains the same
       operator
102 % (annihilation operator "a" acting on the cavity subspace), replicated for
       all
103 % times of t_list.
104
105 switch model
106         case 'A' % Adiabatic model
107
108             Delta = 2*(omega_pulse-omega_c)/kappa; %normalized pulse
                    detuning
109
110             %%%%%%%%% Definition of the adiabatic-model Hamiltonian (in the
                     frame rotating at angular frequency omega_pulse)
111             H_PR = (omega_eff-omega_pulse)*sigma_dag*sigma - 1i*sqrt(
                    Gamma_0*eta_top)*...
112                 ((1-1i*Delta)^(-1)*b_in_fn*sigma_dag-(1+1i*Delta)^(-1)*
                        b_in_fn'*sigma); % Hamiltonian
113
114             %%% Definition of the time-dependent operator "a_vs_time"
                    acting in the QD subspace
115             a_vs_time = -2*(kappa*(1-1i*Delta_QDC))^(-1)*g*sigma*Id_vs_time
                    -2*sqrt(kappa_top)*(kappa*(1-1i*Delta))^(-1)*fsval(b_in_fn,
                    t_list).*Id_vs_time; %annihilation operator a in adiabatic
                    approximation, as fseries
116
117         case 'F'  % Full model
118
119             %%%%%%%%% Definition of the full-model Hamiltonian (in the
                    frame rotating at angular frequency omega_pulse)
120             H_PR = (omega_d-omega_pulse)*sigma_dag*sigma...
121                  + (omega_c-omega_pulse)*a_dag*a...
122                  + 1i*g*(sigma_dag*a-a_dag*sigma)...
123                  - 1i*sqrt(kappa_top)*b_in_fn*(a_dag-a);
124
```

```
125             % Definition of the operator a_vs_time, even though "a" is
                    constant in the full model, to reduce the number of "switch
                    " in the following code
126              a_vs_time = a*Id_vs_time;
127
128 end % end of the "switch model"
129
130 %Superoperator associated to the coherent processes (Hamiltonian)
131 L_coh = -1i * (spre(H_PR) - spost(H_PR));
132
133 %%%%%%%%% Calculation of the Liouvillian superoperator
134 Liouvillian  = L_coh + L_incoh; % Total Liouvillian superoperator including
        both coherent processes (Hamiltonian) and incoherent processes (
    dissipative jumps)
135
136 %%%%%%%%%% Numerical Integration of the Master Equation %%%%%%%%
137
138 % Computation of preliminary time evolution, between 0 (long before the
    pulse)
139 % and t_min (time at which we want to start plotting and integrating the
    physical
140 % quantities. Such a computation can be performed with very low time
    resolution,
141 % i.e. the corresponding t_list_before_t_min has a very low number of
    points,
142 % since the density matrix almost doesn't evolve between 0 and t_min.
143 %
144 % NB: such a preliminary time evolution is mandatory to avoid having
    strictly
145 % zero expectation values for some quantities (such as the input or output
    fields),
146 % during the following time evolution between t_min and t_max). Indeed,
    zero
147 % expectation values lead to NaN errors when used in normalizing physical
148 % quantities, such as conditional density matrices (see below), or
149 % Stokes/Bloch coordinates in the Poincare'/Bloch sphere.
150
151 density_matrix_vs_time_before_t_min = mesolve(Liouvillian,rho0,
    t_list_before_t_min); %master equation solver based on odesolve: first
    evolution of the system
152 density_matrix_at_t_min = density_matrix_vs_time_before_t_min{
    nb_points_time_before_t_min};
153
154 % Computation of the density matrix vs time with t_list, i.e. between t_min
155 % and t_max, requiring a large enough time resolution.
```

```
156 density_matrix_vs_time = mesolve(Liouvillian,density_matrix_at_t_min,t_list
        ); % second evolution of the system
157
158 %%%%%%%%%%%%%%%%%%% Definition of the output operators %%%%%%%%%%%%%%%%%%%%
159 % These are general formulas for both the adiabatic and full model,
        depending on "a_vs_time"
160
161 b_out_vs_time = b_in_vs_time*Id_vs_time + sqrt(kappa_top)*a_vs_time; %
        definition of the operator b_out, i.e. the output operator for the
        reflected light, in ps^(-1/2)
162 c_out_vs_time = sqrt(kappa_bottom)*a_vs_time; % definition of the operator
        c_out_vs_time, i.e. the output operator for the transmitted light, in
        ps^(-1/2)
163 d_out_vs_time = sqrt(kappa_loss)*a_vs_time; % UNUSED HERE: definition of
        the operator d_out_vs_time, i.e. the output operator for the diffracted
        /lost light, in ps^(-1/2)
164 % e_out = sqrt(gamma_sp)*sigma %  output operator for the light
        spontaneously emitted outside the cavity mode, in ps^(-1/2), already
        defined in Init_2level_Hilbert_space_and_operators.m
165
166 % Calculation of the total photon flux as a function of time, for the
        various optical fields
167 flux_injected_photons_vs_time = b_in_vs_time.^2; % total flux of injected
        photons taking into account  eta_in (so only the photons coupled to the
         cavity mode), in ps(-1)
168 flux_reflected_photons_vs_time = real(expect(b_out_vs_time'*b_out_vs_time,
        density_matrix_vs_time)); %flux in ps^(-1)
169 flux_transmitted_photons_vs_time = real(expect(c_out_vs_time'*c_out_vs_time
        ,density_matrix_vs_time)); %flux in ps^(-1)
170 flux_diffracted_photons_vs_time = real(expect(d_out_vs_time'*d_out_vs_time,
        density_matrix_vs_time)); %flux in ps^(-1)
171 flux_emitted_photons_vs_time = real(expect(e_out'*e_out,
        density_matrix_vs_time)); %flux in ps^(-1)
172
173 % Calculation of the total coherent photon flux as a function of time, for
        the various optical fields
174 flux_reflected_photons_pulse_coherent_vs_time= abs(expect(b_out_vs_time,
        density_matrix_vs_time)).^2; % coherent flux = <b_out'(t)><b_out(t)> in
         ps^{-1}
175 flux_transmitted_photons_pulse_coherent_vs_time=abs(expect(c_out_vs_time,
        density_matrix_vs_time)).^2; % coherent flux = <c_out'(t1)><c_out(t2)>
        in ps^{-1}
176 flux_diffracted_photons_pulse_coherent_vs_time=abs(expect(d_out_vs_time,
        density_matrix_vs_time)).^2; % coherent flux = <d_out'(t1)><d_out(t2)>
        in ps^{-1}
```

```
flux_emitted_photons_pulse_coherent_vs_time=abs(expect(e_out,density_matrix_vs_time)).^2; % coherent flux = <e_out'(t1)><e_out(t2)> in ps^{-1}

%Computing the number of reflected/transmitted/emitted photons, by integrating over all times in t_list
Nb_reflected_photons = trapz(t_list,flux_reflected_photons_vs_time);
Nb_transmitted_photons = trapz(t_list,flux_transmitted_photons_vs_time);
Nb_diffracted_photons = trapz(t_list,flux_diffracted_photons_vs_time);
Nb_emitted_photons = trapz(t_list,flux_emitted_photons_vs_time);

%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%  Evaluation of g1(t1,t2) %%%%%%%%%%%%%%%%%%%%%%%%%%%%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% For a given output field operator b, the two-time first-order auto-correlation function,
% g1(t1,t2), also called the "degree of first-order coherence", is defined by:
%      g1(t1,t2)= < b'(t1) b(t2) > / sqrt[ <b'(t1)> <b(t2)> ]
% %
% To calculate such a quantity, one can use the Heisenberg representation of a two-time
% correlation function < A(t1) B(t2) >, where operators A and B are time-dependent, and
% the Schroedinger approach that we have to use here  (where the density matrix varies).
% The "recipe" to deduce < A(t1)B(t2) > consists in:
%     - Letting the density matrix evolve from time 0 to t1
%     - Replacing rho(t1) by a fictitious density matrix rho(t1)*A
%     - Computing the evolution of this fictitious density matrix between times t1 and t2
%     - Calculating the expectation value of operator B using this fictitious density matrix
%
% Note that even if rho(t1)*A is not a real/valid density matrix (it's not even Hermitian),
% we can at least make it of the order of unity, to ensure an optimal numerical convergence
% (especially important in the pulsed regime where for example the operator b_out is
```

```
205 % extremely small at the beginning of the pulse). Looking at the definition
        of g1(t1,t2),
206 % we see  that this is readily obtained by taking the operator B as b/sqrt(
        <b'b>), with
207 % the consequence that operator A has to be taken equal to b'/sqrt( <b'b> )
        .
208
209 % NB1: The approach used in the "g2CW_vs_delay" and "g2PR_vs_t1_t2"
        programs, to calculate
210 % second-order autocorrelation functions, is more focused on the physical/
        experimental
211 % definition of these quantities. It also makes use of real/normalized/
        valid density matrices,
212 % contrary to the fictitious density matrices used below. But the theory
        behind is also
213 % entirely linked to the use of the Quantum Regression Theorem.
214
215 %%%%%%%%%%  Calculation of fictitious density matrix rho(t1)*A  %%%%%%%%%%
216 % In the following the "density_matrix_times_OPERATOR_dag_vs_t1"-s are
217 % defined for each value of time t1 in t_list.
218
219 % NB: notice the normalization  b'/sqrt( <b'b> ) to optimize numerical
220 % convergence
221 density_matrix_times_b_out_dag_vs_t1 = density_matrix_vs_time*b_out_vs_time
        './sqrt(flux_reflected_photons_vs_time);
222 density_matrix_times_c_out_dag_vs_t1 = density_matrix_vs_time*c_out_vs_time
        './sqrt(flux_transmitted_photons_vs_time);
223 density_matrix_times_d_out_dag_vs_t1 = density_matrix_vs_time*d_out_vs_time
        './sqrt(flux_diffracted_photons_vs_time);
224 density_matrix_times_e_out_dag_vs_t1 = density_matrix_vs_time*e_out'./sqrt(
        flux_emitted_photons_vs_time);
225
226 % NB: "tic" is used as a "start" time for the measurement of the computing
        time between "tic" and "toc"
227 %tic
228
229 %Cycle over all times t1 in t_list, corresponding to the moment where a
        first click occurred
230 for t1_index = 1:nb_points_time
231
232     % Both t1 and t2 are values in t_list. However, to compute the
233     % "fictitious density matrix B*rho(t1)" vs t2, we consider
234     % only t2 >= t1, and we need to  deal with the fact that there are less
235     % and less remaining values of t2 in t_list, when t1 increases. To keep
236     % all quantities defined in the full t_list, for each time t1 < t2 we
237     % have a "zero" density matrix, i.e. a fictitious density matrix with
```

```
238     % only zero elements—
239
240     % Incrementing the array of zero density matrices, each time t1_index
241     % is increased, to fill the density matrix for times t1 < t2
242
243     if (t1_index >=2) % No need to include a zero density matrix at the
            first value of t1
244         zero_density_matrix_vs_t2_before_t1{t1_index—1} = 0*Id; %null
                density matrix with the same dimensions of the involved Hilbert
                 space
245     end
246
247     % Array of normalized density matrices, conditioned on the detection of
             a
248     % click at time t1, as a function of t2 >= t1 (and zero otherwise)
249
250     % NB: notice the normalization  b'/sqrt( <b'b> ) to optimize numerical
251     % convergence
252     density_matrix_times_b_out_dag_vs_t2 = [
            zero_density_matrix_vs_t2_before_t1  mesolve(Liouvillian,
            density_matrix_times_b_out_dag_vs_t1{t1_index},t_list(t1_index:end)
            )]./sqrt(flux_reflected_photons_vs_time);
253     density_matrix_times_c_out_dag_vs_t2 = [
            zero_density_matrix_vs_t2_before_t1  mesolve(Liouvillian,
            density_matrix_times_c_out_dag_vs_t1{t1_index},t_list(t1_index:end)
            )]./sqrt(flux_transmitted_photons_vs_time);
254     density_matrix_times_d_out_dag_vs_t2 = [
            zero_density_matrix_vs_t2_before_t1  mesolve(Liouvillian,
            density_matrix_times_d_out_dag_vs_t1{t1_index},t_list(t1_index:end)
            )]./sqrt(flux_diffracted_photons_vs_time);
255     density_matrix_times_e_out_dag_vs_t2 = [
            zero_density_matrix_vs_t2_before_t1  mesolve(Liouvillian,
            density_matrix_times_e_out_dag_vs_t1{t1_index},t_list(t1_index:end)
            )]./sqrt(flux_emitted_photons_vs_time);
256
257
258     % Evaluation of g1(t1,t2) as described in the general notes above, for
            t2 >= t1 (and zero otherwise)
259     % NB: the normalization by the photon flux is already considered in
            density_matrix_times_OPERATOR_dag_vs_t2
260     g1_reflected_vs_t1_t2(t1_index,:) = expect(b_out_vs_time,
            density_matrix_times_b_out_dag_vs_t2);
261     g1_transmitted_vs_t1_t2(t1_index,:) = expect(c_out_vs_time,
            density_matrix_times_c_out_dag_vs_t2);
262     g1_diffracted_vs_t1_t2(t1_index,:) = expect(d_out_vs_time,
            density_matrix_times_d_out_dag_vs_t2);
```

```
263     g1_emitted_vs_t1_t2(t1_index,:) = expect(e_out,
            density_matrix_times_e_out_dag_vs_t2);
264 end
265 %%
266 %%%%% Completion of previous partially-calculated maps vs t1,t2, to include
        the case where t2 < t1
267
268 % Due to the symmetry between t1 and t2, we have to use properties like g1(
        t1,t2)=g1(t2,t1)* to fill
269 % the voids in the quantities that we have only partially calculated yet
270 % (since we systematically considered a zero value when t2 < t1. For each
271 % map function of t1 and t2, this completion is obtained by adding it to
272 % its transpose conjugate (to replace the zeros at t2 < t1) and dividing by
273 % 2 the elements along the diagonal (to avoid counting twice the case where
274 % t2=t1). This is done via an ad-hoc matrix idx below:
275
276 idx = ones(nb_points_time)-0.5*diag(ones(nb_points_time,1)); % to divide by
        2 the elements along the diagonal
277
278 % Completed g1(t1,t2) for the various optical fields
279
280 g1_reflected_vs_t1_t2 = (g1_reflected_vs_t1_t2+g1_reflected_vs_t1_t2').*idx
        ;
281 g1_transmitted_vs_t1_t2 = (g1_transmitted_vs_t1_t2+g1_transmitted_vs_t1_t2
        ').*idx;
282 g1_diffracted_vs_t1_t2 = (g1_diffracted_vs_t1_t2+g1_diffracted_vs_t1_t2').*
        idx;
283 g1_emitted_vs_t1_t2 = (g1_emitted_vs_t1_t2+g1_emitted_vs_t1_t2').*idx;
284
285 % G1 correlation vs (t1,t2), e.g. <b_out'(t1) b_out(t2)>
286 G1_reflected_vs_t1_t2 = g1_reflected_vs_t1_t2.*(
        flux_reflected_photons_vs_time'*flux_reflected_photons_vs_time).^0.5;
287 G1_transmitted_vs_t1_t2 = g1_transmitted_vs_t1_t2.*(
        flux_transmitted_photons_vs_time'*flux_transmitted_photons_vs_time)
        .^0.5;
288 G1_diffracted_vs_t1_t2 = g1_diffracted_vs_t1_t2.*(
        flux_diffracted_photons_vs_time'*flux_diffracted_photons_vs_time).^0.5;
289 G1_emitted_vs_t1_t2 = g1_emitted_vs_t1_t2.*(flux_emitted_photons_vs_time'*
        flux_emitted_photons_vs_time).^0.5;
290
291 % coherent component cross product, e.g <b_out'(t1)> <b_out(t2)>
292 expect_b_out_dag_t1_times_expect_b_out_t2 =  expect(b_out_vs_time',
        density_matrix_vs_time).'*expect(b_out_vs_time,density_matrix_vs_time);
293 expect_c_out_dag_t1_times_expect_c_out_t2 =  expect(c_out_vs_time',
        density_matrix_vs_time).'*expect(c_out_vs_time,density_matrix_vs_time);
```

```
294 expect_d_out_dag_t1_times_expect_d_out_t2 =  expect(d_out_vs_time',
        density_matrix_vs_time).'*expect(d_out_vs_time,density_matrix_vs_time);
295 expect_e_out_dag_t1_times_expect_e_out_t2 =  expect(e_out',
        density_matrix_vs_time).'*expect(e_out,density_matrix_vs_time);
296
297 %% %%%%%%%%%%%%%% Wigner distribution function%%%%%%%%%%%%%%%%%%%%%%%%%
298 %The Wigner distribution function (WDF) is used in signal processing as a
299 %transform in time–frequency analysis. the Wigner distribution function
300 %provides the highest possible temporal vs frequency resolution which is
301 %mathematically possible within the limitations of uncertainty in quantum
302 %wave theory. Given a non–stationary autocorrelation function C(t1,t2), by
303 %defining t=(t1+t2)/2 and tau= t2–t1, Fourier transforming the lag is
304 %obtained :
305 %WDF(t,omega) = \int C(t+tau/2,t–tau/2)*exp(–i*tau*omega)) d\tau.
306 %More info can be found at https://en.wikipedia.org/wiki/
        Wigner_distribution_function
307 %Notice that tau is defined based on the definition of C(t1,t2) in this
308 %script, with opposite sign with respect to the wiki page one.
309 %% Mapping C(t1,t2) into C(t,tau)
310 %The Fourier transform of the WDF will be performed over tau/2 and not
311 %simply tau. This implies that in addition to mapping C(t1,t2) into
312 %C(t,tau), C(t1,t2) must be interpolated also not only over time and tau,
        but also
313 %over the tau/2. If this were not done, aliasing would occur due to
314 %under sampling.  For more information, section "More about" at
315 % https://www.mathworks.com/help/signal/ref/wvd.html
316 interpolation_2level_g1SDPR_vs_t1_t2;
317
318 %Side note:  MATLAB has implemented its own WDF since R2018b, however it is
319 %defined for a single time series signal x(t) and not C(t1,t2). Still, it
320 %can be used for comparison to evaluate the WDF of the coherent component
321 %of the flux. Below it is shown how it is done for the reflected coherent
322 %field.
323
324 % expect_b_out_vs_time_dag = expect(b_out_vs_time',density_matrix_vs_time);
325 % sampling_frequency = 1/t_step; % Sampling frequency ps^(–1)
326 % [WVD_b_out_vs_time_coherent,frequency_WVD,t_list_WVD] = wvd(
        expect_b_out_vs_time_dag,sampling_frequency); %returns the smoothed
        pseudo Wigner–Ville distribution
327 % nb_points_WVD_spectrum = nb_points_time;
328 % omega_step_WVD = 2*pi*(sampling_frequency/2)/nb_points_WVD_spectrum;
329 % omega_step_WVD_muev = omega_step_WVD/ev*hbar/1e–18; %in mueV
330 % omega_spectrum_WVD = omega_pulse + frequency_WVD*2*pi; % rad/ps, arrays
        of angular frequencies. Obs: the zero value from fft corresponds to
        omega_laser
```

```
331 % omega_spectrum_WVD_muev = omega_spectrum_WVD/ev*hbar/1e-18; % mueV,
      arrays of angular frequencies, zero-centered
332 % figure,
333 % surf(t_list_WVD,omega_spectrum_WVD_muev - omega_pulse_ev*1e6,
      WVD_b_out_vs_time_coherent/(2*pi/ev*hbar/1e-18),'EdgeColor','none')
334 % xlabel('time [ps]')
335 % ylabel('\omega-\omega_p [\mueV]')
336 % title('Wigner-Ville distribution reflected coherent photons - MATLAB')
337 % view(2)
338 % colorbar
339
340 %% Fourier Transforming over the delay tau to obtain the WDF
341 % The technique used here for the Fourier Transform is the Fast Fourier
      Transform (FFT) algorithm, which
342 % can provide accurate spectra in a very fast way provided we use a number
      of points that is
343 % a power of 2 (same number of points in the time and frequency domain).
      The spectral width of
344 % the FFT spectrum (here denoted as "FFT_sampling_frequency") and its
      resolution (here related
345 % to the step in angular frequency, "omega_step") are fixed by the width
      and resolution used in
346 % in the time domain (related to "t_max_ps" and "t_delay" characterizing
      the "full_tau_list").
347 % In practice, we are interested only in a spectral window of width "
      width_spectral_window_muev",
348 % and the list of angular frequencies omega in "omega_list" (that we use to
       plot quantities
349 % "versus omega" and calculate integrals) is just a subset of the full FFT
      spectrum, centered on
350 % omega_pulse. Also note that the raw FFT spectra have to be adequately
      transformed into real
351 % physical quantities:
352 %    a) Proper normalization of the FFT result has to be ensured, so that
      the integral of the
353 %       spectral density indeed corresponds to the optical flux in ps^-1 (
      or more precisely
354 %       the incoherent component of the optical flux, as discussed above).
355 %    b) One needs to ensure that the optical spectra are centered on the
      laser frequency/photon
356 %       energy, to take into account the  fact that we worked in the
      rotating frame at this laser
357 %       frequency. Using the "fftshift" function just ensures that the
      first half of the FFT
358 %       spectrum corresponds to negative frequencies, and the second half
      to positive frequencies.
```

```
359 %       In addition, we ensure that the lists of angular frequencies ("
    omega_list_full_spectrum"
360 %       for the complete list in rad/ps, "omega_list" for the selected
    spectral window in rad/ps,
361 %       and "omega_list_eV" for the selected energy window in eV) are
    defined to be centered.
362 %       on the laser frequency/photon energy.
363 %    c) The list of delays in "full_tau_list" include negative and positive
     delays, with the
364 %       zero delay in the middle of the list. But the FFT algorithm
    considers that this is the
365 %       first point of the list which is the time zero, which leads to a
    frequency-dependent
366 %       phase shift along the full FFT spectrum. This phase shift has to be
     compensated (the
367 %       spectral density of flux is a real quantity), which is done through
     a multiplication
368 %       by "phase_shift_compensation_full_FFT_spectrum".
369 % --> All the angular frequency lists and quantities related to the
    calculation of the FFT spectra
370 %     are defined in the "Init_lists_..." subprogram, for clarity.
371 %WDF-s for reflected field
372 WDF_interpolated_G1_reflected_vs_time_tau_full_spectrum = 1/(2*pi)*fftshift
    (fft(interpolated_G1_reflected_vs_time_tau,nb_points_full_spectrum,2)
    ,2)./(FFT_sampling_frequency)./
    phase_shift_compensation_vs_omega_full_spectrum*(ev/hbar*1e-18); % 1/(
    mueV*ps)
373 WDF_interpolated_reflected_coherent_vs_time_tau_full_spectrum = 1/(2*pi)*
    fftshift(fft(interpolated_reflected_coherent_vs_time_tau,
    nb_points_full_spectrum,2),2)./(FFT_sampling_frequency)./
    phase_shift_compensation_vs_omega_full_spectrum*(ev/hbar*1e-18); % 1/(
    mueV*ps)
374 WDF_interpolated_G1_reflected_vs_time_tau =
    WDF_interpolated_G1_reflected_vs_time_tau_full_spectrum(:,
    index_min_zoomed_spectrum:index_max_zoomed_spectrum); % 1/(mueV*ps)
375 WDF_interpolated_reflected_coherent_vs_time_tau =
    WDF_interpolated_reflected_coherent_vs_time_tau_full_spectrum(:,
    index_min_zoomed_spectrum:index_max_zoomed_spectrum); % 1/(mueV*ps)
376
377 %WDF-s for transmitted field
378 WDF_interpolated_G1_transmitted_vs_time_tau_full_spectrum = 1/(2*pi)*
    fftshift(fft(interpolated_G1_transmitted_vs_time_tau,
    nb_points_full_spectrum,2),2)./(FFT_sampling_frequency)./
    phase_shift_compensation_vs_omega_full_spectrum*(ev/hbar*1e-18); % 1/(
    mueV*ps)
```

