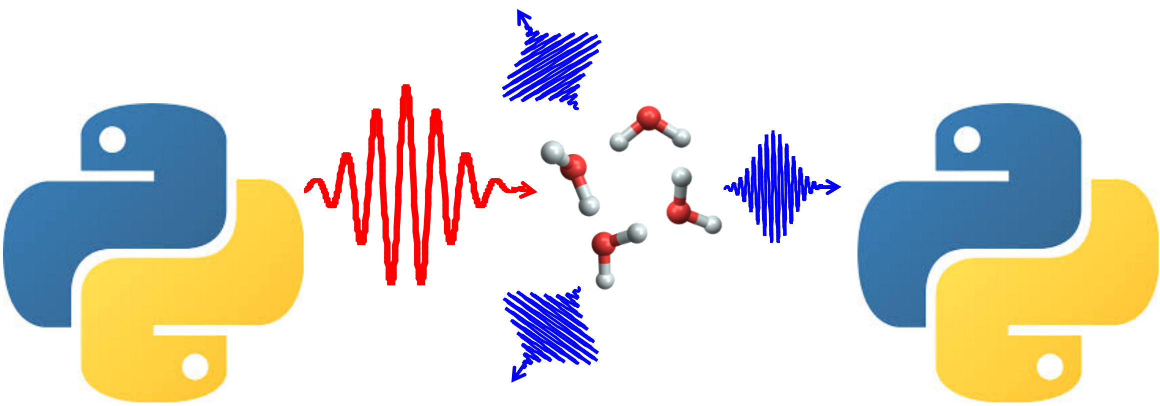
PySHS : a Python Computational Second Harmonic Scattering Software



User Manual 1.2 Written by:

Pierre-Marie Gassin

This user manual is distributed along with the pySHS software to aid in setting up the input files required for carrying out a Second Harmonic Scattering simulation. Every effort is made to release the most updated and complete version of the manual. To report any inconsistencies, errors or missing information, or to suggest improvements, send email to pierre-marie.gassin@enscm.fr.

Acknowledgements

Support for this work was provided by a grant from CAMOMILS Research National French Agency : ANR-15-CE21-0002.

We would like to acknowledge the Prof. Pierre-François Brevet for the numerous stimulating discussions about Second Harmonic Scattering and Dr Julien Duboisset for its HRS_computing program which can be seen as a precursor to PySHS.

The people who have contributed to the development of the code include:

- Dr Pierre-Marie Gassin
- Dr Lotfi Boudjema
- Dr Gaelle Gassin

Contents

Chapter1

Introduction

1) What is PySHS?

PySHS is a collection of different program written in python which can compute the Second Harmonic Intensity Scattered by different class of objects:

- incoherent molecules in solution (see program HRS_polar_90.py , HRS_ polar_180.py, HRS_angle.py)
- supramolecular structures formed by correlated molecules (see program SHS_polar_90.py , SHS_ polar_180.py, SHS_angle.py, SHS_polar_linear_90.py)
- colloidal object like sphere. (other kind of object including cylinder, cuboid are expected for the next version)

2) Distribution

PySHS has been written in python 3.7 version. So, firstly Python has to be installed with its classical dependencies (numpy-pyplot). See https://www.python.org for any question about python installation. After unpacking the pySHS distribution, the parent PySHS directory will contain the following subdirectories:

- Documentation

It contains the user_guide. You may find most answers to your questions here.

- Examples

Contains some examples of calculation and in particular the input and output files.

- Src

Contains the pySHS source code and some script used to generate the input files.

- Work

It is the directory where the user can store the input and output file.

