1 Introduction

Written by Steve Byrnes, 2012. Please email me any feedback. My website is `http://sjbyrnes.com` . Last update/upload: October 2016.

This is a group of programs written in Python / NumPy for simulating light propagation in planar multilayer thin films, including the effects of multiple internal reflections and interference, using the "Transfer Matrix Method". It can also simulate combinations of thin and thick films (e.g. a thick piece of glass with a multi-layer antireflection coating on one side and a mirror on the other side), or purely thick films.

In addition to calculating how much light is transmitted and reflected, the program can calculate, at any given point in the structure, how much light is being absorbed there. This is a very important feature for solar-cell modeling, for example.

It can also calculate the parameters measured in ellipsometry.

I wrote out derivations and discussions of the formulas and calculations implemented by this program at: `http://arxiv.org/abs/1603.02720`. You are encouraged to cite that paper if you publish results that come from this software. :-)

2 Files and API

There are four files: (1) `tmm_core.py` contains all the main programs, (2) `color.py` has additional color-related functions, like calculating the RGB color of a thin film under reflected light, (3) `examples.py` contains a few example calculations to get you started, and (4) `tests.py` contains a number of programs that perform various tests and consistency checks to confirm that everything is coded and running correctly. There is also the standard `__init__.py`, the home of the main `tmm` namespace, into which are imported all the `tmm_core` functions. [To run the color-theory related functions in `color.py`, you need to download the package `colorpy` at `https://pypi.python.org/pypi/colorpy/` . But even without that, you can still run all the other code.]

The API information (list of functions and how to call them) is available at `https://pythonhosted.org/tmm/`. I also suggest to look at the functions in the `examples` submodule to see examples of various kinds of calculations and plots. (If you're not sure how to find this code, it is online at `https://pythonhosted.org/tmm/_modules/tmm/examples.html`)

3 Other people's programs

There are many other free (and non-free) programs that do some or all of what this program does. I have a list at: `http://sjbyrnes.com/multilayer-film-optics-programs/`

I have done a few consistency checks between my program and others. They tend to agree perfectly except in the tricky (and somewhat unusual) case of calculating reflected power or transmitted power when the semi-infinite incoming and/or outgoing medium has a complex index of refraction. Again, see `http://arxiv.org/abs/1603.02720` for further discussion.

4 Installation

Requires SciPy and NumPy to run. The "examples" also use Matplotlib, while the color calculation part requires ColorPy (`https://pypi.python.org/pypi/colorpy`). Tested in Python 2.7 and 3.3–3.5. (For color calculations in Python 3, you need the Python-3-compatible version of ColorPy at `https://github.com/fish2000/ColorPy/` . Install it via `pip` or `easy_install`, or just download it directly (it's pure python, but you need to treat it as a package, not separate files).[1]

4.1 Installation for dummies

If you've never used Python before, check out `http://sjbyrnes.com/python` for advice on installation and getting started.

Let's say you have Windows or Mac, and you just installed the Anaconda Python distribution `https://www.continuum.io/downloads` . The next step is to open the "Anaconda command prompt", connect to the internet, and then run the command `pip install tmm` . All done!

Note that when you install with `pip`, the package source code winds up in a folder somewhere on your computer. If you want to look at it or edit it, you need to find it first! The easy way to find it is to type `import tmm; tmm.__file__` into your Python console.

To get going, try typing the following in IPython (pylab mode) or a Spyder console:

```
>> import tmm.examples
>> tmm.examples.sample1()
```

[1] People don't normally set up packages "by hand", but it's not too difficult in a case like this where the package is pure python, rather than C code etc. See `https://docs.python.org/3/tutorial/modules.html#packages` . You download the package file from `https://pypi.python.org/pypi/tmm` (it's called something like `tmm-0.1.2.tar.gz`), and unzip it. (On Windows you need an unzipping program, `http://www.7-zip.org/`). Your goal is to have a (normal, not zipped) folder named `tmm` (lower-case), in which are all the `.py` files (`__init__.py` and `tmm_core.py` and so on, but not `setup.py`). You may need to move files or rename the folder to get things in this format. Now that you have this folder, put it somewhere that Python can find it. (If you're not sure, type `import sys; sys.path` into your Python interactive console. Any folder on this list is an OK place to put your `tmm` folder. Note that the `tmm` folder itself (i.e., the files inside it) should *not* be on the Python path.) Exit and restart your Python interactive console, and you should have `tmm` working.

A plot should pop up.

5 Units

The program implicitly requires a unit of length. You can use any unit, but keep it consistent. For example, if wavelength is given in nanometers, the thicknesses of the layers should also be given in nanometers, and the absorption at a given point will be in (fraction of incoming light power per nanometer of depth).