```
379 WDF_interpolated_transmitted_coherent_vs_time_tau_full_spectrum = 1/(2*pi)*
        fftshift(fft(interpolated_transmitted_coherent_vs_time_tau,
        nb_points_full_spectrum,2),2)./(FFT_sampling_frequency)./
        phase_shift_compensation_vs_omega_full_spectrum*(ev/hbar*1e-18); % 1/(
        mueV*ps)
380 WDF_interpolated_G1_transmitted_vs_time_tau =
        WDF_interpolated_G1_transmitted_vs_time_tau_full_spectrum(:,
        index_min_zoomed_spectrum:index_max_zoomed_spectrum);% 1/(mueV*ps)
381 WDF_interpolated_transmitted_coherent_vs_time_tau =
        WDF_interpolated_transmitted_coherent_vs_time_tau_full_spectrum(:,
        index_min_zoomed_spectrum:index_max_zoomed_spectrum);% 1/(mueV*ps)
382
383 %WDF-s for emitted field
384 WDF_interpolated_G1_emitted_vs_time_tau_full_spectrum = 1/(2*pi)*fftshift(
        fft(interpolated_G1_emitted_vs_time_tau,nb_points_full_spectrum,2),2)
        ./(FFT_sampling_frequency)./
        phase_shift_compensation_vs_omega_full_spectrum*(ev/hbar*1e-18); % 1/(
        mueV*ps)
385 WDF_interpolated_emitted_coherent_vs_time_tau_full_spectrum = 1/(2*pi)*
        fftshift(fft(interpolated_emitted_coherent_vs_time_tau,
        nb_points_full_spectrum,2),2)./(FFT_sampling_frequency)./
        phase_shift_compensation_vs_omega_full_spectrum*(ev/hbar*1e-18); % 1/(
        mueV*ps)
386 WDF_interpolated_G1_emitted_vs_time_tau =
        WDF_interpolated_G1_emitted_vs_time_tau_full_spectrum(:,
        index_min_zoomed_spectrum:index_max_zoomed_spectrum);% 1/(mueV*ps)
387 WDF_interpolated_emitted_coherent_vs_time_tau =
        WDF_interpolated_emitted_coherent_vs_time_tau_full_spectrum(:,
        index_min_zoomed_spectrum:index_max_zoomed_spectrum);% 1/(mueV*ps)
388
389 %WDF-s for diffracted field
390 WDF_interpolated_G1_diffracted_vs_time_tau_full_spectrum = 1/(2*pi)*
        fftshift(fft(interpolated_G1_diffracted_vs_time_tau,
        nb_points_full_spectrum,2),2)./(FFT_sampling_frequency)./
        phase_shift_compensation_vs_omega_full_spectrum*(ev/hbar*1e-18); % 1/(
        mueV*ps)
391 WDF_interpolated_diffracted_coherent_vs_time_tau_full_spectrum = 1/(2*pi)*
        fftshift(fft(interpolated_diffracted_coherent_vs_time_tau,
        nb_points_full_spectrum,2),2)./(FFT_sampling_frequency)./
        phase_shift_compensation_vs_omega_full_spectrum*(ev/hbar*1e-18); % 1/(
        mueV*ps)
392 WDF_interpolated_G1_diffracted_vs_time_tau =
        WDF_interpolated_G1_diffracted_vs_time_tau_full_spectrum(:,
        index_min_zoomed_spectrum:index_max_zoomed_spectrum);% 1/(mueV*ps)
```

```
393 WDF_interpolated_diffracted_coherent_vs_time_tau =
        WDF_interpolated_diffracted_coherent_vs_time_tau_full_spectrum(:,
        index_min_zoomed_spectrum:index_max_zoomed_spectrum);% 1/(mueV*ps)
394
395 %% Spectral energy density and spectral energy flux
396 % The projection property (see Wiki page linked above) of the WDF(t,omega)
397 % guarantees that its
398 % integral over the spectrum omega gives the photon flux as a function of
399 % time, whereas the integral of WDF over time gives the energy spectral
400 % density of the flux, which is shown in the spectrometer.
401 %
402 % As we saw above, however, the quantity < b'(t,tau) b(t) > is the sum of
        two contributions:
403 %   - A contribution  < b'(t,tau) > < b(t) >, corresponding to the coherent
         part of the flux.
404 %   - Complementary, a contribution < b'(t,tau)b(t) > - < b'(t,tau) > < b(t
        ) >, induced by the incoherent
405 %     part of the flux. This contribution tends towards zero for large
        delays.
406
407 ESD_reflected_photons_vs_omega = real(sum(
        WDF_interpolated_G1_reflected_vs_time_tau,1))*time_step; % 1/mueV
408 ESD_coherent_reflected_photons_laser_vs_omega = real(sum(
        WDF_interpolated_reflected_coherent_vs_time_tau,1))*time_step; % 1/mueV
409 ESD_transmitted_photons_vs_omega = real(sum(
        WDF_interpolated_G1_transmitted_vs_time_tau,1))*time_step; % 1/mueV
410 ESD_coherent_transmitted_photons_laser_vs_omega = real(sum(
        WDF_interpolated_transmitted_coherent_vs_time_tau,1))*time_step; % 1/
        mueV
411 ESD_emitted_photons_vs_omega = real(sum(
        WDF_interpolated_G1_emitted_vs_time_tau,1))*time_step; % 1/mueV
412 ESD_coherent_emitted_photons_laser_vs_omega = real(sum(
        WDF_interpolated_emitted_coherent_vs_time_tau,1))*time_step; % 1/mueV
413 ESD_diffracted_photons_vs_omega = real(sum(
        WDF_interpolated_G1_diffracted_vs_time_tau,1))*time_step; % 1/mueV
414 ESD_coherent_diffracted_photons_laser_vs_omega = real(sum(
        WDF_interpolated_diffracted_coherent_vs_time_tau,1))*time_step; % 1/
        mueV
415
416 %% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
417 %%%%%%%%%%%%%%%%%%%%%%%%    Plots     %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
418 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
419
420 % For plot selection:
421 % - g1 vs (t1,t2) : 'g'
422 % - G1 vs (t1,t2) : 'G'
```

```
% - interpolated G1 vs (time, tau): 'I'
% - WDF vs (time, omega) : 'W'
% - photon fluxes : 'F'

% - plot_choice = ['G'];
plot_choice = ['g';'G';'I';'W';'F'];
Plot2levelG1wdfPR;
```

C.1.5 Second-order correlation in CW

```
clear
clc
close all
%% %%%%%      Common basis to every programs based "two levels" %%%%%%%%%%
% Important note: these paths must be modified if needed
addpath(genpath('..\QotoolboxV015'));
addpath(genpath('..\CQED_subprograms'));
addpath(genpath('..\CQED device parameters'))
savepath

% In addition, for the mesolve function to operate the executable
% files (.exe) and batch files (.bat) contained in  '[...] \QotoolboxV015\
    bin'
% have to be copied to  a folder that is on the Windows system path,
% in the main hard drive where Windows is installed. This can be
% for example in: 'C:\Program Files\Matlab\R2014a\bin'.

% Warning: for the adiabatic version to converge, the tolerance in
% mesolve.m function must be reduced compared to the defaut values. For
% example:
% ode2file('ode_input.dat',L,rho0,t_list,struct('reltol',7e-8,'abstol',8e
    -8));


%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%  g2CW : photon-photon correlations under CW excitation  %%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This section indexed "g2CW" computes the intensity correlations as a
% function of delay, i.e g2(tau), for the various optical fields. One
% checks that values of g2(0) and g2(infty) correspond to the
% expected ones, and that the photons emitted outside the mode are single
% photons (g2(0)=0). Depending on the detuning large bunchings with
% g2(0)>>1 can also be observed, on the transmitted field for example (cf
% PRL 101, 203602 2008) or in the strong-coupling regime (see for example
% Nature 575, 622-627(2019)).

```

```
% Warning: g2CW may not always lead to a proper numerical convergence,
% especially for very low incoming powers: this can be seen when we obtain
% absurd negative values of g(2) and/or noisy values of g2(infty) (instead
% of a smooth convergence to unity), and/or abrupt discontinuities in the
% g2(tau) function. This can usually be solved by adjusting the incoming
% power and/or the Fock space and/or the mesolve function (mainly the
% tolerances, and potentially the calculation algorithm). But it seems that
% the use of normalized density matrices (normalized = unity trace, even
% for conditional density matrices obtained after a detection event) is
% crucial to get the best numerical convergence.

% Choice of full model 'F' or adiabatic 'A'
model = 'F';

%%% Experimental conditions
detuning_QD_C_muev = 0; %Detuning between the QD and cavity frequencies, in
    mueV
eta_in = 1; % Injection efficiency for the incoming photons (depends on
    experimentally-achieved spatial coupling)
P_in_CW_pW=1000;% Incoming continuous-wave power in pW

Init2levelDeviceParametersOngoingTest;
Init2levelHilbertSpaceAndOperators;

%Incoming power
P_in_CW = P_in_CW_pW*1e-12;% Incoming power in W %%
b_in_CW = sqrt(eta_in*P_in_CW*1e-24/(hbar*omega_c)); % square root of the
    photon number per unit time, en ps-1/2

% Angular frequency of the incoming CW laser (fixed)
omega_laser_ev = omega_d_ev; % in eV
omega_laser = omega_laser_ev*ev/hbar*1e-12; % in rad/ps

%Parameters for the calculation of the time dynamics of g(2)(tau)
tau_max=800; % Maximal delay in ps (the minimal delay is fixed to 0 for
    calculations)
nb_points_time_g2CW=20000; %%% Time resolution/number of iterations for the
    curves G(2)(tau)
tau_step_g2CW=(tau_max)/(nb_points_time_g2CW-1); % Duration of a time step
tau_list_g2CW = linspace(0,tau_max,nb_points_time_g2CW); % list of all the
    positive delays considered in the computation and plots
full_tau_list_g2CW = [fliplr(-tau_list_g2CW(2:end)) , tau_list_g2CW ]; %
    list of all negative and positive delays for the plots

% tic NB: "tic" is used as a "start" time for the measurement of the
% computing time between "tic" and "toc"
```

```
74
75 switch model
76         case 'A' % Adiabatic case
77
78             Delta = 2*(omega_laser-omega_c)/kappa; %normalized laser
                   detuning appearing in Eq.12 of the pdf notes
79
80             %%%%%%%%% Definition of the adiabatic-model Hamiltonian (which
                   depends on omega_laser)
81             H_CW = (omega_eff-omega_laser)*sigma_dag*sigma...
82                  - 1i*sqrt(Gamma_0*eta_top)*(b_in_CW*sigma_dag/(1-1i*Delta)
                        -b_in_CW'*sigma/(1+1i*Delta)); % Hamiltonian (Eq.13 of
                         the pdf notes)
83
84
85             %%% For the redefinition of the operator "a" acting in the QD
                   space
86             % (Eq. 10 of the pdf notes
87             a = -2*g*sigma/(kappa*(1-1i*Delta_QDC))-2*sqrt(kappa_top)*
                   b_in_CW*Id/(kappa*(1-1i*Delta)); %annihilation operator a
                   in adiabatic approximation
88
89             % UNUSED HERE: Ansatz for the annihilation operator, obtained
                   by taking the time derivative of "a" equal to 0
90             % (OK for CW but not for PR (pulsed regime) programs)
91             % a = -2*g*sigma/(kappa*(1-1i*Delta))-2*sqrt(kappa_top)*b_in_CW
                   *Id/(kappa*(1-1i*Delta));
92
93         case 'F'  % Full model
94
95             %%%%%%%%% Definition of the full-model Hamiltonian (which
                   depends on omega_laser)
96             H_CW = (omega_d-omega_laser)*sigma_dag*sigma...
97                  + (omega_c-omega_laser)*a_dag*a...
98                  + 1i*g*(sigma_dag*a-a_dag*sigma)...
99                  - 1i*sqrt(kappa_top)*b_in_CW*(a_dag-a);
100
101 end % end of the "switch model"
102
103 %Superoperator associated with the coherent processes (Hamiltonian)
104 L_coh = -1i * (spre(H_CW) - spost(H_CW));
105
106 %%%%%%%%% Calculation of the Liouvillian superoperator
107 Liouvillian  = L_coh + L_incoh; % Total Liouvillian superoperator including
        both coherent processes (Hamiltonian) and incoherent processes (
       dissipative jumps)
```

```
108
109 %%%%%%%%% Calculation of the density matrix corresponding to the stationary
        state
110 rhoss_CW = steady(Liouvillian);
111
112 %%%%%%%%% Definition of the output operators. These are general formulas
113 % for both the adiabatic and full model, depending on the choice of "a"
114 b_out = b_in_CW*Id + sqrt(kappa_top)*a; % definition of the operator b_out,
        i.e. the output operator for the reflected light, in ps^(-1/2)
115 c_out = sqrt(kappa_bottom)*a; % definition of the operator c_out, i.e. the
        output operator for the transmitted light, in ps^(-1/2)
116 d_out = sqrt(kappa_loss)*a; % definition of the operator d_out, i.e. the
        output operator for the diffracted/lost light, in ps^(-1/2)
117 % e_out = sqrt(gamma_sp)*sigma; % definition of the operator e_out, i.e.
        the output operator for the light spontaneously emitted outside the
118 % cavity mode, in ps^(-1/2), already defined in "
        Init_2level_Hilbert_space_and_operators.m"
119
120 % NB: in the adiabatic model the operators could also have been written
        directly as:
121 % b_out = b_in_CW*Id*(1-2*eta_top/(1-1i*Delta))-sqrt(Gamma_0*eta_top)*sigma
        /(1-1i*Delta_QDC); %output flux operator,  (eq.12)
122 % c_out = -2*g*sqrt(kappa_bottom)*sigma/(kappa*(1-1i*Delta_QDC))-2*sqrt(
        kappa_top*kappa_bottom)*b_in_CW*Id/(kappa*(1-1i*Delta));
123 % d_out = -2*g*sqrt(kappa_loss)*sigma/(kappa*(1-1i*Delta_QDC))-2*sqrt(
        kappa_top*kappa_loss)*b_in_CW*Id/(kappa*(1-1i*Delta)); %annihilation
        operator a in adiabatic approximation
124 % Such formulas are obtained by directly replacing the value of "a" from
125 % the adiabatic model, and are thus equivalent to the above, more general
126 % definitions In addition, e_out is independent on the experimental
127 % conditions and thus defined in the subprogram
128 % "Init_2level_Hilbert_space_and operators.m". It is given here for
129 % information and clarity purposes only
130
131 % Calculation of the total photon flux as a function of omega_laser
132 total_flux_injected_photons=abs(b_in_CW)^2; % total flux of injected
        photons taking into account  eta_in (so only the photons coupled to the
        cavity mode), in ps-1
133 total_flux_reflected_photons=expect(b_out'*b_out,rhoss_CW);%flux in ps^(-1)
134 total_flux_transmitted_photons=expect(c_out'*c_out,rhoss_CW);%flux in ps
        ^(-1)
135 total_flux_diffracted_photons=expect(d_out'*d_out,rhoss_CW);%flux in ps
        ^(-1)
136 total_flux_emitted_photons=expect(e_out'*e_out,rhoss_CW);%flux in ps^(-1)
137
138 %%
```

```
%%%%%%%%%% Intensity correlations %%%%%%%%%%
% (method ensuring the best numerical convergence, using normalized density
% matrices in the mesolve functions)

%%% General notes on calculating two-time correlation functions, valid in
%%% PR and CW %%%

% Experimentally, the second order correlation function g2(t1,t2) is the
% ratio between two probabilities:
%   - the probability of detecting a photon at time t2 conditioned by a
%   first photon detected at time t1
%   - the probability of detecting a photon at time t2 without any
%   information on previous detections
% On the theory side, the standard definition of a normalized correlation
% function g2(t1,t2) is:
%     g2(t1,t2) = <b'(t1)b'(t2)b(t2)b(t1)> / <b'(t2)b(t2)> <b'(t1)b(t1)>
% We thus need to make the link between both views, and derive a way to
% practically compute such quantities with the physical interpretation in
% mind.
% As a preliminary remark, the theoretical notations above  consider the
% Heisenberg representation, where the operators vary in time but the
% density matrix is constant (equal to its initial value at t=0). For
% example:
%     <b'(t1)b(t1)> = Trace [ b'(t1)b(t1) * rho(0)]
% Fortunately, to go to the Schrodinger evolution we can use a very useful
% rule, which tells us that we can put the time evolution in the density
% matrix (instead of the operators) to calculate any average value:
%     <b'(t1)b(t1)> = Trace [ b'(0)b(0) * rho(t1) ]
% In this Schrodinger representation the operators are constant and thus
% equal to their value at time 0, so that :
%     <b'(t1)b(t1)> = Trace [ b'b * rho(t1) ]
% This is why, to calculate  <b'(t1)b(t1)> we  just have to start at t=0
% and make rho(t) evolve up to time t1 using mesolve, then calculate the
% expectation value of b'b:
%     <b'(t1)b(t1)> = expect ( b'b, rho(t1) )

% Now, to calculate g2(t1,t2) we also also have to switch from the
% Heisenberg definition to a practical quantity we can calculate, i.e. the
% expectation value of some operator on some density matrix. By definition
% in the Heisenberg representation:
%     < b'(t1) b'(t2) b(t2) b(t1) > = Trace[ b'(t1) b'(t2) b(t2) b(t1) *
    rho(0)]
% But with a circular permutation inside the Trace this can also be seen as
    :
%     < b'(t1) b'(t2) b(t2) b(t1) > = Trace[ b'(t2) b(t2) * b(t1)rho(0)b'(
    t1) ]
```

```
% This gives us some hint that the correlation function, in the Heisenberg
% representation, can be seen as the expectation value of b'b at time t2,
% starting at time t1 from a fictitious density matrix b(t1)rho(0)b'(t1).
% In the Schrodinger representation, however, the operators b and b' are
% constant and only the density matrix evolves between times 0 and t1, and
% between times t1 and t2. But we can also take into account the effect of
% a photon detection event at time t1, which leads to an abrupt change of
% the system density matrix between two times:
%     - time t=t1^(-): after evolution between 0 and t1, but just before a
%     click (photon detection event)
%     - time t=t1^(+): just after a click at time t1
% With this in mind, the idea is to compute < b'(t1) b'(t2) b(t2) b(t1) >
% thanks to three steps:
%     - evolution from 0 to t1^(-), leading to a density matrix rho(t1)
%     "just before a click"
%     - modification of the system state due to the detection event at t1,
%     leading to a different density matrix "just after a click"
%     - further evolution of the system between times t1^(+) and t2

% However, to physically interpret the results and to get a nice numerical
% convergence, we should not work with the fictitious density matrix
% b(t1)rho(0)b'(t1) in the Heisenberg representation, nor its equivalent in
% the Shrodinger representation b rho(t1) b', since it is not normalized
% (its trace is not unity). Instead we define the real/normalized density
% matrix obtained just after a detection event at time t1:
%    rho(t1, just after a click) =  b  rho(t1) b' / expect [ b'b , rho(t1)
    ]
% This is a valid density matrix since the denominator is :
%     expect [ b'b , rho(t1) ] = Trace [ b'b * rho(t1) ] = Trace [ b rho(
   t1) b' ],
% and thus "rho(t1, just after a click)" is well normalized with unity
% trace. This division by expect [ b'b , rho(t1) ] also makes sense since
% this quantity is equal to <b'b(t1)>, i.e. one of the two terms in the
% denominator of g2(t1,t2). From this density matrix at time t1^(+), we can
% then deduce the density matrix at time t2, leading to a density matrix
% "rho(t2, conditioned to a click at t1)". With these notations the
% quantity < b'(t1) b'(t2) b(t2) b(t1) > / < b'b(t1) > is equivalent to :
%     < b'(t1) b'(t2) b(t2) b(t1) > / < b'b'(t1) > = Trace[ b'b * rho(t2,
   conditioned to a click at t1) ]
% And with similar notations the quantity < b'b(t2) > is equivalent to:
%      < b'b'(t2) > = Trace [ b'b  *  rho(t2) ]
% So we find that the normalized correlation function g2(t1,t2) is indeed
% the ratio between two quantities:
%     - the photon flux at time t2, conditioned by a previous photon
%     detection event at time t1
%     - the photon flux at time t2, unconditioned
```

```
225 % This is exactly the experimentalist's definition of g2(t1,t2). Note that
226 % in CW we usually take t1 = 0 and t2 = tau, and we take the stationary
227 % density matrix state both for rho(t1) and rho (t2), since by definition
228 % it does not evolve with time: only the density matrix "rho(tau,
229 % conditioned to a click at time 0)", being different from the stationary
230 % state's density matrix, does evolve with the delay tau
231
232 % Normalized (unity trace) density matrices just after a click at time 0,
233 % for the various optical fields:
234 density_matrix_just_after_reflected_photon_detection=b_out*rhoss_CW*b_out'/
        total_flux_reflected_photons; % Density matrix just after a reflected
        photon click at time 0
235 density_matrix_just_after_transmitted_photon_detection=c_out*rhoss_CW*c_out
        '/total_flux_transmitted_photons; % Density matrix just after a
        transmitted photon click at time 0
236 density_matrix_just_after_emitted_photon_detection=e_out*rhoss_CW*e_out'/
        total_flux_emitted_photons;% Density matrix just after an emitted (
        outside the mode) photon click at time 0
237
238 % Normalized density matrices as a function of the (positive) delay tau,
239 % conditioned on the detection of a click at time 0
240 density_matrix_vs_delay_after_reflected_photon_detection=mesolve(
        Liouvillian,density_matrix_just_after_reflected_photon_detection,
        tau_list_g2CW); %Density matrix at time tau, conditioned on a reflected
         photon click at time 0
241 density_matrix_vs_delay_after_transmitted_photon_detection=mesolve(
        Liouvillian,density_matrix_just_after_transmitted_photon_detection,
        tau_list_g2CW); %Density matrix at time tau, conditioned on a
        transmitted photon click at time 0
242 density_matrix_vs_delay_after_emitted_photon_detection=mesolve(Liouvillian,
        density_matrix_just_after_emitted_photon_detection,tau_list_g2CW); %
        Density matrix at time tau, conditioned on an emitted (outside the mode
        ) photon click at time 0
243
244 % Calculation of the normalized auto-correlation functions g2(tau), for
245 % positive delays and various optical fields
246 g2_reflected_vs_delay=expect(b_out'*b_out,
        density_matrix_vs_delay_after_reflected_photon_detection)/
        total_flux_reflected_photons; %Auto-correlation g(2)(tau) for the
        reflected light
247 g2_transmitted_vs_delay=expect(c_out'*c_out,
        density_matrix_vs_delay_after_transmitted_photon_detection)/
        total_flux_transmitted_photons; %Auto-correlation g(2)(tau) for the
        transmitted light
```

```
248 g2_emitted_vs_delay=expect(e_out'*e_out,
        density_matrix_vs_delay_after_emitted_photon_detection)/
        total_flux_emitted_photons; %Auto-correlation g(2)(tau) for the light
        emitted outside the mode
249
250 % Calculation of g2(tau) for both negative and positive delays
251 full_g2_reflected_vs_delay= [fliplr(g2_reflected_vs_delay(2:end) )
        g2_reflected_vs_delay ];
252 full_g2_transmitted_vs_delay= [fliplr(g2_transmitted_vs_delay(2:end) )
        g2_transmitted_vs_delay ];
253 full_g2_emitted_vs_delay= [fliplr(g2_emitted_vs_delay(2:end) )
        g2_emitted_vs_delay ];
254
255 % Calculation of the conditional occupation probabilities, as a function
256 % of the delay tau after a photon detection event
257 occupation_ground_vs_delay_after_reflected_photon_detection=expect(sigma*
        sigma_dag,density_matrix_vs_delay_after_reflected_photon_detection);
258 occupation_excited_vs_delay_after_reflected_photon_detection=expect(
        sigma_dag*sigma,
        density_matrix_vs_delay_after_reflected_photon_detection);
259 occupation_ground_vs_delay_after_transmitted_photon_detection=expect(sigma*
        sigma_dag,density_matrix_vs_delay_after_transmitted_photon_detection);
260 occupation_excited_vs_delay_after_transmitted_photon_detection=expect(
        sigma_dag*sigma,
        density_matrix_vs_delay_after_transmitted_photon_detection);
261 occupation_ground_vs_delay_after_emitted_photon_detection=expect(sigma*
        sigma_dag,density_matrix_vs_delay_after_emitted_photon_detection);
262 occupation_excited_vs_delay_after_emitted_photon_detection=expect(sigma_dag
        *sigma,density_matrix_vs_delay_after_emitted_photon_detection);
263
264 %toc
265
266 %%% UNUSED HERE: alternative method with non-normalized conditional density
267 %%% matrices
268 % NB: we denote G2(tau) (with a capital "G") the non-normalized intensity
269 % correlations of the form <b'(0) b'(tau) b(tau) b(0)>, where the output
270 % fields are in ps-{1/2}. Complementary, we denote g2(tau) the normalized
271 % intensity correlations <b'(0) b'(tau) b(tau) b(0)>/<b' b><b' b>
272
273 % rho_vs_delay_if_rho0_equal_a_rhoss_a_dag=mesolve(Liouvillian,a*rhoss_CW*
        a_dag,tau_list_g2CW);  % Sufficient to calculate correlations for c_out
        =sqrt(kappa_bottom)a et d_out=sqrt(kappa_loss)a
274 % rho_vs_delay_if_rho0_equal_sigma_rhoss_sigma_dag=mesolve(Liouvillian,
        sigma*rhoss_CW*sigma_dag,tau_list_g2CW);  % Sufficient to calculate
        correlations of e_out=sqrt(gamma_sp) sigma
```

```
275 % rho_vs_delay_if_rho0_equal_rhoss_a_dag=mesolve(Liouvillian,rhoss_CW*a_dag
    ,tau_list_g2CW); % Necessary to calculate correlations of b_out=b_in+
    sqrt(kappa_top) a
276 % rho_vs_delay_if_rho0_equal_a_rhoss=mesolve(Liouvillian,a*rhoss_CW,
    tau_list_g2CW); % Necessary to calculate correlations of b_out=b_in+
    sqrt(kappa_top) a
277 %
278 % G2_reflected_vs_tau=  kappa_top*expect(b_out'*b_out,
    rho_vs_delay_if_rho0_equal_a_rhoss_a_dag)...
279 %                       + abs(b_in_CW)^2*expect(b_out'*b_out,rhoss_CW)...
280 %                       + sqrt(kappa_top)*b_in_CW*expect(b_out'*b_out,
    rho_vs_delay_if_rho0_equal_a_rhoss)...
281 %                       + sqrt(kappa_top)*conj(b_in_CW)*expect(b_out'*b_out,
    rho_vs_delay_if_rho0_equal_rhoss_a_dag);
282 % G2_transmitted_vs_tau=kappa_bottom*expect(c_out'*c_out,
    rho_vs_delay_if_rho0_equal_a_rhoss_a_dag);
283 % G2_diffracted_vs_tau=kappa_loss*expect(d_out'*d_out,
    rho_vs_delay_if_rho0_equal_a_rhoss_a_dag);
284 % G2_emitted_vs_tau=gamma_sp*expect(e_out'*e_out,
    rho_vs_delay_if_rho0_equal_sigma_rhoss_sigma_dag);
285 %
286 %
287 % %%%    Calculation of the G(2) for all delays, through the symmetrization
     G2(tau) = conj(G2(-tau))
288 % full_G2_reflected_vs_tau = [fliplr(G2_reflected_vs_tau(2:end) )
    G2_reflected_vs_tau ];
289 % full_G2_transmitted_vs_tau = [fliplr(G_2_transmitted_vs_tau(2:end) )
    G_2_transmitted_vs_tau ];
290 % full_G2_diffracted_vs_tau = [fliplr(G_2_diffracted_vs_tau(2:end) )
    G_2_diffracted_vs_tau ];
291 % full_G2_emitted_vs_tau = [fliplr(G_2_emitted_vs_tau(2:end) )
    G_2_emitted_vs_tau ];
292 %
293 % %%%    Calculation of the normalized g(2) for all delays
294 % full_g2_reflected_vs_delay = full_G2_reflected_vs_tau/(
    total_flux_reflected_photons)^2;
295 % full_g2_transmitted_vs_tau = full_G_2_transmitted_vs_tau/(
    total_flux_transmitted_photons)^2;
296 % full_g2_diffracted_vs_tau = full_G_2_diffracted_vs_tau/(
    total_flux_diffracted_photons)^2;
297 % full_g2_emitted_vs_tau = full_G_2_emitted_vs_tau/(
    total_flux_emitted_photons^2);
298
299
300 %% %%%%%%%%% Plots %%%%%%%%
301 % For plot selection:
```

```
% - g2 from reflected photons : 'R'
% - g2 from transmitted + diffracted/lost photons : 'T'
% - g2 from photons emitted outside the mode : 'E'
% - associated conditioned occupation probabilities : 'O'

%plot_choice = ['T'];
plot_choice = ['T';'R';'E';'O'];
Plot2LevelG2CWvsDelay;
```

C.1.6 Second-order correlation in PW

```
clear
clc
%close all

%% Important note: these paths must be modified if needed
addpath(genpath('..\QotoolboxV015'));
addpath(genpath('..\CQED subprograms'));
addpath(genpath('..\CQED device parameters'))
savepath

% In addition, for the mesolve function to operate the executable files
% (.exe) and batch files (.bat) contained in  '[...] \QotoolboxV015\bin'
% have to be copied to  a folder that is on the Windows system path, in the
% main hard  drive where Windows is installed. This can be for example in:
% 'C:\Program Files\Matlab\R2014a\bin'.