Chapter 2

General Equations Implemented in PySHS

1) The Notation

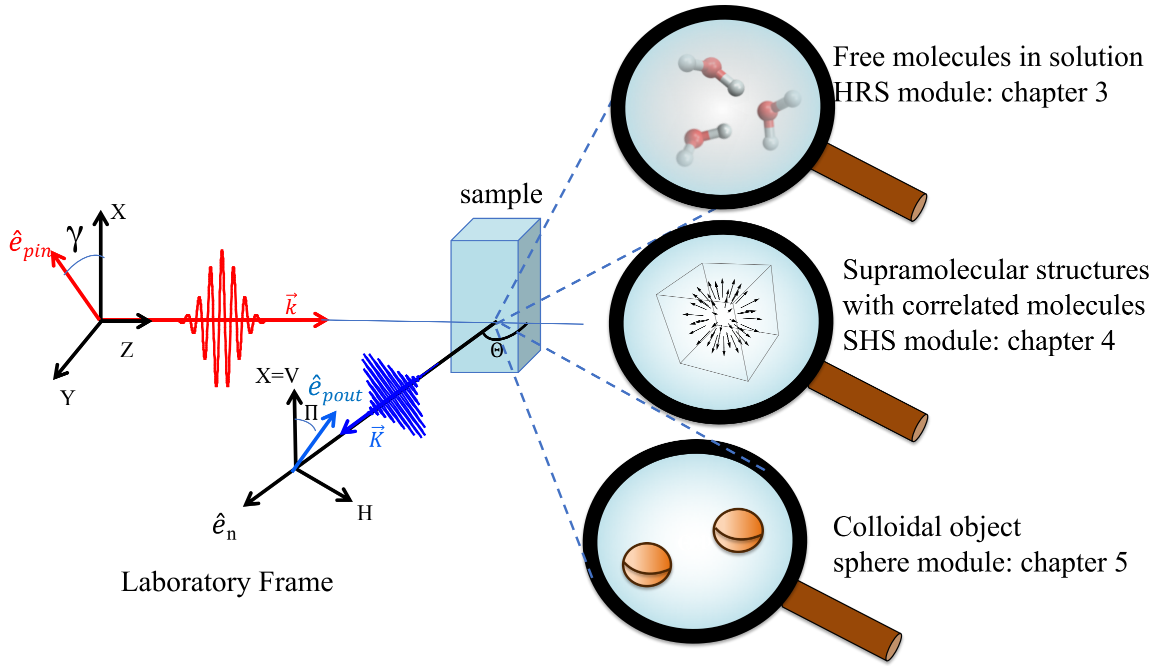


Figure 1: definition and notation

The fundamental light at the ω frequency is defined by its wave vector $\vec{k} = k\hat{e}_z$ and its polarization direction $\hat{e}_{p-in}$. The input polarization angle γ is equal to the angle between $(\hat{e}_X, \hat{e}_{p-in})$

The second harmonic scattered light at the 2ω frequency is defined by its wave vector $\vec{K} = K\hat{e}_n$ and the angle $(\hat{e}_z, \hat{e}_n)$ defines the scattered angle Θ. The polarization of the second harmonic scattered light is defined by $\hat{e}_{p-out}$ and the output polarization angle Π is equal to the angle between $(\hat{e}_X, \hat{e}_{p-out})$. In the laboratory frame, we have the following relation:

$$\hat{e}_n = \begin{pmatrix} 0 \\ \sin(\Theta) \\ \cos(\Theta) \end{pmatrix} \qquad \hat{e}_{p-in} = \begin{pmatrix} \cos(\gamma) \\ \sin(\gamma) \\ 0 \end{pmatrix} \qquad \hat{e}_{p-out} = \begin{pmatrix} \cos(\Pi) \\ \sin(\Pi)\cos(\Theta) \\ \sin(\Pi)\sin(\Theta) \end{pmatrix} \qquad \text{(1-a,b,c)}$$

The scattered wave vector $\overrightarrow{\Delta k}$ is defined as follow: $\overrightarrow{\Delta k}= 2\vec{k}\text{-}\vec{K}=\begin{pmatrix} 0 \\ -K\sin(\Theta) \\ 2\mathrm{k}-\mathrm{K}\cos(\Theta) \end{pmatrix}$ (2)

2) Expression of the SHS Intensity

2.1) Incoherent Second Harmonic Scattering

The Second Harmonic Scattering of uncorrelated molecule in solution is usually referred as Hyper Rayleigh Scattering (HRS) and can be expressed as:

$$I_{HRS}\left(\hat{e}_{p-in}, \hat{e}_{p-out}, \hat{e}_n\right) = I_{HRS}(\gamma, \Pi, \Theta) = \langle \vec{\beta}_{eff}(\gamma, \Pi, \Theta).\vec{\beta}^*_{eff}(\gamma, \Pi, \Theta)\rangle \qquad (3)$$

