ISSN 0281-2762

Coherent Light-Matter Interaction
Pulse propagation in absorbing materials

Emad Hubainy
Master Thesis
LRAP 355, LTH/LU, 2006

Contents

Chapter 1

Introduction:

Since the middle of the twentieth century when information by using computers, was added as a physical resource, many investigations have been performed to improve these computers. In the time of today people talk about quantum computers and the investigations which are done to produce such computers. Such work is not carried out in order to replace the classic computer, instead the quantum computer offers new and exciting prospects.

The most fundamental entity in information science is the bit which is a one of 8 parts of a byte. It is associated to the hardware of computers like hard disks or memories. A bit carries two possible values "0" or "1" which may represent false or true, unfound or found... etc. The quantum analogy of a bit, the qubit is also a two-state system, but the difference is that the qubit can be in a superposition of both states. Mathematically and to explain how a qubit can be in these states we start from the general state which can be written as:

$$|Q\rangle = \alpha|0\rangle + \beta|1\rangle \text{ Where } |\alpha|^2 + |\beta|^2 = 1 \qquad (1.1)$$

The probability of carrying "0" and "1" can be written respectively

$$p(\text{"0"}) = |\alpha|^2,\ p(\text{"1"}) = |\beta|^2 \qquad (1.2)$$

From equations (1.1) & (1.2) and when $|\alpha| = |\beta|$ we find that the probability to find the qubit in the state "0" or "1" is the same. Equation (1.1) describes a *coherent superposition* [1]. To get a better understanding of differences between classical and quantum computers, we assume that we have a classical 3-bit register and a quantum 3-qubit register. A classical 3-bit register can be in one of eight configurations which are 000,010,001..... 111 while a quantum 3-qubit register can be in a superposition of all these configurations i.e. a single quantum 3-qubit register can store all these eight numbers simultuneously. Mathematically the capacity of a quantum register is 2^L where L is the number of qubits [2]. That means the capacity of a quantum register for storing information increases exponentially with the

number of bits. Using light is the best way to send quantum information over large distances (to do this it is necessary to transfer the information between light and matter). Some quantum information processing may be performed directly on the quantum qubits using non-linear or linear quantum optical processes. But before we go through the theory behind coherent light-matter interaction, let us see what can make quantum computers so different from classical computers by taking a very simple example.

We assume that we have an unsorted data base with a million names of stars ($N=10^6$) with some facts in a directory stored in the computer's memory and we are looking for a specific star. By using a classical computer, it requires N/2 memory accesses to find the given name of the star. That because in the process of finding the star of interest, it is necessary to go through all entries one by one. But a quantum computer performs about 1000, that is $\sqrt{N}$, appropriately chosen operations, which will be sufficient for obtaining the name of the star.

Chapter 2

Coherent Transient Theory

In this chapter and starting from Schrödinger's equation of motion, the optical Bloch equations describing light-matter interaction are derived. Then the Maxwell-Bloch equations, where also the electromagnetic field propagation effects are included, are derived in such form that they can be applicable to handle highly absorbing media. The Maxwell-Bloch equations are written in general form so that they can be used in several cases when temporally shaped Gaussian or square pulses are applied or in cases where another type of pulses are applied like frequency-chirped pulses or secant hyperbolic pulses. Finally the theory of secant hyperbolic pulses is discussed as an example of the application of the Maxwell-Bloch equations.

2.1.1 Derivation of Bloch equations

Starting from perturbation theory which tells us that when a perturbation is applied, the Hamiltonian can be expressed as

$$H = H_0 + H' \tag{2.1.1}$$

where H_0 and H' represent unperturbed and perturbed Hamiltonians respectively. Quantum mechanics tells that when the optical electric field is applied on an atomic system, the action of the Hamiltonian can be written as:

$$H_0|a\rangle = E_a|a\rangle \tag{2.1.2}$$

$$H_0|b\rangle = E_b|b\rangle \tag{2.1.3}$$

$$H' = -\overline{M}\cdot\overline{E}(t) \tag{2.1.4}$$

Here $E_{a,b}$ are the energy of ground and excited states respectively, $\overline{M} = -e\overline{r}$ is the dipole operator and $\overline{E}(t)$ is the optical electric field.

The localized quantum state of a two-level system can be described by its density matrix ρ and by using the Schrödinger equation, which gives

$$i\hbar\frac{d\rho}{dt} = [\rho, H] \tag{2.1.5}$$

The density matrix is

$$\rho = \begin{bmatrix} \rho_{aa} & \rho_{ab} \\ \rho^*_{ab} & \rho_{bb} \end{bmatrix} \tag{2.1.6}$$

where $\rho_{aa,bb}$ represent the ground and excited state respectively and $\rho^*_{ab,ab}$ represent the coherence or coupling between states.

By substituting Eq. (2.1.1) into (2.1.5) we obtain

$$i\hbar\frac{d\rho}{dt} = [\rho, (H_0 + H')] \tag{2.1.7}$$

$$i\hbar\frac{d\rho}{dt} = [\rho, H_0] + [\rho, H'] \tag{2.1.8}$$

Now we re-write Eq. (2.1.8)

$$i\hbar\frac{d\rho}{dt} = H_0\rho - \rho H_0 + H'\rho - \rho H' \tag{2.1.9}$$

Our aim of the derivation of the Bloch equation is to manipulate Schrödinger's equation into a set of three numerically partial differential equations which describe the time evolution of the in-phase and in-quadrature polarization components and the population inversion for a two-

level system in response to coherent excitation. Now we derive the equation of motion for ρ_{aa}. In this case Eq. (2.1.9) can be written as

$$\langle a|i\hbar\frac{d\rho}{dt}|a\rangle = \langle a|H_0\rho - \rho H_0 + H'\rho - \rho H'|a\rangle \tag{2.1.10}$$

By taking the density matrix in Eq. (2.1.6) every element for each, Eq. (2.1.10) becomes

$$i\hbar\frac{d\rho_{aa}}{dt} = E_a\rho_{aa} - \rho_{aa}E_a + \langle a|H'\rho|a\rangle - \langle a|\rho H'|a\rangle \tag{2.1.11}$$

The last two terms on the right side of Eq. (2.1.11) and by using unit operator, can reduce to

$$\langle a|H'\rho|a\rangle = \langle a|H'\{|a\rangle\langle a| + |b\rangle\langle b|\}\rho|a\rangle \tag{2.1.12}$$

Now we move the Hamiltonian inside the brackets. That gives

$$\langle a|H'\rho|a\rangle = \langle a|H'|a\rangle\langle a|\rho|a\rangle + \langle a|H'|b\rangle\langle b|\rho|a\rangle \tag{2.1.13}$$

$\langle a|H'|a\rangle = 0$ due to parity (selection rules tell that certain operators connect certain states if the states have same parity and certain operators do not connect other states if the states have opposite parity).

From Eq. (2.1.4) & Eq.(2.1.6) we obtain

$$\langle a|H'\rho|a\rangle = -M_{ab}E(t)\rho^*_{ab} \tag{2.1.14}$$

In a similar way we perform $\langle a|\rho H'|a\rangle$ to obtain

$$\langle a|\rho H'|a\rangle = -\rho_{ab}(E(t)M^*_{ab}) \tag{2.1.15}$$

By substituting Eq (2.1.14) & Eq. (2.1.15) into Eq.(2.1.11) we obtain

$$i\hbar\frac{d\rho_{aa}}{dt} = -E(t)M_{ab}\rho^*_{ab} + E(t)\rho_{ab}M^*_{an} \tag{2.1.16}$$

In the next steps we derivate the transition from $\langle a|$ to $|b\rangle$ an by using Eq. (2.1.9). We obtain

$$\langle a|i\hbar\frac{d\rho}{dt}|b\rangle = \langle a|H_0\rho - \rho H_0 + H'\rho - \rho H'|b\rangle \tag{2.1.17}$$

This becomes

$$i\hbar\frac{d\rho_{ab}}{dt} = E_a\rho_{ab} - \rho_{ab}E_b + \langle a|H'\rho|b\rangle - \langle a|\rho H'|b\rangle \tag{2.1.18}$$

The last two terms in the right side of Eq. (2.1.18) can be handled in the same steps we used in Eqs. (2.1.12-2.1.15) to obtain

$$i\hbar\frac{d\rho_{ab}}{dt} = (E_a - E_b)\rho_{ab} - E(t)M_{ab}(\rho_{aa} - \rho_{bb}) \tag{2.1.19}$$

Let

$$E(t) = \varepsilon\cos(\omega t), \tag{2.1.20}$$

$$E_b - E_a = \hbar\omega_0, \tag{2.1.21}$$

$$\rho_{ab} = \rho'_{ab}e^{i\omega t}, \tag{2.1.22}$$

$$\rho_{ba} = \rho'^{*}_{ab}e^{-i\omega t} \tag{2.1.23}$$

The substituting of Eqs. (2.1.20), (2.1.22) & (2.1.23) into (2.1.16) gives

$$i\hbar\frac{d\rho_{aa}}{dt} = -M_{ab}\rho'_{ba}e^{-i\omega t}\varepsilon\cos(\omega t) + M_{ba}\rho'_{ab}e^{i\omega t}\varepsilon\cos(\omega t) \tag{2.1.24}$$

Using the definition of the function of cosine, we obtain

$$i\hbar\frac{d\rho_{aa}}{dt} = \frac{1}{2}(-M_{ab}\rho'_{ba}\varepsilon e^{-i2\omega t} + M_{ba}\rho'_{ab}\varepsilon e^{i2\omega t} - M_{ab}\rho'_{ba}\varepsilon + M_{ba}\rho'_{ab}\varepsilon) \tag{2.1.25}$$

The integration over any time interval > $\frac{1}{2\omega}$ will tend to average the two left root terms on the right hand side to zero; therefore we can use the so called rotating wave approximation to obtain.

$$i\hbar\frac{d\rho_{aa}}{dt} = -\frac{1}{2}M_{ab}\rho_{ba}^{'}\varepsilon + \frac{1}{2}M_{ba}\rho_{ab}^{'}\varepsilon \tag{2.1.26}$$

By considering that the dipole operator is real such that $M_{ab} = M_{ba}$ and by expressing the complex density matrix elements in terms of their real and imaginary parts ($\rho_{ab}^{'} = \mathrm{Re}\,\rho_{ab}^{'} + \mathrm{Im}\,\rho_{ab}^{'}$, $\rho_{ba}^{'} = \mathrm{Re}\,\rho_{ab}^{'} - \mathrm{Im}\,\rho_{ab}^{'}$) we can re-write Eq. (2.1.26)

$$\hbar\frac{d\rho_{aa}}{dt} = \varepsilon(t)M_{ab}\,\mathrm{Im}\,\rho_{ab}^{'} \tag{2.1.27}$$

The equation for off-diagonal elements of the density matrix can be reduced by the following steps. If we substitute Eq. (2.1.20) – (2.1.23) into Eq. (2.1.19), we obtain

$$i\hbar\frac{d\rho_{ab}^{'}e^{i\omega t}}{dt} = -\rho_{ab}^{'}\hbar\omega_0 e^{i\omega t} - \varepsilon(t)\frac{e^{i\omega t}+e^{-i\omega t}}{2}M_{ab}(\rho_{bb}-\rho_{aa}) \tag{2.1.28}$$

By implementing the chain rule we obtain

$$i\hbar\frac{d\rho_{ab}^{'}}{dt}e^{i\omega t} - \hbar\omega\rho_{ab}^{'}e^{i\omega t} = -\rho_{ab}^{'}\hbar\omega_0 e^{i\omega t} - \varepsilon(t)\frac{e^{i\omega t}+e^{-i\omega t}}{2}M_{ab}(\rho_{bb}-\rho_{aa}) \tag{2.1.29}$$

Again we use the rotating wave approximation to simplify (2.1.29) and that gives

$$\frac{d\rho_{ab}^{'}}{dt} = i(\omega_0-\omega)\rho_{ab}^{'} + i\frac{\varepsilon(t)}{2\hbar}M_{ab}(\rho_{bb}-\rho_{aa}) \tag{2.1.30}$$

By expressing $\rho_{ab}^{'}$ as a complex quantity and balancing then Eq. (2.1.30) in real and imaginary parts we have

$$\frac{d\,\mathrm{Re}\,\rho_{ab}^{'}}{dt} = -(\omega_0-\omega)\,\mathrm{Im}\,\rho_{ab}^{'} \tag{2.1.31}$$

$$i\frac{d\,\mathrm{Im}\,\rho_{ab}^{'}}{dt} = i(\omega_0-\omega)\,\mathrm{Re}\,\rho_{ab}^{'} + i\frac{\varepsilon(t)}{2\hbar}M_{ab}(\rho_{bb}-\rho_{aa}) \tag{2.1.32}$$

Now we let

$$r_1 = 2\,\mathrm{Re}\,\rho_{ab}^{'} \tag{2.1.33}$$

$$r_2 = 2\,\mathrm{Im}\,\rho'_{ab} \tag{2.1.34}$$

$$r_3 = 1 - 2\rho_{aa} \tag{2.1.35}$$

$$\Omega(t) = \frac{\varepsilon(t) M_{ab}}{\hbar} \tag{2.1.36}$$

By substituting Eqs. (2.1.33-36) into Eqs. (21..27), (2.1.31) & (2.1.32) and by letting $\Delta = (\omega - \omega_0)$ we obtain

$$\frac{\partial r_1}{\partial t} = \Delta r_2 \tag{2.1.37}$$

$$\frac{\partial r_2}{\partial t} = \Delta r_1 + \Omega(t) r_3 \tag{2.1.38}$$

$$\frac{\partial r_3}{\partial t} = -\Omega(t) r_2 \tag{2.1.39}$$

The last three equations (2.1.37-39) represent the time development of the in-phase and in-quadrature components and the population inversion respectively and r_1, r_2 and r_3 can be defined as the components of a vector (the so called Bloch vector) in a three dimensional space spanned by Bloch vectors.

Now we include the relaxation and homogenous life time to upper equations to obtain

$$\frac{\partial r_1}{\partial t} = \Delta r_2 - \frac{r_1}{T_2} \tag{2.1.40}$$

$$\frac{\partial r_2}{\partial t} = \Delta r_1 + \Omega(t) r_3 - \frac{r_2}{T_2} \tag{2.1.41}$$

$$\frac{\partial r_3}{\partial t} = -\Omega(t) r_2 - \frac{r_3 + 1}{T_1} \tag{2.1.42}$$

$\Omega(t)$ is denoted the Rabi frequency, T_2 is the homogeneous life time that is due to interactions which disrupt the coherence through irreversible phase disturbances, while T_1 is

the relaxation time of the upper state through spontaneous emission and Δ is the frequency detuning of the electro-magnetic field from the two-level atom resonance frequency.

By applying one Rabi cycle, the atom rotates through an 2π in three-dimensional and considered to be back it its original state. The Bloch equations describe the effects of coherent light on a system of inhomogeneously broadened two level atoms while the Bloch vector describes the polarization state and population of the system.

Yet we have not obtained the propagation effects of the optical field of the input pulses and the various echo signals that are generated by the system. In the next section we will include the Maxwellian wave equation in a numerical integration of the Bloch equation to understand the interplay between the atomic dipoles and the optical field.

2.2 Derivation of Maxwell's wave equation

As stated in the previous section, we need to include Maxwell's wave equation in a numerical integration of Bloch equations to obtain explicit information about the propagation effects. To achieve this we start with a source free medium and a linearly polarized electromagnetic field. The one-dimensional equation becomes

$$\frac{d^2}{dz^2}E(z,t) - \varepsilon_0\mu_0\frac{d^2}{dt^2}E(z,t) = \mu_0\frac{d^2}{dt^2}\overline{P} \tag{2.2.1}$$

where the electric field is

$$E(z,t) = \frac{\varepsilon}{2}\left[e^{i\omega(t-z/c)} + e^{-i\omega(t-z/c)}\right] \tag{2.2.2}$$

The polarization density is

$$P(z,t) = N\int_0^\infty g(\omega_0)d\omega_0\langle M\rangle \tag{2.2.3}$$

N is the number of two-level atoms, $g(\omega_0)$ is the normalized line shape function of the inhomogeneous line width and $\langle M \rangle$ is the expectation of the dipole moment and can be expressed as

$$\langle M \rangle = \rho_{ab} M_{ba} + \rho_{ba} M_{ab} \tag{2.2.4}$$

To derive an expression for the propagation of an optical field that includes the effect of the induced macroscopic polarization we take the second derivation of Eq. (2.2.2) and by using the slowly varying envelope approximation and substituting into Eq. (2.2.1) we obtain

$$-i\frac{\omega}{c}\frac{d}{dz}\varepsilon e^{i\omega(t-z/c)} + i\frac{\omega}{c}\frac{d}{dz}\varepsilon e^{-i\omega(t-z/c)} - i\frac{\omega}{c^2}\frac{d}{dt}\varepsilon e^{i\omega(t-z/c)} + i\frac{\omega}{c^2}\frac{d}{dt}\varepsilon e^{-i\omega(t-z/c)} = \frac{4\pi}{c^2}\frac{d^2}{dt^2}\overline{P}$$

(2.2.5)

We now return to the polarization density and let the density matrix elements oscillate in time such that

$$\langle M \rangle = \rho'_{ab} M_{ba} e^{i\omega(t-z/c)} + \rho'_{ba} M_{ab} e^{-i\omega(t-z/c)} \tag{2.2.6}$$

By substituting Eq. (2.2.6) into Eq. (2.2.3)

$$P(z,t) = N\int_0^\infty g(\omega_0) d\omega_0 (\rho'_{ab} M_{ba} e^{i\omega(t-z/c)} + \rho'_{ba} M_{ab} e^{-i\omega(t-z/c)}) \tag{2.2.7}$$

By using Eq. (2.1.33) – (2.1.35) and letting the density matrix off-diagonal elements be complex, we obtain

$$\rho'_{ab} = \frac{1}{2}(r_1 + ir_2) \tag{2.2.8}$$

$$\rho'_{ba} = \frac{1}{2}(r_1 - ir_2) \tag{2.2.9}$$

Substituting Eq. (2.2.8) & (2.2.9) into (2.2.7), we get

$$P = N\int_0^\infty g(\omega_0)d\omega_0(\frac{1}{2}(r_1 + ir_2)M_{ba}e^{i\omega(t-z/c)} + \frac{1}{2}(r_1 - ir_2)M_{ab}e^{-i\omega(t-z/c)})$$

(2.2.10)

The second time derivation is now taken and assuming that r_1 and r_2 vary slowly in the time, t we get

$$\frac{d^2}{dt^2}P(t) = -\frac{N}{2}\omega^2\int_0^\infty g(\omega_0)d\omega_0((r_1 + ir_2)M_{ba}e^{i\omega(t-z/c)} + (r_1 - ir_2)M_{ab}e^{-i\omega(t-z/c)})$$

(2.2.11)

The almost last step is to substitute the last equation into Eq. (2.2.5) and by matching the $e^{i\omega(t-z/c)}$ components we obtain

$$i\left[\frac{d}{dz}\varepsilon + \frac{1}{c}\frac{d}{dt}\varepsilon\right] = 2N\pi\frac{\omega}{c}M_{ba}\int_0^\infty (r_1 + ir_2)g(\omega_0)d\omega_0 \qquad (2.2.12)$$

The real part of the left hand side in the last equation is equal to zero which means that the electric field responds to out-of-quadrature components of the polarization. By utilizing that and using Eq. (2.1.36), we arrive at the final form of Maxwell's wave equation.

$$\frac{\partial\Omega(z,t)}{\partial z} + \frac{n}{c}\frac{\partial\Omega(z,t)}{\partial t} = \frac{\alpha}{2\pi}\int_0^\infty r_2(z,t,\omega_0)g(\omega_0)d\omega_0 \qquad (2.2.13)$$

where α is the absorption coefficient that incorporates the material properties. We again write the coupled Maxwell-Bloch equations in general form which we in the next section are going to use in our investigation.

$$\frac{\partial r_1(z,t,\Delta)}{\partial t} = \Delta r_2(z,t,\Delta) - \frac{r_1(z,t,\Delta)}{T_2} \qquad (2.2.14)$$

$$\frac{\partial r_2(z,t,\Delta)}{\partial t} = -\Delta r_1(z,t,\Delta) + \Omega(z,t)r_3(z,t,\Delta) - \frac{r_2(z,t,\Delta)}{T_2} \qquad (2.2.15)$$

$$\frac{\partial r_3(z,t,\Delta)}{\partial t} = -\Omega(z,t)r_2(z,t,\Delta) - \frac{r_3(z,t,\Delta)+1}{T_1} \qquad (2.2.16)$$

$$\frac{\partial\Omega(z,t)}{\partial z}+\frac{n}{c}\frac{\partial\Omega(z,t)}{\partial t}=\frac{\alpha}{2\pi}\int_{-\infty}^{\infty}r_2(z,t,\Delta)g(\Delta)d\Delta \tag{2.2.17}$$

The electric field is $E(z,t)=(\hbar/\mu)\Omega(z,t)\cos(\omega_0 t-kz)$ and $\Omega(z,t)$ is the slowly varying Rabi frequency.

