Pythonreflectivity v1.0

M. Zwiebler

July 4, 2016

Preliminary remarks

Installation
Unpack the Pythonreflectivity folder. Open the folder in a console window. Install with "python setup.py install".
The Pythonreflectivity package geometry is the same as the one in *ReMagX*, from which this is inspired and which was used to confirm the reflectivity results.
Run the script "Examples.py" for calculation examples.

Geometry

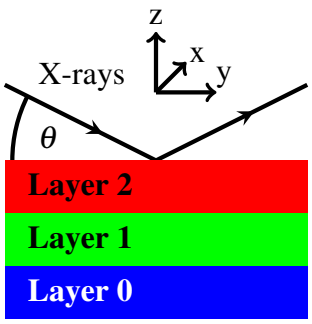

Figure 1: Reflectivity geometry

List of Functions and commands

Pythonreflectivity.Generate_structure(N, (MLstructure))

This command generates a structure on which the reflectivity can be calculated.
Input:

- Number of layer types N (int)
- Multilayer structure (optional), from the lowest layer to the highest. Default is "$0, 1, 2, 3, ..., N-1$". Multilayers syntax is "$0, 100 * (1, 2, 3, 2), 4, ...$". Brackets inside brackets is not supported.

Output:
A list of Layer-type objects with length N

Layers

Setting values:

- Layer.setd(d) - Sets the layer thickness d
- Layer.setsigma(σ) - Sets the roughness σ for the interface on top of this layer
- Layer.setmag(direction) - Sets the magnetization direction for the magneto-optical Kerr effect (MOKE). Allowed inputs are "x", "y", "z" and "0".

- Layer.seteps(ε) - Sets the complex dielectric tensor. Numbers/Length 1 lists set ε_{xx}, ε_{yy} and ε_{zz}. Length 3 lists set ε_{xx}, ε_{yy} and ε_{zz}. Length 4 lists set ε_{xx}, ε_{yy} and ε_{zz} and, depending on the magnetization direction, either $\varepsilon_{yz} = -\varepsilon_{zy}$ (if x), $\varepsilon_{xz} = -\varepsilon_{zx}$ (if y) or $\varepsilon_{xy} = -\varepsilon_{yx}$ (if z).
- Layer.setepsxx(ε_{xx}) sets ε_{xx}
- Layer.setepsyy(ε_{yy}) sets ε_{yy}
- Layer.setepszz(ε_{zz}) sets ε_{zz}
- Layer.setepsg(ε_g) sets either $\varepsilon_{yz} = -\varepsilon_{zy}$ (if x), $\varepsilon_{xz} = -\varepsilon_{zx}$ (if y) or $\varepsilon_{xy} = -\varepsilon_{yx}$ (if z)

Getting values:

- Layer.d() - returns the layer thickness d
- Layer.sigma() - returns the roughness σ for the interface on top of this layer
- Layer.mag() - returns the magnetization direction for MOKE.
- Layer.eps() - returns the complex dielectric tensor as a number, length 3 array or length 4 array depending on the input.
- Layer.epsxx() returns ε_{xx}
- Layer.epsyy() returns ε_{yy}
- Layer.epszz() returns ε_{zz}
- Layer.epsg() returns either $\varepsilon_{yz} = -\varepsilon_{zy}$ (if x), $\varepsilon_{xz} = -\varepsilon_{zx}$ (if y) or $\varepsilon_{xy} = -\varepsilon_{yx}$ (if z)

Pythonreflectivity.Reflectivity(structure, angles, wavelength, (MultipleScattering), (MagneticCutoff), (Output))

Input:

- structure - A list of layers with that contains the Multilayer information generated by Pythonreflectivity.Generate_structure
- angles - array of doubles between 0° and 90°, or a single angle.
- wavelength - double
- MultipleScattering (optional) - Takes True or False. False is $\approx 20\%$ faster. Default is True.
- MagneticCutoff (optional) - double. If an off-diagonal element of ε (ε_g) fulfills $|\varepsilon_g|$<MagneticCutoff, it is set to zero. This is necessary for numerical stability. Default is 10^{-6}
- Output (optional) - Allowed inputs are
  "T" - All Intensities
  If not magnetic, the output will be a matrix of doubles size [2, len(angles)] containing the intensity reflectivity for σ and π polarization. If magnetic the output will be a matrix of doubles size [4, len(angles)] containing the intensity for σ, π, left circular and right circular reflectivity.
  "t" - All Amplitudes
  If not magnetic, the output will be a complex size [2, len(angles)] containing the amplitude reflectivity for σ and π polarization. If magnetic the output will be a complex matrix size [4, len(angles)]

containing the reflectivity matrix elements $[r_{\sigma\to\sigma}, r_{\pi\to\sigma}, r_{\sigma\to\pi}, r_{\pi\to\pi}]$.
"S" σ Intensity reflectivity
- If not magnetic, the output will be an array of doubles containing the intensity reflectivity of σ-polarized light.
"s" σ amplitude reflectivity
- If not magnetic, the output will be a complex array containing the amplitude reflectivity of σ-polarized light.
"P" π Intensity reflectivity
- If not magnetic, the output will be an array of doubles containing the intensity reflectivity of π-polarized light.
"p" π amplitude reflectivity
- If not magnetic, the output will be a complex array containing the amplitude reflectivity of π-polarized light.
The default input it "T".

The function returns the reflectivity at each angle as given by the Output parameter.

Parallelization

The Pythonreflectivity package uses a single processor, but it can be parallelized from the Python side. I found the pp module to be up to the task. Note that Layer instances can not be pickled at this point, which will be an issue if one tries to directly parallelize the Reflectivity function. It is however possible to write an input wrapper and to create the optical structure within the thread from parameters given to the thread.