Here $\vec{\beta}_{eff} = (\hat{e}_n \times \overleftrightarrow{\beta}: \hat{e}_p\hat{e}_p \times \hat{e}_n)$ and $\overleftrightarrow{\beta}$ is the second order hyperpolarisability of the molecule expressed in the laboratory frame. The link between the expression of $\overleftrightarrow{\beta}$ in the laboratory and microscopic frames is:

$$\beta_{IJK,Labo}(\varphi, \theta, \psi) = \sum_i \sum_j \sum_k T_{Ii}(\varphi, \theta, \psi)T_{Jj}(\varphi, \theta, \psi)T_{Kk}(\varphi, \theta, \psi)\, \beta_{ijk,microscopic} \qquad (4)$$

The angle φ, θ, ψ are defined in the figure 2 and the transformation matrix $T(\varphi, \theta, \psi)$ is:

$$T(\varphi, \theta, \psi) = \begin{pmatrix} \cos(\psi)\cos(\varphi) - \cos(\theta)\sin(\varphi)\sin(\psi) & \cos(\theta)\cos(\psi)\sin(\varphi) + \cos(\varphi)\sin(\psi) & \sin(\theta)\sin(\varphi) \\ -\sin(\varphi)\cos(\psi) - \cos(\theta)\cos(\varphi)\sin(\psi) & \cos(\theta)\cos(\psi)\cos(\varphi) - \sin(\varphi)\sin(\psi) & \sin(\theta)\cos(\varphi) \\ \sin(\theta)\sin(\psi) & -\sin(\theta)\cos(\psi) & \cos(\theta) \end{pmatrix} \qquad (5)$$

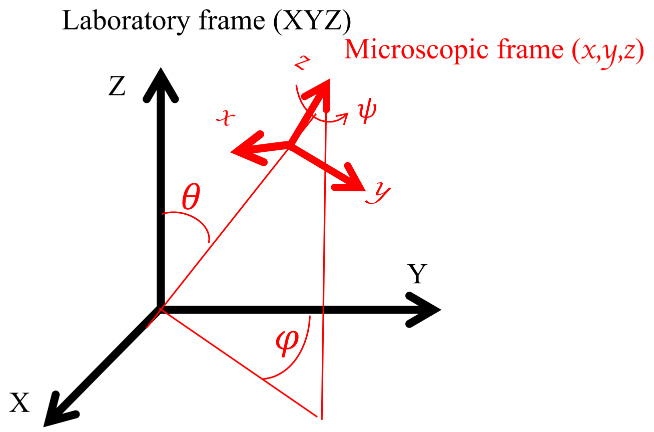


Figure 2: Definition of the angles $\theta,\ \varphi,\ \psi$ which describe the orientation of the scattered object in the laboratory frame.

Finally, the bracket means the average over all the orientation of the scattered object.

$$\langle\,.\,\rangle = \int_{\theta=0}^{\pi}\int_{\varphi=0}^{2\pi}\int_{\psi=0}^{2\pi} .\, sin\theta d\theta d\varphi d\psi \qquad (6)$$

The development of the above equations gives the following expression of $\vec{\beta}_{eff}$ expressed in term of the tensor β components.

$$\vec{\beta}_{eff}(\gamma,\Theta) = \begin{pmatrix} \beta_{XXX}.cos^2(\gamma) + 2\,\beta_{XXY}.\cos(\gamma)\sin(\gamma) + \beta_{XYY}.sin^2(\gamma) \\ [\beta_{YYY}.cos^2(\gamma) + 2\,\beta_{YXY}.\cos(\gamma)\sin(\gamma) + \beta_{YYY}.sin^2(\gamma)].cos^2(\Theta) - [\beta_{ZYY}.cos^2(\gamma) + 2\,\beta_{ZXY}.\cos(\gamma)\sin(\gamma) + \beta_{ZYY}.sin^2(\gamma)].\cos(\Theta)\sin(\Theta) \\ -[\beta_{YYY}.cos^2(\gamma) + 2\,\beta_{YXY}.\cos(\gamma)\sin(\gamma) + \beta_{YYY}.sin^2(\gamma)].\cos(\Theta)\sin(\Theta) + [\beta_{ZYY}.cos^2(\gamma) + 2\,\beta_{ZXY}.\cos(\gamma)\sin(\gamma) + \beta_{ZYY}.sin^2(\gamma)]\,sin^2(\Theta) \end{pmatrix} (7)$$