In all simulations in this thesis $r_3(t=0)$ is assumed to be -1 but in fact it can take any value between +1 & -1 where -1 means the atom in its ground state and +1 means the atom in its excited state and 0 represents equal population in ground and excited states. We also assume that the length in space of the smooth optical pulse is much longer than the medium which means that we can ignore the second term of the left side of Eq. (2.2.17) (i.e. $\frac{n}{c}\frac{\partial\Omega(z,t)}{\partial t}$).That simplifies our numerical simulation to direct integration of Eq. (2.2.17)

If we include the imaginary part of the optical field, the Maxwell-Bloch equations become [9]

$$\frac{\partial r_1(z,t,\Delta)}{\partial t}=\Delta r_2(z,t,\Delta)+\Omega_i(t,z)r_3(t,z,\Delta)-\frac{r_1(z,t,\Delta)}{T_2} \tag{2.2.18}$$

$$\frac{\partial r_2(z,t,\Delta)}{\partial t}=-\Delta r_1(z,t,\Delta)+\Omega_r(z,t)r_3(z,t,\Delta)-\frac{r_2(z,t,\Delta)}{T_2} \tag{2.2.19}$$

$$\frac{\partial r_3(z,t,\Delta)}{\partial t}=-\Omega_r(z,t)r_2(z,t,\Delta)-\Omega_i(t,z)r_1(t,z,\Delta)-\frac{r_3(z,t,\Delta)+1}{T_1} \tag{2.2.20}$$

$$\frac{\partial\Omega_r(z,t)}{\partial z}=\frac{\alpha}{4\pi}\int_{-\infty}^{\infty}r_2(z,t,\Delta)g(\Delta)d\Delta \tag{2.2.21}$$

$$\frac{\partial\Omega_i(z,t)}{\partial z}=\frac{\alpha}{4\pi}\int_{-\infty}^{\infty}r_1(z,t,\Delta)g(\Delta)d\Delta \tag{2.2.22}$$

Here Ω_i and Ω_r are the imaginary and real part of optical field respectively using the atomic transition as referenced field.

Chapter 3

The General Computer Code

In fact there are about twelve different types of computer simulations based on different input data (i.e. different input optical pulses). In all cases the output data are registered and plotted to be investigated. In this section we will explain the computer simulations in general terms to help the readers understand the computer programs in the next sections.

Generally and besides defining of all constants the code can be viewed as consisting of three main parts which are simulation of the input pulses, the inhomogeneous line width $g(\omega_0)$ and the numerical integration of Maxwell-Bloch equations.

At the beginning the variables time t, thickness z of crystal which corresponds to the absorbing medium and the frequency ω are initialized. These variables are chosen to be one microsecond, one centimeter and 310 megahertz respectively. We then divide these variables into dt, dz & $d\omega$ to be later used in our simulations.

In the next step we initialize the input pulses and in this thesis there are three cases. In two of them the input pulses are chosen to be either Gaussian or square pulses while in the third case hyperbolic secant pulses are chosen as input data. We use an *if* sentence to decide which pulses will be applied. In connection with this the amplitude and the length of the input pulses are defined. The time-depending Gaussian pulses are initialized by using a loop function with counter ***I*** where ***I*** represents the temporal window. In case of square pulses, the ***if*** sentence is used within the time loop function ***I*** and by defining time limits for the input pulses, the input pulses are initialized. The result of the operation which are carried out in each time step in this loop are saved in a one dimensional matrix which is called ***"input_pulse"***.

In the next step and as mentioned in Eq. (2.2.17) the function Ω depends on the time and space. Therefore two loops with counters ***I*** and ***J*** are defined which correspond to the consecutive steps in time and space respectively. Into these loops we put the *input_pulse* matrix into a new two-dimensional matrix ***Omega(I,J)*** which represents the output electromagnetic field. The inhomogeneous line width is taken from *Carrie Cornish* [3] simulation routine without any change.

We are now about to describe the integration of the Maxwell-Bloch equations. Firstly the matrices $r_{1,2,3}$ are initialized by using two loops ***J*** & ***K*** representing the space and frequency components respectively. In this step and as we know that r depends on time, space and frequency but we preferred to divide it into 2 matrices, one containing the space and frequency and another containing the time and space over all frequencies. That makes the program run faster and decreases the time of checking the output data. The atoms are considered to be in their ground state such that r_1 & $r_2 = 0$ while $r_3 = -1$. These matrices go through three loops where the inner loop corresponding the small varying in frequency and contains the first three Maxwell-Bloch equations. Into this loop the sum over all frequencies of r is taken and saved into three new two-dimensional matrices u, v & w. These new matrices contain the varying in Maxwell-Block equations during the time and in the medium over all frequencies. The direct integration of Ω is calculated in middle loop and stored as a function of space and time. The matrices r and Ω are then plotted for two cases of having two and three input pulses

That was a description of the general simulation routine that we used in this thesis which corresponds to a typical experiment done in any lab.

Chapter 4

The Numerical Calculations

In this chapter we will go through all the different types of simulations that were done in this thesis one by one. There are three cases in the next section. The first case is when the Bloch vectors *r* depends on time only. That means that the absorption is weak and there is a single resonant frequency. In the second case we include a distribution of absorbing frequencies but the medium is still weakly asorbing. In the third case we include the absorbing medium in our program. In all cases the calculations that were carried out are plotted.

4.1 Application of Gaussian and Square Pulses

In this section we will apply Gaussian and square pulses for each of the three different cases, when the effects of frequency and space are ignored, when the effect of space is ignored only and when the effects of time, space and frequency are all included in the calculations.

4.1.1 The Effect of Time Only

In this simulation routine the Bloch vector *r* depends only on time. That means we ignore the effects of frequency detuning and the absorbing medium in our calculations. In this simulation routine we initialize the input pulse which is one pulse only. As the absorbing medium is not included into this part of the simulation routine, the inhomogeneous line width can be neglected. The application of a Gaussian or a square pulse in this case, cause Bloch vector to change between the upper and lower state exhibiting so called Rabi oscillations. The plotted output data of the Bloch vector *r* is shown in the following figures.

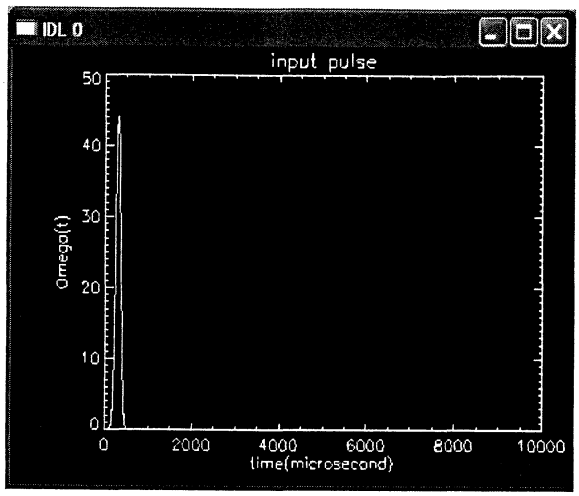
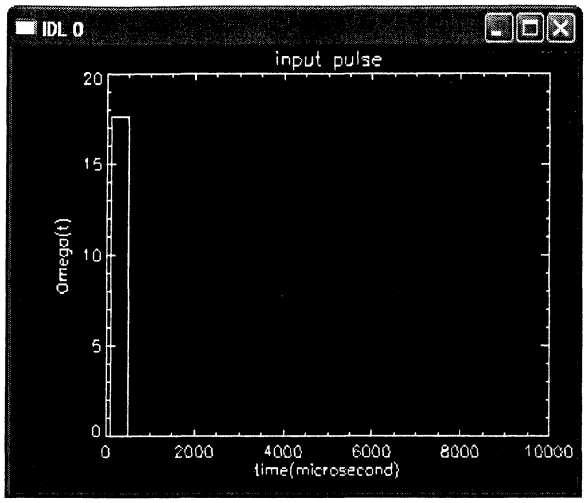


Fig (4.1.1a) The Gaussian input pulse



Fig(4.1.1b) the square input pulse

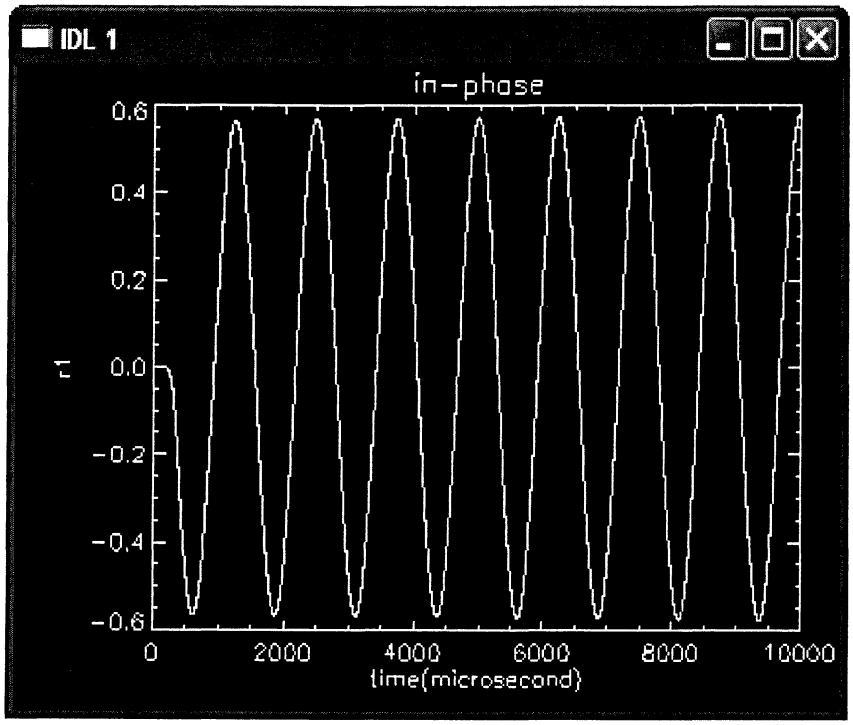
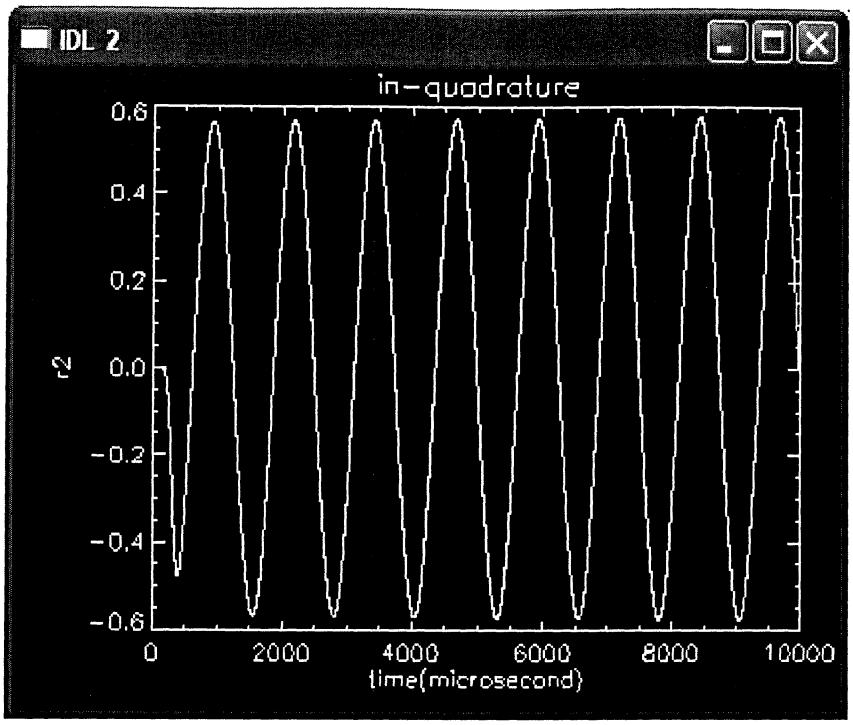


Fig (4.1.2) shows the Bloch vector component r_1 for detuning $\Delta = 8$ MHZ.

r_1 is the component of the atom polarization that is in phase with the driving field.



Fig(4.1.3) shows the in-quadrature component

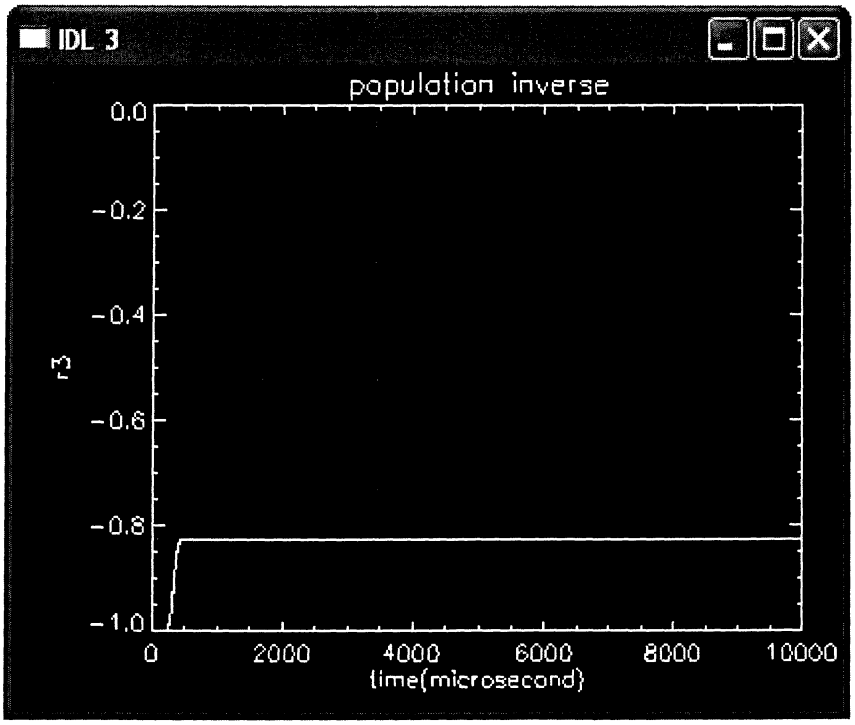


Fig. (4.1.4) the population inverse component r_3 starts from -1 as we assumed that the atoms are in its ground state. As we see in fig. (4.1.2) & (4.1.3), the in-phase and in-quadrature components start from zero while the population inverse component starts from -1 as we assumed that the atoms are in their ground state. By applying an electromagnetic field, the atoms excite to an excited state. The y-axis shows the probability of finding the atom in the excited state while the x-axis shows the interaction time between the coherent light field and the atom.