% Warning: for the adiabatic version to converge, the tolerance in
% mesolve.m function must be reduced compared to the defaut values. For
% example:
% ode2file('ode_input.dat',L,rho0,t_list,struct('reltol',7e-8,'abstol',8e-8));


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%% Intensity correlations in the pulsed regime %%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% This section indexed "g2PR_vs_t1_t2" computes the intensity correlations
% in the pulsed regime, as a function of the detection times t1 and t2 in two
```

```
28 % detectors, for the various optical fields. The normalized function g2(t1,
       t2)
29 % is computed, as well as the coincidence maps determining the probability
       of
30 % having double-clicks, one at time t1 and the other at time t_2, during
       the same
31 % pulse or for uncorrelated pulses. The conditional occupation
       probabilities,
32 % modified by the detection of a first photon at time t1, are also computed
       , as
33 % well as the normalized g2 as a function of delay tau = t2 - t1, and the
       averaged
34 % g2(0), i.e. the area of the zero-delay peak of the normalized g2 function
       .
35 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
36
37
38 %% Choice of full model 'F' or adiabatic model 'A'
39 model = 'F';
40
41 %% Experimental conditions (to be edited)
42 detuning_QD_C_muev = 0; %Detuning between the QD and cavity frequencies, in
        mueV
43 detuning_pulse_QD_muev = 0; %Detuning between the pulse central frequency
       and the QD frequency, in mueV
44 eta_in = 1; % Injection efficiency for the incoming photons (depends on
       experimentally-achieved spatial coupling)
45 Nb_photons_pulse = 1;  % Average number of incoming photons in a pulse.
       This quantity should be multiplied by eta_in to know the number of
       incoming photons actually coupled to the optical mode
46 FWHM_pulse = 15; %in ps, full width at half-maximum of the incoming
       Gaussian pulse intensity (unit: ps since angular frequencies are in rad
       /ps)
47 t_delay = 2*FWHM_pulse; % Time at which the pulse is maximally intense, so
       that the computation starts when the pulse has not arrived yet
48 t_max_ps = (t_delay + 4*FWHM_pulse)*5; % Final time where we stop the
       computation and plots of time evolutions
49 nb_points_time = 100; % Time resolution/Number of iterations / <100000
       otherwise the integrating the master equation gets difficult (odesolve)
       )
50 t_min = 0.4*FWHM_pulse; % Initial time considered for the computations and
       plots of time evolutions
51
52
53 %%
54 % Initialization of parameters, operators, arrays, etc...
```

```
55 Init2levelDeviceParametersOngoingTest;
56 Init2levelHilbertSpaceAndOperators;
57 InitMaps2LevelG2PRvsT1T2;
58
59 % Definition of the input field in ps^(-1/2), in the form of a fseries (
       necessary for integrating the master equation)
60 Standard_deviation_b_in_PR = FWHM_pulse/(2*sqrt(log(2))); %Deduced from the
        properties of a gaussian function
61 b_in_fn = fn('gauss',t_delay,Standard_deviation_b_in_PR) * sqrt( eta_in*
       Nb_photons_pulse / ( sqrt(pi) * Standard_deviation_b_in_PR ) ); %
       square root of the incoming photon number per time unit, in ps-1/2
62 b_in_vs_time = fsval(b_in_fn,t_list); %scalar array representing b_in vs
       time
63
64 % Initial density matrix before the pulse has started
65 switch model
66     case 'F' %Full model
67         psi0 = tensor(Vacuum_state,g_ket); % Initial state: tensorial
               product of photonic vacuum and QD ground state
68         rho0 = psi0*psi0'; % Density matrix corresponding to the initial
               pure state
69     case 'A' % Adiabatic model
70         rho0 = g_ket*g_ket'; % Density matrix corresponding to the initial
               pure state
71 end
72
73 %% %%%%%%%% System Hamiltonian and time-dependent operators  %%%%%%%%%%%%
74 %
75 % The system Hamitonian is time-dependent due to the function b_in_fn
76 % describing the input field b_in(t).
77
78 % In addition, in the case of adiabatic elimination of the cavity mode an
       effective
79 % operator $a$ is defined, acting on the QD subspace, based on the formula
       for
80 % adiabatic elimination (Eq. 10 of the pdf notes). Since this formula
       depends
81 % on b_in(t), we define a time-dependent quantity "a_vs_time", which is an
82 % array containing, for each time of t_list, the corresponding operator
83 % "a".
84 % In the full model case, to simplify the following calculations, we define
        the
85 % same quantity a_vs_time, yet this time this array contains the same
       operator
86 % (annihilation operator "a" acting on the cavity subspace), replicated for
        all
```

```
% times of t_list.


switch model
        case 'A' % Adiabatic model

            Delta = 2*(omega_pulse-omega_c)/kappa; %normalized laser
                detuning appearing in Eq.12 of the pdf notes

            %%%%%%%%% Definition of the adiabatic-model Hamiltonian (in the
                 frame rotating at angular frequency omega_pulse)
            H_PR = (omega_eff-omega_pulse)*sigma_dag*sigma - 1i*sqrt(
                Gamma_0*eta_top)*...
                    ((1-1i*Delta)^(-1)*b_in_fn*sigma_dag-(1+1i*Delta)^(-1)*
                        b_in_fn'*sigma); % Hamiltonian (eq.13)

            %%% Definition of the time-dependant operator "a_vs_time"
                acting in the QD subspace (Eq. 10 of the pdf notes)
            a_vs_time = -2*(kappa*(1-1i*Delta_QDC))^(-1)*g*sigma*Id_vs_time
                -2*sqrt(kappa_top)*(kappa*(1-1i*Delta))^(-1)*fsval(b_in_fn,
                t_list).*Id_vs_time; %annihilation operator a in adiabatic
                approximation (eq.10), as fseries

        case 'F'  % Full model

            %%%%%%%%% Definition of the full-model Hamiltonian (in the
                frame rotating at angular frequency omega_pulse)
            H_PR = (omega_d-omega_pulse)*sigma_dag*sigma...
                 + (omega_c-omega_pulse)*a_dag*a...
                 + 1i*g*(sigma_dag*a-a_dag*sigma)...
                 - 1i*sqrt(kappa_top)*b_in_fn*(a_dag-a);

            % Definition of the operator a_vs_time, even though "a" is
                constant in the full model, to reduce the number of "switch
                " in the following code
             a_vs_time = a*Id_vs_time;

end % end of the "switch model"

%Superoperator associated to the coherent processes (Hamiltonian)
L_coh = -1i * (spre(H_PR) - spost(H_PR));

%%%%%%%%% Calculation of the Liouvillian superoperator
Liouvillian  = L_coh + L_incoh; % Total Liouvillian superoperator including
     both coherent processes (Hamiltonian) and incoherent processes (
    dissipative jumps)
```

```
120
121 %%%%%%%%%% Numerical Integration of the Master Equation %%%%%%%%
122
123 % Computation of preliminary time evolution, between 0 (long before the
    pulse)
124 % and t_min (time at which we want to start plotting and integrating the
    physical
125 % quantities. Such a computation can be performed with very low time
    resolution,
126 % i.e. the corresponding t_list_before_t_min has a very low number of
    points,
127 % since the density matrix almost doesn't evolve between 0 and t_min.
128 %
129 % NB: such a preliminary time evolution is mandatory to avoid having
    strictly
130 % zero expectation values for some quantities (such as the input or output
    fields),
131 % during the following time evolution between t_min and t_max). Indeed,
    zero
132 % expectation values lead to NaN errors when used in normalizing physical
133 % quantities, such as conditional density matrices (see below), or
134 % Stokes/Bloch coordinates in the Poincare'/Bloch sphere.
135
136 density_matrix_vs_time_before_t_min = mesolve(Liouvillian,rho0,
    t_list_before_t_min); %master equation solver based on odesolve: first
    evolution of the system
137 density_matrix_at_t_min = density_matrix_vs_time_before_t_min{
    nb_points_time_before_t_min};
138
139
140 % Computation of the density matrix vs time with t_list, i.e. between t_min
141 % and t_max, requiring a large enough time resolution.
142 density_matrix_vs_time = mesolve(Liouvillian,density_matrix_at_t_min,t_list
    ); % second evolution of the system
143
144
145 %%%%%%%%%%%%%%%%%% Definition of the output operators %%%%%%%%%%%%%%%%%%
146 % These are general formulas for both the adiabatic and full model,
    depending on "a_vs_time"
147
148 b_out_vs_time = b_in_vs_time*Id_vs_time + sqrt(kappa_top)*a_vs_time; %
    definition of the operator b_out, i.e. the output operator for the
    reflected light, in ps^(-1/2)
149 c_out_vs_time = sqrt(kappa_bottom)*a_vs_time; % definition of the operator
    c_out_vs_time, i.e. the output operator for the transmitted light, in
    ps^(-1/2)
```

```
% d_out_vs_time = sqrt(kappa_loss)*a_vs_time; % UNUSED HERE: definition of the operator d_out_vs_time, i.e. the output operator for the diffracted/lost light, in ps^(-1/2)
% e_out = sqrt(gamma_sp)*sigma %  output operator for the light spontaneously emitted outside the cavity mode, in ps^(-1/2), already defined in Init_2level_Hilbert_space_and_operators.m

% Calculation of the total photon flux as a function of time, for the various optical fields
flux_injected_photons_vs_time = b_in_vs_time.^2; % total flux of injected photons taking into account  eta_in (so only the photons coupled to the cavity mode), in ps(-1)
flux_reflected_photons_vs_time = real(expect(b_out_vs_time'*b_out_vs_time,density_matrix_vs_time)); %flux in ps^(-1)
flux_transmitted_photons_vs_time = real(expect(c_out_vs_time'*c_out_vs_time,density_matrix_vs_time)); %flux in ps^(-1)
% flux_diffracted_photons_vs_time = real(expect(d_out_vs_time'*d_out_vs_time,rho_vs_time)); %flux in ps^(-1)
flux_emitted_photons_vs_time = real(expect(e_out'*e_out,density_matrix_vs_time)); %flux in ps^(-1)

%Computing the number of reflected/transmitted/emitted photons, by integrating over all times in t_list
Nb_reflected_photons = trapz(t_list,flux_reflected_photons_vs_time);
Nb_transmitted_photons = trapz(t_list,flux_transmitted_photons_vs_time);
% Nb_diffracted_photons = trapz(t_list,flux_diffracted_photons_vs_time);
Nb_emitted_photons = trapz(t_list,flux_emitted_photons_vs_time);


%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%% Intensity correlations and conditional probabilities %%%%%%%%%%%%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (method ensuring the best numerical convergence, using normalized density
% matrices in the mesolve functions)

%%% General notes on calculating two-time correlation functions, valid in
%%% PR (pulsed regime) and CW
%
% Experimentally, the second order correlation function g2(t1,t2) is the
% ratio between two probabilities:
```

```
% -  the probability of detecting a photon at time t2 conditioned by a first
%    photon detected at time t1
% -  the probability of detecting a photon at time t2 without any information
%    on previous detections
%
% On the theory side, the standard definition of a normalized correlation
% function g2(t1,t2) is:
%     g2(t1,t2) = <b'(t1)b'(t2)b(t2)b(t1)> / <b'(t2)b(t2)> <b'(t1)b(t1)>
%
% We thus need to make the link between both views, and derive a way to
% practically compute such quantities with the physical interpretation in
% mind.


% As a preliminary remark, the theoretical notations above  consider the
% Heisenberg representation, where the operators vary in time but the
% density matrix is constant (equal to its initial value at t=0). For
% example:
%     <b'(t1)b(t1)> = Trace [ b'(t1)b(t1) * rho(0)]
% Fortunately, to go to the Schrodinger evolution we can use a very useful
% rule, which tells us that we can put the time evolution in the density
% matrix (instead of the operators) to calculate any average value:
%     <b'(t1)b(t1)> = Trace [ b'(0)b(0) * rho(t1) ]
% In this Schrodinger representation the operators are constant and thus
% equal to their value at time 0, so that :
%     <b'(t1)b(t1)> = Trace [ b'b * rho(t1) ]
% This is why, to calculate  <b'(t1)b(t1)> we  just have to start at t=0
% and make rho(t) evolve up to time t1 using mesolve, then calculate the
% expectation value of b'b:
%     <b'(t1)b(t1)> = expect ( b'b, rho(t1) )

% Now, to calculate g2(t1,t2) we also also have to switch from the
% Heisenberg definition to a practical quantity we can calculate, i.e. the
% expectation value of some operator on some density matrix. By definition
% in the Heisenberg representation:
%     < b'(t1) b'(t2) b(t2) b(t1) > = Trace[ b'(t1) b'(t2) b(t2) b(t1) * rho(0)]
% But with a circular permutation inside the Trace this can also be seen as :
%     < b'(t1) b'(t2) b(t2) b(t1) > = Trace[ b'(t2) b(t2) * b(t1)rho(0)b'(t1) ]

% This gives us some hint that the correlation function, in the Heisenberg
% representation, can be seen as the expectation value of b'b at time t2,
```

```
% starting at time t1 from a fictitious density matrix b(t1)rho(0)b'(t1).
% In the Schrodinger representation, however, the operators b and b' are
% constant and only the density matrix evolves between times 0 and t1, and
% between times t1 and t2. But we can also take into account the effect of
% a photon detection event at time t1, which leads to an abrupt change of
% the system density matrix between two times:
%  -   time t=t1^(-): after evolution between 0 and t1, but just before a
%      click (photon detection event)
%  -   time t=t1^(+): just after a click at time t1
% With this in mind, the idea is to compute < b'(t1) b'(t2) b(t2) b(t1) >
% thanks to three steps:
%  -   evolution from 0 to t1^(-), leading to a density matrix rho(t1)
%      "just before a click"
%  -   modification of the system state due to the detection event at t1,
%      leading to a different density matrix "just after a click"
%  -   further evolution of the system between times t1^(+) and t2

% However, to physically interpret the results and to get a nice numerical
% convergence, we should not work with the fictitious density matrix
% b(t1)rho(0)b'(t1) in the Heisenberg representation, nor its equivalent in
% the Shrodinger representation b rho(t1) b', since it is not normalized
% (its trace is not unity). Instead we define the real/normalized density
% matrix obtained just after a detection event at time t1:
%      rho(t1, just after a click) =  b  rho(t1) b' / expect [ b'b , rho(t1
    ) ]
% This is a valid density matrix since the denominator is :
%      expect [ b'b , rho(t1) ] = Trace [ b'b * rho(t1) ] = Trace [ b rho(
    t1) b' ],
% and thus "rho(t1, just after a click)" is well normalized with unity
% trace. This division by expect [ b'b , rho(t1) ] also makes sense since
% this quantity is equal to <b'b(t1)>, i.e. one of the two terms in the
% denominator of g2(t1,t2). From this density matrix at time t1^(+), we can
% then deduce the density matrix at time t2, leading to a density matrix
% "rho(t2, conditioned to a click at t1)". With these notations the
% quantity < b'(t1) b'(t2) b(t2) b(t1) > / < b'b(t1) > is equivalent to :
%      < b'(t1) b'(t2) b(t2) b(t1) > / < b'b(t1) > = Trace[ b'b * rho(t2,
%      conditioned to a click at t1) ]
% And with similar notations the quantity < b'b(t2) > is equivalent to:
%       < b'b(t2) > = Trace [ b'b  *  rho(t2) ]
% So we find that the normalized correlation function g2(t1,t2) is indeed
% the ratio between two quantities:
%  -  the photon flux at time t2, conditioned by a previous photon
    detection
%     event at time t1
%  -  the photon flux at time t2, unconditioned
% This is exactly the experimentalist's definition of g2(t1,t2). Note that
```

```
% in CW we usually take t1 = 0 and t2 = tau, and we take the stationary
% density matrix state "rhoss" both for rho(t1) and rho (t2), since by
% definition rhoss does not evolve with time: only the density matrix
% "rho(tau, conditioned to a click at time 0)", being different from rhoss,
% does evolve with the delay tau
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%  Calculation of conditional density matrices %%%%%%%%%%%%%%
% In the following the "density_matrix_vs_t1_just_after_click_OPERATOR"s are
% defined for each value of time t1 in t_list. Their N-th element represent
% the normalized density matrix just after a click has occurred at the N-th time.

density_matrix_vs_t1_just_after_click_b_out = b_out_vs_time*density_matrix_vs_time*b_out_vs_time'/flux_reflected_photons_vs_time; % Density matrix just after a reflected photon click at time t1
density_matrix_vs_t1_just_after_click_c_out = c_out_vs_time*density_matrix_vs_time*c_out_vs_time'/flux_transmitted_photons_vs_time; % Density matrix just after a transmitted photon click at time t1
density_matrix_vs_t1_just_after_click_e_out = e_out*density_matrix_vs_time*e_out'/flux_emitted_photons_vs_time;% Density matrix just after an emitted (outside the mode) photon click at time t1

% NB: "tic" is used as a "start" time for the measurement of the computing time between "tic" and "toc"
%tic

%Cycle over all times t1 in t_list, corresponding to the moment where a first click occurred
for t1_index = 1:nb_points_time

    % Both t1 and t2 are values in t_list. However, to compute the
    % normalized density matrices vs t2 after a click at t1, we consider
    % only t2 >= t1, and we need to  deal with the fact that there are less
    % and less remaining values of t2 in t_list, when t1 increases. To keep
    % all quantities defined in the full t_list, for each time t1 < t2 we
    % have a "zero" density matrix, i.e. a fictitious density matrix with
    % only zero elements-

    % Incrementing the array of zero density matrices, each time t1_index
    % is increased, to fill the density matrix for times t1 < t2

    if (t1_index >=2) % No need to include a zero density matrix at the first value of t1
```

```
296         zero_density_matrix_vs_t2_before_t1{t1_index-1} = 0*Id; %null
                density matrix with the same dimensions of the involved Hilbert
                 space
297     end
298
299     % Array of normalized density matrices, conditioned on the detection of
            a
300     % click at time t1, as a function of t2 >= t1 (and zero otherwise)
301     density_matrix_vs_t2_after_click_b_out_at_t1 = [
            zero_density_matrix_vs_t2_before_t1  mesolve(Liouvillian,
            density_matrix_vs_t1_just_after_click_b_out{t1_index},t_list(
            t1_index:end))];
302     density_matrix_vs_t2_after_click_c_out_at_t1 = [
            zero_density_matrix_vs_t2_before_t1  mesolve(Liouvillian,
            density_matrix_vs_t1_just_after_click_c_out{t1_index},t_list(
            t1_index:end))];
303     density_matrix_vs_t2_after_click_e_out_at_t1 = [
            zero_density_matrix_vs_t2_before_t1  mesolve(Liouvillian,
            density_matrix_vs_t1_just_after_click_e_out{t1_index},t_list(
            t1_index:end))];
304
305     % Evaluation of g2(t1,t2) as described in the general notes above, for
            t2 >= t1 (and zero otherwise)
306     g2_reflected_vs_t1_t2(t1_index,:) = expect(b_out_vs_time'*b_out_vs_time
            ,density_matrix_vs_t2_after_click_b_out_at_t1)./
            flux_reflected_photons_vs_time; %Auto-correlation g(2)(tau) for the
             reflected light
307     g2_transmitted_vs_t1_t2(t1_index,:) = expect(c_out_vs_time'*
            c_out_vs_time,density_matrix_vs_t2_after_click_c_out_at_t1)./
            flux_transmitted_photons_vs_time; %Auto-correlation g(2)(tau) for
            the transmitted light
308     g2_emitted_vs_t1_t2(t1_index,:) = expect(e_out'*e_out,
            density_matrix_vs_t2_after_click_e_out_at_t1)./
            flux_emitted_photons_vs_time; %Auto-correlation g(2)(tau) for the
            light emitted outside the mode
309
310     % Calculation of the conditional occupation probabilities at time t2
            after
311      % a photon detection event at t1, for t2 >= t1 (and zero otherwise)
312     occupation_ground_vs_t1_vs_t2_after_click_b_out_at_t1(t1_index,:) =
            expect(sigma*sigma_dag,density_matrix_vs_t2_after_click_b_out_at_t1
            );
313     occupation_excited_vs_t1_vs_t2_after_click_b_out_at_t1(t1_index,:) =
            expect(sigma_dag*sigma,density_matrix_vs_t2_after_click_b_out_at_t1
            );
```

```
314     occupation_ground_vs_t1_vs_t2_after_click_c_out_at_t1(t1_index,:) =
            expect(sigma*sigma_dag,density_matrix_vs_t2_after_click_c_out_at_t1
            );
315     occupation_excited_vs_t1_vs_t2_after_click_c_out_at_t1(t1_index,:) =
            expect(sigma_dag*sigma,density_matrix_vs_t2_after_click_c_out_at_t1
            );
316     occupation_ground_vs_t1_vs_t2_after_click_e_out_at_t1(t1_index,:) =
            expect(sigma*sigma_dag,density_matrix_vs_t2_after_click_e_out_at_t1
            );
317     occupation_excited_vs_t1_vs_t2_after_click_e_out_at_t1(t1_index,:) =
            expect(sigma_dag*sigma,density_matrix_vs_t2_after_click_e_out_at_t1
            );
318 end
319
320 %toc
321
322 %%
323 %%%%% Completion of previous partially-calculated maps vs t1,t2, to include
        the case where t2 < t1
324
325 % Due to the symmetry between t1 and t2 (we don't know which detector will
326 % click first), we have to use properties like g2(t1,t2)=g2(t2,t1) to fill
327 % the voids in the quantities that we have only partially calculated yet
328 % (since we systematically considered a zero value when t2 < t1. For each
329 % map function of t1 and t2, this completion is obtained by adding it to
330 % its transpose (to replace the zeros at t2 < t1) and dividing by 2 the
331 % elements along the diagonal (to avoid counting twice the case where
332 % t2=t1). This is done via an ad-hoc matrix idx below:
333
334 idx = ones(nb_points_time)-0.5*diag(ones(nb_points_time,1)); % to divide by
        2 the elements along the diagonal
335
336 % Completed g2(t1,t2) for the various optical fields
337 g2_emitted_vs_t1_t2 = real((g2_emitted_vs_t1_t2+g2_emitted_vs_t1_t2.')).*
        idx;
338 g2_reflected_vs_t1_t2 = real((g2_reflected_vs_t1_t2+g2_reflected_vs_t1_t2
        .')).*idx;
339 g2_transmitted_vs_t1_t2 = real((g2_transmitted_vs_t1_t2+
        g2_transmitted_vs_t1_t2.')).*idx;
340
341 %Completed conditional occupation probabilities for the excited and ground
        state
342 occupation_ground_vs_t1_vs_t2_after_click_b_out_at_t1 = (
        occupation_ground_vs_t1_vs_t2_after_click_b_out_at_t1+
        occupation_ground_vs_t1_vs_t2_after_click_b_out_at_t1.').*idx;
```

```
343 occupation_excited_vs_t1_vs_t2_after_click_b_out_at_t1 = (
        occupation_excited_vs_t1_vs_t2_after_click_b_out_at_t1+
        occupation_excited_vs_t1_vs_t2_after_click_b_out_at_t1.').*idx;
344 occupation_ground_vs_t1_vs_t2_after_click_c_out_at_t1 = (
        occupation_ground_vs_t1_vs_t2_after_click_c_out_at_t1+
        occupation_ground_vs_t1_vs_t2_after_click_c_out_at_t1.').*idx;
345 occupation_excited_vs_t1_vs_t2_after_click_c_out_at_t1 = (
        occupation_excited_vs_t1_vs_t2_after_click_c_out_at_t1+
        occupation_excited_vs_t1_vs_t2_after_click_c_out_at_t1.').*idx;
346 occupation_ground_vs_t1_vs_t2_after_click_e_out_at_t1 = (
        occupation_ground_vs_t1_vs_t2_after_click_e_out_at_t1+
        occupation_ground_vs_t1_vs_t2_after_click_e_out_at_t1.').*idx;
347 occupation_excited_vs_t1_vs_t2_after_click_e_out_at_t1 = (
        occupation_excited_vs_t1_vs_t2_after_click_e_out_at_t1+
        occupation_excited_vs_t1_vs_t2_after_click_e_out_at_t1.').*idx;
348
349
350 %% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
351 %%%%%%%%%%%%%%% Coincidence maps as a function of t1, t2 %%%%%%%%%%%%%%%%%
352 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
353
354 % This part computes quantities such as :
355 %   - < b_out'(t1) b_out'(t2) b_out(t2) b_out(t1) >, which is proportional
356 %     to the probability of detecting two reflected photons at time t1 and
        t2
357 %     in two detectors, during the same pulse (correlated clicks)
358 %   - < b_out'(t1) b_out(t1) > < b_out'(t2) b_out(t2) >, which is
        proportional
359 %     to the probability of detecting two reflected photons at time t1 and
        t2
360 %     in two detectors, but for different pulses (uncorrelated clicks)
361
362 % Uncorrelated photon_coincidences vs (t1,t2), e.g. <b_out'(t1)b_out(t1)> <
        b_out'(t2)b_out(t2)>
363 emitted_photon_coincidences_vs_t1_vs_t2_uncorrelated_pulses = real(expect(
        e_out'*e_out,density_matrix_vs_time))'*real(expect(e_out'*e_out,
        density_matrix_vs_time));
364 reflected_photon_coincidences_vs_t1_vs_t2_uncorrelated_pulses = real(expect
        (b_out_vs_time'*b_out_vs_time,density_matrix_vs_time))'*real(expect(
        b_out_vs_time'*b_out_vs_time,density_matrix_vs_time));
365 transmitted_photon_coincidences_vs_t1_vs_t2_uncorrelated_pulses = real(
        expect(c_out_vs_time'*c_out_vs_time,density_matrix_vs_time))'*real(
        expect(c_out_vs_time'*c_out_vs_time,density_matrix_vs_time));
366
367 % Correlated photon_coincidences vs (t1,t2), e.g. <b_out'(t1) b_out'(t2)
        b_out(t2) b_out(t1)>
```

```
emitted_photon_coincidences_vs_t1_vs_t2 = g2_emitted_vs_t1_t2.*emitted_photon_coincidences_vs_t1_vs_t2_uncorrelated_pulses;
reflected_photon_coincidences_vs_t1_vs_t2 = g2_reflected_vs_t1_t2.*reflected_photon_coincidences_vs_t1_vs_t2_uncorrelated_pulses;
transmitted_photon_coincidences_vs_t1_vs_t2 = g2_transmitted_vs_t1_t2.*transmitted_photon_coincidences_vs_t1_vs_t2_uncorrelated_pulses;

%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%  Normalized g2(tau)  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% As in a standard HBT experiment in the pulsed regime, g2(tau) is obtained
% by defining an histogram integrating all the coincidences corresponding
% to a given delay tau (with tau = t2 - t1 being positive or negative). The
% normalization choice here is that the area of the g2(tau) peak is unity
% for uncorrelated coincidences, i.e. for all peaks except the zero-delay
% peak.