2.2) Coherent Second Harmonic Scattering of N fully correlated molecules

The Second Harmonic Scattering of N fully correlated molecule in solution can be expressed as:

$$I_{SHS}(\hat{e}_{p-in}, \hat{e}_{p-out}, \hat{e}_n) = I_{SHS}(\gamma, \Pi, \Theta) = \langle \vec{\beta}_{eff}(\gamma, \Pi, \Theta).\vec{\beta}^{*}_{eff}(\gamma, \Pi, \Theta)\rangle \qquad (8)$$

Here $\vec{\beta}_{eff} = (\hat{e}_n \times \overleftrightarrow{\beta_t} : \hat{e}_p\hat{e}_p \times \hat{e}_n)$ with $\overleftrightarrow{\beta_t} = \sum_j^N \overleftrightarrow{\beta}_j e^{i\overrightarrow{\Delta k}.\vec{r}_j}$ (9-10)

where $\overleftrightarrow{\beta_j}$ is the second order hyperpolarisability of the molecule j located at the position $\vec{r}_j$ and $\overrightarrow{\Delta k}$ has been defined above. To perform the I_{SHS} calculation, the user needs to specify the position $\vec{r}_j = (X'_j, Y'_j, Z'_j)$ and orientation $(\varphi'_j, \theta'_j, \psi'_j)$ of each molecule in the mesoscopic frame.

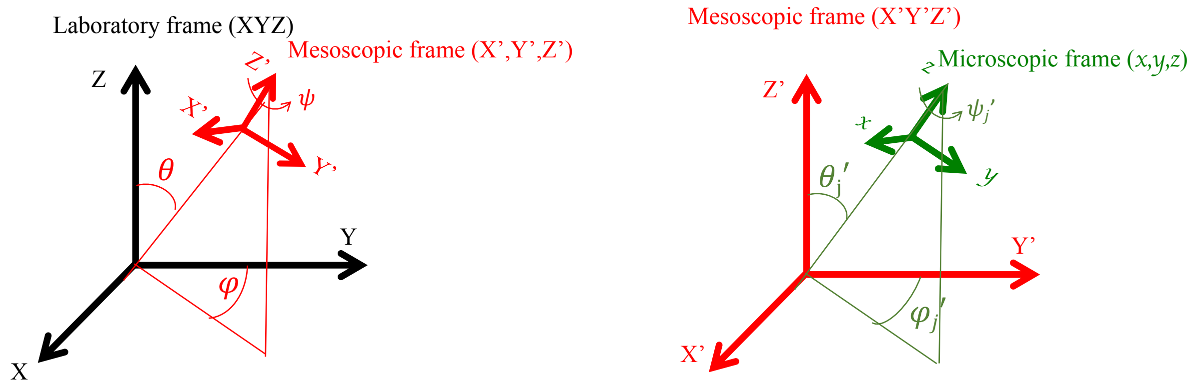


Figure 3: Definition of the angles θ, φ, ψ which describe the orientation of the scattered object in the laboratory frame.

Two ways are implemented to perform the I_{SHS} calculation:

- The first way explicitly compute I_{SHS} according to equations (8-10). This is implemented in program "SHS_polar_90.py", SHS_polar_180.py" and SHS_angle.py"
- The second way used the exponential Taylor development ($e^{i\overrightarrow{\Delta k}.\vec{r}_j} \cong 1 + \overrightarrow{\Delta k}.\vec{r}_j + \cdots$) to compute I_{SHS}. (see the following references to further detail[1].) This approximation is implemented in program "SHSlinear_polar_90.py" and "SHSlinear_polar_180.py"

2.3) Second Harmonic Scattering by a sphere

In the case of colloidal object in solution like a sphere, we can expressed the I_{SHS} intensity by introducing an effective nonlinear susceptibility $\overleftrightarrow{\Gamma^{(2)}}$, which represents the nonlinear response of the entire particle surface[2] :

$$I_{SHS}\left(\hat{e}_{p-in}, \hat{e}_{p-out}, \hat{e}_n\right) = I_{SHS}(\gamma, \Pi, \Theta) = \langle \vec{\beta}_{eff}(\gamma, \Pi, \Theta). \vec{\beta}^*_{eff}(\gamma, \Pi, \Theta) \rangle \qquad (11)$$