If we put the T_1 & T_2 equal to 1 (i.e. we include the relaxation time), the probability of finding the atom in the excited state becomes less and less. That is shown in the figures below

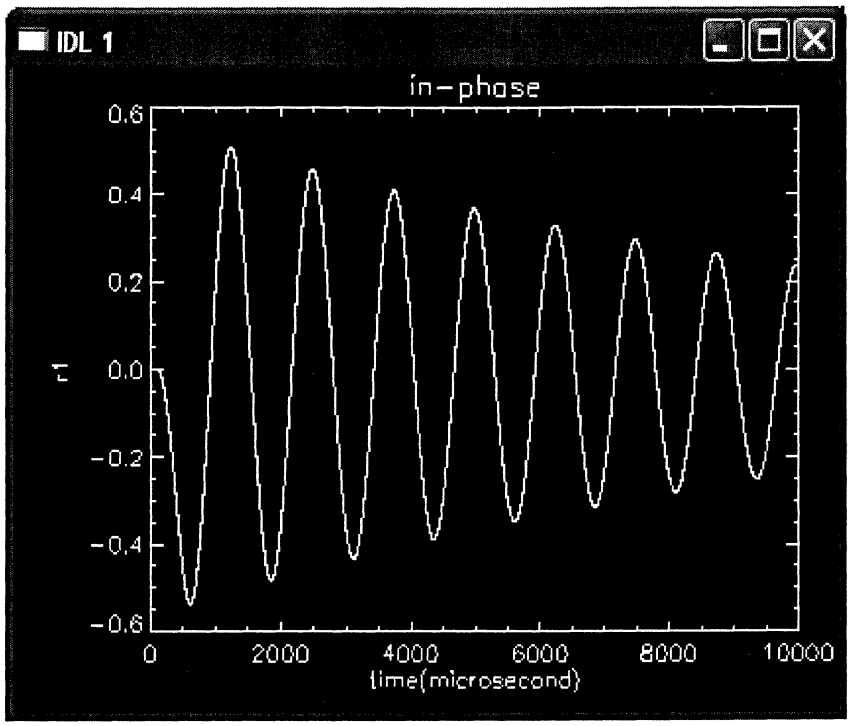


Fig. (4.1.5) shows the in-phase for detuning $\Delta = 8$ MHZ and how the probability decreases with time. The same thing happens to the other components (i.e. the in-quadrature and population inversion components)

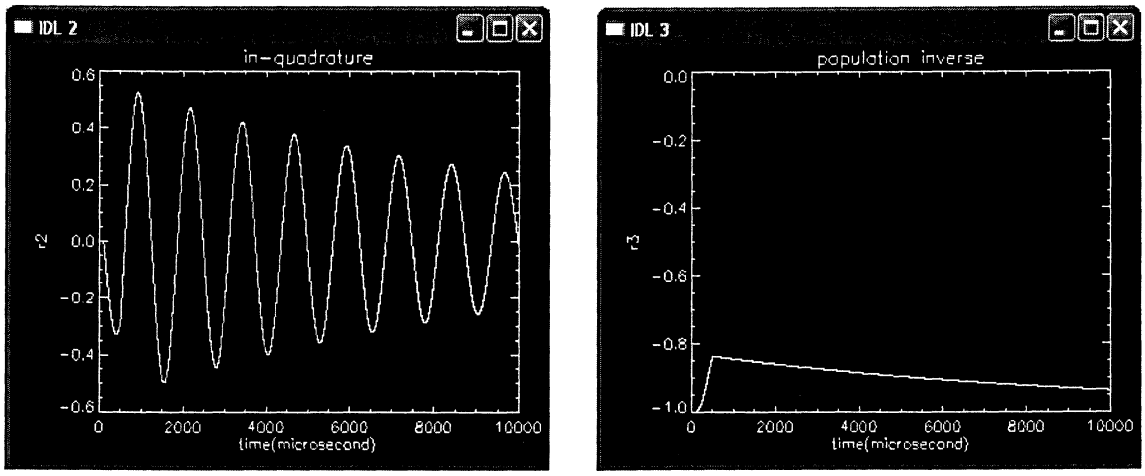


Fig (4.1.6a, b) shows the in-quadrature component and the population inversion respectively.

4.1.2 The Effect of Time & Frequency

Now we include the fact that each input pulse will interact with a range of atoms with different frequencies in the Maxwell-Bloch equations. For that we introduce of frequency interval in the program. In this case there are two and three input pulses included in the calculation respectively. The inhomogeneous line width is also included in the program. The integration is done by using two loops ***I*** & ***K*** which represent time and frequency variables respectively. The slowly varying Bloch vector is calculated for detuning frequencies $-\Delta$ to $+\Delta$ by taking a small step of order of *df*. The output data are stored over all frequencies into new one-dimensional matrices ***u, v*** & ***w*** which correspond to r_1, r_2 & r_3 respectively, while the slowly varying Rabi frequency Ω is calculated by direct integration into ***I*** loop and stored over all time.

We firstly apply two Gaussian pulses with a pulse area of order of $\pi/2$ and π respectively as shown in fig. (4.2.1)

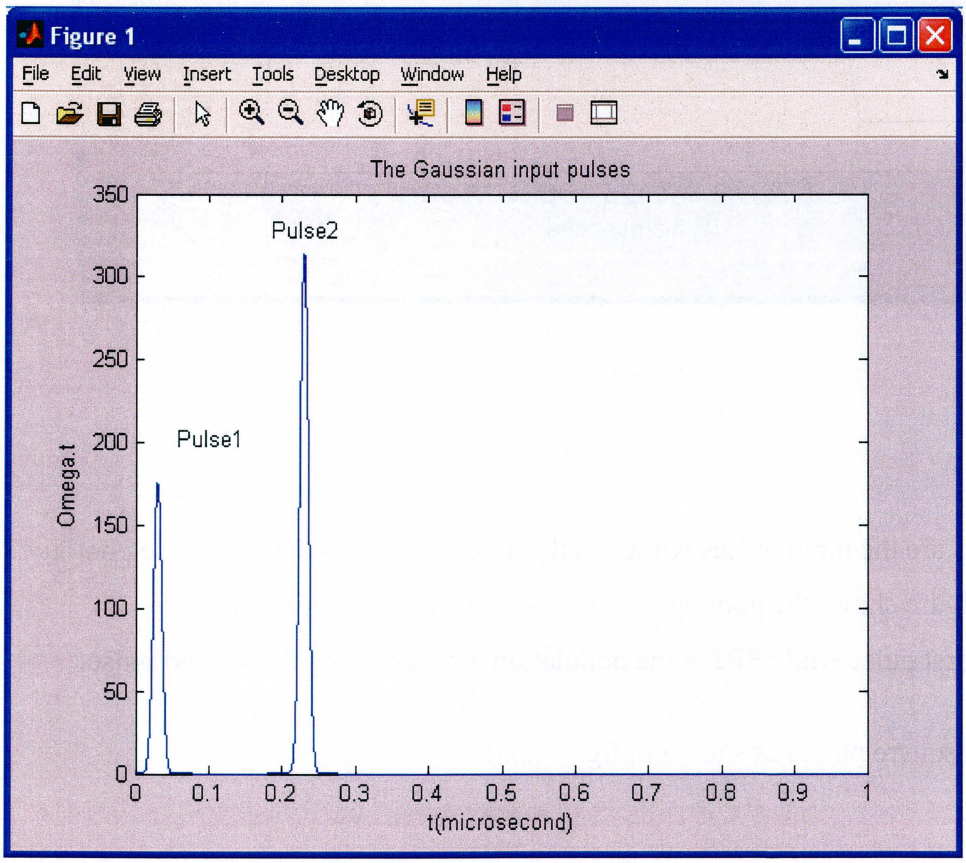


Fig. (4.2.1) Gaussian pulses

The application of these two Gaussian pulses leads to the emission of a photon echo as shown in the next figures

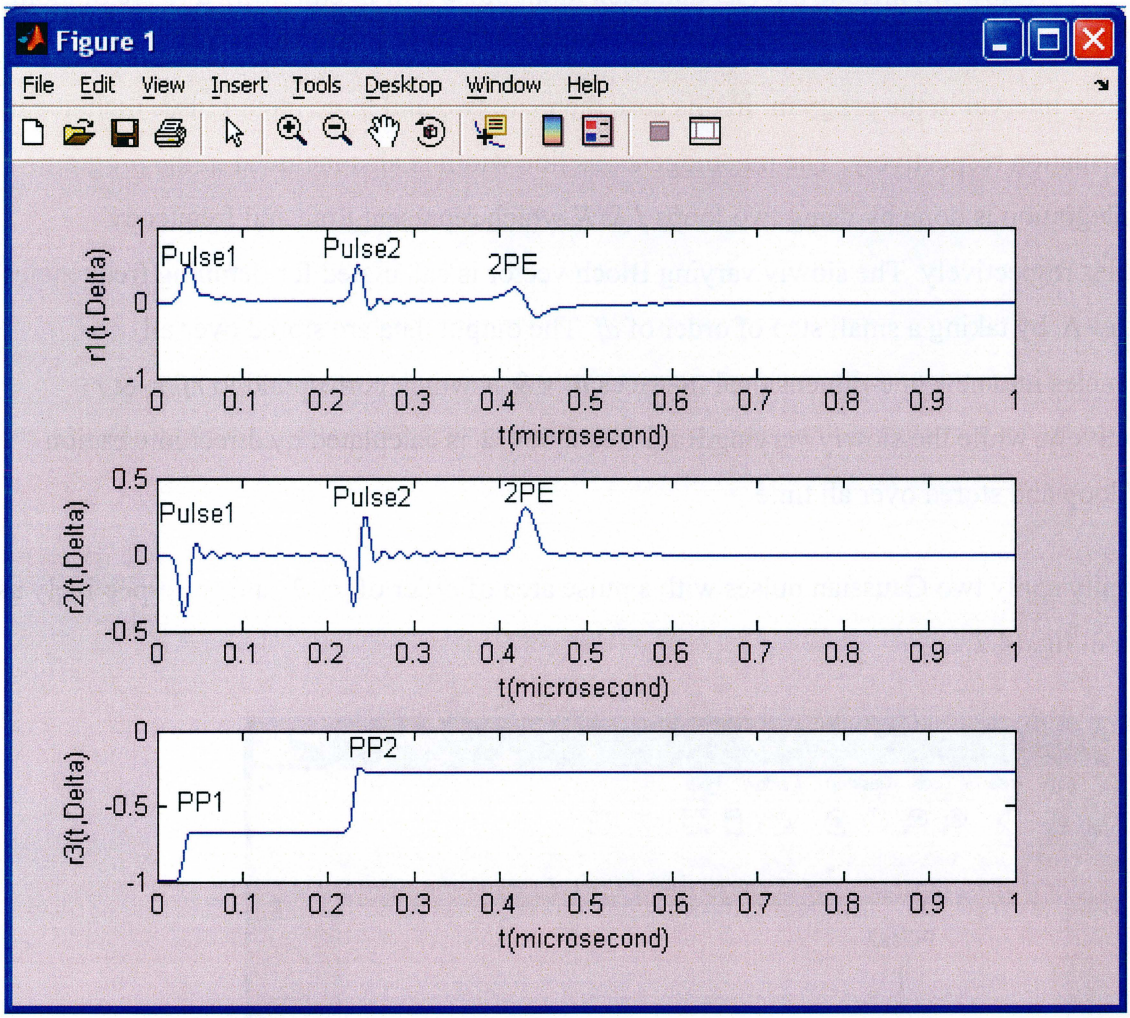


Fig.(4.2.2)

The first two pulses are the input pulses while the third pulse (2PE) is the two-pulse photon echo. The lowest figure shows the population where PP1 is the population after the application of the first pulse while PP2 is the population after applying the second pulse.

We now apply two square pulses as shown in fig. (4.2.3)

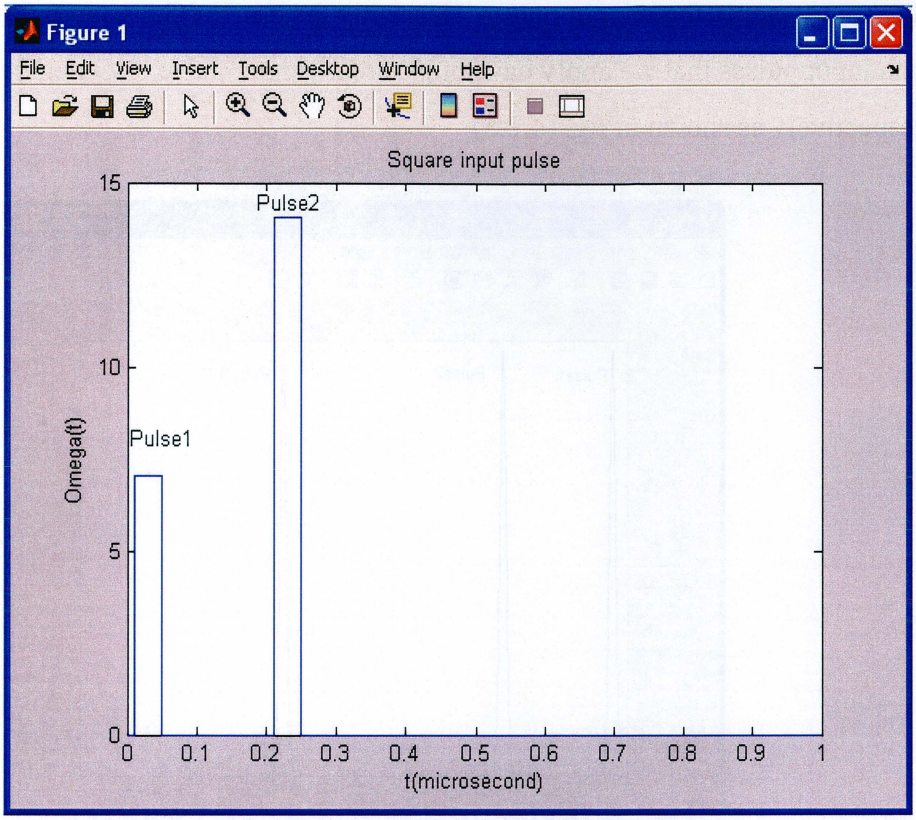


Fig. (4.2.3)

By applying two square pulses, we obtain

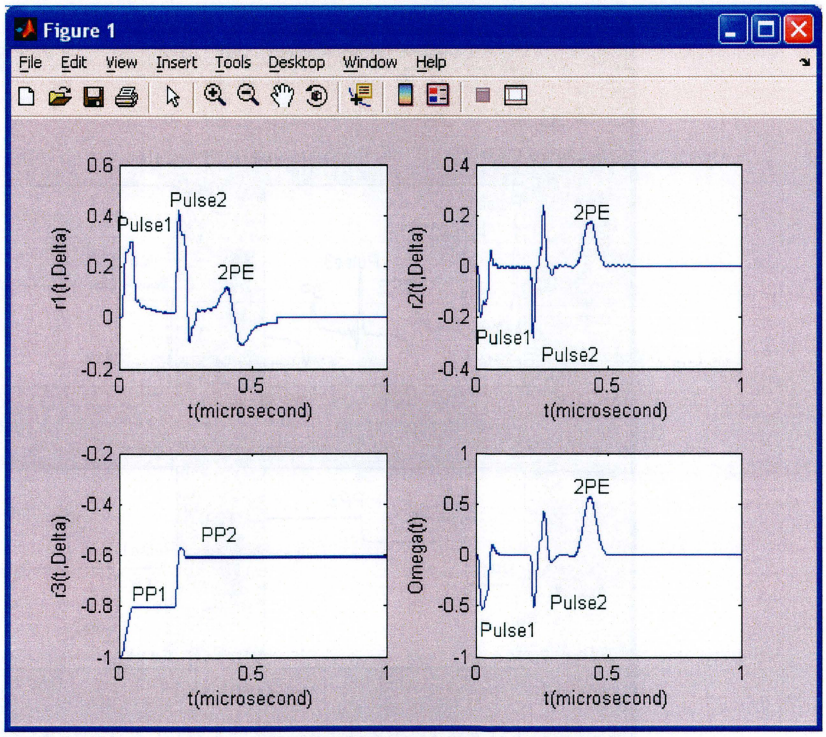


Fig. (4.2.4) (To see the 2PE clearer every component can be plotted using the program code in appendix II)

We are now about to apply three input pulses. Here only the case of Gaussian pulses is shown. The Gaussian input pulses that we apply have pulse areas (input_pulse or Omega) $\pi/2$, $\pi/2$, and $\pi/2$ respectively as shown in fig. (4.2.5)