% Normalized g2 (tau) for the zero-delay peak, corresponding to photons emitted
% within the same pulse. It is obtained from the  correlated coincidence map
% <b'(t1)b'(t2)b(t2)b(t1)>, by integrating over all values corresponding to a
% given delay tau = t2 - t1, with a time bin t_step, and normalizing by Nb_photons^2.
% Note that tau = 0 for t1=t2, corresponding to  the diagonal elements of the
% g2(t1,t2) map, while non-zero delays are obtained outside the diagonal.
for j = 1:nb_points_time
    normalized_g2_vs_delay_emitted(j) = real(sum(diag(emitted_photon_coincidences_vs_t1_vs_t2,j)))*t_step/Nb_emitted_photons^2;
    normalized_g2_vs_delay_reflected(j) = real(sum(diag(reflected_photon_coincidences_vs_t1_vs_t2,j)))*t_step/Nb_reflected_photons^2;
    normalized_g2_vs_delay_transmitted(j) = real(sum(diag(transmitted_photon_coincidences_vs_t1_vs_t2,j)))*t_step/Nb_transmitted_photons^2;
end

% Normalized g2 (tau) for the other peaks, corresponding to photons emitted
% within different pulses. It is obtained from the uncorrelated coincidence map
% <b'(t2)b(t2)><b'(t1)b(t1)>, by integrating over all values corresponding to a
```

```
398 % given delay tau = t2 - t1, with a time bin t_step, and normalizing by
        Nb_photons^2.
399 for j = 1:nb_points_time
400     normalized_g2_vs_delay_uncorrelated_emitted(j) = real(sum(diag(
            emitted_photon_coincidences_vs_t1_vs_t2_uncorrelated_pulses,j)))*
            t_step/Nb_emitted_photons^2;
401     normalized_g2_vs_delay_uncorrelated_reflected(j) = real(sum(diag(
            reflected_photon_coincidences_vs_t1_vs_t2_uncorrelated_pulses,j)))*
            t_step/Nb_reflected_photons^2;
402     normalized_g2_vs_delay_uncorrelated_transmitted(j) = real(sum(diag(
            transmitted_photon_coincidences_vs_t1_vs_t2_uncorrelated_pulses,j))
            )*t_step/Nb_transmitted_photons^2;
403 end
404
405 %% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
406 %%%%%%%%%%%%%%%% Mean g2 for the zero delay peak %%%%%%%%%%%%%%%%%%%%%%%%
407 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
408
409 % In the pulsed regime, what experimentalists usually call the g2(0) is in
410 % fact the mean value of g2(tau) for the zero-delay peak, or more precisely
411 % the area of the normalized g2(tau) curve for the zero-delay peak, as
        compared
412 % to the unity area obtained for the other uncorrelated peaks.
413
414 % Area of normalized g2(tau) for the zero-delay/correlated peak
415 mean_g2_zero_delay_peak_reflected_photons = trapz(full_tau_list,[flip(
        normalized_g2_vs_delay_reflected(2:end))
        normalized_g2_vs_delay_reflected]);
416 mean_g2_zero_delay_peak_transmitted_photons = trapz(full_tau_list,[flip(
        normalized_g2_vs_delay_transmitted(2:end))
        normalized_g2_vs_delay_transmitted]);
417 mean_g2_zero_delay_peak_emitted_photons = trapz(full_tau_list,[flip(
        normalized_g2_vs_delay_emitted(2:end)) normalized_g2_vs_delay_emitted])
        ;
418
419 % Area of normalized g2(tau) for the zero-delay/correlated peak
420 mean_g2_uncorrelated_peaks_reflected_photons = trapz(full_tau_list,[flip(
        normalized_g2_vs_delay_uncorrelated_reflected(2:end))
        normalized_g2_vs_delay_uncorrelated_reflected]);
421 mean_g2_uncorrelated_peaks_transmitted_photons = trapz(full_tau_list,[flip(
        normalized_g2_vs_delay_uncorrelated_transmitted(2:end))
        normalized_g2_vs_delay_uncorrelated_transmitted]);
422 mean_g2_uncorrelated_peaks_emitted_photons = trapz(full_tau_list,[flip(
        normalized_g2_vs_delay_uncorrelated_emitted(2:end))
        normalized_g2_vs_delay_uncorrelated_emitted]);
423
```

```
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%     Plots     %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% For plot selection:
% - g2 vs (t1,t2) : 'G'
% - photon coincidences (correlated and not) vs (t1,t2) : 'C'
% - ground state occupations vs (t1,t2) : 'O'
% - photon fluxes vs time & g2 vs delay  : 'F'

% plot_choice = ['G'];
plot_choice = ['G';'C';'O';'F'];
Plot2LevelG2PRvsT1T2;
```

C.2 CQED device parameters

```
if ~ismember(model,['F'; 'A'])
    f = errordlg('Not valid input for full / adiabatic model selection.','Error');
end

%%%% Physical constants (do not change) %%%%
ev=1.60217646e-19;h=6.626068e-34; hbar=h/(2*pi);

% Parameters: energies of the mode ("c" for "cavity" and of the QD ("d" for
% "dot") in eV. NB: all variable names have to end in "_ev" when they are
% in electron-volts. Afterwards the energies in electron-volts will all be
% converted in angular frequencies in rad/ps for subsequent calculations.
omega_c_ev = 1.329810; % Cavity mode energy
omega_d_ev = omega_c_ev + detuning_QD_C_muev*1e-6; % QD transition energy

% Parameters of the QD (in ueV as indicated by the name ending by "_muev"
g_muev = 17; % Light matter coupling in mueV
gamma_sp_muev = 0.6; %Spontaneous emission of leaky modes (i.e. not in the
                        %cavity mode)
gamma_puredephasing_muev = 0; %Pure dephasing in mueV
gamma_decoherence_muev = gamma_sp_muev/2 + gamma_puredephasing_muev;

%Parameters of the cavity
kappa_muev = 400; % Cavity intensity decay rate, in ueV
eta_top = 0.7; %Extraction efficiency for the top Bragg mirror
eta_bottom = 0.1; %Extraction efficiency for the bottom Bragg mirror
```

```
26 eta_loss = 1-eta_top-eta_bottom; %Losses induced by lateral diffraction or
       absorption
27
28 % Conversions in rad/ps
29 omega_c = omega_c_ev*ev/hbar*1e-12; %in rad/ps
30 omega_d = omega_d_ev*ev/hbar*1e-12; %in rad/ps
31 g = g_muev*10^-6*ev/hbar*1e-12;  %in rad/ps
32 gamma_decoherence = gamma_decoherence_muev*1e-6*ev/hbar*1e-12;  %in rad/ps
33 gamma_sp = gamma_sp_muev*1e-6*ev/hbar*1e-12;  %in rad/ps
34 gamma_puredephasing = gamma_puredephasing_muev*1e-6*ev/hbar*1e-12;  %in rad
       /ps
35 kappa = kappa_muev*1e-6*ev/hbar*1e-12;  %in rad/ps
36 kappa_top = kappa*eta_top;  %in rad/ps
37 kappa_bottom = kappa*eta_bottom;  %in rad/ps
38 kappa_loss = kappa*eta_loss;  %in rad/ps
39
40 if strcmp(model,'F')
41     N=10; %Maximal number of photons in the cavity mode (troncature of the
           Fock space if complete model is used)
42 end
43
44 % Parameters useful for physical interpretation of data, and required for
45 % the "adiabatic elimination" model
46 Delta_QDC = 2*(omega_d-omega_c)/kappa; %normalized QD-cavity detuning
47 Gamma_0 = 4*g^2/kappa; %Purcell-enhanced emission rate at zero detuning
48 Gamma_m = Gamma_0/(1 + Delta_QDC^2); % Purcell-enhanced emission rate
49 Gamma_tot = Gamma_m + gamma_sp; %total emission rate
50 omega_eff = omega_d + 0.5*Gamma_0*Delta_QDC/(1 + Delta_QDC^2); %cavity
       induced frequency shift
```

C.3 Subprograms

C.3.1 mesolve

```
1  % Function developed by Sze Meng Tan. A quantum optics toolbox, 1999
2  function [rho] = mesolve(L,rho0,t_list)
3  %%%%%%%%%%%%%%%%%%% L : Liouvillien use in the Master Equation
4  %%%%%%%%%%%%%%%%%%% rho 0 : Initial Condition for the solving of the ME
5  %%%%%%%%%%%%%%%%%%% t_list : time list containing each time value where the
6  %%%%%%%%%%%%%%%%%%% Density Matrix will be calculated thanks to the routine
7
8  %%%%%%%%%%% Numerical Integration of the Master Equation %%%%%%%%
9
10
11 ode2file('ode_input.dat',L,rho0,t_list,struct('reltol',7e-8,'abstol',8e-7))
       ; % Writes the data into a file
```

```
odesolve('ode_input.dat','ode_output.dat'); % Solve/Integration (here Adams
    method by default)
fid = fopen('ode_output.dat','rb');
rho = qoread(fid,dims(rho0),size(t_list));
fclose(fid);
%%%%%%%%%%%%%%%%%%% rho : list of the Density Matrix calculated for each
%%%%%%%%%%%%%%%%%%% time specified in the list
```

C.3.2 Init 2level Hilbert space and operators

```
switch model
    case 'F'
        %%%%%%%%%%%%%%%%%%%% Sub-space "quantum dot"   %%%%%%%%%%%%%%%%%%%

        %Basis states for the quantum dot (g for "ground", e for "excited")
        g_ket=qo([1;0]); %%% Quantum object "ket" associated to the ground
            state
        g_bra=g_ket'; %%% Quantum object "bra" associated to the ground
            state
        e_ket=qo([0;1]); %%% Quantum object "ket" associated to the excited
             state
        e_bra=e_ket';   %%% Quantum object "bra" associated to the excited
            state

        %Operators acting in the sub-space of the 2-level QD system (2x2
            matrixs)
        id_QD=e_ket*e_bra+g_ket*g_bra; % Quantum object "Identity operator"
        sigma_QD=g_ket*e_bra; % Quantum object "de-excitation of the 2-
            level system"
        sigma_dag_QD=sigma_QD'; % Quantum object "excitation of the 2-level
             system"

        %%%%%%%%%%%%%%%%%%%% Sub-space "cavity"
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        %N denotes the maximal number of photons in the truncated Fock
            space (see above)
        id_cav = identity(N);  % Quantum object "Identity operator"  (
            matrix NxN)
        destroy_cav=destroy(N);% Quantum object "annihilation operator"  (
            matrix NxN)
        Vacuum_state=basis(N,1); % State corresponding to photon vacuum |0>
             (Nx1)

        last_Fock_state_ket = basis(N,N);% State corresponding to the last
            Fock state |N-1>, to control its occupation probability

```

```
        % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        %%%%%%%%% Operators acting in the product space cavity-quantum dot (matrices 2Nx2N)
        Id=tensor(id_cav,id_QD); % Tensorial product of identities in the cavity and QD subspace
        sigma = tensor(id_cav,sigma_QD); % De-excitation operator from the excited QD state to the ground QD state
        sigma_dag = sigma'; % De-excitation operator from the ground QD state to the excited QD state

        a=tensor(destroy_cav,id_QD); % Annihilation operator for a photon in the cavity mode
        a_dag=a'; % Creation operator for a photon in the cavity mode
        n=a_dag*a; % Intracavity photon number operator

        occupation_last_Fock_state=tensor(last_Fock_state_ket*last_Fock_state_ket',id_QD);% operator for the occupation of the last Fock state, to control that it remains low


        %%%%%%%%%%%%%%%% Definition of Collapse operators %%%%%%%%%%%%%%%%%%%%%
        C_cav = sqrt(kappa)*a; %Collapse operator for the cavity
        C_sp = sqrt(gamma_sp)*sigma; % Collapse operator for a spontaneous emission outside the cavity mode
        C_pure_dephasing = sqrt(2*gamma_puredephasing)*sigma_dag*sigma; % Collapse operator for pure dephasing

        % Lindblad operators associated to each incoherent process
        L_cav = 1/2 * (2*spre(C_cav)*spost(C_cav') - spre(C_cav'*C_cav) - spost(C_cav'*C_cav));% 1st terme: cavity dumping
        L_sp = 1/2 * (2*spre(C_sp)*spost(C_sp') - spre(C_sp'*C_sp) - spost(C_sp'*C_sp));% 2nd terme: exciton lifetime
        L_pure_dephasing = 1/2 * (2*spre(C_pure_dephasing)*spost(C_pure_dephasing') - spre(C_pure_dephasing'*C_pure_dephasing) - spost(C_pure_dephasing'*C_pure_dephasing));% 3rd term: pure dephasing

        % Lindblad operator associated to all the incoherent processes
        L_incoh=L_cav+L_sp+L_pure_dephasing;

    case 'A'
        %%%%%%%%%%%%%%%%%%%% space "quantum dot"   %%%%%%%%%%%%%%%%%%%%
```

```
53
54          %Basis states for the quantum dot (g for "ground", e for "excited")
55          g_ket=qo([1;0]); %%% Quantum object "ket" associated to the ground
                state
56          g_bra=g_ket'; %%% Quantum object "bra" associated to the ground
                state
57          e_ket=qo([0;1]); %%% Quantum object "ket" associated to the excited
                 state
58          e_bra=e_ket';   %%% Quantum object "bra" associated to the excited
                state
59
60          %Operators acting in the sub-space of the 2-level QD system (2x2
                matrixs)
61          Id=e_ket*e_bra+g_ket*g_bra; % Quantum object "Identity operator"
62          sigma=g_ket*e_bra; % Quantum object "de-excitation of the 2-level
                system"
63          sigma_dag=sigma'; % Quantum object "excitation of the 2-level
                system"
64
65          %%%%%%%%%%%%%%%% Definition of Collapse operators
                %%%%%%%%%%%%%%%%%%%%%%
66          C_QD = sqrt(Gamma_tot)*sigma; % Collapse operator for a spontaneous
                 emission outside the cavity mode (eq.15)
67          C_pure_dephasing = sqrt(2*gamma_puredephasing)*sigma_dag*sigma; %
                Collapse operator for pure dephasing
68
69          % Lindblad operators associated to each incoherent process
70          L_QD = 1/2 * (2*spre(C_QD)*spost(C_QD') - spre(C_QD'*C_QD) ...
71              - spost(C_QD'*C_QD));% exciton lifetime
72          L_pure_dephasing = 1/2 * (2*spre(C_pure_dephasing)*spost(
                C_pure_dephasing') - spre(C_pure_dephasing'*C_pure_dephasing) -
                 spost(C_pure_dephasing'*C_pure_dephasing));% 3rd term: pure
                dephasing
73
74          % Lindblad operator associated to all the incoherent processes
75          L_incoh=L_QD+L_pure_dephasing;
76
77          % NB: in the adiabatic model L_QD includes both the emission
                outside the mode
78          % and the Purcell-enhanced emission through the cavity mode, hence
                the
79          % "Gamma_tot" term in the definition of C_QD
80
81  end
82
```

```
83 %%%%%%%%%%  Constant output operator used to describe the field emitted
       outside the mode
84 e_out=sqrt(gamma_sp)*sigma; % Output flux operator in ps^(-1/2)
85
86 % NB: the other output flux operators can be power-dependent or frequency-
       dependent, depending on the model used
87 % (full or adiabatic). They are thus defined in the main file. Also note
       that e_out is equal to C_sp, but we use
88 % a different notation to insist on its use as an output operator,
       analogous to the other output operators b_out,
89 % c_out, and d_out, respectively describing the reflected, transmitted, and
        diffracted/lost photon field.
```

C.3.3 Init lists 2level CW scan laser frequency

```
1 %Parameters for the calculation of spectra
2 omega_laser_min_ev=omega_d_ev+min_detuning_muev*1e-6;
3 omega_laser_max_ev=omega_d_ev+max_detuning_muev*1e-6;
4
5 %%%% Initialization of lists to calculate and plot the spectra as a
       function of omega_laser
6 omega_laser_list_ev = linspace(omega_laser_min_ev,omega_laser_max_ev,
       nb_points_spectrum);% list of laser photon energies for the plots in eV
7 omega_laser_list = omega_laser_list_ev*ev/hbar*1e-12;% list of laser
       angular frequencies in rad/ps, the unit used for calculations
8 omega_step = (max(omega_laser_list) - min(omega_laser_list)) / (
       nb_points_spectrum-1); %in rad/ps, step for the calculation of
       integrals
9
10 % List to plot the spectra as a function of the detuning omega_laser-
       omega_d, in  mueV
11 detuning_list_muev = (omega_laser_list_ev - omega_d_ev)*1e6;
12
13 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
14 % Memory preallocation (allows gaining in calculation time)
15 % The  "_vs_omega" indicates here that this is a list of values related to
       the defferent_values of omega_laser
16
17 total_flux_reflected_photons_vs_omega = zeros(1,length(omega_laser_list));
       %flux in ps^(-1)
18 total_flux_transmitted_photons_vs_omega = zeros(1,length(omega_laser_list))
       ; %flux in ps^(-1)
19 total_flux_diffracted_photons_vs_omega = zeros(1,length(omega_laser_list));
        %flux in ps^(-1)
20 total_flux_emitted_photons_vs_omega = zeros(1,length(omega_laser_list)); %
       flux in ps^(-1)
```

```
21
22 occupation_excited_state_vs_omega = zeros(1,length(omega_laser_list)); %
       for the occupation of the excited state
23 occupation_ground_state_vs_omega = zeros(1,length(omega_laser_list)); % for
        the occupation of the ground state
24
25
26 %%%% OPTIONAL : Memory preallocation for the coherent part of the output
       fields
27 % Here the term "laser_coherent" means that it corresponds to the part of
       the flux that is
28 % coherent with the incoming excitation laser, and not the "total" flux.
       Obviously the incoherent part
29 % is just given by substrating the "laser_coherent" part from the "total"
       flux
30
31 flux_reflected_photons_laser_coherent_vs_omega  = zeros(1,length(
       omega_laser_list)); %flux in ps^(-1)
32 flux_transmitted_photons_laser_coherent_vs_omega = zeros(1,length(
       omega_laser_list)); %flux en s^(-1)
33 flux_diffracted_photons_laser_coherent_vs_omega = zeros(1,length(
       omega_laser_list)); %flux en s^(-1)
34 flux_emitted_photons_laser_coherent_vs_omega  = zeros(1,length(
       omega_laser_list)); %flux en s^(-1)
```

C.3.4 Init lists 2level PW vs time

```
1 % Parameters of the incoming gaussian pulse
2 omega_pulse_ev=omega_d_ev+detuning_pulse_QD_muev*1e-6; %center energy of
       the incoming pulse, in eV
3 omega_pulse=omega_pulse_ev*ev/hbar*1e-12; %in rad/ps
4
5 % Parameters for the computation of time evolutions
6 t_delay = 2*FWHM; % Time at which the pulse is maximally intense, so that
       the computation starts when the pulse has not arrived yet
7 t_max_ps=t_delay + 4*FWHM +0.5/gamma_sp; % Final time where we stop the
       computation and plots of time evolutions
8 nb_points_time = 1000; % Time resolution/Number of iterations / <100000
       otherwise the integrating the master equation gets difficult (odesolve)
       )
9 t_min = 0*FWHM; % Initial time considered for the computations and plots of
        time evolutions
10 t_step=(t_max_ps-t_min)/(nb_points_time-1); % Duration of a time step
11 t_list=linspace(t_min,t_max_ps,nb_points_time); % list of all the times
       considered in the computation and plots
12
```

```
% Initialization of qo array of identity operator
Id_vs_time = qo;
for time_index = 1:nb_points_time
    Id_vs_time{time_index} = Id;
end
```

C.3.5 Init lists 2level g1SD CW vs delay and frequency

```
%M Laser frequency (fixed)

omega_laser_ev = omega_d_ev + detuning_laser_QD_muev*1e-6;
omega_laser = omega_laser_ev*ev/hbar*1e-12; % laser angular frequency in
    rad/ps, the unit used for calculations


%% Parameters and lists for the evaluation of the temporal evolution as a
    function of the delay tau

tau_step = (tau_max)/(nb_points_delay-1); %size of the time step, in ps
tau_list = linspace(0,tau_max,nb_points_delay); % array containg the non-
    negative time steps
full_tau_list = [fliplr(-tau_list(2:end-1)) , tau_list ];% array containing
     both positive and negative time steps


%% Parameters for the calculation of spectral densities through the Fast
    Fourier Transform (FFT) algorithm
%
% NB1: See comments in the main program for a definition of spectral
    densities and main concepts involved
%
% NB2: For more information in the matlab FFT function, see:
%  https://fr.mathworks.com/help/matlab/ref/fft.html
% A discussion on the proper normalization of the FFT signal (which should
    be performed by
% dividing the fft function's result by the signal's sampling frequency)
    can be found here:
% https://math.stackexchange.com/questions/636847/understanding-fourier-
    transform-example-in-matlab
% Such a normalization choice allows respecting the Parseval's theorem, i.e
    .:
%       sum_{n=0}^{N-1} |x[n]|^2 = 1/N*sum_{k=0}^{N-1} |X[k]|^2,
% with x[n] the signal and X[k] its discrete Fourier Transform, n and k
    being positive
% indices between 0 and N-1. Indeed, Parseval's theorem is at the heart of
    our normalization
```

