Solving Maxwell-Schrodinger Equations to Simulate a New Lasing Scheme

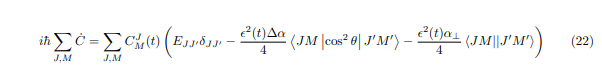
文献



**第一部分**

根据文献内容，运行提供的代码，获得原来无修正项的理论模型。

代码基于下面方程，具体情况见文献



得出的理论模型和实验对比如下：

图表, 直方图

描述已自动生成

提供了原有的代码和注释。

**第二部分**

在第一部分的基础上引入一个Dipole self-interaction correction，变为解下面方程：

图形用户界面, 文本, 应用程序, Word

描述已自动生成

并未有提供理论模型的图像和代码，实验数据和图表仍是之前的。

**第一部分提供的代码和额外注释信息如下**：

**python代码：**



**Python代码额外注释：**

Major changes in the code:

I use a dictionary rather than an input file

molecule\_dictionary = {

    'molecule\_name': 'N2',

    'T':10,                # Temperature K

    'Jmax': 30,            # Max J value

    'B':1.9895,             # B cm-1

    'centrifugal':0.0,                # Centrifugal distorsion

    'alphaper':9.8,         # Bohr^3

    'alphapar':15,          # Bohr^3

    'reneven':0.6667,         # g even J

    'renodd':0.3333,          # g off J

    'tau': 50,              # first pulse duration in fs

    'I': 3,                 # peak intensity in TW/cm2

    'tau2': 50,             # pulse duration in fs

    'I02': 3,                # pulse 2 peak intensity in TW/cm2

    'Tfree1': 9000,              # evolution after first pulse  in fs

    'Tfree2': 10,              # evolution after second pulse  in fs

    'tau3': 10,                # pulse duration of third pulse in fs

    'Tfree3': 10,              # evolution after third pulse in fs

    'I03': 0.0,                 # Intesity of third pulse (W/cm2)

    'NOPlaser': 25,            # field points in fs

    'stepfree': 20             # step size in fs

    }

This is then fed into the main function

cos2thetaaverage,D2,D4,D6,D8,time,Eout,symm = alignlinearD(molecule\_dictionary)

**Addition of local polarizabilities ：**

% To add the non-local polarizabilities, we need to add the cos^4 which is

% already programmed in alignlinearmultipulses.m. Matrix elements are built

% in Vprime.m

% In addition, we need to add a correction to the weights of the new

% elements, that is a term that is proportional to lambda^2 where lambda is

% the non-local perturbation parameter

% The weights are added in the VXX parameters. V03 is a new parameter that

% needs to be added

% V00(1,:)=-0.25\*alphaper\*E1.^2;

% V02(1,:)=-0.25\*E1.^2\*(alphapar-alphaper);

% V00(1,:)=-0.25\*E1.^2\*(1-lambda\_nonlocal^2\*alphaper)\*alphaper;

% V02(1,:)=-0.25\*E1.^2\*(alphapar-alphaper)\*(1-2\*lambda\_nonlocal^2\*alphaper);

% V03(1,:)=0.25\*E1.^2\*(alphapar-alphaper)^2\*lambda\_nonlocal^2

Addition of V03 was done in alignlinearD\_CT, alignlinearmultipulse and fieldode

**Matlab代码：**



**Python代码额外注释：**

The code and the example input file are in 'J:\ctgroup\Zhanna\Molecular alignment from Varun\Linear Molecule Alignment Code'. Use the 'alignlinearD.m' function. Example input file is 'input\_co2\_orig\_Varun.txt'

Function 'alignlinearD.m' can simulate a rotational wavepacket for two pulses. Input variables are 'input\_textfile\_name', tau, I,tau2 and I2. I, tau, I2 and tau2 are intensity (I) and duration for each of these pulses.

In the 'input\_co2\_orig\_Varun.txt' file, Tfree1, Tfree2, Tfree3 and tau3 and I03 are needed to apply additional alignment pulses. Up to three can be applied, and tau3 and I03 are the duration and intensity of the third pulse. Each ‘Tfree’ is the field free propagation time after each pulse. For one pulse, Trfee1 will be the entire time window of the calculation, Tfree2 and Tfree3 are set to be very short.

For my simulations, I use Tfree1=9000 fs (for N2, 80000 for CO2, depends on the revival time of the molecule), Tfree2=10 fs, Tfree3=10 fs,  tau3=10 fs and I03=0.0 W/cm2.