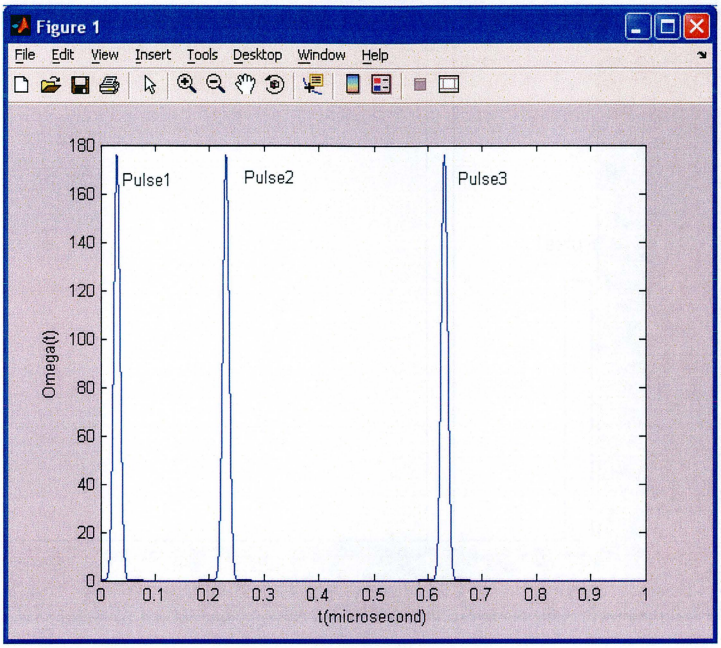


Fig. (4.2.5) Gaussian input pulses

The application of these pulses gives

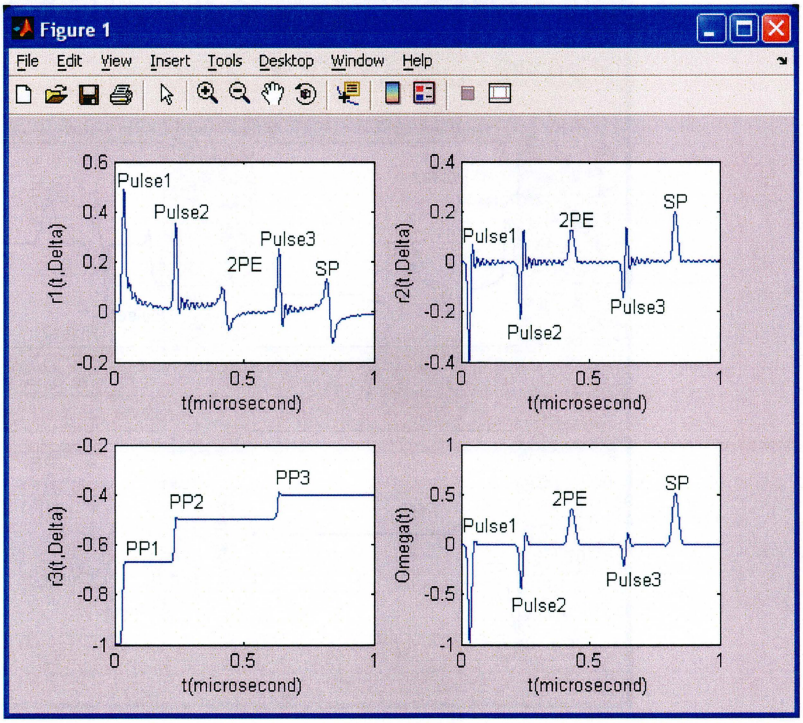
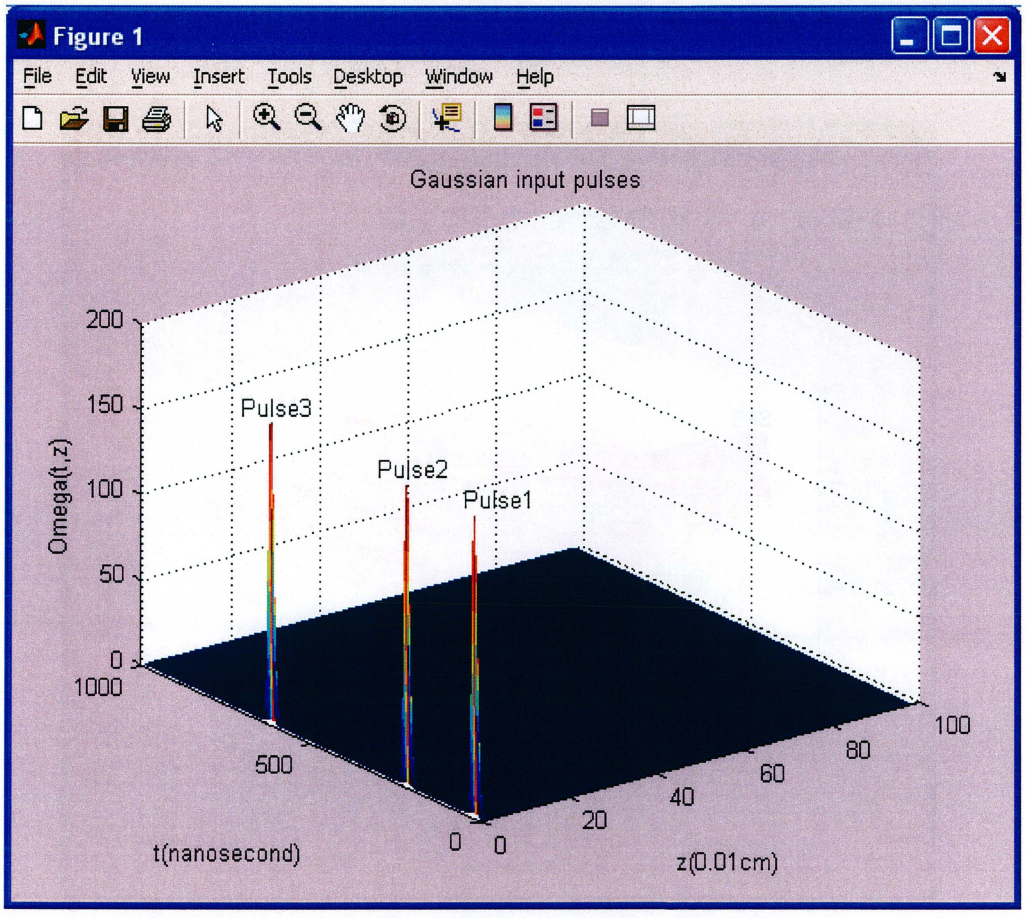


Fig (4.2.6) Shows the plot of Block vector's components *r* by applying three pulses

Fig. (4.2.6) the pulse1, pulse2 and pulse3 are the input pulses, 2PE is the two-pulse photon echo and SP is stimulated photon echo. In analogy to Fig(4.2.4) PP1, PP2 and PP3 corresponds to the population after applying pulses1, 2 and 3 respectively.

4.1.3 The Effect of Time, Space and Frequency

In this section we include the absorbing medium in our simulation routine which means that the Maxwell-Bloch equations depend on time *t*, position ,s, in the absorbing medium which is assumed to be a crystal of length of 1cm and the frequency detuned off resonance. That leads to including of an extra loop with counter ***J*** in the integration part, an extra loop to the part of initializing Ω and an extra loop to the part of initializing the Bloch vector *r*. We will discuss the case of applying three pulses. However the case of applying two pulses can be treated by running the program in appendix III. The application of three Gaussian pulses (Fig. 4.3.1) gives



Fig(4.3.1) shows Gaussian input pulses

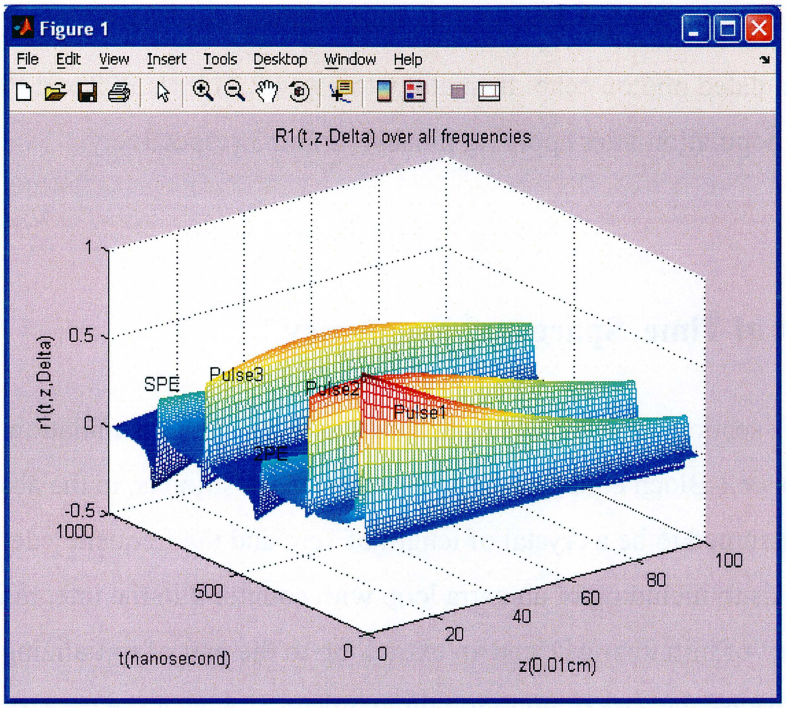


Fig. (4.3.2) shows the in-phase component. The pulse 1, 2 and 3 are our input pulses while 2PE and SPE are the two-pulse photon echo and stimulated photon echo. As we expected, the application of pulses of order of π /2 and π /2 cause the emission of a photon echo while the application of an additional pulse of order of π /2 causes another photon echo. These photon echoes called two-pulse photon echo and stimulated photon echo respectively. The figure shows how the pulses propagate through the crystal.

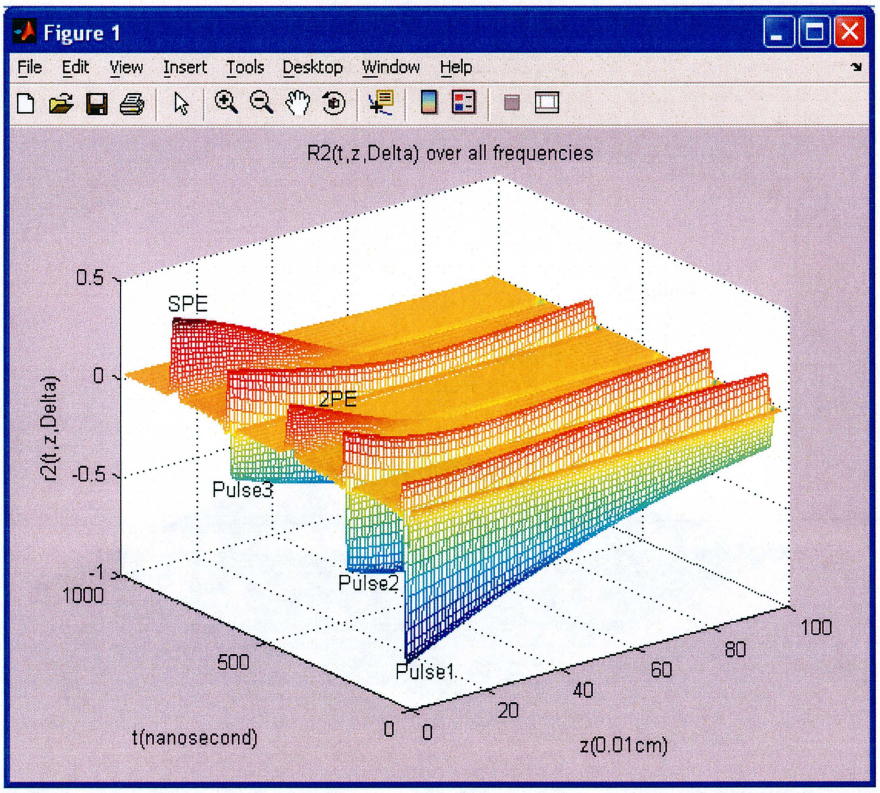


Fig. (4.3.3) shows the average of in-quadrature component over all frequencies

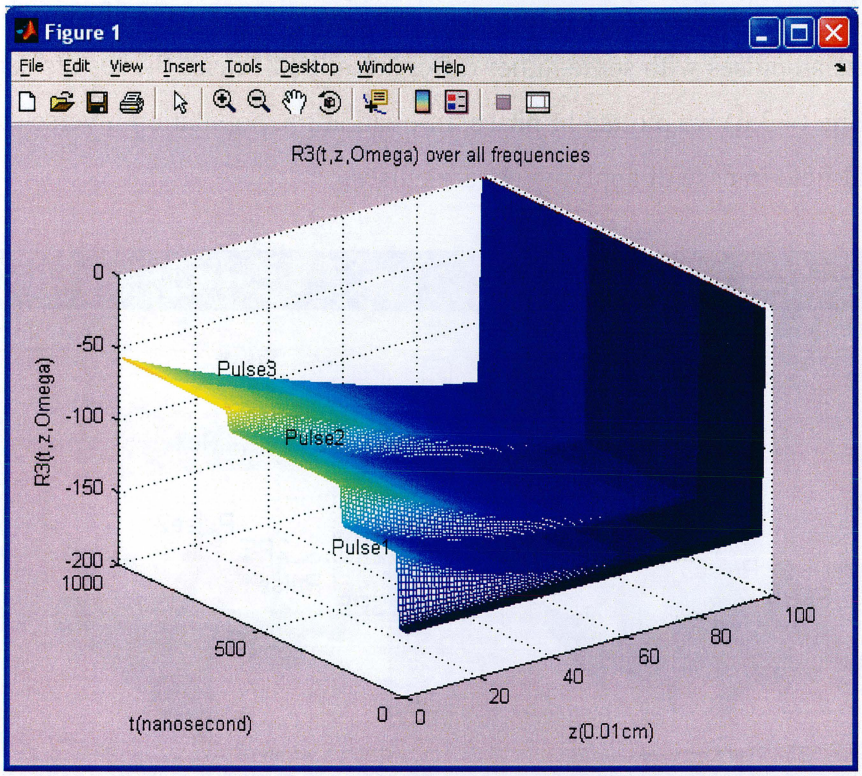


Fig. (4.3.4) the average population integrated over all frequencies.

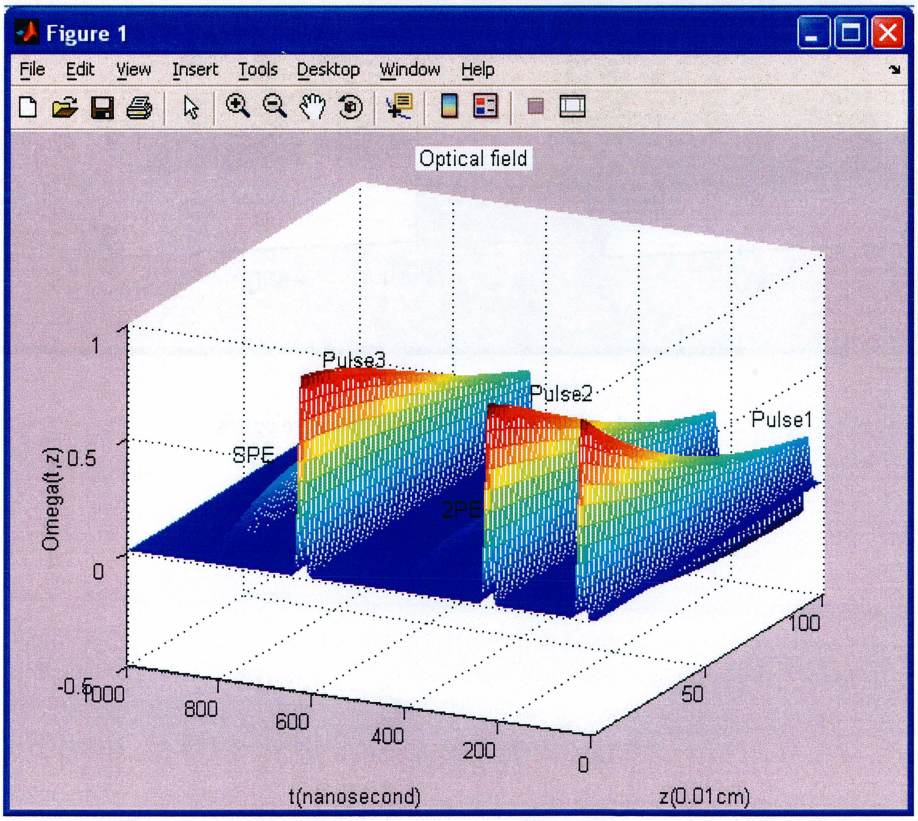


Fig. (4.3.5) $\Omega(t,z)$ integrated over all frequencies

The application of three square pulses gives the same result as we have got by applying Gaussian pulses with some minor differences. In all cases the results came as we expected. In Fig. (4.3.6) we plotted in-phase, in-quadrature, population inverse components and the optical field in case of applying square pulses.

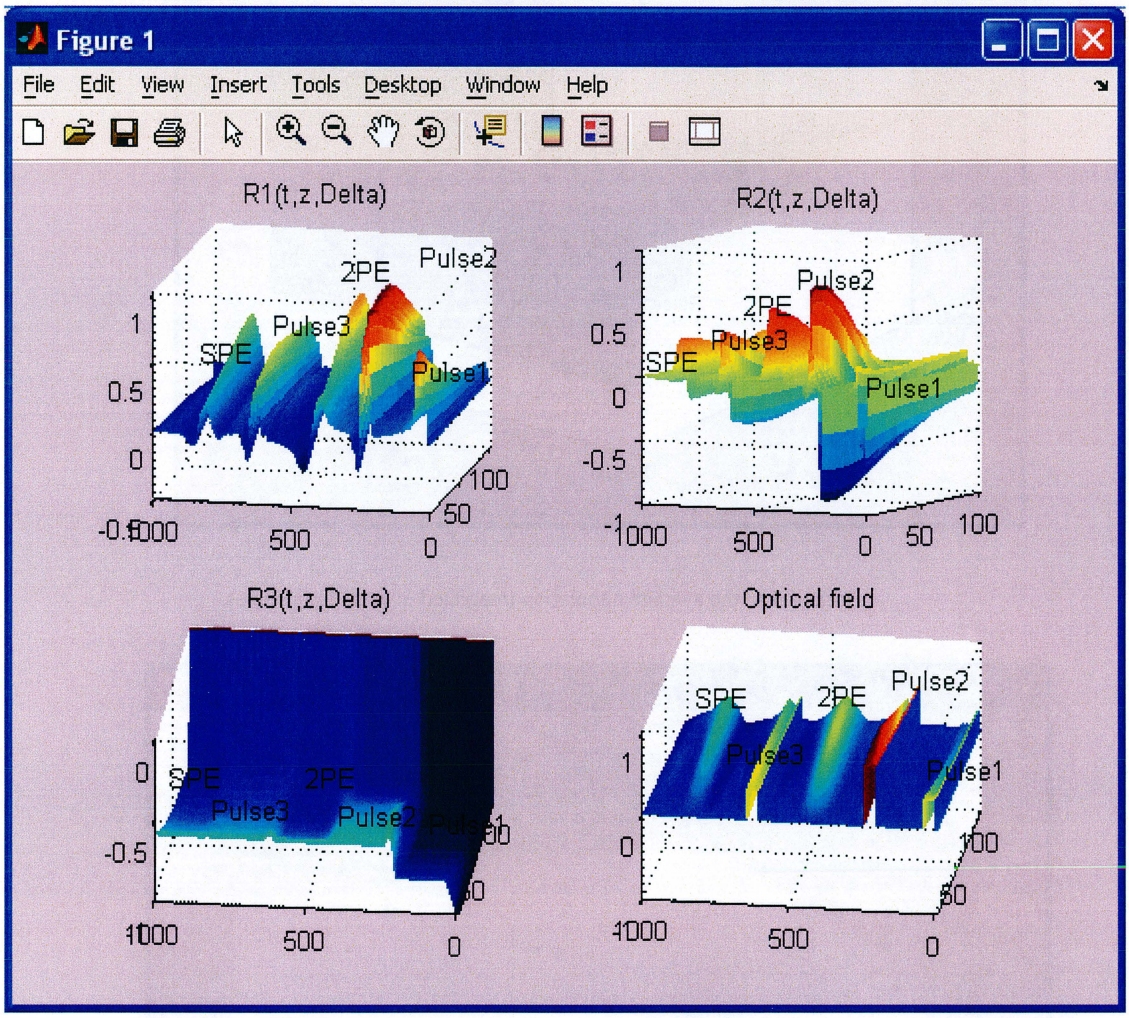


Fig. (4.3.6) shows the application of square pulses

4.2 The Application of Complex Hyperbolic Secant pulses

In this section we are interested in the output data of the population inverse r_3 that we get by applying complex hyperbolic secant pulses. The equation that describes such kind of pulse can be written as:

$$\Omega_R(t) = \Omega_0[\sec h(\beta(t-t_0))]^{1+i\mu}$$

where Ω_R is Rabi frequency, Ω_0 is the maximum Rabi frequency, μ is a real constant and β can be written as $\beta = FWHM / 2.6$

Firstly and as we did in the previous section, we ignore the effect of the absorbing medium and check the Bloch vectors as functions of time and frequency and the optical field Ω as a function of time. We then add the effects of absorbing medium to our calculations and plot the r_3 and the other components in three dimensions (t,z, Δ). The figure below shows a complex hyperbolic secant pulse that we use in our calculation.