Here $\vec{\beta}_{eff} = (\hat{e}_n \times \overleftrightarrow{\Gamma^{(2)}} : \hat{e}_p\hat{e}_p \times \hat{e}_n)$ (12)

and $\overleftrightarrow{\Gamma^{(2)}} = \oiint_{surface\ of\ the\ particle} \overleftrightarrow{\chi^{(2)}} . e^{i\overrightarrow{\Delta k}.\vec{r}} d\vec{r}$ (13)

where $\overleftrightarrow{\chi^{(2)}}$ is the surface second order nonlinear susceptibility of the particle.

[1] Gassin and al Langmuir 2018, 34, 12206−1221 and Duboisset and al Phys. Rev. Lett. **120**, 263001

[2] Roke and al Phys. Rev. B **70**, 115106 (2004)

Chapter 3

Incoherent Second Harmonic Scattering : HRS Module

1) The different input/output

The program “HRS_polar_90.py” and “HRS_polar_180.py” compute the polarization resolved incoherent SHS Intensity at an angle of respectively Θ = 90° and Θ = 180°. The only input parameter of these programs is the microscopic hyperpolarizability tensor. Those components are passed through the input_beta file.

Format of the input_beta file:

line 1: comments

line 2:”xxx” space value

line3: “xxy” space value

And so on…

The output calculation are the coefficient $a_{\Pi}^{\Theta},\ b_{\Pi}^{\Theta},\ c_{\Pi}^{\Theta}$ where Π =V or H and Θ =90° or 180.

They are defined by:

$$I_{SHS}{}^{\Theta}(\gamma,\Pi) = a_{\Pi}^{\Theta}.cos^4(\gamma) + b_{\Pi}^{\Theta}.cos^2(\gamma)sin^2(\gamma) + c_{\Pi}^{\Theta}.sin^4(\gamma) \quad (14)$$

The comparison with equation (7) gives:

$$a_{\mathrm{V}}^{90} = a_{\mathrm{V}}^{180} = \langle \beta_{XXX}\beta_{XXX}^{*}\rangle \quad (15)$$

$$c_{\mathrm{V}}^{90} = c_{\mathrm{V}}^{180} = \langle \beta_{XYY}\beta_{XYY}^{*}\rangle \quad (16)$$

$$b_{\mathrm{v}}^{90} = b_{\mathrm{v}}^{180} = \langle 4\beta_{XXY}\beta_{XXY}^{*} + \beta_{XXX}\beta_{XYY}^{*} + \beta_{XXX}^{*}\beta_{XYY}\rangle \quad (17)$$

$$a_{\mathrm{H}}^{90} = \langle \beta_{zXX}\beta_{ZXX}^{*}\rangle \quad (18)$$

$$a_{\mathrm{H}}^{180} = \langle \beta_{YXX}\beta_{YXX}^{*}\rangle \quad (19)$$

$$c_{\mathrm{H}}^{90} = \langle \beta_{ZYY}\beta_{ZYY}^{*}\rangle \quad (20)$$

$$c_{\mathrm{H}}^{180} = \langle \beta_{YYY}\beta_{YYY}^{*}\rangle \quad (21)$$

$$b_{\mathrm{H}}^{90} = \langle 4\beta_{ZXY}\beta_{ZXY}^{*} + \beta_{ZXX}\beta_{ZYY}^{*} + \beta_{ZXX}^{*}\beta_{ZYY}\rangle \quad (22)$$

$$b_{\mathrm{H}}^{180} = \langle 4\beta_{YXY}\beta_{YXY}^{*} + \beta_{YXX}\beta_{YYY}^{*} + \beta_{YXX}^{*}\beta_{YYY}\rangle \quad (23)$$

Additionnal coefficient I_2 and I_4 are also given. They are defined by:

$$I_2^{\Theta,\Pi} = \frac{4(a_\Pi^\Theta - c_\Pi^\Theta)}{(3a_\Pi^\Theta + b_\Pi^\Theta + 3c_\Pi^\Theta)} \quad (24)$$
$$I_4^{\Theta,\Pi} = \frac{(a_\Pi^\Theta - b_\Pi^\Theta + c_\Pi^\Theta)}{(3a_\Pi^\Theta + b_\Pi^\Theta + 3c_\Pi^\Theta)} \quad (25)$$

The program will also show a graphic with the polarization plot.