```
27 % choice that the integral of the spectra density of flux should be the
28 % total flux.
29 %
30 % NB3: To get a really fast algorithm, the number of points used in the
31 % time and frequency domain should be a power of 2.
32
33 FFT_sampling_frequency = 1/tau_step; % sampling frequency, also called "
       sampling rate", in ps^(-1)
34 nb_points_full_spectrum = 2^nextpow2(length(full_tau_list)); % ensuring a
       power of 2 for for optimized FFT performance
35
36 omega_step = 2*pi*FFT_sampling_frequency/nb_points_full_spectrum; % angular
        frequency step in the spectrum, in rad/ps
37 omega_step_muev = omega_step/ev*hbar/1e-18; % photon energy step in mueV
38
39 omega_list_full_spectrum = omega_laser + (-nb_points_full_spectrum/2:
       nb_points_full_spectrum/2-1)*omega_step; % array of angular frequencies
        in rad/ps
40 % spectrum centered around omega_laser, since we work in the frame rotating
        at this angular frequency)
41
42
43 %% Parameters for the zoomed spectrum, i.e. the list of angular frequencies
        of interest in the slected spectral window.
44 %
45 % NB: This zoomed spectrum is simply a subset of the previous one, between
       a minimal index
46 % and a maximal one that are defined below.
47
48 index_min_zoomed_spectrum = nb_points_full_spectrum/2+1-round(
       width_spectral_window_muev/omega_step_muev/2);
49 index_max_zoomed_spectrum = nb_points_full_spectrum/2+1+round(
       width_spectral_window_muev/omega_step_muev/2);
50
51 omega_list = linspace(omega_list_full_spectrum(index_min_zoomed_spectrum),
       omega_list_full_spectrum(index_max_zoomed_spectrum),
       index_max_zoomed_spectrum-index_min_zoomed_spectrum+1); % list of
       angular frequencies, in rad/ps over the full FFT spectrum
52
53 nb_points_spectrum = length(omega_list);
54
55
56 %% Parameters for the zoomed spectrum expressed in photon energy
57
58 omega_list_ev = omega_list/ev*hbar/1e-12; % selected list of photon
       energies, in eV
```

```
59
60
61 %% Parameters for the compensation of the FFT phase shift compensation
62 %
63 % NB: a first-order autocorrelation function is always anti-symmetric in
       the sense that:
64 %          g1(-tau) = g1(tau)*
65 % This antisymmetry ensures that its Fourier Transform, and thus the
       corresponding spectral
66 % density, gives a real physical quantity. However, as discussed in the
       main program the fft
67 % function considers that the first signal point corresponds to time 0,
       while in our case
68 % the full_tau_list contains both negative and positive delays, and the
       antisymmetry point
69 % (tau=0) is shifted to the middle of the spectrum. This is a very general
       issue arising
70 % from the fact that MATLAB arrays have only positive indices, so a signal
       x[n] defined over
71 % n =  -N, -N+1, ..., 0, ... N-1, N is treated by the fft function as it
       were defined over
72 % n = 1 , 2 , ... , 2*N+1. Such a translation of the x[n] signal leads, in
       its Fourier
73 % Transform X[k], to a phase shift which linearly increases with the index
       k. In our case,
74 % the phase shift will depend on the angular frequency omega, and has to be
        compensated by
75 % by a phase term denoted "phase_shift_compensation_vs_omega_full_spectrum
       ".
76 % For more info on the phase shift compensation, see:
77 % https://www.mathworks.com/matlabcentral/answers/94874-why-is-the-fft-of-
       an-anti-symmetric-signal-not-correct
78
79 k = 0:(nb_points_full_spectrum-1);
80 phase_shift_compensation_vs_omega_full_spectrum = exp(-1j*2*pi*(
       nb_points_full_spectrum/2-1)*k/length(k));
81
82
83 %% Memory preallocation (allows gaining in calculation time)
84 % NB: The  "_vs_omega" indicates here that this is a list of values related
        to the different
85 % values of omega, in rad/ps, in omega_list
86
87 spectral_density_flux_reflected_photons_incoh_vs_omega = zeros(1,length(
       omega_list));
```

```
spectral_density_flux_transmitted_photons_incoh_vs_omega = zeros(1,length(
    omega_list));
spectral_density_flux_emitted_photons_incoh_vs_omega = zeros(1,length(
    omega_list));
```

C.3.6 Init maps 2level g1 WDF PW

```
% Parameters of the incoming gaussian pulse
omega_pulse_ev=omega_d_ev+detuning_pulse_QD_muev*1e-6; %center energy of
    the incoming pulse, in eV
omega_pulse=omega_pulse_ev*ev/hbar*1e-12; %in rad/ps

% Parameters for the computation of time evolutions
t_step=(t_max_ps-t_min)/(nb_points_time-1); % Duration of a time step
t_list=linspace(t_min,t_max_ps,nb_points_time); % list of all the times
    considered in the computation and plots

% Steps in time and delay for the Wigner Distribution Function (WDF).
% Notice that even though tau = t2-t1, its step here is half the step in
% the density matrix evolution to correctly Fourier transform over
% frequency later, according to the WDF definition. More info in the "main"
% script.

tau_step = t_step/2; %important to avoid aliasing
time_step = t_step/2;

time_list = (t_list(1):time_step:t_list(end));
tau_list = (t_list(1)-t_list(end):tau_step:t_list(end)-t_list(1));
%%
% Parameters for the computation of preliminary time evolution,
% between 0 (long before the pulse) and t_min (time at which we want
% to start plotting and integrating the physical quantities
nb_points_time_before_t_min = 5; % (Low) time resolution for first
    evolution of the system for initialization
t_list_before_t_min = linspace(0,t_min,nb_points_time_before_t_min); % time
     array for first evolution of the sistem

% Initialization of qo array of identity operator
Id_vs_time = qo;
for t1_index = 1:nb_points_time
    Id_vs_time{t1_index} = Id;
end

%%% Preallocation of the memory to save computing time

% Initialization of g1 vs (t1,t2)
```

```
g1_reflected_vs_t1_t2 = zeros(nb_points_time,nb_points_time);
g1_transmitted_vs_t1_t2 = zeros(nb_points_time,nb_points_time);
g1_diffracted_vs_t1_t2 = zeros(nb_points_time,nb_points_time);
g1_emitted_vs_t1_t2 = zeros(nb_points_time,nb_points_time);

% Initialization of the "zero" density matrix which will be used to fill the
% conditional density matrices, using 0 values for t2 < t1
zero_density_matrix_vs_t2_before_t1 = qo;
%% Parameters for the calculation of Fourier Transform over tau through the Fast Fourier Transform (FFT) algorithm
%
% NB1: See comments in the main program for a definition of spectral densities and main concepts involved
%
% NB2: For more information in the matlab FFT function, see:
%  https://fr.mathworks.com/help/matlab/ref/fft.html
% A discussion on the proper normalization of the FFT signal (which should be performed by
% dividing the fft function's result by the signal's sampling frequency) can be found here:
% https://math.stackexchange.com/questions/636847/understanding-fourier-transform-example-in-matlab
% Such a normalization choice allows respecting the Parseval's theorem, i.e.:
%       sum_{n=0}^{N-1} |x[n]|^2 = 1/N*sum_{k=0}^{N-1} |X[k]|^2,
% with x[n] the signal and X[k] its discrete Fourier Transform, n and k being positive
% indices between 0 and N-1. Indeed, Parseval's theorem is at the heart of our normalization
% choice that the integral of the spectra density of flux should be the
% total flux.
%
% NB3: To get a really fast algorithm, the number of points used in the
% time and frequency domain should be a power of 2. However, in this
% specific program the number of elements is 4*nb_points_time-3, so zero
% padding is performed by the fft-s defined in the main.

FFT_sampling_frequency = 1/t_step; % sampling frequency, also called "sampling rate", in ps^(-1)
nb_points_full_spectrum = 2^nextpow2(length(tau_list)); % ensuring a power of 2 for for optimized FFT performance
omega_step = 2*pi*FFT_sampling_frequency/nb_points_full_spectrum; % angular frequency step in the spectrum, in rad/ps
omega_step_muev = omega_step/ev*hbar/1e-18; % photon energy step in mueV

```

```
70 omega_list_full_spectrum = omega_pulse + (-nb_points_full_spectrum/2:
       nb_points_full_spectrum/2-1)*omega_step; % array of angular frequencies
        in rad/ps
71 omega_list_full_spectrum_muev = omega_list_full_spectrum/ev*hbar/1e-18;
72 % spectrum centered around omega_pulse, since we work in the frame rotating
        at this angular frequency)
73
74 %% Parameters for the zoomed spectrum, i.e. the list of angular frequencies
        of interest in the slected spectral window.
75 %
76 % NB: This zoomed spectrum is simply a subset of the previous one, between
       a minimal index
77 % and a maximal one that are defined below.
78
79 index_min_zoomed_spectrum = nb_points_full_spectrum/2+1-round(
       width_spectral_window_muev/omega_step_muev/2);
80 index_max_zoomed_spectrum = nb_points_full_spectrum/2+1+round(
       width_spectral_window_muev/omega_step_muev/2);
81
82 omega_list = linspace(omega_list_full_spectrum(index_min_zoomed_spectrum),
       omega_list_full_spectrum(index_max_zoomed_spectrum),
       index_max_zoomed_spectrum-index_min_zoomed_spectrum+1); % list of
       angular frequencies, in rad/ps over the full FFT spectrum
83
84 nb_points_spectrum = length(omega_list);
85
86 %% Parameters for the zoomed spectrum expressed in photon energy
87
88 omega_list_ev = omega_list/ev*hbar/1e-12; % selected list of photon
       energies, in eV
89
90 %% Parameters for the compensation of the FFT phase shift compensation
91 %
92 % NB: WVD gives a real quantity. However, as discussed in the main program
       the fft
93 % function considers that for each row, the first signal point corresponds
       to delay 0, while in our case
94 % each row contains both negative and positive delays. This is a very
       general issue arising
95 % from the fact that MATLAB arrays have only positive indices, so a signal
       x[n] defined over
96 % n =  -N, -N+1, ..., 0, ... N-1, N is treated by the fft function as it
       were defined over
97 % n = 1 , 2 , ... , 2*N+1. Such a translation of the x[n] signal leads, in
       its Fourier
```

```
98  % Transform X[k], to a phase shift which linearly increases with the index
        k. In our case,
99  % the phase shift will depend on the angular frequency omega, and has to be
         compensated by
100 % by a phase term denoted "phase_shift_compensation_vs_omega_full_spectrum
        ".
101 % For more info on the phase shift compensation, see:
102 % https://www.mathworks.com/matlabcentral/answers/94874-why-is-the-fft-of-
        an-anti-symmetric-signal-not-correct
103
104 k = 0:(nb_points_full_spectrum-1);
105 %NB 1: the command "repmat" is used since the fft returns a matrix to be
106 %phaseshifted only along the rows. Therefore, "repmat" is used to generate
107 %a matrix with identical rows.
108 %NB 2: "length(tau_list)-1)/2" is the number of negative delay elements for
109 %each row
110 phase_shift_compensation_vs_omega_full_spectrum = repmat(exp(-1j*2*pi*((
        length(tau_list)-1)/2)*k/length(k)),2*nb_points_time-1,1);
```

C.3.7 Interpolation 2level G1 WDF PR

```
1  %% Memory preallocation (allows gaining in calculation time)
2  interpolated_G1_reflected_vs_time_tau = 0*time_list'*tau_list;
3  interpolated_reflected_coherent_vs_time_tau = 0*time_list'*tau_list;
4  interpolated_G1_transmitted_vs_time_tau = 0*time_list'*tau_list;
5  interpolated_transmitted_coherent_vs_time_tau = 0*time_list'*tau_list;
6  interpolated_G1_emitted_vs_time_tau = 0*time_list'*tau_list;
7  interpolated_emitted_coherent_vs_time_tau = 0*time_list'*tau_list;
8  interpolated_G1_diffracted_vs_time_tau = 0*time_list'*tau_list;
9  interpolated_diffracted_coherent_vs_time_tau = 0*time_list'*tau_list;
10
11 %From the definition of time = (t1+t2)/2 and tau=t2-t1, it is found that
       for a
12 %matrix tau_index = t2_index-t1_index + N and time_index =
13 %t1_index+t2_index-1. Therefore, it is obtained that t1_index =
14 %(time_index-tau_index+1+1)/2 and t2_index = (time_index+tau_index+1-N)/2.
15
16 %NB: The following interpolation is based on a simple arithmetic mean
       between
17 % consecutives values at half-integer values. "Fancier" interpolations
18 % would surely increase the overall accuracy.
19
20 for time_index = 1:length(time_list)
21     for tau_index = 1:length(tau_list)
```

```
22          t1_index = (time_index-((tau_index+1)/2)+1+nb_points_time)/2; %obs:
                tau_index from the previous formula is acqually "(tau_index+1)
               /2" since we have set tau_step = t_step/2
23          t2_index = (time_index+((tau_index+1)/2)+1-nb_points_time)/2; %same
                as above
24
25          if t1_index <= nb_points_time && t1_index >= 1 && t2_index <=
               nb_points_time && t2_index >= 1
26
27              if floor(t1_index)==t1_index && floor(t2_index)==t2_index
28                  interpolated_G1_reflected_vs_time_tau(time_index, tau_index
                        ) = G1_reflected_vs_t1_t2(t1_index, t2_index);
29                  interpolated_reflected_coherent_vs_time_tau(time_index,
                        tau_index) = expect_b_out_dag_t1_times_expect_b_out_t2(
                        t1_index, t2_index);
30                  interpolated_G1_transmitted_vs_time_tau(time_index,
                        tau_index) = G1_transmitted_vs_t1_t2(t1_index, t2_index
                        );
31                  interpolated_transmitted_coherent_vs_time_tau(time_index,
                        tau_index) = expect_c_out_dag_t1_times_expect_c_out_t2(
                        t1_index, t2_index);
32                  interpolated_G1_emitted_vs_time_tau(time_index, tau_index)
                        = G1_emitted_vs_t1_t2(t1_index, t2_index);
33                  interpolated_emitted_coherent_vs_time_tau(time_index,
                        tau_index) = expect_e_out_dag_t1_times_expect_e_out_t2(
                        t1_index, t2_index);
34                  interpolated_G1_diffracted_vs_time_tau(time_index,
                        tau_index) = G1_diffracted_vs_t1_t2(t1_index, t2_index)
                        ;
35                  interpolated_diffracted_coherent_vs_time_tau(time_index,
                        tau_index) = expect_d_out_dag_t1_times_expect_d_out_t2(
                        t1_index, t2_index);
36
37              elseif floor(t1_index)==t1_index && floor(t2_index)~= t2_index
38                  interpolated_G1_reflected_vs_time_tau(time_index, tau_index
                        ) = (G1_reflected_vs_t1_t2(t1_index, floor(t2_index)) +
                         G1_reflected_vs_t1_t2(t1_index, ceil(t2_index)))/2;
39                  interpolated_reflected_coherent_vs_time_tau(time_index,
                        tau_index) = (expect_b_out_dag_t1_times_expect_b_out_t2
                        (t1_index, floor(t2_index)) +
                        expect_b_out_dag_t1_times_expect_b_out_t2(t1_index,
                        ceil(t2_index)))/2;
40                  interpolated_G1_transmitted_vs_time_tau(time_index,
                        tau_index) = (G1_transmitted_vs_t1_t2(t1_index, floor(
                        t2_index)) + G1_transmitted_vs_t1_t2(t1_index, ceil(
                        t2_index)))/2;
```

```
41                interpolated_transmitted_coherent_vs_time_tau(time_index,
                      tau_index) = (expect_c_out_dag_t1_times_expect_c_out_t2
                      (t1_index, floor(t2_index)) +
                      expect_c_out_dag_t1_times_expect_c_out_t2(t1_index,
                      ceil(t2_index)))/2;
42                interpolated_G1_emitted_vs_time_tau(time_index, tau_index)
                      = (G1_emitted_vs_t1_t2(t1_index, floor(t2_index)) +
                      G1_emitted_vs_t1_t2(t1_index, ceil(t2_index)))/2;
43                interpolated_emitted_coherent_vs_time_tau(time_index,
                      tau_index) = (expect_e_out_dag_t1_times_expect_e_out_t2
                      (t1_index, floor(t2_index)) +
                      expect_e_out_dag_t1_times_expect_e_out_t2(t1_index,
                      ceil(t2_index)))/2;
44                interpolated_G1_diffracted_vs_time_tau(time_index,
                      tau_index) = (G1_diffracted_vs_t1_t2(t1_index, floor(
                      t2_index)) + G1_diffracted_vs_t1_t2(t1_index, ceil(
                      t2_index)))/2;
45                interpolated_diffracted_coherent_vs_time_tau(time_index,
                      tau_index) = (expect_d_out_dag_t1_times_expect_d_out_t2
                      (t1_index, floor(t2_index)) +
                      expect_d_out_dag_t1_times_expect_d_out_t2(t1_index,
                      ceil(t2_index)))/2;
46
47            elseif floor(t2_index)==t2_index && floor(t1_index)~= t1_index
48                interpolated_G1_reflected_vs_time_tau(time_index, tau_index
                      ) = (G1_reflected_vs_t1_t2(floor(t1_index), t2_index)+
                      G1_reflected_vs_t1_t2(ceil(t1_index), t2_index))/2;
49                interpolated_reflected_coherent_vs_time_tau(time_index,
                      tau_index) = (expect_b_out_dag_t1_times_expect_b_out_t2
                      (floor(t1_index), t2_index)+
                      expect_b_out_dag_t1_times_expect_b_out_t2(ceil(t1_index
                      ), t2_index))/2;
50                interpolated_G1_transmitted_vs_time_tau(time_index,
                      tau_index) = (G1_transmitted_vs_t1_t2(floor(t1_index),
                      t2_index)+G1_transmitted_vs_t1_t2(ceil(t1_index),
                      t2_index))/2;
51                interpolated_transmitted_coherent_vs_time_tau(time_index,
                      tau_index) = (expect_c_out_dag_t1_times_expect_c_out_t2
                      (floor(t1_index), t2_index)+
                      expect_c_out_dag_t1_times_expect_c_out_t2(ceil(t1_index
                      ), t2_index))/2;
52                interpolated_G1_emitted_vs_time_tau(time_index, tau_index)
                      = (G1_emitted_vs_t1_t2(floor(t1_index), t2_index)+
                      G1_emitted_vs_t1_t2(ceil(t1_index), t2_index))/2;
```

```
53                  interpolated_emitted_coherent_vs_time_tau(time_index,
                        tau_index) = (expect_e_out_dag_t1_times_expect_e_out_t2
                        (floor(t1_index), t2_index)+
                        expect_e_out_dag_t1_times_expect_e_out_t2(ceil(t1_index
                        ), t2_index))/2;
54                  interpolated_G1_diffracted_vs_time_tau(time_index,
                        tau_index) = (G1_diffracted_vs_t1_t2(floor(t1_index),
                        t2_index)+G1_diffracted_vs_t1_t2(ceil(t1_index),
                        t2_index))/2;
55                  interpolated_diffracted_coherent_vs_time_tau(time_index,
                        tau_index) = (expect_b_out_dag_t1_times_expect_d_out_t2
                        (floor(t1_index), t2_index)+
                        expect_b_out_dag_t1_times_expect_d_out_t2(ceil(t1_index
                        ), t2_index))/2;
56              else
57                  interpolated_G1_reflected_vs_time_tau(time_index, tau_index
                        ) = (G1_reflected_vs_t1_t2(floor(t1_index), floor(
                        t2_index))+G1_reflected_vs_t1_t2(ceil(t1_index), floor(
                        t2_index))+G1_reflected_vs_t1_t2(floor(t1_index), ceil(
                        t2_index))+G1_reflected_vs_t1_t2(ceil(t1_index), ceil(
                        t2_index)))/4;
58                  interpolated_reflected_coherent_vs_time_tau(time_index,
                        tau_index) = (expect_b_out_dag_t1_times_expect_b_out_t2
                        (floor(t1_index), floor(t2_index))+
                        expect_b_out_dag_t1_times_expect_b_out_t2(ceil(t1_index
                        ), floor(t2_index)) +
                        expect_b_out_dag_t1_times_expect_b_out_t2(floor(
                        t1_index), ceil(t2_index))+
                        expect_b_out_dag_t1_times_expect_b_out_t2(ceil(t1_index
                        ), ceil(t2_index)))/4;
59                  interpolated_G1_transmitted_vs_time_tau(time_index,
                        tau_index) = (G1_transmitted_vs_t1_t2(floor(t1_index),
                        floor(t2_index))+G1_transmitted_vs_t1_t2(ceil(t1_index)
                        , floor(t2_index))+G1_transmitted_vs_t1_t2(floor(
                        t1_index), ceil(t2_index))+G1_transmitted_vs_t1_t2(ceil
                        (t1_index), ceil(t2_index)))/4;
60                  interpolated_transmitted_coherent_vs_time_tau(time_index,
                        tau_index) = (expect_c_out_dag_t1_times_expect_c_out_t2
                        (floor(t1_index), floor(t2_index))+
                        expect_c_out_dag_t1_times_expect_c_out_t2(ceil(t1_index
                        ), floor(t2_index)) +
                        expect_c_out_dag_t1_times_expect_c_out_t2(floor(
                        t1_index), ceil(t2_index))+
                        expect_c_out_dag_t1_times_expect_c_out_t2(ceil(t1_index
                        ), ceil(t2_index)))/4;
```

```
61                  interpolated_G1_emitted_vs_time_tau(time_index, tau_index)
                        = (G1_emitted_vs_t1_t2(floor(t1_index), floor(t2_index)
                        )+G1_emitted_vs_t1_t2(ceil(t1_index), floor(t2_index))+
                        G1_emitted_vs_t1_t2(floor(t1_index), ceil(t2_index))+
                        G1_emitted_vs_t1_t2(ceil(t1_index), ceil(t2_index)))/4;
62                  interpolated_emitted_coherent_vs_time_tau(time_index,
                        tau_index) = (expect_e_out_dag_t1_times_expect_e_out_t2
                        (floor(t1_index), floor(t2_index))+
                        expect_e_out_dag_t1_times_expect_e_out_t2(ceil(t1_index
                        ), floor(t2_index)) +
                        expect_e_out_dag_t1_times_expect_e_out_t2(floor(
                        t1_index), ceil(t2_index))+
                        expect_e_out_dag_t1_times_expect_e_out_t2(ceil(t1_index
                        ), ceil(t2_index)))/4;
63                  interpolated_G1_diffracted_vs_time_tau(time_index,
                        tau_index) = (G1_diffracted_vs_t1_t2(floor(t1_index),
                        floor(t2_index))+G1_diffracted_vs_t1_t2(ceil(t1_index),
                         floor(t2_index))+G1_diffracted_vs_t1_t2(floor(t1_index
                        ), ceil(t2_index))+G1_diffracted_vs_t1_t2(ceil(t1_index
                        ), ceil(t2_index)))/4;
64                  interpolated_diffracted_coherent_vs_time_tau(time_index,
                        tau_index) = (expect_d_out_dag_t1_times_expect_d_out_t2
                        (floor(t1_index), floor(t2_index))+
                        expect_d_out_dag_t1_times_expect_d_out_t2(ceil(t1_index
                        ), floor(t2_index)) +
                        expect_d_out_dag_t1_times_expect_d_out_t2(floor(
                        t1_index), ceil(t2_index))+
                        expect_d_out_dag_t1_times_expect_d_out_t2(ceil(t1_index
                        ), ceil(t2_index)))/4;
65              end
66          end
67      end
68  end
```

C.3.8 Init maps 2level g2 PR vs t1 t2

```
1   % Parameters of the incoming gaussian pulse
2   omega_pulse_ev=omega_d_ev+detuning_pulse_QD_muev*1e-6; %center energy of
        the incoming pulse, in eV
3   omega_pulse=omega_pulse_ev*ev/hbar*1e-12; %in rad/ps
4
5   % Parameters for the computation of time evolutions
6   t_step=(t_max_ps-t_min)/(nb_points_time-1); % Duration of a time step
7   t_list=linspace(t_min,t_max_ps,nb_points_time); % list of all the times
        considered in the computation and plots
8
```

```
9  % Parameters for the computation of preliminary time evolution,
10 % between 0 (long before the pulse) and t_min (time at which we want
11 % to start plotting and integrating the physical quantities
12 nb_points_time_before_t_min = 5; % (Low) time resolution for first
       evolution of the system for initialization
13 t_list_before_t_min = linspace(0,t_min,nb_points_time_before_t_min); % time
        array for first evolution of the sistem
14
15
16
17 % Initialization of qo array of identity operator
18 Id_vs_time = qo;
19 for time_index = 1:nb_points_time
20     Id_vs_time{time_index} = Id;
21 end
22
23
24 full_tau_list = [-flip(t_list(2:end)) t_list(1:end)]; %t_list with both
       negative and positive delays for plotting
25
26 %%% Preallocation of the memory to save computing time
27
28 % Initialization of g2 vs (t1,t2)
29 g2_reflected_vs_t1_t2 = zeros(nb_points_time,nb_points_time);
30 g2_transmitted_vs_t1_t2 = zeros(nb_points_time,nb_points_time);
31 g2_emitted_vs_t1_t2 = zeros(nb_points_time,nb_points_time);
32
33 % Initialization of conditional occupation probabilities vs (t1,t2)
34 occupation_ground_vs_t1_vs_t2_after_click_b_out_at_t1 = zeros(
       nb_points_time,nb_points_time);
35 occupation_excited_vs_t1_vs_t2_after_click_b_out_at_t1 = zeros(
       nb_points_time,nb_points_time);
36 occupation_ground_vs_t1_vs_t2_after_click_c_out_at_t1 = zeros(
       nb_points_time,nb_points_time);
37 occupation_excited_vs_t1_vs_t2_after_click_c_out_at_t1 = zeros(
       nb_points_time,nb_points_time);
38 occupation_ground_vs_t1_vs_t2_after_click_e_out_at_t1 = zeros(
       nb_points_time,nb_points_time);
39 occupation_excited_vs_t1_vs_t2_after_click_e_out_at_t1 = zeros(
       nb_points_time,nb_points_time);
40
41 % Initialization of the "zero" density matrix which will be used to fill
       the
42 % conditional density matrices, using 0 values for t2 < t1
43 zero_density_matrix_vs_t2_before_t1 = qo;
44
```

```
% Initialization of correlated normalized g2(tau)
normalized_g2_vs_delay_emitted = zeros(1,nb_points_time);
normalized_g2_vs_delay_reflected = zeros(1,nb_points_time);
normalized_g2_vs_delay_transmitted = zeros(1,nb_points_time);

% Initialization of uncorrelated normalized g2(tau)
normalized_g2_vs_delay_uncorrelated_emitted = zeros(1,nb_points_time);
normalized_g2_vs_delay_uncorrelated_reflected = zeros(1,nb_points_time);
normalized_g2_vs_delay_uncorrelated_transmitted = zeros(1,nb_points_time);
```

C.3.9 plot 2level CW vs laser frequency

```
switch model
    case 'F'
        text_legend_model='full model';
    case 'A'
        text_legend_model='adiabatic model';
end

% Parameters for the text displayed in figure legends
text_legend_QD=['g=', num2str(g_muev) ' muev    \gamma_{sp}=' num2str(
    gamma_sp_muev) 'muev     \gamma^*=' num2str(gamma_puredephasing_muev) '
    muev'];
text_legend_cav=['\kappa=' num2str(kappa_muev) '\muev   \eta_{top}='
    num2str(eta_top)  '   C=' num2str(g^2/kappa/gamma_decoherence,3)];
text_legend_P_in=['P_{in}=' num2str(P_in_CW_pW) 'pW  n_0=' num2str(4*
    eta_top*abs(b_in_CW)^2/kappa,2) '  n_c=' num2str(gamma_decoherence*
    gamma_sp/(4*g^2),2)];
text_legend={text_legend_QD,text_legend_cav,text_legend_P_in};
% NB: C is the cooperativity and n_c the critical photon number, both
    depend only on the cavity-QED parameters.
% On the contrary, n_0 depends on the incoming power P_in since it is the
    calculated number of intracavity photons
% in the absence of QD (i.e. when g=0). When n_0 is much lower than n_c we
    are in the weak excitation limit.


