SCOptC

Private

main 1 Branch 1 Tags Go to file t Go to file + About Add file Code ...

manromagu Update README.... bd49365 · now 7 Commits

src	Update and rename ReTr...	22 minutes ago
LICENSE.md	Add files via upload	43 minutes ago
README.md	Update README.md	now

README License

SCOptC

Solar Cell Optical Calculator.

Instalation:

Add `src` folder to Matlab path.

Suported Models:

- Constant refractive index model: $n(\lambda) = n0$
- Real Cauchy Model: $n(\lambda) = A_1 + 10^4 \frac{A_2}{\lambda^2} + 10^9 \frac{A_3}{\lambda^4}$
- Imaginary Cauchy Model:

Solar Cell Optical Calculator

Readme
View license
Activity
Custom properties
0 stars
0 watching
0 forks

Releases 1

SCOptC v0.1 Latest
21 minutes ago

Packages

No packages published
Publish your first package

Languages

MATLAB 100.0%

$$n(\lambda) = A_1 + 10^4 \frac{A_2}{\lambda^2} + 10^9 \frac{A_3}{\lambda^4} + A_4 \cdot i + 10^4 \frac{A_5}{\lambda^2} \cdot i + 10^9 \frac{A_6}{\lambda^4} \cdot i$$

- Forouhi-Bloomer model: $n(\lambda) = n_0 + \sum_{j=1,2,3,4} \frac{B_j \cdot (E(\lambda)-E_j)+C_j}{(E(\lambda)-E_j)^2+G_j^2} + \frac{f_j \cdot (E(\lambda)-E_g)^2 \cdot \delta(E(\lambda)-E_g)}{(E(\lambda)-E_j)^2+G_j^2} \cdot i$, being δ the step function.
- Linear gradient model: $n_j(\lambda) = \frac{n_2-n_1}{N-1} \cdot j$, being N the number of sublayers.
- DBR: Distributed Bragg Reflector with N periods.
- File: Load n and k data from `mat` file.

Usage:

SCOptC function call takes the following form:

```
[models_out,N,D,results,foptions_out] =
SCOptC(wl,theta,models,foptions)
```

Input arguments:

- `wl` wavelength (in nm) vector
- `theta` incident angles struct:
  - `theta.values` vector containing the incident angles (in degrees) of the measurements to calculate.
- `models` cell of structs containing the information of each layer.
- `foptions` options struct.

Models struct:

All the required information of each layer is stored inside a `model` struct. We make a distinction between known layers and unknown layers.

- Known layers:
  - Forouhi-Bloomer model:
    - `model.type = "Fh-N"`
    - `model.Eg` Bandgap in Ev
    - `model.n0` low frequency refractive index
    - `model.fi` fi parameter (length should be equal to the number of oscillators)
    - `model.Ei` Ei parameter (length should be equal to the number of oscillators)
    - `model.Gi` Gi parameter (length should be equal to the number of oscillators)
    - `model.D` layer thickness in nm
  - Real Cauchy model
    - `model.type = "Ch-n"`
    - `model.A` vector with real Cauchy parameters `[ A1 , A2 , A3 ]`
    - `model.D` layer thickness in nm
  - Complex Cauchy model
    - `model.type = "Ch-nk"`
    - `model.A` vector with real Cauchy parameters `[ A1 , A2 , A3 , A4 , A5 , A6 ]`
    - `model.D` layer thickness in nm
  - Linear refractive index gradient
    - `model.type = "lin-grad"`
    - `model.n1` refractive index of the first layer

        - `model.n1` refractive index of the first layer
        - `model.n2` refractive index of the last layer
        - `model.nlayers` number of layers
        - `model.D` total tickness in nm
    - Constant refractive index
        - `model.type = "cnst"`
        - `model.n` refractive index
        - `model.D` layer thickness in nm
    - DBR (distributed Bragg reflector):
        - `model.type = "DBR"`
        - `model.n1` refractive index of the first layer
        - `model.n2` refractive index of the second layer
        - `model.D1` layer thickness in nm of the first layer
        - `model.D2` layer thickness in nm of the second layer
        - `model.nperiod` number of periods
    - DBRf (distributed Bragg reflector with nk data from file):
        - `model.type = "DBR"`
        - `model.filename1` refractive index file of the first layer
        - `model.filename2` refractive index file of the second layer
        - `model.D1` layer thickness in nm of the first layer
        - `model.D2` layer thickness in nm of the second layer
        - `model.nperiod` number of periods
    - Load from .mat file
        - `model.type = "file"`
        - `model.filename` full path to the nk .mat file. The variables inside this file must be:

this file must be:

        - `wl` wavelenth in nm
        - `n` real part of the refractive index
        - `k` imaginary part of the refractive index

All the models shoud have a property called `active`. If `model.active="true"` the absorption in that layer will contribute to the Jsc. Once all the models are properly defined, they must be packed in a cell array: `models = {model_1 model_2 model_3 model_4}`.

Options struct:

- `foptions.lcoher` coherence length (`1e4` is recommended).
- `foptions.backwards` calculate the cell from the oposite illumination (`"true"` or `"false"`).
- `foptions.zstep` z step for electric field calculation (< `1nm` is recommended).
- `foptions.plot` plot R%T results (`"true"` or `"false"`).