The program "HRS_angle.py" computes the angle resolved incoherent SHS Intensity for different polarization states:

- $\mathrm{IH}(0°) = I_{HRS}(\gamma = 0°, \Pi = 90°, \Theta)$ also referred as I_{PSS} to be coherent with some author[3]
- $\mathrm{IH}(90°) = I_{HRS}(\gamma = 90°, \Pi = 90°, \Theta)$ also referred as I_{PPP}
- $\mathrm{IV}(0°) = I_{HRS}(\gamma = 0°, \Pi = 0°, \Theta)$ also referred as I_{SSS}
- $\mathrm{IV}(90°) = I_{HRS}(\gamma = 90°, \Pi = 0°, \Theta)$ also referred as I_{SPP}

The user has to specify the number of point between 0 and 180° that he want to compute.

2) Command line

In a terminal, enter the following command:

>python $PATH/HRS_polar_90.py $PATH/input_beta $PATH/output_filename
where $PATH is the path where the program HRS_polar_90 is located.

Exemple: if you are in the parent PySHS directory:
>python Src/HRS_polar_90.py work/input_beta work/output_filename

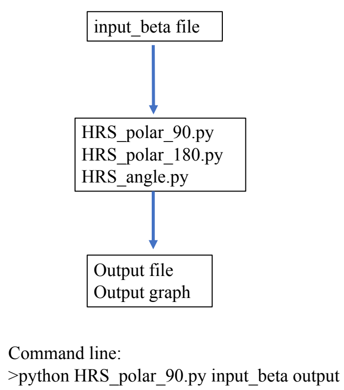


Figure 4: Flow diagram representing a typical HRS simulation

[3] See for example: THE JOURNAL OF CHEMICAL PHYSICS **132**, 234702 (2010)

Chapter 4

Coherent Second Harmonic Scattering of Supramolecular structures : SHS Module

1) The different input/output

Program "SHS_polar_90.py" and "HRS_polar_180.py" compute the polarization resolved coherent SHS Intensity at an angle of respectively Θ = 90° and Θ = 180°.

Different input parameters have to be given:

- microscopic hyperpolarizability tensor components are expressed in a file input_beta like in chap 3.
- the position and orientation of each dipole of the aggregate are expressed in the mesoscopic frame in an input_file2. The format of this input file is a 6 columns per line containing in the order($\varphi'_j, \theta'_j, \psi'_j, x'_j, y'_j, z'_j$) which represent the orientation and position of the molecule j. The number of line is the number of molecule in the aggregate. The angles $\varphi'_j, \theta'_j, \psi'_j$ **have to be given in radian.**

Some specific script also given in ./Scr folder can generate these input files for different geometry (sphere, cylinder …)

- the refractive index, the laser wavelength (expressed in nanometer), the number of dipole in the aggregate. All these argument are asked by the program.

The output calculation are the coefficient a_{Π}^{Θ}, b_{Π}^{Θ}, c_{Π}^{Θ}, $I_2^{\Theta,\Pi}$ and $I_4^{\Theta,\Pi}$ as explained in the HRS module. Additionnal coefficient $b1_{\Pi}^{\Theta}$ and $b2_{\Pi}^{\Theta}$ are also given. Indeed, in the developpment of equation (3) appears 2 terms that are zero in HRS calculus but may be eventually no vanished in some special case in SHS calculus, so their values are also given in the output. Those term are defined by:

$$I_{SHS}{}^{\Theta}(\gamma,\Pi) = a_{\Pi}^{\Theta}.cos^4(\gamma) + b_{\Pi}^{\Theta}.cos^2(\gamma)sin^2(\gamma) + c_{\Pi}^{\Theta}.sin^4(\gamma) + b1_{\Pi}^{\Theta}cos^3(\gamma)sin\ \ (\gamma) + b2_{\Pi}^{\Theta}cos\ \ (\gamma)sin^3(\gamma)$$

(26)

Program “SHS_angle.py” computes the angle resolved incoherent SHS Intensity for different polarization state as explained in chap 3. The user has to specify the number of point to be compute.