% Verification of the conservation of total photon flux
fprintf(['CW - ' text_legend_model ' ' ': Maximal relative error on the
    conservation of photon flux: ' num2str(max(abs(1-R_vs_omega-T_vs_omega-
    D_vs_omega-E_vs_omega))) ' \n \n'])

%%%%%%%%%%%%%%%%%%%% Plots %%%%%%%%%%%%%%%%%%%%


if ismember('R',plot_choice)
```

```
25      %Color code : Refl in red, Transm + Diffr/lost in blue, Spont. Em. in
            magenta
26      figure('Name',['CW - ' text_legend_model ' ' ' - Reflectivity vs laser
            photon energy - Pin = ' num2str(P_in_CW_pW) ' pW'],'NumberTitle','
            off')
27      %%% UNUSED HERE: allows comparing total part and coherent part
28      % plot(omega_laser_list_ev_CW,real(R_vs_omega),'r',
            omega_laser_list_ev_CW,real(R_coh_vs_omega),'r--');
29      % legend('Fraction of reflected photons','Fraction of coherent
            reflected photons')
30      plot(detuning_list_muev,real(R_vs_omega),'r');
31      xlabel('\omega_{laser}-\omega_{d} [\mueV]'); ylabel('Reflectivity');
32      ylim([0 1]);
33      text(detuning_list_muev(ceil(nb_points_spectrum*0.55)),0.8,text_legend,
            'FontSize',9)
34      legend('Reflected photons','Location','Northeast')
35  end
36
37
38  if ismember('T',plot_choice)
39      figure('Name',['CW - ' text_legend_model ' ' ' - Transmission vs laser
            photon energy - Pin = ' num2str(P_in_CW_pW) ' pW'],'NumberTitle','
            off')
40      %%% UNUSED HERE: allows comparing total part and coherent part
41      % plot(omega_laser_list_ev_CW,real(T_vs_omega+D_vs_omega),'b',
            omega_laser_list_ev_CW,real(T_coh_vs_omega+D_coh_vs_omega),'b--');
42      % legend('Fraction of transmitted + diffracted photons','Fraction of
            coherent transmitted/diffracted photons')
43      plot(detuning_list_muev,real(T_vs_omega+D_vs_omega),'b');
44
45      xlabel('\omega_{laser}-\omega_{d} [\mueV]'); ylabel('Transmission +
            diffraction/losses');
46      ylim([0 1]);
47      text(detuning_list_muev(ceil(nb_points_spectrum*0.55)),0.12,text_legend
            ,'FontSize',9)
48      legend('Transmitted + diffracted photons','Location','Northeast')
49  end
50
51
52  if ismember('E',plot_choice)
53      figure('Name',['CW - ' text_legend_model ' ' ' - Spontaneous emission
            outside the mode vs laser photon energy - Pin = ' num2str(
            P_in_CW_pW) ' pW'],'NumberTitle','off')
54      % %%% UNUSED HERE: allows comparing total part and coherent part
55      % plot(omega_laser_list_ev_CW,real(E_vs_omega),'m',
            omega_laser_list_ev_CW,real(E_coh_vs_omega),'m--');
```

```
56      % legend('Fraction of spontaneously emitted photons','Fraction of
            coherent spontaneously-emitted photons')
57      plot(detuning_list_muev,real(E_vs_omega),'m');
58  %     xlabel('\omega_{laser}-\omega_{d} [\mueV]'); ylabel('Fraction of
        photons emitted outside the mode');
59      ylim([0 1]);
60      text(detuning_list_muev(ceil(nb_points_spectrum*0.55)),0.8,text_legend,
            'FontSize',9)
61      legend('Photons emitted outside the mode','Location','Northeast')
62  end
63
64
65
66  if ismember('O',plot_choice)
67      figure('Name',['CW - ' text_legend_model ' ' ' - Occupation
            probabilities vs laser photon energy - Pin = ' num2str(P_in_CW_pW)
            ' pW'],'NumberTitle','off')
68      plot(detuning_list_muev,real(occupation_excited_state_vs_omega),'r',...
69           detuning_list_muev,real(occupation_ground_state_vs_omega),'b');
70      xlabel('\omega_{laser}-\omega_{d} [\mueV]'); ylabel('Occupation
            probability');
71      ylim([-0.05 1.05]);
72      text(detuning_list_muev(ceil(nb_points_spectrum*0.55)),0.7,text_legend,
            'FontSize',9)
73      legend('Occupation state |e>','Occupation state |g>','Location','best')
74  end
```

C.3.10 plot 2level PW vs time

```
1   switch model
2       case 'F'
3           text_legend_model='Full model';
4       case 'A'
5           text_legend_model='Adiabatic model';
6   end
7
8   fprintf([text_legend_model ' - Average number of injected photons per pulse
        : ' num2str(abs(sum(flux_injected_photons_vs_time)*t_step)) ' \n'])
9   fprintf([text_legend_model ' - Average number of reflected photons per
        pulse : ' num2str(abs(sum(flux_reflected_photons_vs_time)*t_step)) ' \n
        '])
10  fprintf([text_legend_model ' - Average number of transmitted photons per
        pulse : ' num2str(abs(sum(flux_transmitted_photons_vs_time)*t_step)) '
        \n'])
```

```
11 fprintf([text_legend_model ' - Average number of diffracted photons per
       pulse : ' num2str(abs(sum(flux_diffracted_photons_vs_time)*t_step)) ' \
       n'])
12 fprintf([text_legend_model ' - Average number of spontaneously-emitted
       photons per pulse : ' num2str(abs(sum(flux_emitted_photons_vs_time)*
       t_step)) ' \n \n'])
13
14 %Verification of the conservation of photon number
15 fprintf([text_legend_model ' - Verification - relative error on the
       conservation of photon number : ' num2str(abs(sum(
       flux_reflected_photons_vs_time+flux_transmitted_photons_vs_time+
       flux_diffracted_photons_vs_time+flux_emitted_photons_vs_time-
       flux_injected_photons_vs_time)/sum(flux_injected_photons_vs_time))) ' \
       n \n'])
16
17 %Verification of the maximal photon number in the last Fock state (for full
18 %model only)
19 if model == 'F'
20     last_Fock_state_ket = basis(N,N);
21     occupation_last_Fock_state=tensor(last_Fock_state_ket*
           last_Fock_state_ket',id_QD);
22     expect_occupation_last_Fock_state_vs_time=expect(
           occupation_last_Fock_state,rho_vs_time); %
23     fprintf(['Full model - Verification - maximal occupation of the last
           Fock state : ' num2str(max(abs(
           expect_occupation_last_Fock_state_vs_time))) ' \n'])
24 end
25
26 %%%%%%%%%% Plots %%%%%%%%%%
27 % Parameters for the text displayed in figure legends
28 text_legend_QD = ['g=', num2str(g_muev) ' muev    \gamma_{sp}=' num2str(
       gamma_sp_muev) 'muev     \gamma^*=' num2str(gamma_puredephasing_muev) '
       muev'];
29 text_legend_cav = ['\kappa=' num2str(kappa_muev) 'muev   \eta_{top}='
       num2str(eta_top)  '   C=' num2str(g^2/kappa/gamma_decoherence,3)];
30 text_legend_pulse = ['N_{in}=' num2str(Nb_photons) '       FWHM_{pulse}='
       num2str(FWHM) 'ps'];
31 text_legend = {text_legend_QD,text_legend_cav,text_legend_pulse};
32 % NB: C is the cooperativity
33
34 if ismember('F',plot_choice)
35     figure('Name',['PR - ' text_legend_model ' - Photon flux vs time - Nin
           = ' num2str(Nb_photons) ' - Pulse FWHM = ' num2str(FWHM) ' ps -
           pulsation laser = ' num2str(omega_pulse_ev) ' eV'],'NumberTitle','
           off')
```

```
36      plot(t_list,real(flux_reflected_photons_vs_time),'r',t_list,real(
            flux_transmitted_photons_vs_time+flux_diffracted_photons_vs_time),'
            b',t_list,real(flux_emitted_photons_vs_time),'g')
37      legend('Flux of reflected photons','Flux of transmitted + diffracted/
            lost photons', 'Flux of photons spontaneously-emitted outside the
            mode')
38      xlabel('Time t [ps]');ylabel('Photon flux [ps^{-1}]');
39      text(t_list(floor(nb_points_time*0.6)),0.7*max(
            flux_reflected_photons_vs_time),text_legend,'FontSize',9)
40  end
41
42  if ismember('O',plot_choice)
43      figure('Name',['PR - ' text_legend_model ' - Occupation probabilities
            vs time - Nin = ' num2str(Nb_photons) ' - Pulse FWHM = ' num2str(
            FWHM) ' ps - pulsation laser = ' num2str(omega_pulse_ev) ' eV'],'
            NumberTitle','off')
44      plot(t_list,real(expect_sigma_dag_sigma_vs_time),'r',t_list,real(
            expect_sigma_sigma_dag_vs_time),'k')
45      legend('Occupation of excited state |e>','Occupation of ground state |g
            >')
46      xlabel('Time t [ps]');ylabel('Occupation probability');
47      ylim([0 1])
48      text(t_list(floor(nb_points_time*0.6)),0.7,text_legend,'FontSize',9)
49  end
```

C.3.11 plot 2level g1SD CW vs delay and frequency

```
1   switch model
2       case 'F'
3           text_legend_model='full model';
4       case 'A'
5           text_legend_model='adiabatic model';
6   end
7
8   % Parameters for the text displayed in figure legends
9   text_legend_QD = ['g=', num2str(g_muev) ' muev    \gamma_{sp}=' num2str(
        gamma_sp_muev) 'muev     \gamma^*=' num2str(gamma_puredephasing_muev) '
        muev'];
10  text_legend_cav = ['\kappa=' num2str(kappa_muev) 'muev   \eta_{top}='
        num2str(eta_top)  '   C=' num2str(g^2/kappa/gamma_decoherence,3)];
11  text_legend_P_in = ['P_{in}=' num2str(P_in_CW_pW) 'pW  n_0=' num2str(4*
        eta_top*abs(b_in_CW)^2/kappa,2) '  n_c=' num2str(gamma_decoherence*
        gamma_sp/(4*g^2),2)];
12  text_legend_omega = [' \omega_{laser}-\omega_d=' num2str(
        detuning_laser_QD_muev) ' muev'];
```

```
13 text_legend = {text_legend_QD,text_legend_cav,text_legend_P_in,
       text_legend_omega};
14 % NB: C is the cooperativity and n_c the critical photon number, both
       depend only on the cavity-QED parameters.
15 % On the contrary, n_0 depends on the incoming power P_in since it is the
       calculated number of intracavity photons
16 % in the absence of QD (i.e. when g=0). When n_0 is much lower than n_c we
       are in the weak excitation limit.
17
18 %Verification of the maximal photon number in the last Fock state (for full
        model only)
19 if model == 'F'
20    fprintf(['Full model - Verification - maximal occupation of the last
          Fock state : ' num2str(abs(expect(occupation_last_Fock_state,
          density_matrix_stationary_state))) ' \n \n'])
21 end
22
23 % Verification of photon flux conservation
24 fprintf(['g1SDCW - ' text_legend_model ': Relative error on photon flux
       conservation: ' num2str(abs(flux_injected_photons-
       flux_reflected_photons-flux_transmitted_photons-flux_diffracted_photons
       -flux_emitted_photons)/flux_injected_photons) ' \n \n'])
25
26 % Verifications on the extreme values of g1(tau) functions at zero and "
       infinite" delay
27 fprintf(['g1SDCW - ' text_legend_model ': Relative error on g1(0) = 1: '
       num2str(abs(1-g_1_reflected_vs_tau(1))) ' \n'])
28 fprintf(['g1SDCW - ' text_legend_model ': Relative error on g1(0) = 1: '
       num2str(abs(1-g_1_transmitted_vs_tau(1))) ' \n'])
29 fprintf(['g1SDCW - ' text_legend_model ': Relative error on g1(0) = 1: '
       num2str(abs(1-g_1_emitted_vs_tau(1))) ' \n  \n'])
30 fprintf(['g1SDCW - ' text_legend_model ': Relative error on g1(infty) =
       coherent fraction for reflected photons: ' num2str(abs(
       flux_reflected_photons_laser_coherent/flux_reflected_photons-
       g_1_reflected_vs_tau(nb_points_delay))) ' \n'])
31 fprintf(['g1SDCW - ' text_legend_model ': Relative error on g1(infty) =
       coherent fraction for transmitted photons: ' num2str(abs(
       flux_transmitted_photons_laser_coherent/flux_transmitted_photons-
       g_1_transmitted_vs_tau(nb_points_delay))) ' \n'])
32 fprintf(['g1SDCW - ' text_legend_model ': Relative error on g1(infty) =
       coherent fraction for emitted photons: ' num2str(abs(
       flux_emitted_photons_laser_coherent/flux_emitted_photons-
       g_1_emitted_vs_tau(nb_points_delay))) ' \n \n'])
33
34 % Verification of spectral density normalization for the various fields (
       incoherent part only). Its integral over
```

```
35 % the whole spectrum must correspond to the incoherent photon flux,
       considering that each spectral density is measured
36 % in ps-1/muev, which gives a flux in ps-1 when multiplying by the photon
       energy step in mueV (omega_step_ev*1e-6)
37 fprintf(['g1SDCW - ' text_legend_model ': Relative error on the spectral
       density normalization - reflected photons (incoherent part) : ' num2str
       (abs(sum(spectral_density_flux_reflected_photons_incoh_vs_omega*(
       omega_step_muev))-flux_reflected_photons_incoh)/
       flux_reflected_photons_incoh) ' \n'])
38 fprintf(['g1SDCW - ' text_legend_model ': Relative error on the spectral
       density normalization - transmitted photons (incoherent part) :'
       num2str(abs(sum(
       spectral_density_flux_transmitted_photons_incoh_vs_omega*(
       omega_step_muev))-flux_transmitted_photons_incoh)/
       flux_reflected_photons_incoh) ' \n'])
39 fprintf(['g1SDCW - ' text_legend_model ': Relative error on the spectral
       density normalization - emitted photons (incoherent part) : ' num2str(
       abs(sum(spectral_density_flux_emitted_photons_incoh_vs_omega*(
       omega_step_muev))-flux_emitted_photons_incoh)/
       flux_emitted_photons_incoh) ' \n \n'])
40
41
42 %%%%%%%%%%%%%%%%%%  Plots of g(1)(tau)           %%%%%%%%%%%%%%%%%%%%%%
43 if ismember('G',plot_choice)
44     figure('Name',['g1SDCW - ' text_legend_model ' -  |(g1)| vs tau  -
           reflected photons - Pin = ' num2str(P_in_CW_pW) ' pW - Pulsation
           laser = ' num2str(omega_laser_ev) ' eV' ],'NumberTitle','off')
45     plot(full_tau_list,abs(full_g_1_reflected_vs_tau),'r');
46     xlabel('Delay \tau [ps]'); ylabel('|(g1)| - reflected photons');
47     xlim([min(full_tau_list) max(full_tau_list)]);ylim([0 real(max(
           full_g_1_reflected_vs_tau))]);
48     text(full_tau_list(ceil(nb_points_delay/20)),0.2*max(
           full_g_1_reflected_vs_tau),text_legend,'FontSize',9)
49     annotation('textbox',[.15 .65 .2 .2],'String',['Coherent   : ' num2str(
           abs(100*flux_reflected_photons_laser_coherent/
           flux_reflected_photons)) ' %'],'FitBoxToText','on');
50     annotation('textbox',[.15 .55 .2 .2],'String',['Incoherent : ' num2str(
           abs(100*flux_reflected_photons_incoh/flux_reflected_photons)) ' %'
           ],'FitBoxToText','on');
51
52     figure('Name',['g1SDCW - ' text_legend_model ' -  |(g1)| vs tau -
           transmitted photons - Pin = ' num2str(P_in_CW_pW) ' pW - Pulsation
           laser = ' num2str(omega_laser_ev) ' eV' ],'NumberTitle','off')
53     plot(full_tau_list,abs(full_g_1_transmitted_vs_tau),'r');
54     xlabel('Delay \tau [ps]'); ylabel('|(g1)| - transmitted photons');
```

```
55      xlim([min(full_tau_list) max(full_tau_list)]);ylim([0 real(max(
            full_g_1_transmitted_vs_tau))]);
56      text(full_tau_list(ceil(nb_points_delay/20)),0.2*max(
            full_g_1_transmitted_vs_tau),text_legend,'FontSize',9)
57      annotation('textbox',[.15 .65 .2 .2],'String',['Coherent   : ' num2str(
            abs(100*flux_transmitted_photons_laser_coherent/
            flux_transmitted_photons)) ' %'],'FitBoxToText','on');
58      annotation('textbox',[.15 .55 .2 .2],'String',['Incoherent : ' num2str(
            abs(100*flux_transmitted_photons_incoh/flux_transmitted_photons)) '
             %'],'FitBoxToText','on');
59
60      figure('Name',['g1SDCW - ' text_legend_model ' -real |(g1)| vs tau -
            emitted photons in leaky modes - Pin = ' num2str(P_in_CW_pW) ' pW -
             Pulsation laser = ' num2str(omega_laser_ev) ' eV' ],'NumberTitle',
            'off')
61      plot(full_tau_list,abs(full_g_1_emitted_vs_tau),'r');
62      xlabel('Delay \tau [ps]'); ylabel('|(g1)| - emitted photons');
63      xlim([min(full_tau_list) max(full_tau_list)]);ylim([0 real(max(
            full_g_1_emitted_vs_tau))]);
64      text(full_tau_list(ceil(nb_points_delay/20)),0.2*max(
            full_g_1_emitted_vs_tau),text_legend,'FontSize',9)
65      annotation('textbox',[.15 .65 .2 .2],'String',['Coherent   : ' num2str(
            abs(100*flux_emitted_photons_laser_coherent/flux_emitted_photons))
            ' %'],'FitBoxToText','on');
66      annotation('textbox',[.15 .55 .2 .2],'String',['Incoherent : ' num2str(
            abs(100*flux_emitted_photons_incoh/flux_emitted_photons)) ' %'],'
            FitBoxToText','on');
67  end
68  %%%%%%%%%%%%%%%%  Plotting of spectra densities  %%%%%%%%%%%%%%%%%%
69  if ismember('S',plot_choice)
70
71      figure('Name',['g1SDCW - ' text_legend_model ' - Spectral density of
            flux - Incoherent reflected photons - Pin = ' num2str(P_in_CW_pW) '
             pW - Pulsation laser = ' num2str(omega_laser_ev) ' eV' ],'
            NumberTitle','off')
72      plot((omega_list_ev-omega_laser_ev)*1e6,abs(
            spectral_density_flux_reflected_photons_incoh_vs_omega));
73      xlabel('\omega-\omega_{laser}  [muev]');ylabel('Flux spectral density [
            photons / ps / mueV]');
74      text(1e6*(omega_list_ev(ceil(nb_points_spectrum/20))-omega_laser_ev)
            ,0.9*real(max(abs(
            spectral_density_flux_reflected_photons_incoh_vs_omega))),
            text_legend,'FontSize',9)
75      title('Flux spectral density - incoherent part of the reflected field')
76
```

```
77      figure('Name',['g1SDCW - ' text_legend_model ' - Spectral density of
            flux - Incoherent transmitted photons - Pin = ' num2str(P_in_CW_pW)
             ' pW - Pulsation laser = ' num2str(omega_laser_ev) ' eV' ],'
            NumberTitle','off')
78      plot((omega_list_ev-omega_laser_ev)*1e6,abs(
            spectral_density_flux_transmitted_photons_incoh_vs_omega));
79      xlabel('\omega-\omega_{laser}  [muev]');ylabel('Flux spectral density [
            photons / ps / mueV]');
80      text(1e6*(omega_list_ev(ceil(nb_points_spectrum/20))-omega_laser_ev)
            ,0.9*real(max(abs(
            spectral_density_flux_transmitted_photons_incoh_vs_omega))),
            text_legend,'FontSize',9)
81      title('Flux spectral density - incoherent part of the transmitted field
            ')
82
83      figure('Name',['g1SDCW - ' text_legend_model ' - Spectral density of
            flux - Incoherent emitted photons - Pin = ' num2str(P_in_CW_pW) '
            pW - Pulsation laser = ' num2str(omega_laser_ev) ' eV' ],'
            NumberTitle','off')
84      plot((omega_list_ev-omega_laser_ev)*1e6, abs(
            spectral_density_flux_emitted_photons_incoh_vs_omega));
85      xlabel('\omega-\omega_{laser}  [muev]');ylabel('Flux spectral density [
            photons / ps / mueV]');
86      text(1e6*(omega_list_ev(ceil(nb_points_spectrum/20))-omega_laser_ev)
            ,0.9*real(max(abs(
            spectral_density_flux_emitted_photons_incoh_vs_omega))),text_legend
            ,'FontSize',9)
87      title('Flux spectral density - incoherent part of the emitted field')
88  end
```

C.3.12 plot 2level g1 WDF PW

```
1   %Verification of the photon number conservation
2   fprintf(['RFTR: relative error of the photon number conservation: ' num2str
        (abs(Nb_reflected_photons+Nb_transmitted_photons+Nb_diffracted_photons+
        Nb_emitted_photons-Nb_photons_pulse)/Nb_photons_pulse) ' \n'])
3   %
4   switch model
5       case 'F'
6           text_legend_model='full model';
7       case 'A'
8           text_legend_model='adiabatic model';
9   end
10  %% plotting g1(t1,t2)
11
12  if ismember('g',plot_choice)
```

```
13
14      figure('Name',['g1 reflected vs (t1,t2) - ' text_legend_model],'
            NumberTitle','off')
15      surf(flip(t_list),t_list,flip(abs(g1_reflected_vs_t1_t2),2),'LineStyle'
            ,'none')
16      xlabel('t_1 [ps]')
17      ylabel('t_2 [ps]')
18      title('reflected |g^{(1)}(t_1,t_2)|')
19      view(2)
20      colorbar
21
22      figure('Name',['g1 diffracted vs (t1,t2) - ' text_legend_model],'
            NumberTitle','off')
23      surf(flip(t_list),t_list,flip(abs(g1_diffracted_vs_t1_t2),2),'LineStyle
            ','none')
24      xlabel('t_1 [ps]')
25      ylabel('t_2 [ps]')
26      title('diffracted |g^{(1)}(t_1,t_2)|')
27      view(2)
28      colorbar
29
30      figure('Name',['g1 transmitted vs (t1,t2) - ' text_legend_model],'
            NumberTitle','off')
31      surf(flip(t_list),t_list,flip(abs(g1_transmitted_vs_t1_t2),2),'
            LineStyle','none')
32      xlabel('t_1 [ps]')
33      ylabel('t_2 [ps]')
34      title('transmitted |g^{(1)}(t_1,t_2)|')
35      view(2)
36      colorbar
37
38      figure('Name',['g1 emitted vs (t1,t2) - ' text_legend_model],'
            NumberTitle','off')
39      surf(flip(t_list),t_list,flip(abs(g1_emitted_vs_t1_t2),2),'LineStyle','
            none')
40      xlabel('t_1 [ps]')
41      ylabel('t_2 [ps]')
42      title('emitted |g^{(1)}(t_1,t_2)|')
43      view(2)
44      colorbar
45  end
46  %% plottig G1 vs(t1,t2)
47  if ismember('G',plot_choice)
48
49      figure('Name',['G1 reflected vs (t1,t2) - ' text_legend_model],'
            NumberTitle','off')
```