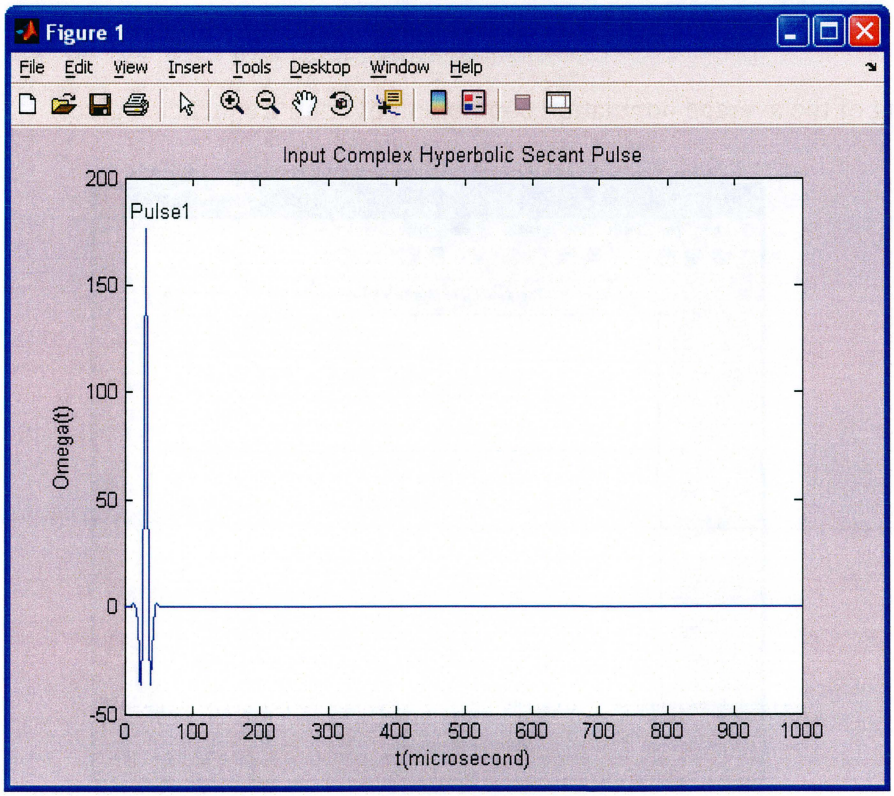


Fig. (4.4.1) the input complex hyperbolic secant pulse

The application of such a pulse leads to a Gaussian shape as a result by plotting the population inverse as shown in fig.(4.4.2)

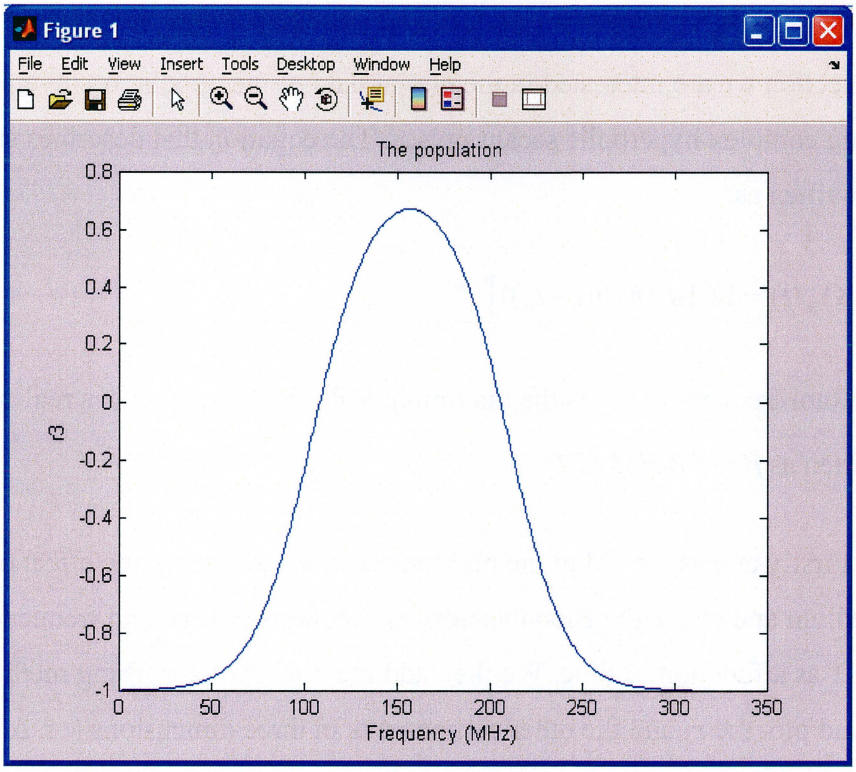


Fig. (4.4.2) shows the output data of the population inverse component r_3 that carried out by applying complex hyperbolic secant pulse

The plotting of the average population component over all frequencies gives

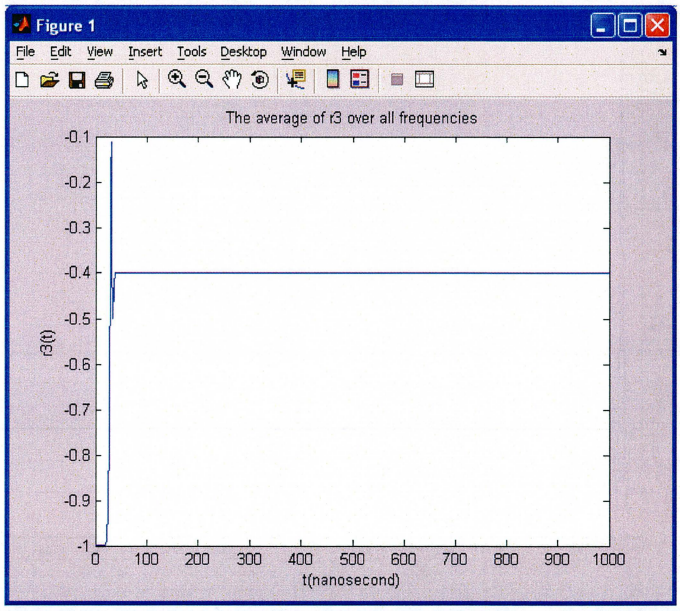


Fig (4.4.3) shows the average of the population integrated over all frequencies

We now include the effect of absorbing medium to our calculating and plot the output data that carried out of population inverse r_3. The input complex hyperbolic secant pulses are plotted in the fig. (4.4.4).

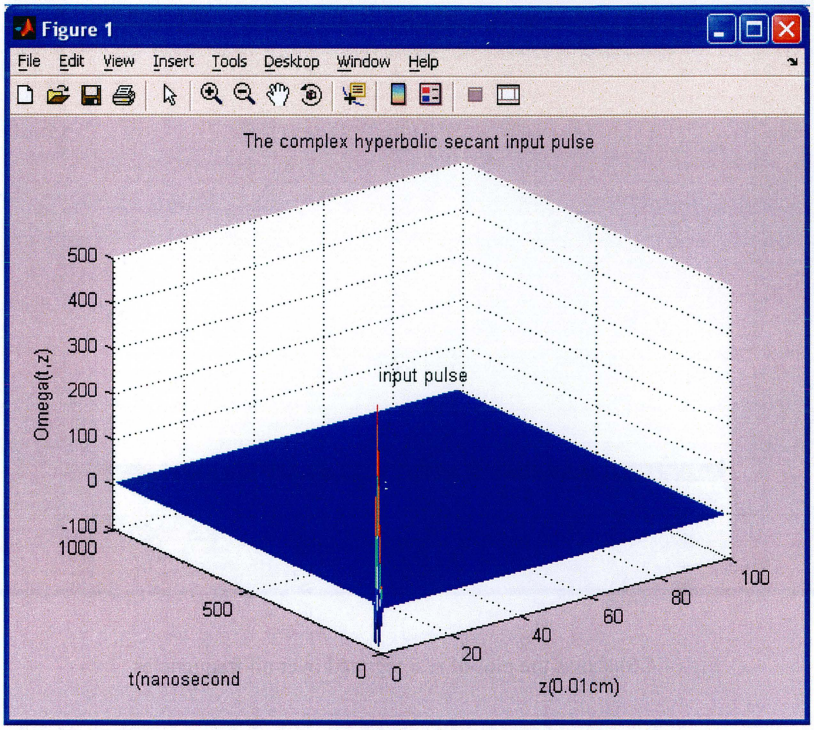
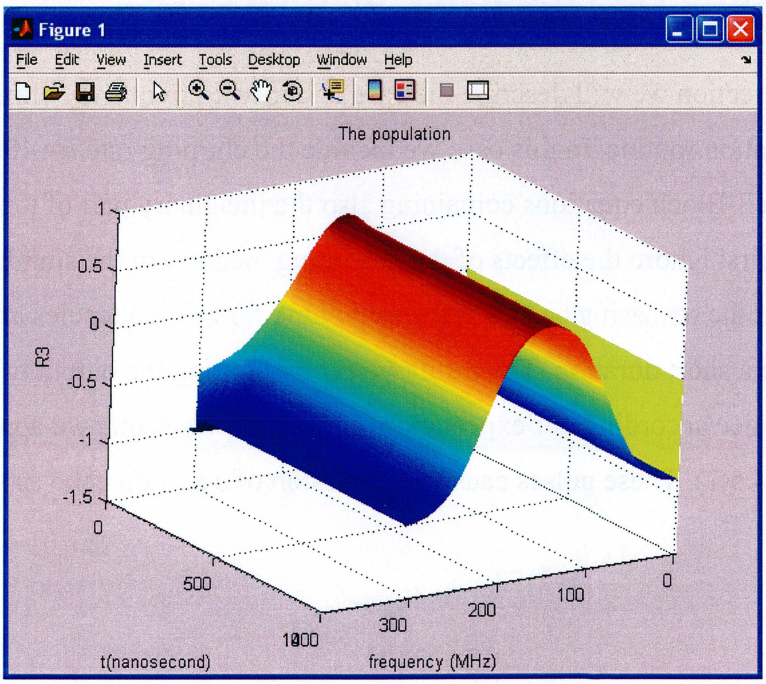


Fig. (4.4.4) shows the complex hyperbolic secant pulse used as input data



Fig(4.4.5) shows the population r_3 that carried out as a function of position in the sample over all frequencies by applying complex hyperbolic secant pulse

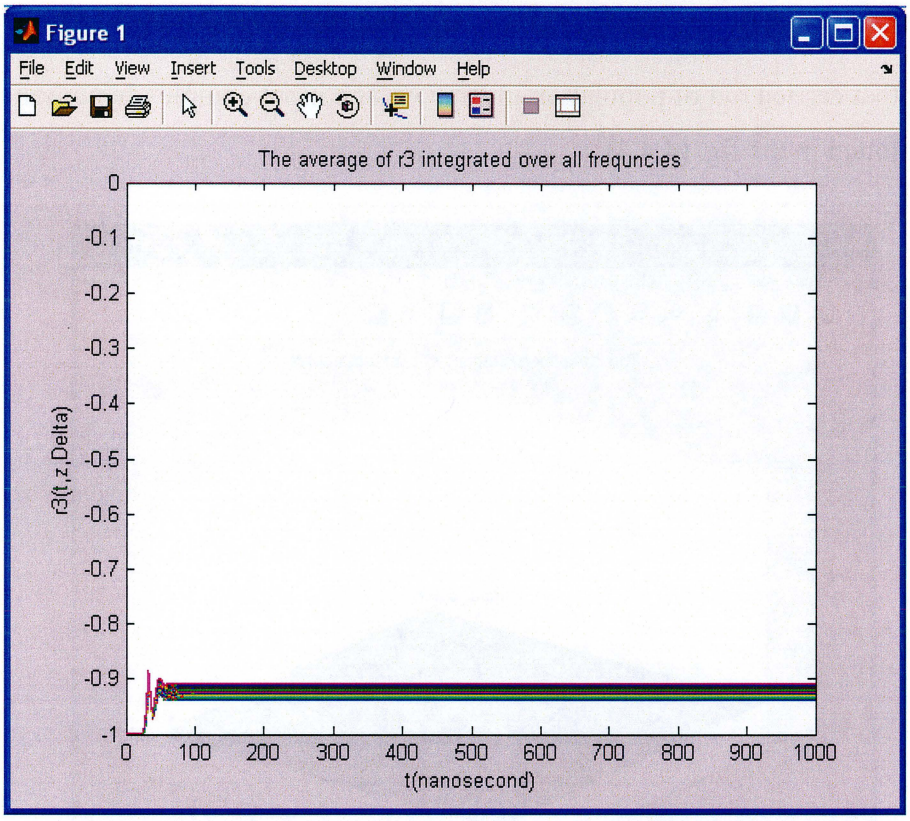


Fig (4.4.6) shows the plot of r_3 integrated over all frequencies

4.3 The Application of Frequency-Chirped pulses

In this section we will describe the case of applying frequency-chirped pulses into our simulation routine. In this case we include the chirping rate k= 400 MHz and we use Maxwell-Bloch equations containing also the imaginary part of the optical field. Doing so, we first ignore the effects of the absorbing medium in the simulation routine. We expect that the rephasing occurs at the same time for all frequencies and thus the echo will have a short duration. The result we get by plotting the output of the optical field is pretty nice according our expectation. The input pulses that we applied are shown in fig.(4.5.1). These pulses cause the emission of a photon echo as shown in fig.(4.5.2)