2) The SHSlinear Module

As explained in the chap 2, the I_{SHS} calculation can be performed using an exponential Taylor development ($e^{i\overrightarrow{\Delta k}.\vec{r}_j} \cong 1 + \overrightarrow{\Delta k}.\vec{r}_j$). This approach only works in the case of a small size aggregate in regards with the input laser wavelength. This approach reduces the computational time needs to perform the calculation in the case of large number molecules in aggregate (typically N>1000). The inputs and outputs of this module are the same as for the SHS module.

3) Command line

In a terminal, enter the following command:

>python $PATH/SHS_polar_90.py $PATH/input_beta $PATH/input_aggregate $PATH/output

where $PATH is the path where the program SHS_polar_90 is located and so on.

Exemple: if you are in the parent PySHS directory:

>python Src/SHS_polar_90.py Work/input_beta work/input_aggregate Work/ouput_file

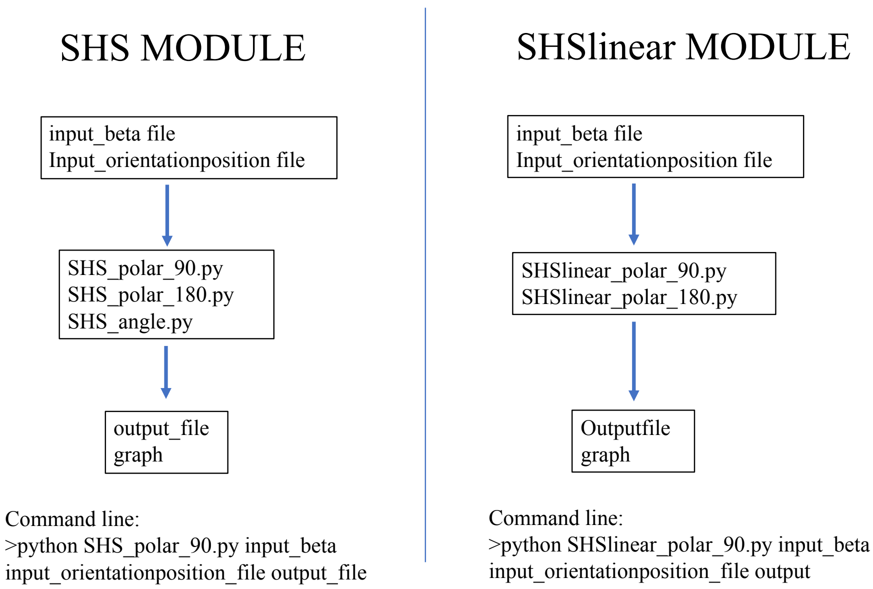


Figure 5: Flow diagram representing a typical setup of a SHS simulation

Chapter 5

Coherent Second Harmonic Scattering of Colloidal object : Sphere Module

1) The different input/output

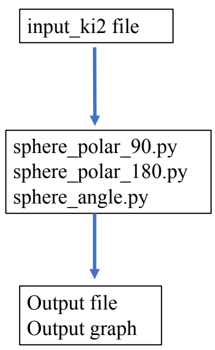
Program “sphere_polar_90.py” , “sphere_polar_180.py” compute the polarization resolved coherent SHS Intensity of a sphere with a surface nonlinear susceptibility at an angle of respectively Θ = 90° and Θ = 180°.

Different input parameters have to be given:

- Surface second order susceptibility tensor components are expressed in a file input_ki(2) like in chap 3.
- the refractive index, the laser wavelength (expressed in nm), the radius of the sphere. All these argument are asked by the program.

The output calculation are the coefficient a_{Π}^{Θ}, b_{Π}^{Θ}, c_{Π}^{Θ} , $I_2^{\Theta,\Pi}$ and $I_4^{\Theta,\Pi}$ as explained in the HRS module.

Program “sphere_angle.py” computes the angle resolved incoherent SHS Intensity for different polarization state as explained in chapter 3. The user has to specify the number of point to be compute.



Command line:
>python sphere_polar_90.py input_ki2 output

Figure 6 : Flow diagram representing a typical setup of a SHS simulation

Chapter 6

Additional Scripts and Perspectives

1) Scripts to generate orientation-position inputfile in SHS module

- "script_sphere.py" : It generates the position of N dipoles onto a sphere of specified radius and with a radial orientation. This script is based on this code[4]. The input parameters asked by the program are the radius of the sphere and number of dipoles.