```
50      surf(flip(t_list),t_list,flip(abs(G1_reflected_vs_t1_t2),2),'LineStyle'
            ,'none')
51      xlabel('t_1 [ps]')
52      ylabel('t_2 [ps]')
53      title('|<b_{out}''(t_1) b_{out}(t_2)>|')
54      view(2)
55      colorbar
56
57      figure('Name',['G1 transmitted vs (t1,t2) - ' text_legend_model],'
            NumberTitle','off')
58      surf(flip(t_list),t_list,flip(abs(G1_transmitted_vs_t1_t2),2),'
            LineStyle','none')
59      xlabel('t_1 [ps]')
60      ylabel('t_2 [ps]')
61      title('|<c_{out}''(t_1) c_{out}(t_2)>|')
62      view(2)
63      colorbar
64
65      figure('Name',['G1 diffracted vs (t1,t2) - ' text_legend_model],'
            NumberTitle','off')
66      surf(flip(t_list),t_list,flip(abs(G1_diffracted_vs_t1_t2),2),'LineStyle
            ','none')
67      xlabel('t_1 [ps]')
68      ylabel('t_2 [ps]')
69      title('|<d_{out}''(t_1) d_{out}(t_2)>|')
70      view(2)
71      colorbar
72
73      figure('Name',['G1 emitted vs (t1,t2) - ' text_legend_model],'
            NumberTitle','off')
74      surf(flip(t_list),t_list,flip(abs(G1_emitted_vs_t1_t2),2),'LineStyle','
            none')
75      xlabel('t_1 [ps]')
76      ylabel('t_2 [ps]')
77      title('|<e_{out}''(t_1) e_{out}(t_2)>|')
78      view(2)
79      colorbar
80  end
81  % Parameters for the text displayed in figure legends
82  text_legend_QD = ['g=', num2str(g_muev) ' muev    \gamma_{sp}=' num2str(
        gamma_sp_muev) 'muev     \gamma^*=' num2str(gamma_puredephasing_muev) '
        muev'];
83  text_legend_cav = ['\kappa=' num2str(kappa_muev) 'muev   \eta_{top}='
        num2str(eta_top)  '   C=' num2str(g^2/kappa/gamma_decoherence,3)];
84  text_legend_pulse = ['N_{in}=' num2str(Nb_photons_pulse) '       FWHM_{
        pulse}=' num2str(FWHM_pulse) 'ps'];
```

```
85 text_legend = {text_legend_QD,text_legend_cav,text_legend_pulse};
86 % NB: C is the cooperativity
87
88 %% plotting interpolated G1(time,tau)
89 if ismember('I',plot_choice)
90     figure('Name',['interpolated G1 reflected vs time tau - '
           text_legend_model ' - Nb_photons = ' num2str(Nb_photons_pulse)],'
           NumberTitle','off')
91     surf(time_list,tau_list, abs(interpolated_G1_reflected_vs_time_tau'),'
           LineStyle','none')
92     axis xy
93     colorbar
94     xlabel('t, origin at beginning of 1st evolution [ps]','FontSize',14)
95     ylabel('\tau [ps]','FontSize',14)
96     title('interpolated G1 reflected vs time tau')
97     view(2)
98
99     figure('Name',['interpolated G1 transmitted vs time tau - '
           text_legend_model ' - Nb_photons = ' num2str(Nb_photons_pulse)],'
           NumberTitle','off')
100    surf(time_list,tau_list, abs(interpolated_G1_transmitted_vs_time_tau'),
           'LineStyle','none')
101    axis xy
102    colorbar
103    xlabel('t, origin at beginning of 1st evolution [ps]','FontSize',14)
104    ylabel('\tau [ps]','FontSize',14)
105    title('interpolated G1 transmitted vs time tau')
106    view(2)
107
108    figure('Name',['interpolated G1 emitted vs time tau - '
           text_legend_model ' - Nb_photons = ' num2str(Nb_photons_pulse)],'
           NumberTitle','off')
109    surf(time_list,tau_list, abs(interpolated_G1_emitted_vs_time_tau'),'
           LineStyle','none')
110    axis xy
111    colorbar
112    xlabel('t, origin at beginning of 1st evolution [ps]','FontSize',14)
113    ylabel('\tau [ps]','FontSize',14)
114    title('interpolated G1 emitted vs time tau')
115    view(2)
116
117    figure('Name',['interpolated G1 diffracted vs time tau - '
           text_legend_model ' - Nb_photons = ' num2str(Nb_photons_pulse)],'
           NumberTitle','off')
118    surf(time_list,tau_list, abs(interpolated_G1_diffracted_vs_time_tau'),'
           LineStyle','none')
```

```
119     axis xy
120     colorbar
121     xlabel('t, origin at beginning of 1st evolution [ps]','FontSize',14)
122     ylabel('\tau [ps]','FontSize',14)
123     title('interpolated G1 diffracted vs time tau')
124     view(2)
125 end
126 %% Plotting WDF
127 if ismember('W',plot_choice)
128     figure('Name',['WDF reflected photons - ' text_legend_model ' -
            Nb_photons = ' num2str(Nb_photons_pulse)],'NumberTitle','off')
129     imagesc(time_list,(omega_list_full_spectrum_muev - omega_pulse_ev*1e6),
              real(WDF_interpolated_G1_reflected_vs_time_tau_full_spectrum)');
130     axis xy
131     colorbar
132     xlabel('t, origin at beginning of 1st evolution [ps]')
133     ylabel('\omega-\omega_p [\mueV]','FontSize',14)
134     title('Wigner distribution reflected [1/(\mueV\cdotps)]')
135
136     figure('Name',['WDF reflected photons - zoomed spectrum - '
            text_legend_model ' - Nb_photons = ' num2str(Nb_photons_pulse)],'
            NumberTitle','off')
137     imagesc(time_list,(omega_list_ev - omega_pulse_ev)*1e6,  real(
            WDF_interpolated_G1_reflected_vs_time_tau'));
138     axis xy
139     colorbar
140     xlabel('t, origin at beginning of 1st evolution [ps]')
141     ylabel('\omega-\omega_p [\mueV]','FontSize',14)
142     title('zoomed Wigner distribution reflected [1/(\mueV\cdotps)]')
143
144     figure('Name',['WDF transmitted photons - ' text_legend_model ' -
            Nb_photons = ' num2str(Nb_photons_pulse)],'NumberTitle','off')
145     imagesc(time_list,(omega_list_full_spectrum_muev - omega_pulse_ev*1e6),
              real(WDF_interpolated_G1_transmitted_vs_time_tau_full_spectrum)')
            ;
146     axis xy
147     colorbar
148     xlabel('t, origin at beginning of 1st evolution [ps]')
149     ylabel('\omega-\omega_p [\mueV]','FontSize',14)
150     title('Wigner distribution transmitted [1/(\mueV\cdotps)]')
151
152     figure('Name',['WDF transmitted photons - zoomed spectrum -'
            text_legend_model ' - Nb_photons = ' num2str(Nb_photons_pulse)],'
            NumberTitle','off')
153     imagesc(time_list,(omega_list_ev - omega_pulse_ev)*1e6,  real(
            WDF_interpolated_G1_transmitted_vs_time_tau'));
```

```
154     axis xy
155     colorbar
156     xlabel('t, origin at beginning of 1st evolution [ps]')
157     ylabel('\omega-\omega_p [\mueV]','FontSize',14)
158     title('zoomed Wigner distribution transmitted [1/(\mueV\cdotps)]')
159
160     figure('Name',['WDF emitted photons - ' text_legend_model ' -
            Nb_photons = ' num2str(Nb_photons_pulse)],'NumberTitle','off')
161     imagesc(time_list,(omega_list_full_spectrum_muev - omega_pulse_ev*1e6),
              real(WDF_interpolated_G1_emitted_vs_time_tau_full_spectrum)');
162     axis xy
163     colorbar
164     xlabel('t, origin at beginning of 1st evolution [ps]')
165     ylabel('\omega-\omega_p [\mueV]','FontSize',14)
166     title('Wigner distribution emitted [1/(\mueV\cdotps)]')
167
168     figure('Name',['WDF emitted photons - zoomed spectrum - '
            text_legend_model ' - Nb_photons = ' num2str(Nb_photons_pulse)],'
            NumberTitle','off')
169     imagesc(time_list,(omega_list_ev - omega_pulse_ev)*1e6,  real(
            WDF_interpolated_G1_emitted_vs_time_tau'));
170     axis xy
171     colorbar
172     xlabel('t, origin at beginning of 1st evolution [ps]')
173     ylabel('\omega-\omega_p [\mueV]','FontSize',14)
174     title('zoomed Wigner distribution emitted [1/(\mueV\cdotps)]')
175
176     figure('Name',['WDF diffracted photons - ' text_legend_model ' -
            Nb_photons = ' num2str(Nb_photons_pulse)],'NumberTitle','off')
177     imagesc(time_list,(omega_list_full_spectrum_muev - omega_pulse_ev*1e6),
              real(WDF_interpolated_G1_diffracted_vs_time_tau_full_spectrum)');
178     axis xy
179     colorbar
180     xlabel('t, origin at beginning of 1st evolution [ps]')
181     ylabel('\omega-\omega_p [\mueV]','FontSize',14)
182     title('Wigner distribution diffracted  [1/(\mueV\cdotps)]')
183
184     figure('Name',['WDF diffracted photons - zoomed spectrum - '
            text_legend_model ' - Nb_photons = ' num2str(Nb_photons_pulse)],'
            NumberTitle','off')
185     imagesc(time_list,(omega_list_ev - omega_pulse_ev)*1e6,  real(
            WDF_interpolated_G1_diffracted_vs_time_tau'));
186     axis xy
187     colorbar
188     xlabel('t, origin at beginning of 1st evolution [ps]')
189     ylabel('\omega-\omega_p [\mueV]','FontSize',14)
```

```
190     title('zoomed Wigner distribution diffracted [1/(\mueV\cdotps)]')
191 end
192 %% Plotting photon fluxes
193 if ismember('F',plot_choice)
194     figure('Name',['Flux reflected photons vs time - Nb_photons = ' num2str
            (Nb_photons_pulse)],'NumberTitle','off')
195     plot(t_list, flux_reflected_photons_vs_time,'r', time_list, real(sum(
            WDF_interpolated_G1_reflected_vs_time_tau_full_spectrum,2))*
            omega_step_muev,'g--', t_list,
            flux_reflected_photons_pulse_coherent_vs_time,'k',time_list, real(
            sum(WDF_interpolated_reflected_coherent_vs_time_tau_full_spectrum
            ,2))*omega_step_muev,'m--');
196     xlabel('Time t [ps]');ylabel('Flux reflected photon [ps^{-1}]');
197     legend('<b_{out}''b_{out}>','total flux by WDF(t,\omega)','<b_{out}''><
            b_{out}>','total coherent flux by WDF(t,\omega)');
198     text(t_list(floor(nb_points_time*0.6)),0.2*max(
            flux_reflected_photons_vs_time),text_legend,'FontSize',9)
199
200     figure('Name',['Flux transmitted photons vs time - Nb_photons = '
            num2str(Nb_photons_pulse)],'NumberTitle','off')
201     plot(t_list, flux_transmitted_photons_vs_time,'r', time_list, real(sum(
            WDF_interpolated_G1_transmitted_vs_time_tau_full_spectrum,2))*
            omega_step_muev,'g--', t_list,
            flux_transmitted_photons_pulse_coherent_vs_time,'k',time_list, real
            (sum(
            WDF_interpolated_transmitted_coherent_vs_time_tau_full_spectrum,2))
            *omega_step_muev,'m--');
202     xlabel('Time t [ps]');ylabel('Flux transmitted photon [ps^{-1}]');
203     legend('<c_{out}''c_{out}>','total flux by WDF(t,\omega)','<c_{out}''><
            c_{out}>','total coherent flux by WDF(t,\omega)');
204     text(t_list(floor(nb_points_time*0.6)),0.2*max(
            flux_transmitted_photons_vs_time),text_legend,'FontSize',9)
205
206     figure('Name',['Flux emitted photons vs time - Nb_photons = ' num2str(
            Nb_photons_pulse)],'NumberTitle','off')
207     plot(t_list, flux_emitted_photons_vs_time,'r', time_list, real(sum(
            WDF_interpolated_G1_emitted_vs_time_tau_full_spectrum,2))*
            omega_step_muev,'g--', t_list,
            flux_emitted_photons_pulse_coherent_vs_time,'k',time_list, real(sum
            (WDF_interpolated_emitted_coherent_vs_time_tau_full_spectrum,2))*
            omega_step_muev,'m--');
208     xlabel('Time t [ps]');ylabel('Flux emitted photon [ps^{-1}]');
209     legend('<d_{out}''d_{out}>','total flux by WDF(t,\omega)','<d_{out}''><
            d_{out}>','total coherent flux by WDF(t,\omega)');
210     text(t_list(floor(nb_points_time*0.6)),0.2*max(
            flux_emitted_photons_vs_time),text_legend,'FontSize',9)
```

```
211
212     figure('Name',['Flux diffracted photons vs time - Nb_photons = '
            num2str(Nb_photons_pulse)],'NumberTitle','off')
213     plot(t_list, flux_diffracted_photons_vs_time,'r', time_list, real(sum(
            WDF_interpolated_G1_diffracted_vs_time_tau_full_spectrum,2))*
            omega_step_muev,'g--', t_list,
            flux_diffracted_photons_pulse_coherent_vs_time,'k',time_list, real(
            sum(WDF_interpolated_diffracted_coherent_vs_time_tau_full_spectrum
            ,2))*omega_step_muev,'m--');
214     xlabel('Time t [ps]');ylabel('Flux diffracted photon [ps^{-1}]');
215     legend('<e_{out}''e_{out}>','total flux by WDF(t,\omega)','<e_{out}''><
            e_{out}>','total coherent flux by WDF(t,\omega)');
216     text(t_list(floor(nb_points_time*0.6)),0.2*max(
            flux_diffracted_photons_vs_time),text_legend,'FontSize',9)
217
218
219     figure('Name',['Spectral density reflected photons vs omega -
            Nb_photons = ' num2str(Nb_photons_pulse)],'NumberTitle','off')
220     plot((omega_list_ev - omega_pulse_ev)*1e6, real(
            ESD_reflected_photons_vs_omega) ,'r',(omega_list_ev -
            omega_pulse_ev)*1e6, real(
            ESD_coherent_reflected_photons_laser_vs_omega),'g',(omega_list_ev -
             omega_pulse_ev)*1e6, real(ESD_reflected_photons_vs_omega)-real(
            ESD_coherent_reflected_photons_laser_vs_omega),'b');
221     xlim([min(omega_list_ev - omega_pulse_ev)*1e6 max(omega_list_ev -
            omega_pulse_ev)*1e6]);
222     xlabel('\omega-\omega_p [\mueV]');ylabel('Reflected field spectral
            density');
223     legend('WDF(\omega) - total reflected flux', 'WDF(\omega) coherent flux
            ', 'WDF(\omega) - incoherent flux')
224     text(omega_list_ev(1) - omega_pulse_ev,0.7*max(
            ESD_reflected_photons_vs_omega),text_legend,'FontSize',9)
225
226     figure('Name',['Spectral density transmitted photons vs omega -
            Nb_photons = ' num2str(Nb_photons_pulse)],'NumberTitle','off')
227     plot((omega_list_ev - omega_pulse_ev)*1e6, real(
            ESD_transmitted_photons_vs_omega) ,'r',(omega_list_ev -
            omega_pulse_ev)*1e6, real(
            ESD_coherent_transmitted_photons_laser_vs_omega),'g',(omega_list_ev
             - omega_pulse_ev)*1e6, real(ESD_transmitted_photons_vs_omega)-real
            (ESD_coherent_transmitted_photons_laser_vs_omega),'b');
228     xlim([min(omega_list_ev - omega_pulse_ev)*1e6 max(omega_list_ev -
            omega_pulse_ev)*1e6]);
229     xlabel('\omega-\omega_p [\mueV]');ylabel('Reflected field spectral
            density');
```

```
230     legend('WDF(\omega) - total transmitted flux', 'WDF(\omega) coherent
            flux', 'WDF(\omega) - incoherent flux')
231     text(omega_list_ev(1) - omega_pulse_ev,0.7*max(
            ESD_transmitted_photons_vs_omega),text_legend,'FontSize',9)
232
233     figure('Name',['Spectral density emitted photons vs omega - Nb_photons
            = ' num2str(Nb_photons_pulse)],'NumberTitle','off')
234     plot((omega_list_ev - omega_pulse_ev)*1e6, real(
            ESD_emitted_photons_vs_omega) ,'r',(omega_list_ev - omega_pulse_ev)
            *1e6, real(ESD_coherent_emitted_photons_laser_vs_omega),'g',(
            omega_list_ev - omega_pulse_ev)*1e6, real(
            ESD_emitted_photons_vs_omega)-real(
            ESD_coherent_emitted_photons_laser_vs_omega),'b');
235     xlim([min(omega_list_ev - omega_pulse_ev)*1e6 max(omega_list_ev -
            omega_pulse_ev)*1e6]);
236     xlabel('\omega-\omega_p [\mueV]');ylabel('Reflected field spectral
            density');
237     legend('WDF(\omega) - total emitted flux', 'WDF(\omega) coherent flux',
             'WDF(\omega) - incoherent flux')
238     text(omega_list_ev(1) - omega_pulse_ev,0.7*max(
            ESD_emitted_photons_vs_omega),text_legend,'FontSize',9)
239
240     figure('Name',['Spectral density diffracted photons vs omega - '
            text_legend_model ' - Nb_photons = ' num2str(Nb_photons_pulse)],'
            NumberTitle','off')
241     plot((omega_list_ev - omega_pulse_ev)*1e6, real(
            ESD_diffracted_photons_vs_omega) ,'r',(omega_list_ev -
            omega_pulse_ev)*1e6, real(
            ESD_coherent_diffracted_photons_laser_vs_omega),'g',(omega_list_ev
            - omega_pulse_ev)*1e6, real(ESD_diffracted_photons_vs_omega)-real(
            ESD_coherent_diffracted_photons_laser_vs_omega),'b');
242     xlim([min(omega_list_ev - omega_pulse_ev)*1e6 max(omega_list_ev -
            omega_pulse_ev)*1e6]);
243     xlabel('\omega-\omega_p [\mueV]');ylabel('Reflected field spectral
            density');
244     legend('WDF(\omega) - total diffracted flux', 'WDF(\omega) coherent
            flux', 'WDF(\omega) - incoherent flux')
245     text(omega_list_ev(1) - omega_pulse_ev,0.7*max(
            ESD_diffracted_photons_vs_omega),text_legend,'FontSize',9)
246 end
247
248 %% Checking normalization of ESDs
249 fprintf(['Relative error over normalization of spectral density - reflected
        field: ' num2str(abs(sum(ESD_reflected_photons_vs_omega)*
    omega_step_muev-Nb_reflected_photons)/Nb_reflected_photons) ' \n'])
```

```
250 fprintf(['Relative error over normalization of spectral density -
        transmitted field: ' num2str(abs(sum(ESD_transmitted_photons_vs_omega)*
        omega_step_muev-Nb_transmitted_photons)/Nb_transmitted_photons) ' \n'])
251 fprintf(['Relative error over normalization of spectral density - emitted
        field: ' num2str(abs(sum(ESD_emitted_photons_vs_omega)*omega_step_muev-
        Nb_emitted_photons)/Nb_emitted_photons) ' \n'])
252 fprintf(['Relative error over normalization of spectral density -
        diffracted field: ' num2str(abs(sum(ESD_diffracted_photons_vs_omega)*
        omega_step_muev-Nb_diffracted_photons)/Nb_diffracted_photons) ' \n'])
```

C.3.13 plot 2level g2 CW vs delay

```
1  switch model
2      case 'F'
3          text_legend_model='full model';
4      case 'A'
5          text_legend_model='adiabatic model';
6  end
7
8  % Parameters for the text displayed in figure legends
9  text_legend_QD=['g=', num2str(g_muev) ' muev    \gamma_{sp}=' num2str(
       gamma_sp_muev) 'muev     \gamma^*=' num2str(gamma_puredephasing_muev) '
       muev'];
10 text_legend_cav=['\kappa=' num2str(kappa_muev) 'muev   \eta_{top}=' num2str
       (eta_top)  '   C=' num2str(g^2/kappa/gamma_decoherence,3)];
11 text_legend_P_in=['P_{in}=' num2str(P_in_CW_pW) 'pW   n_c=' num2str(
       gamma_decoherence*gamma_sp/(4*g^2),2)];
12 text_legend_omega=['\omega_{laser}-\omega_d=' num2str(1e6*(omega_laser_ev-
       omega_d_ev),3) 'muev    \omega_d-\omega_c=' num2str(1e6*(omega_d_ev-
       omega_c_ev),3)  'muev'];
13 text_legend={text_legend_QD,text_legend_cav,text_legend_P_in,
       text_legend_omega};
14 % NB: C is the cooperativity and n_c the critical photon number, both
       depend only on the cavity-QED parameters.
15 % On the contrary, n_0 depends on the incoming power P_in since it is the
       calculated number of intracavity photons
16 % in the absence of QD (i.e. when g=0). When n_0 is much lower than n_c we
       are in the weak excitation limit.
17
18
19 %%%%%%%%%%%%%%%%%% Verifications %%%%%%%%%%%%%%%%%
20
21 % Verification on the conservation of photon flux
```

```
22 fprintf(['Verification - relative error on the conservation of photon flux
       : ' num2str(abs(total_flux_injected_photons-
       total_flux_reflected_photons-total_flux_transmitted_photons-
       total_flux_diffracted_photons-total_flux_emitted_photons)/
       total_flux_injected_photons) ' \n'])
23
24 if model=='F'
25     %Verification of the maximal photon number in the last Fock state
26     expect_occupation_last_Fock_state=expect(occupation_last_Fock_state,
           rhoss_CW); %
27     fprintf(['Verification - occupation of the last Fock state : ' num2str(
           max(abs(expect_occupation_last_Fock_state))) ' \n \n'])
28 end
29
30
31 % Basic verifications on the intensity correlations
32 % (one also has to verify that the g(2) curves are smooth - irregular
       curves probably indicate a wrong numerical convergence)
33 if min(g2_reflected_vs_delay)<0 || min(g2_transmitted_vs_delay)<0 || min(
       g2_emitted_vs_delay)<0
34     fprintf(' \n \n !!!!!!!!! Warning: non-physical negative values
           !!!!!!!!!! \n \n');
35 end
36
37 fprintf(['Verification: relative error on g2_reflected(0)=<b_out'' b_out''
       b_out b_out>/(<b_out'' b_out>^2) : ' num2str(abs(g2_reflected_vs_delay
       (1) - (expect(b_out'*b_out'*b_out*b_out,rhoss_CW)/(expect(b_out'*b_out,
       rhoss_CW)^2) )))  ' \n'])
38 fprintf(['Verification: relative error on g2_transmitted(0)=<c_out'' c_out'
       ' c_out c_out>/(<c_out'' c_out>^2) : ' num2str(abs(
       g2_transmitted_vs_delay(1) - (expect(c_out'*c_out'*c_out*c_out,rhoss_CW
       )/(expect(c_out'*c_out,rhoss_CW)^2) )))  ' \n'])
39 fprintf(['Verification: relative error on g2_reflected(0)=<e_out'' e_out''
       e_out e_out>/(<e_out'' e_out>^2) : ' num2str(abs(g2_emitted_vs_delay(1)
        - (expect(e_out'*e_out'*e_out*e_out,rhoss_CW)/(expect(e_out'*e_out,
       rhoss_CW)^2) )))  ' \n'])
40
41 fprintf(['Verification: relative error on g2(infty)=1 for reflected photons
        : ' num2str(abs(g2_reflected_vs_delay(nb_points_time_g2CW)-1)) ' \n'])
42 fprintf(['Verification: relative error on g2(infty)=1 for transmitted
       photons : ' num2str(abs(g2_transmitted_vs_delay(nb_points_time_g2CW)-1)
       ) ' \n'])
43 fprintf(['Verification: relative error on g2(infty)=1 for emitted photons :
        ' num2str(abs(g2_emitted_vs_delay(nb_points_time_g2CW)-1)) ' \n'])
44
45
```

```
46
47 %%%%%%%%%%%%%%%%%%  Plots of the normalized g(2)(tau)
       %%%%%%%%%%%%%%%%%%%%%%%
48 if ismember('R',plot_choice)
49
50     figure('Name',['g2CW ' text_legend_model ' - g2 vs tau  - reflected
           photons - Pin = ' num2str(P_in_CW_pW) ' pW - Laser photon energy =
           ' num2str(omega_laser_ev) ' eV' ],'NumberTitle','off')
51     plot(full_tau_list_g2CW,full_g2_reflected_vs_delay,'r');
52     xlabel('Delay \tau [ps]'); ylabel('g^{2} - reflected photons');
53     xlim([min(full_tau_list_g2CW) max(full_tau_list_g2CW)]);
54     text(full_tau_list_g2CW(ceil(nb_points_time_g2CW/20)),0.8*max(
           full_g2_reflected_vs_delay),text_legend,'FontSize',9)
55
56     if ismember('O',plot_choice)
57         figure('Name',['g2CW ' text_legend_model ' - occupation
               probabilities conditioned to a reflected photon detecton - Pin
               = ' num2str(P_in_CW_pW) ' pW - Laser photon energy = ' num2str(
               omega_laser_ev) ' eV' ],'NumberTitle','off')
58         plot(tau_list_g2CW,
               occupation_ground_vs_delay_after_reflected_photon_detection,'r'
               ,...
59              tau_list_g2CW,
                   occupation_excited_vs_delay_after_reflected_photon_detection
                   ,'b');
60         xlabel('Delay \tau [ps]'); ylabel('Conditionnal occupation
               probabilities');
61         xlim([min(tau_list_g2CW) max(tau_list_g2CW)]); ylim([0 1]);
62         text(tau_list_g2CW(ceil(nb_points_time_g2CW/10)),0.7,text_legend,'
               FontSize',9)
63         legend('Occupation of |g> conditioned on a reflected photon
               detection','Occupation of |e> conditioned on a reflected photon
                detection','Location','best');
64     end
65 end
66
67 if ismember('T',plot_choice)
68     figure('Name',['g2CW ' text_legend_model ' - g2 vs tau - transmitted
           photons - Pin = ' num2str(P_in_CW_pW) ' pW - Pulsation laser = '
           num2str(omega_laser_ev) ' eV' ],'NumberTitle','off')
69     plot(full_tau_list_g2CW,full_g2_transmitted_vs_delay,'b');
70     xlabel('Delay \tau [ps]'); ylabel('g^{2} - transmitted photons');
71     xlim([min(full_tau_list_g2CW) max(full_tau_list_g2CW)]);
72     text(full_tau_list_g2CW(ceil(nb_points_time_g2CW/20)),0.8*max(
           full_g2_transmitted_vs_delay),text_legend,'FontSize',9)
73
```