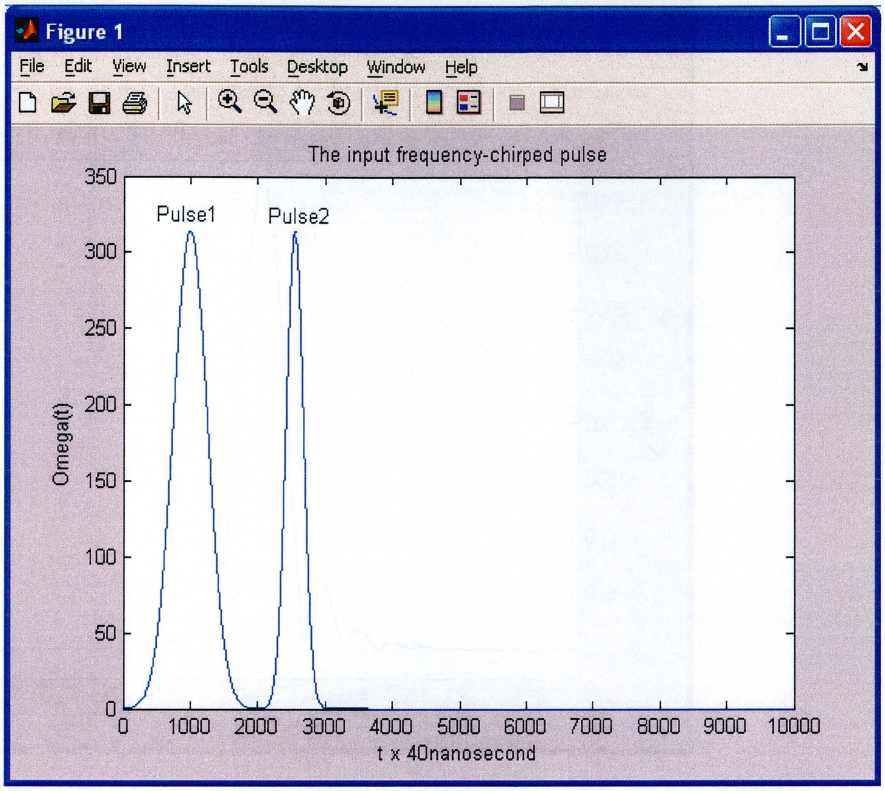


Fig. (4.5.1) shows Gaussian input frequency-chirped pulses

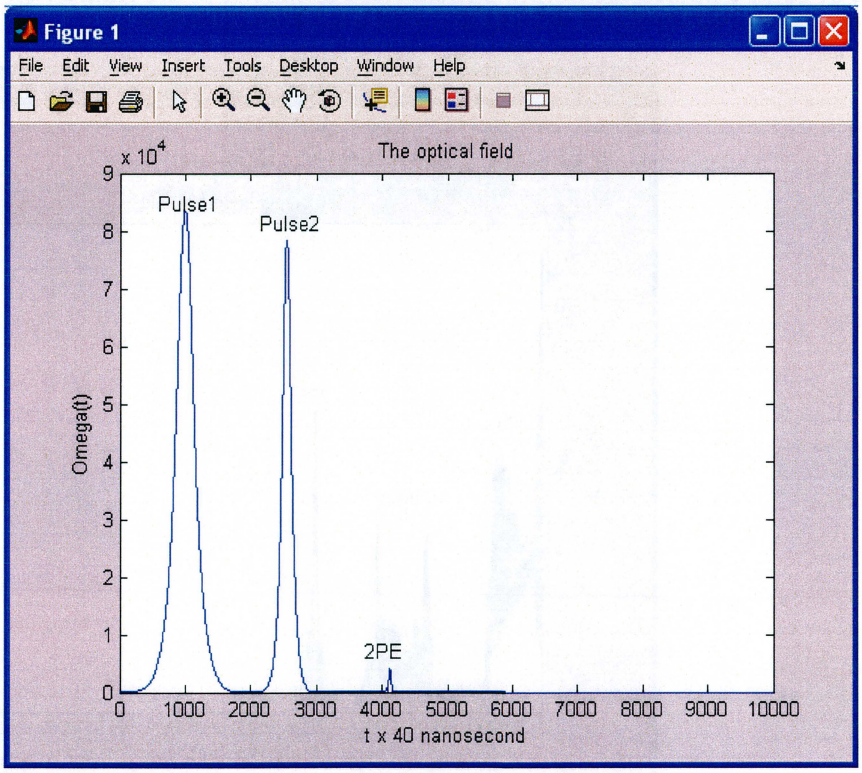


Fig. (4.5.2) shows the photon echo after applying frequency-chirped pulse

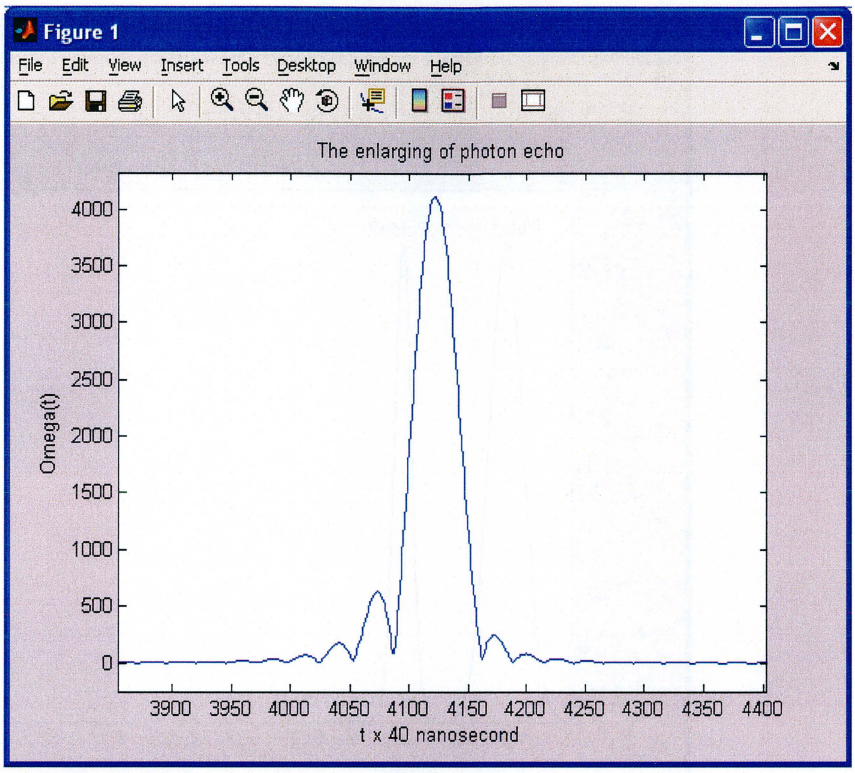


Fig. (4.5.3) shows a magnified view over the two-pulse photon echo

In case of square pulses the output optical field is shown in the figure below

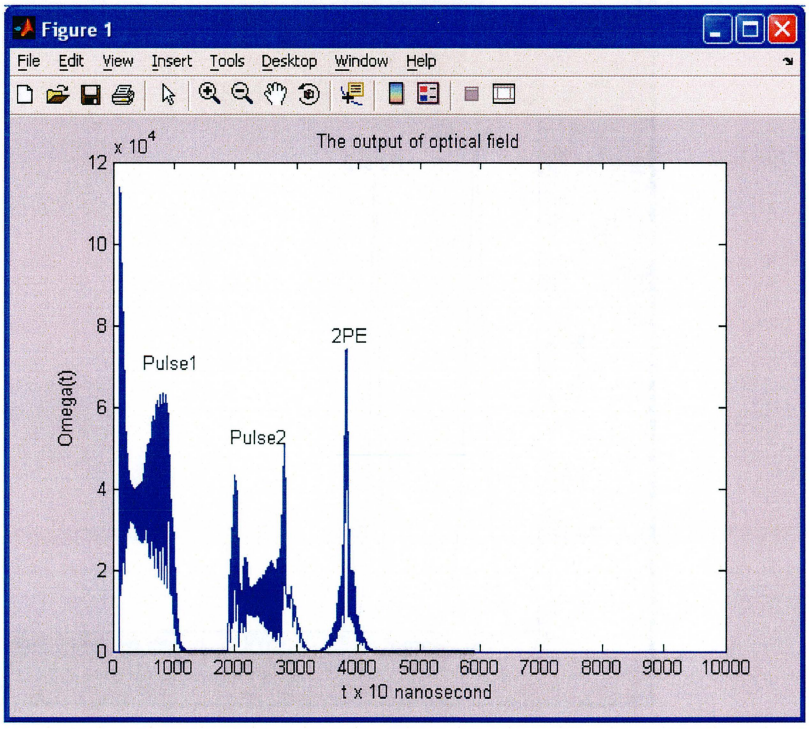


Fig. (4.5.4) shows the output optical field obtained by applying square pulses.

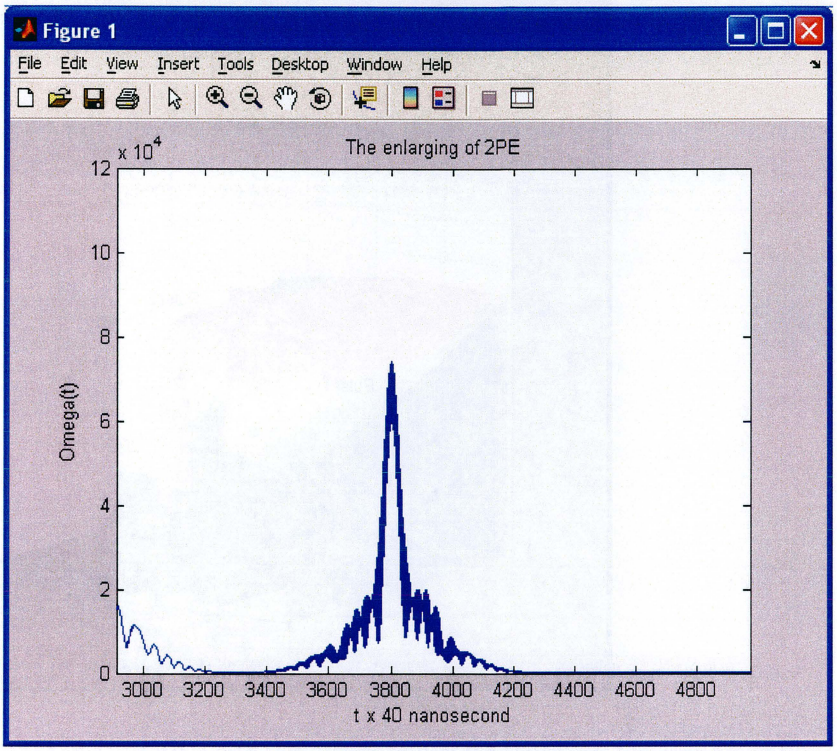


Fig. (4.5.5) an enlarging of 2PE occuring by applying square pulses

In case of including the effect of an absorbing medium and by applying Gaussian frequency-chirped pulses, we obtain

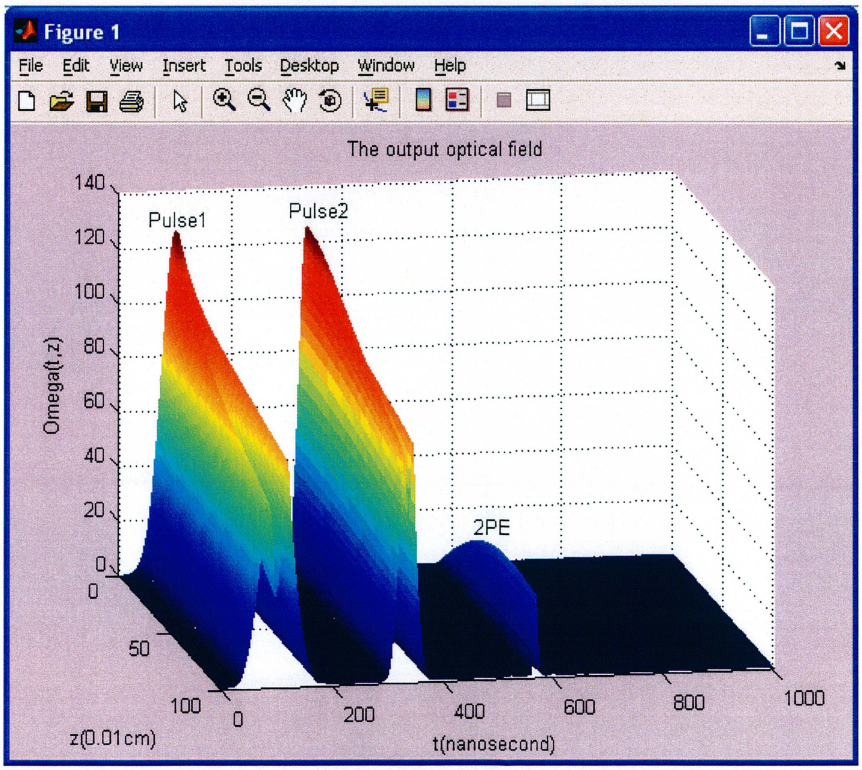


Fig. (4.5.6) the photon echoes are emitted at the same time for all frequencies and the two-photon echo is consequently compressed in time

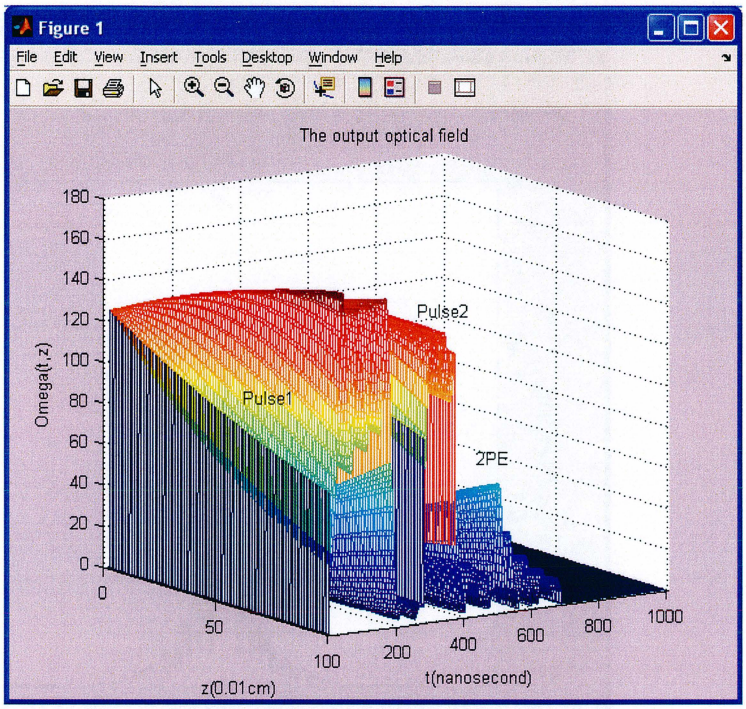


Fig. (4.5.7) shows the photon echo that occurs by applying square pulses.

Chapter 5

Conclusions and Comments

The aim of this thesis was to investigate the propagation of several kinds of coherent pulses in coherent media. At the beginning we used the old version of Matlab which was very slow. Therefore we used IDL which is analogous to Matlab but much faster. Matlab offers many things that IDL doesn't and it is easier than IDL. Later a new version of Matlab was obtained which is faster than the old version. As mentioned this thesis contains three main simulation routines to handle Gaussian and square pulses, complex hyperbolic secant pulses and frequency-chirped Gaussian and square pulses respectively.

In the first part and by applying Gaussian and square pulses for each, population and polarization obtained when absorption is neglected and when a single frequency group is investigated. The relaxation time was included into and excluded from the code respectively. Then, by also including atoms absorbing at frequencies detuned from the resonance frequency, the photon echoes appear when input pulse areas of order of $\pi/2$ or π. Later we included the time, absorbing medium and the resonance frequency (t, z, Δ), and got photon echoes with Gaussian shape.

In the second part, the complex hyperbolic secant pulses were applied in two cases, with and without the effect of absorbing medium. We obtain that the atoms in the middle of the crystal got excited more than the atoms at the edges. In this part we included the imaginary part of the optical field to handle such pulses.

In the last part we applied frequency-chirped pulses with and without the effect of absorbing medium. This part took many weeks without getting a nice result that matches our expectations. We didn't know why the simulation routine was unable to handle the frequency-chirped pulses. At last it was decided only to discuss the attempt of applying frequency-chirped pulses. However by mistake it happened that I lost all work that was made and even the text that I had already written in this thesis. So I had no other choice than to rewrite the programs. The surprise was that now when running the code the plotted output data became as we expected theoretically i.e. that the photon echoes were compressed in time.

Koherent växelverkan mellan ljus och materia

Om ljuspulsers utbredning i absorberande material

Emad Hubainy

Examensarbetet har utförts inom Atomfysiks fotonekogrupp. Gruppen arbetar med att utveckla hårdvara för kvantdatorer. En kritisk egenskap för att ett material skall vara lämpligt som kvantdatorhårdvara är att det skall vara möjligt att skapa väldefinierade kvantmekaniska superpositionstillstånd med en livstid mycket längre än tiden för en elementär operation i materialet. En elementär operation kan exempelvis vara att överföra de atomer som representerar en bit information (en kvantbit) från ett kvantmekaniskt tillstånd till ett annat. Gruppen använder elektromagnetiska pulser (laserpulser) för att kontrollera och styra operationerna som utförs av kvantdatorhårdvaran. Resonanta optiska pulser som utbreder sig i material där de kvantmekaniska tillstånden har lång superpositionstid ger upphov till flera icke-intuitiva fenomen. Den polarisation som pulserna skapar i materialet återverkar kraftigt på de propagerande pulserna och ändrar dess form. Materialet kan också själv också börja emittera ljuspulser.

Examensarbetet har bestått av att utveckla ett program som simulerar utbredningen av elektromagnetiska pulser i den typ av material som gruppen studerar. För detta har två olika programpaket, IDL och Matlab, använts. IDL är ett snabbt program som använder ungefär samma koder som Matlab. IDL användes i första delen av arbetet därför att det var snabbare än den version av Matlab som var tillgänglig då. Senare användes i huvudsak Matlab version 7.1. Beroende på vilken slags operation man vill utföra så används olika excitationspulser. I detta arbete undersöks dels pulser där frekvensen är fix medan intensiteten varierar i tiden, dels mer komplicerade pulser där såväl pulsens frekvens som intensitet varierar med tiden.

Först simulerades utbredning av pulser med fix frekvens där intensitetens tidsberoende kunde representeras med en gausskurva eller fyrkantpuls. Programmet beräknar besättningen i övre och nedre tillståndet, polarisationen i materialet samt den elektromagnetiska strålningen som emitteras från materialet. Dessa beräkningar görs för såväl svagt absorberande medier som för medier med kraftig absorption. När absorptionen är kraftig tar programmet hänsyn till att pulsintensiteten varierar allteftersom pulsen utbreder sig i materialet och population och polarisation beräknas som funktion av position i materialet. När flera koherenta excitationspulser samverkar för att skapa en polarisation i materialet kan materialet emittera strålning. Programmet har speciellt skrivits så att det kan simulera materialets respons på excitation av flera koherenta pulser. Speciellt har så kallade fotonekoprocesser studerats. Fotoneko är analogt till kärnspinnresonans (NMR) men optiska pulser används till excitationen istället för radiopulser och de övergångar som drivs är optiska övergångar istället för kärnspinnövergångar. Programmets simulerade resultat stämde här väl överens med teoretiskt förväntade och experimentella resultat.

Programmet modifierades därefter så att det även kunde behandla pulser där såväl frekvens som amplitud ändras med tiden. I de flesta fall uppnåddes här överensstämmelse med teoretiskt förväntade resultat. Programmet kan således testa effekten av olika excitationspulser. Eftersom experimenten ofta är komplicerade och tidsödande kan denna typ av program vara värdefulla hjälpmedel för experimentplaneringen.

References:

1. Bouwmeister, Ekert and Zeilinger, *The physics of quantum information* Springer, (2000)

2. Niklas Ohlsson, *Quantum optics and quantum information processing in rare-earth-ion-doped crystal,* PhD. Thesis, Lunds raports on atomic physics, LRAP-300,(2003)

3. Carrie Cornish, *Highly efficient photon echo generation and a study of the energy source of the photon echoes*, PhD Thesis, LS. of Washington (2000)

4. Roos Ingela, *Theoretical investigation of robust quantum computing in rare-earth-ion doped crystal,* (Master thesis 2003) LRAP-298

5. Afzelius Mikael, *Theoretical modeling of temporal compression of optical pulses and pulse sequences using photon echoes,* (Master thesis 1999) LRAP-251

6. L. Mandel and E. Wolf, *Optical coherence and quantum optics,* The Press Sydicate of the University of Cambridge, USA (1995)

7. L. Allen and J.H. Eberly, *Optical resonance and two-level atoms,* John Wiley and Sons, Inc.,New York (1974)

8. Vladimir S. Malinovsky and Jeffrey L. Krause, *Efficiency and robustness of coherent population transfer with intense, chirped laser pulses*, University of Florida, Florida (2001)

9. Geoffrey W. Burr, Todd L. Harris, Wm. Randall Babbitt, C. Michael Jefferson, *Incorporating excitation-induced dephasing into the Maxwell–Bloch numerical modeling of photon echoes,* IBM Almaden Research Center, Montana state University, USA (2004)

10. Jeffrey L. Krause and Vladimir S. Malinovsky, *Selective Population Transfer Intense, Chirped Laser Pulses,* University of Florida, Florida

11. Matthew Robert Fetterman, *Ultrafast pulse shaping amplification, characterization and applications*, Princeton University Center of Ultrafast Laser Application, Princeton University, Princeton (1999)

12. http://www-ee.eng.hawaii.edu/~sasaki/EE693F/Spring03/Powerpoint/ONFiberB.PDF

13. http://hyperphysics.phy-astr.gsu.edu/hbase/hframe.html

Appendix I (The Effect of Time Only by Applying Gaussian/Square Pulses)

```
PRO C                                   ;The name of program
tmax=10000                              ;The number of steps of the time
tfinal=1.0                              ;The time in microsecond
dt = tfinal/tmax                        ;Define the value of Δt
pi=3.1416                               ;The constant π
del=8                                   ;The  resonance frequency
del=del*pi*2.0                          ;The resonance frequency in radian
pulse_shape=1.0                         ;By this commando we decide if it is a
                                        ;Gaussian or square pulses
TR=1.000                                ;The value of T1
TR2=1.0                                 ;The relaxation time
t_deph=0.59*tfinal                      ;The dephasing time
r1=fltarr(tmax)                         ;Create a matrix to store the data of r1
r2=fltarr(tmax)                         ;Create a matrix to store the data of r2
r3=fltarr(tmax)                         ;Create a matrix to store the data of r3
omega1=fltarr(tmax)                     ;Create a matrix to store the data of Ω
r1(1)=0.0                               ;Initialize Bloch vectors
r2(1)=0.0
r3(1)=-1.0
;**********************The sequence of square pulse*****************************
if (pulse_shape eq 1.0) then begin
Amp1=70                                 ;Rabi frequency inMHZ
```

```
Amp1=Amp1*2.0*pi                    ;Rabi frequency in radian
Amp2=140*0                          ;Rabi frequency of the second pulse
Amp2=Amp2*2*pi                      ;Second pulse in radian
t1=0.010*tfinal
t2=0.050*tfinal
t3=0.30*tfinal
t4=0.34*tfinal
t=0.0
for i=1,tmax-1 do begin
t=t+dt
if (t ge t1 and t le t2) then begin ;Initialize the first input pulse
a1=Amp1*(t2-t1)
omega1(i)=a1
endif
if (t ge t3 and t le t4) then begin ;Initialize the second input pulse
a2=Amp2*(t4-t3)
omega1(i)=a2
endif
endfor                              ;The end of the time loop
endif                               ;The end of the initializing square pulse
;**********************The sequence of the Gaussian pulses**********************
if (pulse_shape eq 0) then begin    ;The loop of the time
Amp1=7.0                            ;Rabi frequency of first pulse in MHZ
Amp1=Amp1*pi*2.0                    ;Rabi frequency in radial
Amp2=56*0                           ;Rabi frequency of second pulse
```

```
Amp2=Amp2*2.0*pi                                ;Rabi frequency in radial
t1i=0.010*tfinal
t1f=0.050*tfinal
t2i=0.21*tfinal
t2f=0.25*tfinal
tau1=0.2*(t1f-t1i)
tau2=0.2*(t2f-t2i)
t=0.0
for i=1,tmax-1 do begin
a1=Amp1*exp(-((t-((t1i+t1f)/2.0))/tau1)^2.0)     ;The first input pulse
a2=Amp2*exp(-((t-((t2i+t2f)/2.0))/tau2)^2.0)     ;The second input pulse
omega1(i)=a1+a2
t=t+dt
endfor                                          ;The end of the time loop
endif                                           ;The end of initializing Gaussain pulses
;*****************************The integration*****************************
for i=2,tmax-1 do begin                         ;The loop of the time
r1(i)=dt*(((del*r2(i-1))-(r1(i-1)/TR2)))+r1(i-1)  ;Maxwell-Bloch equations
r2(i)=dt*(((-del)*r1(i-1)+omega1(i)*r3(i-1)))-dt*(r2(i-1)/TR2)+r2(i-1)
r3(i)=dt*(((-omega1(i)*r2(i-1))-(r3(i-1)+1)/TR))+r3(i-1)
endfor                                          ;The end of the integration
;*************************The plotting of results*************************
window,0,xsize=420,ysize=320
plot,omega1,$                                   ;Plot the input pulses
title='input pulse',xtitle='time(microsecond)',ytitle='Omega(t)'
```

```
window,1,xsize=420,ysize=320
plot,r1,$                                   ;Plot the output data of r1
title='in-phase',xtitle='time(microsecond)', ytitle='r1'
window,2,xsize=420,ysize=320
plot,r2,$                                   ;Plot the output data of r2
title='in-quadrature',xtitle='time(microsecond)', ytitle='r2'
window,3,xsize=420,ysize=320
plot,r3,$                                   ;Plot the output data of r3
title='population inverse',xtitle='time(microsecond)', ytitle='r3'
end                                         ;The End of program
```