- "script_sphere _random.py": It generates the position of N dipoles onto a sphere of specified radius and with a specified normal distribution orientation around the radial direction. The input parameters asked by the program are the radius of the sphere, number of dipoles and the standard deviation (sigma) of the normal distribution around the radial direction.

- "script_spheroid.py": It generates the position of N dipoles onto a spheroid of specified radius a and c and with a normal orientation.

- "script_spheroid_random.py": It generates the position of N dipoles onto a spheroid of specified a and c radius and with a specified normal distribution orientation around the normal direction.

- "script_cylinder.py": It generates the position of dipoles uniformally spaced onto a cylinder of specified length and radius with a normal orientation.

- "script_cube.py": It generates the position of dipoles uniformally spaced onto a cube of specified length with a normal orientation.

[4] https://bduvenhage.me/geometry/2019/07/31/generating-equidistant-vectors.html
https://www.openprocessing.org/sketch/41142

In all script, the command line is:

>python script_XXX.py output_file

Where “XXX” is “sphere” or “sphere_random” or ”spheroid” or “cylinder” or “cube”.

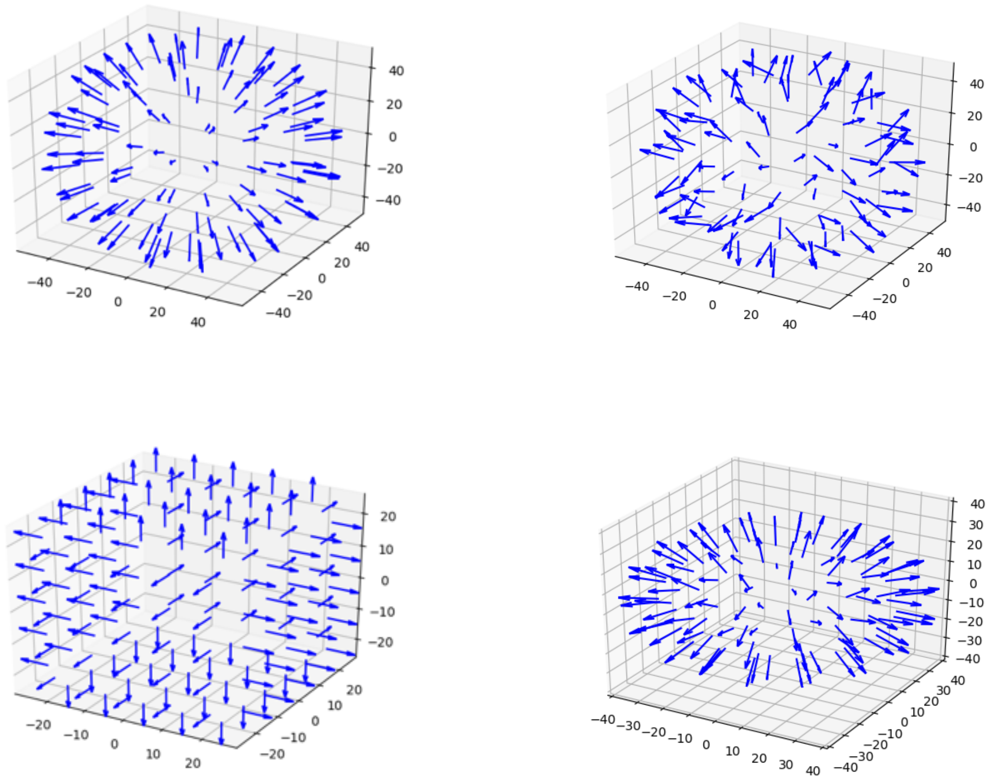


Figure 7 : Some dipoles organization obtained with the various script, in the order: script_sphere.py (radius =50, N=100), script_sphere_random.py (radius =50, N=100, sigma=0.5), script_cube.py (length=50, N=100), script_spheroid.py (a=40,c=20, N=100).

2) Perspectives

PySHS will likely evolved. This evolution will include other kind of geometry in sphere module, like cylinder, cubic geometry etc…

Moreover ki(3) effect, which are not taken into account in PySHS V1.1 will probably be added in a future version.