```
74      if ismember('O',plot_choice)
75          figure('Name',['g2CW ' text_legend_model ' - occupation
                probabilities conditioned to a transmitted photon detecton -
                Pin = ' num2str(P_in_CW_pW) ' pW - Laser photon energy = '
                num2str(omega_laser_ev) ' eV' ],'NumberTitle','off')
76          plot(tau_list_g2CW,
                occupation_ground_vs_delay_after_transmitted_photon_detection,'
                r',...
77                tau_list_g2CW,
                    occupation_excited_vs_delay_after_transmitted_photon_detection
                    ,'b');
78          xlabel('Delay \tau [ps]'); ylabel('Conditionnal occupation
                probabilities');
79          xlim([min(tau_list_g2CW) max(tau_list_g2CW)]); ylim([0 1]);
80          text(tau_list_g2CW(ceil(nb_points_time_g2CW/10)),0.7,text_legend,'
                FontSize',9)
81          legend('Occupation of |g> conditioned on a transmitted photon
                detection','Occupation of |e> conditioned on a transmitted
                photon detection','Location','best');
82      end
83  end
84  if ismember('E',plot_choice)
85
86      figure('Name',['g2CW ' text_legend_model ' - g2 vs tau - photons
            emitted outside the mode - Pin = ' num2str(P_in_CW_pW) ' pW -
            Pulsation laser = ' num2str(omega_laser_ev) ' eV' ],'NumberTitle','
            off')
87      plot(full_tau_list_g2CW,full_g2_emitted_vs_delay,'g');
88      xlabel('Delay \tau [ps]'); ylabel('g^{2} - photons emitted outside the
            mode');
89      xlim([min(full_tau_list_g2CW) max(full_tau_list_g2CW)]);
90      text(full_tau_list_g2CW(ceil(nb_points_time_g2CW/20)),0.3*max(
            full_g2_emitted_vs_delay),text_legend,'FontSize',9)
91
92      if ismember('O',plot_choice)
93          figure('Name',['g2CW ' text_legend_model ' - occupation
                probabilities conditioned to an emitted photon detecton - Pin =
                 ' num2str(P_in_CW_pW) ' pW - Laser photon energy = ' num2str(
                omega_laser_ev) ' eV' ],'NumberTitle','off')
94          plot(tau_list_g2CW,
                occupation_ground_vs_delay_after_emitted_photon_detection,'r',
                ...
95                tau_list_g2CW,
                    occupation_excited_vs_delay_after_emitted_photon_detection
                    ,'b');
```

```
96          xlabel('Delay \tau [ps]'); ylabel('Conditionnal occupation
                probabilities');
97          xlim([min(tau_list_g2CW) max(tau_list_g2CW)]); ylim([0 1]);
98          text(tau_list_g2CW(ceil(nb_points_time_g2CW/10)),0.7,text_legend,'
                FontSize',9)
99          legend('Occupation of |g> conditioned on an emitted photon
                detection','Occupation of |e> conditioned on an emitted photon
                detection','Location','best');
100     end
101 end
```

C.3.14 plot 2level g2 PR vs t1 t2

```
1  switch model
2      case 'F'
3          text_legend_model='full model';
4      case 'A'
5          text_legend_model='adiabatic model';
6  end
7
8  % Parameters for the text displayed in figure legends
9  text_legend_QD = ['g=', num2str(g_muev) ' muev    \gamma_{sp}=' num2str(
       gamma_sp_muev) 'muev     \gamma^*=' num2str(gamma_puredephasing_muev) '
       muev'];
10 text_legend_cav = ['\kappa=' num2str(kappa_muev) 'muev   \eta_{top}='
       num2str(eta_top)  '   C=' num2str(g^2/kappa/gamma_decoherence,3)];
11 text_legend_pulse = ['N_{in}=' num2str(Nb_photons_pulse) '       FWHM_{
       pulse}=' num2str(FWHM_pulse) 'ps'];
12 text_legend = {text_legend_QD,text_legend_cav,text_legend_pulse};
13 % NB: C is the cooperativity
14 %% g2 vs (t1,t2)
15 if ismember('G',plot_choice)
16     figure('Name',['g2 emitted vs (t1,t2) - ' text_legend_model],'
           NumberTitle','off')
17     surf(flip(t_list),t_list,flip(real(g2_emitted_vs_t1_t2),2))
18     xlabel('t_1 [ps]')
19     ylabel('t_2 [ps]')
20     title('emitted photon g^{(2)}(t_1,t_2)')
21     view(2)
22     colorbar
23
24     figure('Name',['g2 reflected vs (t1,t2) - ' text_legend_model],'
           NumberTitle','off')
25     surf(flip(t_list),t_list,flip(real(g2_reflected_vs_t1_t2),2))
26     xlabel('t_1 [ps]')
27     ylabel('t_2 [ps]')
```

```
28      title('reflected photon g^{(2)}(t_1,t_2)')
29      view(2)
30      colorbar
31
32      figure('Name',['g2 transmitted vs (t1,t2) - ' text_legend_model],'
            NumberTitle','off')
33      surf(flip(t_list),t_list,flip(real(g2_transmitted_vs_t1_t2),2))
34      xlabel('t_1 [ps]')
35      ylabel('t_2 [ps]')
36      title('transmitted photon g^{(2)}(t_1,t_2)')
37      view(2)
38      colorbar
39  end
40  %% Photon coincidences
41  if ismember('C',plot_choice)
42      % photon coincidences uncorrelated
43      figure('Name',['Uncorrelated coincidences emitted photons vs (t1,t2) -
            ' text_legend_model],'NumberTitle','off')
44      surf(flip(t_list),t_list,flip(real(
            emitted_photon_coincidences_vs_t1_vs_t2_uncorrelated_pulses),2))
45      xlabel('t_1 [ps]')
46      ylabel('t_2 [ps]')
47      title('<e_{out}''(t_1) e_{out}(t_1)> <e_{out}''(t_2) e_{out}(t_2)>')
48      view(2)
49      colorbar
50
51      figure('Name',['Uncorrelated coincidences reflected photons vs (t1,t2)
            - ' text_legend_model],'NumberTitle','off')
52      surf(flip(t_list),t_list,flip(real(
            reflected_photon_coincidences_vs_t1_vs_t2_uncorrelated_pulses),2))
53      xlabel('t_1 [ps]')
54      ylabel('t_2 [ps]')
55      title('<b_{out}''(t_1) b_{out}(t_1)> <b_{out}''(t_2) b_{out}(t_2)>')
56      view(2)
57      colorbar
58
59      figure('Name',['Uncorrelated coincidences transmitted photons vs (t1,t2
            ) - ' text_legend_model],'NumberTitle','off')
60      surf(flip(t_list),t_list,flip(real(
            transmitted_photon_coincidences_vs_t1_vs_t2_uncorrelated_pulses),2)
            )
61      xlabel('t_1 [ps]')
62      ylabel('t_2 [ps]')
63      title('<c_{out}''(t_1) c_{out}(t_1)> <c_{out}''(t_2) c_{out}(t_2)>')
64      colorbar
65
```

```
66      % photon coincidences correlated
67      figure('Name',['Correlated coincidences emitted photons vs (t1,t2) - '
            text_legend_model],'NumberTitle','off')
68      surf(flip(t_list),t_list,flip(real(
            emitted_photon_coincidences_vs_t1_vs_t2),2))
69      xlabel('t_1 [ps]')
70      ylabel('t_2 [ps]')
71      title('<e_{out}''(t_1) e_{out}''(t_2) e_{out}(t_2) e_{out}(t_1)>')
72      view(2)
73      colorbar
74
75      figure('Name',['Correlated coincidences reflected photons vs (t1,t2) -
            ' text_legend_model],'NumberTitle','off')
76      surf(flip(t_list),t_list,flip(real(
            reflected_photon_coincidences_vs_t1_vs_t2),2))
77      xlabel('t_1 [ps]')
78      ylabel('t_2 [ps]')
79      title('<b_{out}''(t_1) b_{out}''(t_2) b_{out}(t_2) b_{out}(t_1)>')
80      view(2)
81      colorbar
82
83      figure('Name',['Correlated coincidences transmitted photons vs (t1,t2)
            - ' text_legend_model],'NumberTitle','off')
84      surf(flip(t_list),t_list,flip(real(
            transmitted_photon_coincidences_vs_t1_vs_t2),2))
85      xlabel('t_1 [ps]')
86      ylabel('t_2 [ps]')
87      title('<c_{out}''(t_1) c_{out}''(t_2) c_{out}(t_2) c_{out}(t_1)>')
88      view(2)
89      colorbar
90  end
91  %% conditioned occupations
92  if ismember('O',plot_choice)
93      figure('Name',['Occupation ground after reflected photon vs (t1,t2) - '
             text_legend_model],'NumberTitle','off')
94      surf(flip(t_list),t_list,flip(real(
            occupation_ground_vs_t1_vs_t2_after_click_b_out_at_t1),2))
95      xlabel('t_1 [ps]')
96      ylabel('t_2 [ps]')
97      title('Occupation ground after reflected photon')
98      view(2)
99      colorbar
100
101 %     figure('Name',['Occupation excited after reflected photon vs (t1,t2)
        - ' text_legend_model],'NumberTitle','off')
```

```
102 %     surf(flip(t_list),t_list,flip(real(
    occupation_excited_vs_t1_vs_t2_after_click_b_out_at_t1),2))
103 %     xlabel('t_1 [ps]')
104 %     ylabel('t_2 [ps]')
105 %     title('Occupation excited after reflected photon')
106 %     view(2)
107 %     colorbar
108
109       figure('Name',['Occupation ground after transmitted photon vs (t1,t2) -
              ' text_legend_model],'NumberTitle','off')
110       surf(flip(t_list),t_list,flip(real(
             occupation_ground_vs_t1_vs_t2_after_click_c_out_at_t1),2))
111       xlabel('t_1 [ps]')
112       ylabel('t_2 [ps]')
113       title('Occupation ground after transmitted photon')
114       view(2)
115       colorbar
116
117 %     figure('Name',['Occupation excited after transmitted photon vs (t1,t2
    ) - ' text_legend_model],'NumberTitle','off')
118 %     surf(flip(t_list),t_list,flip(real(
    occupation_excited_vs_t1_vs_t2_after_click_c_out_at_t1),2))
119 %     xlabel('t_1 [ps]')
120 %     ylabel('t_2 [ps]')
121 %     title('Occupation excited after transmitted photon')
122 %     view(2)
123 %     colorbar
124
125       figure('Name',['Occupation ground after emitted photon vs (t1,t2) - '
             text_legend_model],'NumberTitle','off')
126       surf(flip(t_list),t_list,flip(real(
             occupation_ground_vs_t1_vs_t2_after_click_e_out_at_t1),2))
127       xlabel('t_1 [ps]')
128       ylabel('t_2 [ps]')
129       title('Occupation ground after emitted photon')
130       view(2)
131       colorbar
132
133 %     figure('Name',['Occupation excited after emitted photon vs (t1,t2) -
    ' text_legend_model],'NumberTitle','off')
134 %     surf(flip(t_list),t_list,flip(real(
    occupation_excited_vs_t1_vs_t2_after_click_e_out_at_t1),2))
135 %     xlabel('t_1 [ps]')
136 %     ylabel('t_2 [ps]')
137 %     title('Occupation excited after emitted photon')
138 %     view(2)
```

```
%     colorbar
end
%% photon fluxes
if ismember('F',plot_choice)
    figure('Name',['Photons flux vs t1 - ' text_legend_model],'NumberTitle'
        ,'off')
    plot(t_list,flux_injected_photons_vs_time,'k','Displayname','incoming')
    hold on
    plot(t_list,flux_reflected_photons_vs_time,'r','Displayname','reflected
        ')
    plot(t_list,flux_transmitted_photons_vs_time,'b','Displayname','
        transmitted')
    plot(t_list,flux_emitted_photons_vs_time,'g','Displayname','emitted')
    hold off
    title(['Photon flux - Nb =' num2str(Nb_photons_pulse)])
    xlabel('t_1 [ps]')
    ylabel('[1/ps]')
    legend
    text(t_list(ceil(nb_points_time*0.05)),0.95*max(
        flux_injected_photons_vs_time),text_legend,'FontSize',9)

    figure('Name',['Normalized g2 vs delay, photon emission - '
        text_legend_model],'NumberTitle','off')
    plot(full_tau_list,[flip(normalized_g2_vs_delay_emitted(2:end))
        normalized_g2_vs_delay_emitted],'g','Displayname', 'correlated')
    hold on
    plot(full_tau_list,[flip(normalized_g2_vs_delay_uncorrelated_emitted(2:
        end)) normalized_g2_vs_delay_uncorrelated_emitted],'--g','
        Displayname', 'uncorrelated')
    title('Histogram < g^{(2)}(\tau) > emitted photons')
    xlabel('\tau [ps]')
    ylabel('counts')
    legend('Location','northeast');
    text(full_tau_list(ceil(nb_points_time*0.05)),max(
        normalized_g2_vs_delay_uncorrelated_emitted)*0.9,text_legend,'
        FontSize',9)

    figure('Name',['Normalized g2 vs delay, photon reflection - '
        text_legend_model],'NumberTitle','off')
    plot(full_tau_list,[flip(normalized_g2_vs_delay_reflected(2:end))
        normalized_g2_vs_delay_reflected],'r','Displayname', 'correlated')
    hold on
    plot(full_tau_list,[flip(normalized_g2_vs_delay_uncorrelated_reflected
        (2:end)) normalized_g2_vs_delay_uncorrelated_reflected],'--r','
        Displayname', 'uncorrelated')
    title('Histogram < g^{(2)}(\tau) > reflected photons')
```

```
171     xlabel('\tau [ps]')
172     ylabel('counts')
173     legend('Location','northeast');
174     text(full_tau_list(ceil(nb_points_time*0.05)),max(
            normalized_g2_vs_delay_uncorrelated_reflected)*0.95,text_legend,'
            FontSize',9)
175
176     figure('Name',['Normalized g2 vs delay, photon transmission - '
            text_legend_model],'NumberTitle','off')
177     plot(full_tau_list,[flip(normalized_g2_vs_delay_transmitted(2:end))
            normalized_g2_vs_delay_transmitted],'b','Displayname', 'correlated'
            )
178     hold on
179     plot(full_tau_list,[flip(
            normalized_g2_vs_delay_uncorrelated_transmitted(2:end))
            normalized_g2_vs_delay_uncorrelated_transmitted],'--b','Displayname
            ', 'uncorrelated')
180     title('Histogram < g^{(2)}(\tau) > transmitted photons')
181     xlabel('\tau [ps]')
182     ylabel('counts')
183     legend('Location','northeast');
184     text(full_tau_list(ceil(nb_points_time*0.05)),max(
            normalized_g2_vs_delay_uncorrelated_transmitted)*0.95,text_legend,'
            FontSize',9)
185 end
186 %%
187
188 % Displaying the mean g2(0), i.e. the area of the correlated HBT peak
189 fprintf(['PR - ' text_legend_model ' ' ': Area of mean g2(0) from reflected
        photons: ' num2str(mean_g2_zero_delay_peak_reflected_photons) '\n'])
190 fprintf(['PR - ' text_legend_model ' ' ': Area of mean g2(0) from
        transmitted photons: ' num2str(
        mean_g2_zero_delay_peak_transmitted_photons) '\n'])
191 fprintf(['PR - ' text_legend_model ' ' ': Area of mean g2(0) from emitted
        photons: ' num2str(mean_g2_zero_delay_peak_emitted_photons) '\n'])
192
193
194 % Verification of the area of the uncorrelated HBT peaks, that should be
195 % normalized to unity
196 fprintf(['PR - ' text_legend_model ' ' ': Relative error on the area of
        mean g2 for uncorrelated peaks, for reflected photons: ' num2str((-1 +
        mean_g2_uncorrelated_peaks_reflected_photons)*100 ) '%% \n'])
197 fprintf(['PR - ' text_legend_model ' ' ': Relative error on the area of
        mean g2 for uncorrelated peaks, for transmitted photons: ' num2str((-1
        + mean_g2_uncorrelated_peaks_transmitted_photons)*100 ) '%% \n'])
```

```
198 fprintf(['PR — ' text_legend_model ' ' ': Relative error on the area of
        mean g2 for uncorrelated peaks, for emitted photons: ' num2str((-1 +
        mean_g2_uncorrelated_peaks_emitted_photons)*100 ) '%% \n'])
199
200
201 % Basic verifications on the intensity correlations
202 if min(normalized_g2_vs_delay_uncorrelated_transmitted)<0 || min(
        normalized_g2_vs_delay_reflected)<0 || min(
        normalized_g2_vs_delay_emitted)<0
203     fprintf(' \n \n !!!!!!!!! Warning: non-physical negative values in g2
            vs delay !!!!!!!!!! \n \n');
204 end
```

Bibliography

[1] Jonathan P. Dowling and Gerard J. Milburn. Quantum technology: the second quantum revolution. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 361(1809):1655–1674, jun 2003. doi: 10.1098/rsta.2003.1227.

[2] H. J. Kimble. The quantum internet. *Nature*, 453(7198):1023–1030, jun 2008. doi: 10.1038/nature07127.

[3] Hatim Azzouz, Sander N. Dorenbos, Daniel De Vries, Esteban Bermúdez Ureña, and Valery Zwiller. Efficient single particle detection with a superconducting nanowire. *AIP Advances*, 2(3):032124, sep 2012. doi: 10.1063/1.4740074.

[4] Peter Michler, editor. *Quantum Dots for Quantum Information Technologies*. Springer-Verlag GmbH, 2017. ISBN 978-3-319-56378-7.

[5] David Press, Thaddeus D. Ladd, Bingyang Zhang, and Yoshihisa Yamamoto. Complete quantum control of a single quantum dot spin using ultrafast optical pulses. *Nature*, 456(7219):218–221, nov 2008. doi: 10.1038/nature07530.

[6] A. Dousse, L. Lanco, J. Suffczyński, E. Semenova, A. Miard, A. Lemaître, I. Sagnes, C. Roblin, J. Bloch, and P. Senellart. Controlled light-matter coupling for a single quantum dot embedded in a pillar microcavity using far-field optical lithography. *Physical Review Letters*, 101(26), dec 2008. doi: 10.1103/physrevlett.101.267404.

[7] A. Dousse, J. Suffczyński, R. Braive, A. Miard, A. Lemaître, I. Sagnes, L. Lanco, J. Bloch, P. Voisin, and P. Senellart. Scalable implementation of strongly coupled cavity-quantum dot devices. *Applied Physics Letters*, 94(12):121102, mar 2009. doi: 10.1063/1.3100781.

[8] Adrien Dousse, Jan Suffczyński, Alexios Beveratos, Olivier Krebs, Aristide Lemaître, Isabelle Sagnes, Jacqueline Bloch, Paul Voisin, and Pascale Senellart. Ultrabright source of entangled photon pairs. *Nature*, 466(7303):217–220, jul 2010. doi: 10.1038/nature09148.

[9] O. Gazzano, S. Michaelis de Vasconcellos, C. Arnold, A. Nowak, E. Galopin, I. Sagnes, L. Lanco, A. Lemaître, and P. Senellart. Bright solid-state sources of indistinguishable single photons. *Nature Communications*, 4(1), feb 2013. doi: 10.1038/ncomms2434.

[10] N. Somaschi, V. Giesz, L. De Santis, J. C. Loredo, M. P. Almeida, G. Hornecker, S. L. Portalupi, T. Grange, C. Antón, J. Demory, C. Gómez, I. Sagnes, N. D. Lanzillotti-Kimura, A. Lemaítre, A. Auffeves, A. G. White, L. Lanco, and P. Senellart. Near-optimal single-photon sources in the solid state. *Nature Photonics*, 10(5):340–345, mar 2016. doi: 10.1038/nphoton.2016.23.

[11] Hélène Ollivier, Ilse Maillette de Buy Wenniger, Sarah Thomas, Stephen C. Wein, Abdelmounaim Harouri, Guillaume Coppola, Paul Hilaire, Clément Millet, Aristide Lemaître, Isabelle Sagnes, Olivier Krebs, Loïc Lanco, Juan C. Loredo, Carlos Antón, Niccolo Somaschi, and Pascale Senellart. Reproducibility of high-performance quantum dot single-photon sources. *ACS Photonics*, 7(4):1050–1059, mar 2020. doi: 10.1021/acsphotonics.9b01805.

[12] Jürgen T. Stockburger. Stochastic and numerical approaches to dissipative quantum dynamics: path integrals and beyond. *physica status solidi (b)*, 237(1):146–158, may 2003. doi: 10.1002/pssb.200301796.

[13] Pierre Meystre. *Elements of quantum optics*. Springer, Berlin New York, 2007. ISBN 9783540742111.

[14] Howard Carmichael. *An open systems approach to quantum optics : lectures presented at the Université libre de Bruxelles, October 28 to November 4, 1991*. Springer-Verlag, Berlin New York, 1993. ISBN 9783540566342.

[15] Rémi Azouit. *Adiabatic elimination for open quantum systems*. PhD thesis, PSL Research University, 2017.

[16] C. C. Gerry. *Introductory quantum optics*. Cambridge University Press, Cambridge, UK New York, 2005. ISBN 9780511791239.

[17] Mark Fox. *Quantum Optics*. Oxford University Press, 2006. ISBN 0198566735.

[18] Peter Michler, editor. *Single Semiconductor Quantum Dots*. Springer-Verlag GmbH, 2009. ISBN 3540874461.

[19] Lorenzo De Santis. *Single photon generation and manipulation with semiconductor quantum dot devices*. PhD thesis, Université Paris-Saclay, 03 2018.

[20] Howard Carmichael. *Statistical Methods in Quantum Optics 1 : Master Equations and Fokker-Planck Equations*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999. ISBN 9783662038758.

[21] C. W. Gardiner and M. J. Collett. Input and output in damped quantum systems: Quantum stochastic differential equations and the master equation. *Physical Review A*, 31(6):3761–3774, jun 1985. doi: 10.1103/physreva.31.3761.

[22] Loïc Lanco. *Enhancing photon-photon and spin-photon interactionswith quantum-dot based light-matter interfaces*. Habilitation à diriger des recherches, University Paris Diderot - Paris VII, 2017.

[23] Sze Meng Tan. A quantum optics toolbox for matlab 5, 1999.

[24] Vivien Loo. *Excitation résonante et non-linéarité à faible nombre de photons d'une boîte quantique en microcavité*. PhD thesis, Paris 7, 2012.

[25] D.A. Steck. Quantum and atom optics. 2007.

[26] Baldassare Bartolo. *Spectroscopy of systems with spatially confined structures : Proceedings of the NATO Advanced Study Institute on Spectroscopy of Systems with Spatially Confined Structures Erice, Sicily, Italy 15-30 June 2001*. Springer, Dordrecht, 2003. ISBN 9789401002875.

[27] Ilan Shlesinger, Pascale Senellart, Loic Lanco, and Jean-Jacques Greffet. Time-frequency encoded single-photon sources and broadband quantum memories based on a tunable one-dimensional atom. 05 2019.

[28] Howard J. Carmichael. *Statistical Methods in Quantum Optics 2*. Springer-Verlag GmbH, 2009. ISBN 9783540713203.

[29] Carlos Antón, Paul Hilaire, Christian A. Kessler, Justin Demory, Carmen Gómez, Aristide Lemaître, Isabelle Sagnes, Norberto Daniel Lanzillotti-Kimura, Olivier Krebs, Niccolo Somaschi, Pascale Senellart, and Loïc Lanco. Tomography of the optical polarization rotation induced by a single quantum dot in a cavity. *Optica*, 4(11):1326, oct 2017. doi: 10.1364/optica.4.001326.

[30] Chris Gustin, Ross Manson, and Stephen Hughes. Spectral asymmetries in the resonance fluorescence of two-level systems under pulsed excitation. *Optics Letters*, 43(4):779, feb 2018. doi: 10.1364/ol.43.000779.

[31] T. Nakanishi, K. Yamane, and M. Kitano. Absorption-free optical control of spin systems: The quantum zeno effect in optical pumping. *Physical Review A*, 65(1), dec 2001. doi: 10.1103/physreva.65.013404.

[32] S. Hughes. Enhanced single-photon emission from quantum dots in photonic crystal waveguides and nanocavities. *Optics Letters*, 29(22):2659, nov 2004. doi: 10.1364/ol.29.002659.

[33] J.R. Johansson, P.D. Nation, and Franco Nori. QuTiP 2: A python framework for the dynamics of open quantum systems. *Computer Physics Communications*, 184(4):1234–1240, apr 2013. doi: 10.1016/j.cpc.2012.11.019.

[34] A. Ulhaq, S. Weiler, S. M. Ulrich, R. Roßbach, M. Jetter, and P. Michler. Cascaded single-photon emission from the mollow triplet sidebands of a quantum dot. *Nature Photonics*, 6(4):238–242, feb 2012. doi: 10.1038/nphoton.2012.23.