Appendix II

(The Effect of Time & Frequency by Applying Gaussian/Square Pulses)

```
pulse_shape=0.0;                    %The pulse shape
TR=1000;                            %The T1
TR2=10;                             %The relaxation time
alpha=500;                          %The constant α
pi=3.14159265;                      %The constant π
inversion=-1.0;
tmax=10000;                         %The number of the steps of the time
sta=alpha/(2*pi);
omega_index=310;
tfinal=1.0;                         %The time in μs
dt=(tfinal/tmax);                   %Δt
 del=100;                           %The resonance frequency Δ in MHZ
 del=del*2.0*pi;                    %The resonance frequency in Radial
 profile=1.0;
 procent=0.5;
 omega0=0.0;                        %The value of ω0
 domega=2*del/omega_index;          %dω
if (pulse_shape == 0.0)             %Initialize the input Gaussian pulses
Amp1=28;                            %Rabi frequency of first pulse in MHZ
Amp1=Amp1*2*pi;                     %Rabi frequency in radial
Amp2=50;                            %Rabi frequency of second pulse
Amp2=Amp2*2*pi;                     %Rabi frequency in radial
Amp3=28;                            %Rabi frequency of third pulse
Amp3=Amp3*2.0*pi;                   %Rabi frequency in Radial
t1i=0.01*tfinal;
 t1f=0.05*tfinal;
 t2i=0.21*tfinal;
 t2f=0.25*tfinal;
```

```
t3i=0.61*tfinal;
t3f=0.65*tfinal;
t_deph=0.59*tfinal;
t1v=(t1i+t1f)/2.0;
t2v=(t2i+t2f)/2.0;
t3v=(t3i+t3f)/2.0;
tau1=0.2*(t1f-t1i);
tau2=0.2*(t2f-t2i);
tau3=0.2*(t3f-t3i);
t=0;
for i=1:tmax                                %The loop of the time
a1=Amp1*exp(-(((t-t1v)/tau1).^2));          %The first input pulse
a2=Amp2*exp(-(((t-t2v)/tau2).^2));          %The second input pulse
a3=Amp3*exp(-(((t-t3v)/tau3).^2));          %The third input pulse
input_pulse1(i)=(a1+a2+a3);
t=t+dt;
if (abs(t-(t_deph)) < (5*dt))               %Time dephasing applied
   i_deph=i;
end
end                                         %The end of loop
end                                         %The end of initializing Gaussian
pulse
if (pulse_shape == 1)                       %The sequence of square pulses
   Amp1=280;                                %Rabi frequency of first pulse in MHZ
   Amp1=Amp1*(2.0*pi);                      %Rabi frequency in Radial
   Amp2=560;                                %Rabi frequency of second pulse
   Amp2=Amp2*(2.0*pi);
   Amp3=280*0;                              %Rabi frequency of third pulse
   Amp3=Amp3*(2.0*pi);
   t1=0.010*tfinal;
   t2=0.050*tfinal;
   t3=0.21*tfinal;
   t4=0.25*tfinal;
   t5=0.610*tfinal;
```

```
   t6=0.650*tfinal;

   t1v=(t1+t2)/2.0;
   t2v=(t3+t4)/2.0;
   t3v=(t5+t6)/2.0;
   t_deph=0.59*tfinal;
   t=0;

   for i=1:tmax                                  %The loop of the time
       input_pulse1(i)=0;
      if (t >= t1 & t <= t2)                     %Initialize the first input pulse
         a=Amp1*(t2-t1);
         input_pulse1(i)=(a);
      end
      if (t >= t3 & t <= t4)                     %Initialize the second input pulse
         a=Amp2*(t4-t3);
         input_pulse1(i)=(a);
      end
      if (t >= t5 & t <= t6)                     %initialize the third input pulse
         a=Amp3*(t4-t3);
         input_pulse1(i)=(a);
      end
        t=t+dt;
      if (abs(t-t_deph) < 5*dt)                  %Time diphasing applied
         i_deph=i;
      end

   end                                           %The end of the loop
end                                              %The end of initializing of square
pulse

for i=1:tmax
    time(i)=tfinal*i/tmax;
end
```

```
for i=1:tmax
   Omega1(i)=input_pulse1(i);
end
%********************The initializing of inhomogenous lineshape*****************
A=procent*omega_index;
C=(1.0-procent)*omega_index;
for k=1.0:omega_index
   if(profile == 0.0)
     G(k)=1.0;
   end
   if(profile == 1.0)
     B=pi*0.5*k/A;
     if(k >= 0.0 & k < A)
       G(k)=sin(B);
       G(k)=G(k)*G(k);
     end
     if(k >= A & k < C)
       G(k)=1.0;
     end
     if(k >= C)
       GR=sin(pi*0.5*(k-C)/A);
       G(k)=1.0-GR*GR;
     end
  end
end
%*********************The initializing of r1, r2 and r3*****************************

for k=1:omega_index

   r2(k)=0;
   r1(k)=0;
   r3(k)=-1;

   r11(k)=0;
```

```
    r22(k)=0;
    r33(k)=inversion;
end
%**********************The initializing of the average of r1, r2 and r3***************
for i=1:tmax
    u(i)=0;
    v(i)=0;
    w(i)=-1;
end
%*****************************The integration*********************************
t=0;
for i=1:tmax
     if(i == (i_deph))
        for M=1:omega_index
            r1(M)=0;
            r2(M)=0;
            r11(M)=0;
            r22(M)=0;
        end
    end
     omegas=-del;
     t=t+dt;
     for k=1:omega_index
         omegas=omegas+domega;
         roold1=r11(k);
       roold2=r22(k);
       roold3=r33(k);
       rold1=r1(k);
       rold2=r2(k);
       rold3=r3(k);

       rnew1=dt*((-(omegas-omega0)*rold2)-(rold1/TR2))+roold1;
       rnew2=dt*(((omegas-omega0)*rold1)+(Omega1(i)*rold3)-(rold2/TR2))+roold2;
       rnew3=dt*((-Omega1(i)*rold2)-((rold3+1)/TR))+roold3;
```

```
        r11(k)=r1(k);
        r22(k)=r2(k);
        r33(k)=r3(k);
        r1(k)=rnew1;
        r2(k)=rnew2;
        r3(k)=rnew3;
        if (k >= 155)
          u(i)=(u(i)+r1(k));
          v(i)=(v(i)+r2(k));
          w(i)=(w(i)+r3(k));
        end
    end
    int1=0;
    for k=2.0:omega_index-1
        int1=int1+((domega*G(k-1))*(r2(k-1.0)+r2(k+1.0)));
    end
    Omega1(i)=Omega1(i)+(int1*sta);
end
subplot(3,1,1)
plot(time,u/157)
subplot(3,1,2)
plot(time,v/157)
subplot(3,1,3)
plot(time,w/157)
```

Appendix III

(The Effect of Time, Space and Frequency by Applying Gaussian/Square Pulses)

```
pulse_shape=1.0;
TR2=1000;
TR=10;
alpha=500;
pi=3.14159265;
inversion=-1.0;
tmax=1000;
zmax=98;
omega_index=310;
tfinal=1.0;
dt=(tfinal/tmax);
zfinal=0.01;                                %The thickness of crystal
dz=zfinal/zmax;
del=100;
del=del*2.0*pi;
profile=1.0;
procent=0.5;
omega0=0.0;
domega=2*del/omega_index;
sta=alpha*dz/(2.0*pi);

%*********The initializing of Gaussian input pulses**********

if (pulse_shape == 0.0)

   Amp1=28;
   Amp1=Amp1*(2*pi);
   Amp2=50;
   Amp2=Amp2*(2*pi);
   Amp3=28;
   Amp3=Amp3*2.0*pi;

   t1i=0.01*tfinal;
   t1f=0.05*tfinal;

   t2i=0.21*tfinal;
   t2f=0.25*tfinal;

   t3i=0.61*tfinal;
   t3f=0.65*tfinal;

   t_deph=0.59*tfinal;

   tau1=0.2*(t1f-t1i);
   tau2=0.2*(t2f-t2i);
   tau3=0.2*(t3f-t3i);

   t=0.0;

   for i=1:tmax
```

```
        a1=Amp1*exp(-((t-((t1i+t1f)/2.0))/tau1)^2.0);
        a2=Amp2*exp(-((t-((t2i+t2f)/2.0))/tau2)^2.0);
        a3=Amp3*exp(-((t-((t3i+t3f)/2.0))/tau3)^2.0);

       input_pulse(i)=a1+a2+a3;
       t=t+dt;

        if (abs(t-(t_deph)) <= (5*dt))

           i_deph=i;
        end
    end
end

%**************The initializing of square pulses***************

if (pulse_shape == 1)

    Amp1=280;
    Amp1=Amp1*(2.0*pi);
    Amp2=560;
    Amp2=Amp2*(2.0*pi);
    Amp3=280;
    Amp3=Amp3*(2.0*pi);

    t1=0.010*tfinal;
    t2=0.050*tfinal;
    t3=0.210*tfinal;
    t4=0.250*tfinal;
    t5=0.610*tfinal;
    t6=0.650*tfinal;

    t_deph=0.59*tfinal;

    t=0;

    for i=1:tmax

        t=t+dt;
        input_pulse(i)=0;

        if (t >= t1 & t <= t2)

            input_pulse(i)=Amp1*(t2-t1);
        end

        if (t >= t3 & t <= t4)

            input_pulse(i)=Amp2*(t4-t3);
        end

        if  (t >= t5 & t <= t6)

            input_pulse(i)=Amp3*(t6-t5);
        end

        if (abs(t-t_deph) <= 5*dt)
```

```
            i_deph=i;
        end
    end
end
%*****************The initializing of input pulses*************

for i=1:tmax

    for j=1:zmax      %The including of effect of absorbing medium

        if (j == 1.0)

            Omega(i,j)=input_pulse(i);
        else
            Omega(i,j)=0.0;
        end
    end
end

%************initialize the inhomogoneous lineshape***********

A=procent*omega_index;
C=(1.0-procent)*omega_index;
for k=1:omega_index

    if(profile == 0.0)

        G(k)=1.0;
    end
    if(profile == 1.0)

        B=pi*0.5*k/A;

        if(k >= 0.0 & k < A)
            G(k)=sin(B);
            G(k)=G(k)*G(k);
        end
        if(k >= A & k < C)
            G(k)=1.0;
        end
        if(k >= C)
            GR=sin(pi*0.5*(k-C)/A);
            G(k)=1.0-GR*GR;
        end
    end
end

%*******************Inistialize Block vectors*****************
for i=1:tmax
    for j=1:zmax

        u(i,j)=0.0;
        v(i,j)=0.0;
        w(i,j)=0.0;
    end

end
```

```
   for j=1:zmax

       for k=1:omega_index

           r2(j,k)=0;
           r1(j,k)=0;
           r3(j,k)=-1;

           r11(j,k)=0;
           r22(j,k)=0;
           r33(j,k)=inversion;

       end
   end

%**********************The integration************************

for i=1:tmax

    if(i == (i_deph))

        for L=1:zmax
            for M=1:omega_index
                r1(L,M)=0.0;
                r2(L,M)=0.0;

                r11(L,M)=0;
                r22(L,M)=0;
            end
        end
    end

    for j=1:zmax-1

        omegas=omega0-del;

        for k=2:omega_index

            omegas=omegas+domega;

            roold1=r11(j,k);
            roold2=r22(j,k);
            roold3=r33(j,k);

            rold1=r1(j,k);
            rold2=r2(j,k);
            rold3=r3(j,k);

            rnew1=dt*((omega0-omegas)*rold2-(rold1/TR2))+roold1;
            rnew2=dt*((omegas-omega0)*rold1+(Omega(i,j)*rold3)-(rold2/TR2
))+roold2;
            rnew3=-1*dt*(Omega(i,j)*rold2-(rold3+1)/TR)+roold3;

            r11(j,k)=r1(j,k);
            r22(j,k)=r2(j,k);
            r33(j,k)=r3(j,k);

            r1(j,k)=rnew1;
```

```
            r2(j,k)=rnew2;
            r3(j,k)=rnew3;

            if (k >= 155)
                u(i,j)=u(i,j)+r1(j,k);  %The sum of r1 over all frequencies
                v(i,j)=v(i,j)+r2(j,k);  %The sum of r2 over all frequencies
                w(i,j)=w(i,j)+r3(j,k);  %The sum of r3 over all frequencies

            end
        end

        %**********The direct integration of Ω(t,z)*************

        int=0;

        for k=2.0:omega_index-1.0

            int=int+domega*0.5*G(k)*(r2(j,k-1.0)+r2(j,k+1.0));
        end

        Omega(i,j+1.0)=Omega(i,j)+(int*sta);
    end
end

%*************The plotting of the output data*****************

subplot(2,2,1)
mesh(u/70)
subplot(2,2,2)
mesh(v/50)
subplot(2,2,3)
mesh(w/156)
subplot(2,2,4)
mesh(Omega/200)
```

Appendix IV
(The Effect of Time & Frequency by Applying Frequency-Chirped Pulses)

```
pulse_shape=0.0;
TR=1000;
TR2=10;
alpha=500;
pi=3.14159265;
inversion=-1.0;
tmax=10000;
sta=alpha/(2*pi);
omega_index=310;
tfinal=4.0;
dt=(tfinal/tmax);
del=100;
del=del*2.0*pi;
profile=1.0;
procent=0.5;
omega0=0.0;
domega=2*del/omega_index;


%********Initialize Gaussian frequency-chirped pulses*********

if (pulse_shape == 0.0)

 Amp1=50;
 Amp1=Amp1*2*pi;
 Amp2=50;
 Amp2=Amp2*2*pi;
 Amp3=28*0;
 Amp3=Amp3*2.0*pi;

 t1i=0.01*tfinal;
 t1f=0.19*tfinal;

 t2i=0.21*tfinal;
 t2f=0.30*tfinal;

 t3i=0.61*tfinal;
 t3f=0.65*tfinal;

 t_deph=0.59*tfinal;

 t1v=(t1i+t1f)/2.0;
 t2v=(t2i+t2f)/2.0;
 t3v=(t3i+t3f)/2.0;

 tau1=0.2*(t1f-t1i);
 tau2=0.2*(t2f-t2i);
 tau3=1.414*(t3f-t3i);
 t=0;
 kb=2*pi*400;                                %The chirping rate
 x=-j*kb;
```

```
 for i=1:tmax

     a1=Amp1*exp(-(((t-t1v)/tau1).^2))*exp(x*((t-t1v).^2));
     a2=Amp2*exp(-(((t-t2v)/tau2).^2))*exp(2*x*((t-t2v).^2));
     a3=Amp3*exp(-(((t-t3v)/tau3).^2))*exp(x*((t-t3v).^2));

     input_pulse1(i)=real(a1)+real(a2)+real(a3);  %The real part of pulses
     input_pulse2(i)=imag(a1)+imag(a2)+imag(a3);  %The imag part of pulses

     t=t+dt;

     if (abs(t-(t_deph)) < (5*dt))
         i_deph=i;
     end
 end
end

%********Initialize square frequency-chirped pulses*********

if (pulse_shape == 1)
    Amp1=400;
    Amp1=Amp1*(2.0*pi);
    Amp2=400;
    Amp2=Amp2*(2.0*pi);

    t1=0.010*tfinal;
    t2=0.190*tfinal;
    t3=0.29*tfinal;
    t4=0.38*tfinal;
    t1v=(t1+t2)/2.0;
    t2v=(t3+t4)/2.0;

    t_deph=0.59*tfinal;
    t=0;
    kb=2*pi*200;                                %The chirping rate
    x=-j*kb

    for i=1:tmax

        input_pulse1(i)=0;
        input_pulse2(i)=0;

        if (t >= t1 & t <= t2)
            a1=Amp1*exp(x*((t-t1v).^2));
            input_pulse1(i)=real(a1);
            input_pulse2(i)=imag(a1);
        end
        if (t >= t3 & t <= t4)
            a2=Amp2*exp(2*x*((t-t2v).^2));
            input_pulse1(i)=real(a2);
            input_pulse2(i)=imag(a2);
        end

        t=t+dt;
        if (abs(t-t_deph) < 5*dt)
           i_deph=i;
        end
```

```
    end
end



for i=1:tmax
    Omega1(i)=input_pulse1(i);   %Store the real part of pulses
    Omega2(i)=input_pulse2(i);   %Store the real part of pulses
End

%************initialize the inhomogoneous lineshape***********

A=procent*omega_index;
C=(1.0-procent)*omega_index;
for k=1.0:omega_index
    if(profile == 0.0)
       G(k)=1.0;
    end
    if(profile == 1.0)
       B=pi*0.5*k/A;
       if(k >= 0.0 & k < A)
          G(k)=sin(B);
          G(k)=G(k)*G(k);
       end
       if(k >= A & k < C)
          G(k)=1.0;
       end
       if(k >= C)
          GR=sin(pi*0.5*(k-C)/A);
          G(k)=1.0-GR*GR;
      end
  end
end

%******************initialize Bloch vectors*********************

for k=1:omega_index

    r2(k)=0;
    r1(k)=0;
    r3(k)=-1;

    r11(k)=0;
    r22(k)=0;
    r33(k)=inversion;
end

%************************The integration************************

t=0;
for i=1:tmax

    if(i == (i_deph))


     for M=1:omega_index
```

```
            r1(M)=0;
            r2(M)=0;

            r11(M)=0;
            r22(M)=0;

        end
    end

    omegas=-del;

   t=t+dt;

    for k=1:omega_index

        omegas=omegas+domega;    %Detuned frequency from -Δ to +Δ

        roold1=r11(k);
        roold2=r22(k);
        roold3=r33(k);

        rold1=r1(k);
        rold2=r2(k);
        rold3=r3(k);


%The Maxwell-Bloch equations that contain the imaginary part

rnew1=dt*((-(omegas-omega0)*rold2)-(Omega2(i)*rold3)-(rold1/TR2))+roold1;
rnew2=dt*(((omegas-omega0)*rold1)+(Omega1(i)*rold3)-(rold2/TR2))+roold2;
rnew3=dt*((-Omega1(i)*rold2)+(Omega2(i)*rold1)-((rold3+1)/TR))+roold3;

        r11(k)=r1(k);
        r22(k)=r2(k);
        r33(k)=r3(k);

        r1(k)=rnew1;
        r2(k)=rnew2;
        r3(k)=rnew3;

    end

%************The direct integration of Ω_r(t) & Ω_i(t)************

    int1=0;
    int2=0;

    for k=2.0:omega_index-1

        int1=int1+((domega*G(k-1))*(r2(k-1.0)+r2(k+1.0)));
        int2=int2-((domega*G(k-1))*(r1(k-1.0)+r1(k+1.0)));
    end

    Omega1(i)=Omega1(i)+(int1*sta);
    Omega2(i)=Omega2(i)+(int2*sta);
 End
Plot ((Omega1.^2+Omega2.^2).^0.5)
```

Appendix V
(The Effect of Time, Space & Frequency by Applying Frequency-Chirped Pulses)

```
pulse_shape=1;
TR=10000;
TR2=10;
alpha=300;
pi=3.14159265;
inversion=-1.0;
tmax=1000;
zfinal=0.01;                                %The thickness of crystal
zmax=98;
dz=zfinal/zmax;
sta=alpha*dz/(2*pi);
omega_index=310;
tfinal=2;
dt=(tfinal/tmax);
del=100;
del=del*2.0*pi;
profile=1.0;
procent=0.5;
omega0=0.0;
domega=2*del/omega_index;

%******Initializing of Gaussian frequency-chirped pulses******
if (pulse_shape == 0.0)

 Amp1=20;
 Amp1=Amp1*2*pi;
 Amp2=20;
 Amp2=Amp2*2*pi;

 t1i=0.01*tfinal;
 t1f=0.19*tfinal;

 t2i=0.29*tfinal;
 t2f=0.38*tfinal;

 t_deph=0.69*tfinal;

 t1v=(t1i+t1f)/2.0;
 t2v=(t2i+t2f)/2.0;

 tau1=0.2*(t1f-t1i);
 tau2=0.2*(t2f-t2i);
 t=0;
 kb=2*pi*800;                               %The chirping rate in radial
 x=-j*kb;

 for i=1:tmax

     a1=Amp1*exp(-(((t-t1v)/tau1).^2))*exp(x*((t-t1v).^2));
     a2=Amp2*exp(-(((t-t2v)/tau2).^2))*exp(2*x*((t-t2v).^2));

     input_pulse1(i)=real(a1)+real(a2);
```

```
      input_pulse2(i)=imag(a1)+imag(a2);

      t=t+dt;

      if (abs(t-(t_deph)) < (5*dt))
          i_deph=i;
      end
 end
end

%**********Initialize square frequency-chirped pulses*********

if (pulse_shape == 1)

    Amp1=20;
    Amp1=Amp1*(2.0*pi);
    Amp2=20;
    Amp2=Amp2*(2.0*pi);

    t1=0.010*tfinal;
    t2=0.190*tfinal;
    t3=0.29*tfinal;
    t4=0.38*tfinal;

    t1v=(t1+t2)/2.0;
    t2v=(t3+t4)/2.0;

    t_deph=0.69*tfinal;
    t=0;
    kb=2*pi*300;
    x=-j*kb

    for i=1:tmax



        input_pulse1(i)=0;
        input_pulse2(i)=0;

        if (t >= t1 & t <= t2)
            a1=Amp1*exp(x*((t-t1v).^2));
            input_pulse1(i)=real(a1);
            input_pulse2(i)=imag(a1);
        end
        if (t >= t3 & t <= t4)
            a2=Amp2*exp(2*x*((t-t2v).^2));
            input_pulse1(i)=real(a2);
            input_pulse2(i)=imag(a2);
        end

        t=t+dt;
        if (abs(t-t_deph) < 5*dt)
           i_deph=i;
        end

    end
end
```

```
%This part is to initialize the input pulses in time and space
%dimension

for i=1:tmax
    for j=1:zmax
        if (j==1)

            Omega1(i,j)=input_pulse1(i);
            Omega2(i,j)=input_pulse2(i);
        else
            Omega1(i,j)=0;
            Omega2(i,j)=0;
        end
    end
end
%**********The initializing of inhomogenous lineshape*********

A=procent*omega_index;
C=(1.0-procent)*omega_index;
for k=1.0:omega_index
    if(profile == 0.0)
       G(k)=1.0;
    end
    if(profile == 1.0)
       B=pi*0.5*k/A;
       if(k >= 0.0 & k < A)
          G(k)=sin(B);
          G(k)=G(k)*G(k);
       end
       if(k >= A & k < C)
          G(k)=1.0;
       end
       if(k >= C)
          GR=sin(pi*0.5*(k-C)/A);
          G(k)=1.0-GR*GR;
       end
  end
end

for j=1:zmax
    for k=1:omega_index

        r2(j,k)=0;
        r1(j,k)=0;
        r3(j,k)=-1;

        r11(j,k)=0;
        r22(j,k)=0;
        r33(j,k)=inversion;
    end
end

%************************The integration**********************
t=0;
for i=1:tmax
```

```
if(i == (i_deph))


 for M=1:zmax
     for L=1:omega_index

         r1(M,L)=0;
         r2(M,L)=0;

         r11(M,L)=0;
         r22(M,L)=0;
     end
 end
end
for j=1:zmax-1                %The effect of absorbing medium included

    omegas=-del;

    for k=1:omega_index

        omegas=omegas+domega;

        roold1=r11(j,k);
        roold2=r22(j,k);
        roold3=r33(j,k);

        rold1=r1(j,k);
        rold2=r2(j,k);
        rold3=r3(j,k);

%****Maxwell-Bloch equations including the imaginary part*****

rnew1=dt*((-(omegas-omega0)*rold2)-(Omega2(i,j)*rold3)-(rold1/TR2))+roold1;
rnew2=dt*(((omegas-omega0)*rold1)+(Omega1(i,j)*rold3)-(rold2/TR2))+roold2;
rnew3=dt*((-Omega1(i,j)*rold2)+(Omega2(i,j)*rold1)-((rold3+1)/TR))+roold3;

        r11(j,k)=r1(j,k);
        r22(j,k)=r2(j,k);
        r33(j,k)=r3(j,k);

        r1(j,k)=rnew1;
        r2(j,k)=rnew2;
        r3(j,k)=rnew3;
    end
%**********The direct integration of Ω_r(t,z) & Ω_i(t,z)**********

    int1=0;
    int2=0;

    for k=2.0:omega_index-1

        int1=int1+((domega*0.5*G(k-1))*(r2(j,k-1)+r2(j,k+1)));
        int2=int2-((domega*0.5*G(k-1))*(r1(j,k-1)+r1(j,k+1)));
    end
   Omega1(i,j+1)=Omega1(i,j)+(int1*sta);
   Omega2(i,j+1)=Omega2(i,j)+(int2*sta);
```

```
    end
end

mesh((Omega1.^2+Omega2.^2).^0.5)
```

Appendix VI
(The Effect of Time & Frequency by Applying Complex Hyperbolic Secant Pulse)

```
TR=1000;
TR2=10;
alpha=100;
pi=3.14159265;
inversion=-1.0;
tmax=1000;
omega_index=310;
tfinal=3.0;
dt=(tfinal/tmax);
del=100;
del=del*2.0*pi;
profile=1.0;
procent=0.5;
omega0=0.0;
domega=2*del/omega_index;
alpha=100;
sta=alpha/2*pi;

%*********Initialize complex hyperbolic secant pulses*********
mi=1;                                          %The real constant
x=1+j*mi
Amp1=28;
Amp1=Amp1*(2*pi);
Amp2=28;
Amp2=Amp2*(2*pi);
Amp3=28;
Amp3=Amp3*2.0*pi;
t1i=0.01*tfinal;
t1f=0.05*tfinal;
t2i=0.21*tfinal;
t2f=0.25*tfinal;
t3i=0.61*tfinal;
t3f=0.65*tfinal;
t1v=(t1i+t1f)/2;
t2v=(t2i+t2f)/2;
t3v=(t3i+t3f)/2
t_deph=0.59*tfinal;
tau1=0.2*(t1f-t1i);
tau2=0.2*(t2f-t2i);
tau3=0.2*(t3f-t3i);
t=0.0;
beta1=2.6/tau1;
beta2=2.6/tau2;
beta3=2.6/tau3;

for i=1:tmax

     a11=(cosh(beta1*(t-t1v)))^x;
     a22=(cosh(beta2*(t-t2v)))^x;
     a33=(cosh(beta3*(t-t3v)))^x;

     a1(i)=Amp1*(1/a11);
     a2(i)=Amp2*(1/a22);
     a3(i)=Amp3*(1/a33);
```

```
      input_pulse(i)=a1(i)+a2(i)+a3(i);
      input_pulse1(i)=real(input_pulse(i));
      input_pulse2(i)=imag(input_pulse(i));


      t=t+dt;
      if (abs(t-(t_deph)) <= (5*dt))
          i_deph=i;
      end
 end
```

%Initialize the real and imaginary parts of input pulses $\Omega_r(t)$ & %$\Omega_i(t)$ respectively**********

```
 for i=1.0:tmax
      Omega1(i)=input_pulse1(i);
      Omega2(i)=input_pulse2(i);
 end

%**********The sequence of inhomogoneous lineshape**********

A=procent*omega_index;
C=(1.0-procent)*omega_index;

for k=1.0:omega_index
    if(profile == 0.0)
        G(k)=1.0;
    end
    if(profile == 1.0)
        B=pi*0.5*k/A;
        if(k >= 0.0 & k < A)
            G(k)=sin(B);
            G(k)=G(k)*G(k);
        end
        if(k >= A & k < C)
            G(k)=1.0;
        end
        if(k >= C)
            GR=sin(pi*0.5*(k-C)/A);
            G(k)=1.0-GR*GR;
        end
    end
end

 for i=1.0:tmax
      u(i)=0.0;
      v(i)=0.0;
      w(i)=1.0;
 end

 for k=1:omega_index
      r2(k)=0;
      r1(k)=0;
      r3(k)=-1;

      r11(k)=0;
      r22(k)=0;
```

```
    r33(k)=inversion;
end

%************************The integration************************

for i=1:tmax

    if(i == (i_deph))

        for M=1:omega_index

            r1(M)=0.0;
            r2(M)=0.0;

            r11(M)=0;
            r22(M)=0;

        end
    end

    omegas=omega0-del;

    for k=2:omega_index

        omegas=omegas+domega;

        roold1=r11(k);
        roold2=r22(k);
        roold3=r33(k);

        rold1=r1(k);
        rold2=r2(k);
        rold3=r3(k);

rnew1=dt*(((omega0-omegas))*rold2)-(Omega2(i)*rold3)-(rold1/TR2))+roold1;
rnew2=dt*((((omegas-omega0))*rold1)+(Omega1(i)*rold3)-(rold2/TR2))+roold2;
rnew3=dt*((-Omega1(i)*rold2)+(Omega2(i)*rold1)-((rold3+1)/TR))+roold3;

        r11(k)=r1(k);
        r22(k)=r2(k);
        r33(k)=r3(k);

        r1(k)=rnew1;
        r2(k)=rnew2;
        r3(k)=rnew3;

        if (k >= 155)

            u(i)=u(i)+r1(k);
            v(i)=v(i)+r2(k);
            w(i)=w(i)+r3(k);

        end
    end

    int1=0;
    int2=0;
```

```
    for k=2.0:omega_index-2.0

        int1=int1+domega*0.5*G(k)*(r2(k-1.0)+r2(k+1.0));
        int2=int2+domega*0.5*G(k)*(r1(k-1.0)+r1(k+1.0));
    end

      Omega1(i)=Omega1(i)+(int1*sta);
      Omega2(i)=Omega2(i)+(-int2*sta);

end
```

Appendix VII
(The Effect of Time, Space & Frequency by Applying Complex Hyperbolic Secant Pulse)

```
TR=1000;
TR2=10;
alpha=500;
pi=3.14159265;
inversion=-1.0;
tmax=1000;
zmax=98;
omega_index=310;
tfinal=1.0;
dt=(tfinal/tmax);
zfinal=0.01;
dz=zfinal/zmax;
del=100;
del=del*2.0*pi;
profile=1.0;
procent=0.5;
omega0=0.0;
domega=2*del/omega_index;
sta=alpha*dz/(2.0*pi);
mi=3;
x=1+j*mi;

%******Initialize the complex secant hyperbolic pulses *******

Amp1=28;
Amp1=Amp1*(2*pi);
Amp2=28;
Amp2=Amp2*(2*pi);
Amp3=28;
Amp3=Amp3*2.0*pi;

t1i=0.01*tfinal;
t1f=0.05*tfinal;
t2i=0.21*tfinal;
t2f=0.25*tfinal;
t3i=0.61*tfinal;
t3f=0.65*tfinal;

t1v=(t1i+t1f)/2;
t2v=(t2i+t2f)/2;
t3v=(t3i+t3f)/2;

t_deph=0.59*tfinal;

tau1=0.2*(t1f-t1i);
tau2=0.2*(t2f-t2i);
tau3=0.2*(t3f-t3i);
t=0.0;
beta1=2.6/tau1;
beta2=2.6/tau2;
beta3=2.6/tau3;

for i=1:tmax
```

```
    a11=(cosh(beta1*(t-t1v)))^x;
    a22=(cosh(beta2*(t-t2v)))^x;
    a33=(cosh(beta3*(t-t3v)))^x;

    a1(i)=Amp1*(1/a11);
    a2(i)=Amp2*(1/a22);
    a3(i)=Amp3*(1/a33);

    input_pulse(i)=a1(i)+a2(i)+a3(i);
    input_pulse1(i)=real(input_pulse(i));
    input_pulse2(i)=imag(input_pulse(i));


    t=t+dt;
    if (abs(t-(t_deph)) <= (5*dt))
        i_deph=i;
    end
 end

%***Initialize the input pulse in time and space dimension ***

 for i=1.0:tmax

     for j=1:zmax
         if (j == 1.0)
            Omega1(i,j)=input_pulse1(i);
            Omega2(i,j)=input_pulse2(i);
        else
            Omega1(i,j)=0.0;
            Omega2(i,j)=0.0;
        end
    end
 end

%***********The sequence of inhomogoneous lineshape***********

A=procent*omega_index;
C=(1.0-procent)*omega_index;
for k=1.0:omega_index
    if(profile == 0.0)
        G(k)=1.0;
    end
    if(profile == 1.0)
        B=pi*0.5*k/A;
        if(k >= 0.0 & k < A)
            G(k)=sin(B);
            G(k)=G(k)*G(k);
        end
        if(k >= A & k < C)
            G(k)=1.0;
        end
        if(k >= C)
            GR=sin(pi*0.5*(k-C)/A);
            G(k)=1.0-GR*GR;
        end
    end
end
```

```
%***********************The integration************************

  for j=1:zmax
   for k=1:omega_index

        r2(j,k)=0;
        r1(j,k)=0;
        r3(j,k)=-1;

        r11(j,k)=0;
        r22(j,k)=0;
        r33(j,k)=inversion;

    end
end

for i=1:tmax

    if(i == (i_deph))

       for L=1:zmax
           for M=1:omega_index

           r1(L,M)=0.0;
           r2(L,M)=0.0;


           r11(L,M)=0;
           r22(L,M)=0;
       end
   end
 end

 for j=1:zmax-2.0
        omegas=omega0-del;
       for k=2:omega_index-1.0
           omegas=omegas+domega;

           roold1=r11(j,k);
           roold2=r22(j,k);
           roold3=r33(j,k);
           rold1=r1(j,k);
           rold2=r2(j,k);
           rold3=r3(j,k);
rnew1=dt*(omega0-omegas)*rold2+(dt*Omega2(i,j)*rold3)-(dt*rold1/TR2)+
roold1;
rnew2=dt*((omegas-omega0)*rold1+(Omega1(i,j)*rold3))-(dt*rold2/TR2)+roold2;
rnew3=(-dt*Omega1(i,j)*rold2)-(Omega2(i,j)*rold1*dt)-(dt*(rold3+1)/TR)+
roold3;
r11(j,k)=r1(j,k);
r22(j,k)=r2(j,k);
r33(j,k)=r3(j,k);
r1(j,k)=rnew1;
r2(j,k)=rnew2;
r3(j,k)=rnew3;

w(i,k)=r3(j,k);

end
```

```
%**********The direct integration of Ω_r(t,z) & Ω_i(t,z) **********

    int1=0;
    int2=0;
       for k=2.0:omega_index-2.0

           int1=int1+domega*0.5*G(k)*(r2(j,k-1.0)+r2(j,k+1.0));
           int2=int2+domega*0.5*G(k)*(r1(j,k-1.0)+r1(j,k+1.0));

       end

       Omega1(i,j+1.0)=Omega1(i,j)+(int1*sta);
       Omega2(i,j+1.0)=Omega2(i,j)+(-int2*sta);


   end

end
```