simetuc User Manual

Pedro Villanueva-Delgado

January 12, 2017

Contents

Chapter 1

Features

- Command line interface program.
  - Run with

    ```
    simetuc config_file.txt [options]
    ```

  - See all options below and with

    ```
    simetuc -h
    ```

- The simulation is controlled by a configuration text file that the user can edit with the parameters adequate to its system of study. It includes:
  - Information about the host lattice.
  - Energy states labels.
  - Absorption and excitation (including ESA).
  - Decay (including branching ratios).
  - Energy transfer.
  - Other settings for the power and concentration dependence or optimization.
- *simetuc* works with any sensitizer and activator ion kind.
  - The examples are given for the Yb-Tm system.
- All kinds of energy transfer processes are supported:
  - Energy migration.
  - Upconversion (ETU).
  - Downconversion.
  - Cross-relaxation.
  - Cooperative processes.
  - Energy transfer from sensitizers to activators.
  - Back transfer from activators to sensitizers.

- See the example configuration file in the simetuc folder.
  - The configuration file sections are detailed and discussed in this document.
  - See also the full example in the appendix.
- Add decay experimental data as two column text data, separated by tabs or spaces.
- Different options:
  - Create the lattice.
  - Simulate the dynamics (rise and decay).
  - Optimize the energy transfer parameters.
    - Minimize the deviation between experiment and simulation.
  - Simulate the steady state.
  - Simulate the power dependence of each emission.
  - Simulate the concentration dependence of the dynamics or the steady state.
- All results are plotted and saved in the .hdf5 format.
- For all options `--average` uses standard average rate equations instead of microscopic ones.

Part I

User Guide

Chapter 2

Installation

Python 3.5 is required. Installing Anaconda is recommended; it works with Windows (64/32 bits), Linux (64/32 bits) and Mac (64 bits).

After installing Anaconda execute the following command at the command prompt (*cmd.exe* for Windows, *shell* for Linux and Mac):[1]

```
conda install -c pedvide simetuc
```

or

```
pip install simetuc
```

That will download and install all necessary files.

2.1 Update

If you installed it using *conda*, update with:

```
conda update -c pedvide simetuc
```

If you installed it with *pip*, update with:

```
pip install -U simetuc
```

[1] Some OSX users report problems using *conda*, if after installing you can't use the program (i.e., *simetuc -h* fails because *simetuc* wasn't recognized as a command), use *pip install simetuc*.

Chapter 3

The configuration file

Some sections of the configuration file are mandatory (Sections 3.1-3.5), some are optional (Sections 3.6-3.9), and some are only mandatory when performing a particular simulation type (Sections 3.10 and 3.11).

Comments can be placed anywhere in the file starting with a hash symbol (#).

The different sections of the file begin by a section name, for example lattice or states, followed by a colon (:). The different subsections and values are in a new line and indented with four spaces (not tabs!). A section can contain a single value or subsections; for example:

```
# comment1
section1: value1
section2:
    subsection1: value2 # comment2
    subsection2:
        subsubsection1: value3
```

Some of the values may be text or numbers, others, however, may be lists or a list of lists. A list is written as a comma-separated values between two square brackets:

```
section: [item1, item2, item3]
```

And for a list of lists:

```
section: [[lst1_item1, lst1_item2], [lst2_item1, lst2_item2]]
```

A full example of a configuration file, showing all options, is shown in Listing 9.1.

3.1 Version

At the moment, there is only one format for the configuration file; however, in the future, it is possible that an improved format is introduced that is not compatible with the current one. For this reason the version of the format has to be included, the only value allowed is 1:

```
version: 1
```

3.2 Lattice

This section defines the name of the lattice, the unit cell parameters, the concentration of sensitizer and activator ions, and optionally, a maximum distance of interaction.

This section starts with

```
lattice:
   # values and subsections...
```

Those values and subsections are described now.

The name is an arbitrary text that describes the lattice. It will be used as a folder name under latticeData (Section 5.1.1), expData (Section 5.3.1), and results (Section 4.2). Therefore it should only contain letters, numbers, and underscore (_) characters (no spaces).

The number of unit cells is an integer greater than zero. It specifies how many unit cells along the three directions a lattice will consist on.

The concentration of sensitizer and activator, in percentages from 0 to 100%. An ion with 0% concentration will not participate in the equations and therefore will not make the simulation take longer.

```
   S_conc: 0
   A_conc: 0.3
```

The unit cell parameters consist on the three lattice distances and three angles. The distances are given in Angstrom and the angles in degrees (between 0 and 360º).

```
   # distances in Angstrom
   a: 5.9738
   b: 5.9738
   c: 3.5297
   # angles in degree
   alpha: 90
   beta: 90
   gamma: 120
```

The spacegroup determines the symmetry of the lattice. It can be given as the international short symbol (e.g., P-6) or as the International Union of Crystallography number (e.g. 174).

The sites positions and occupancies determine the positions of the doped ions (in units of the lattice distances, from 0 to 1) and their fractional occupancy (between 0 and 1). For a lattice with just one site:

```
   sites_pos: [0, 0, 0]
   sites_occ: 1
```

For a lattice with two sites, the second of which is half occupied:

```
   sites_pos: [[0, 0, 0], [2/3, 1/3, 1/2]]
   sites_occ: [1, 1/2]
```

The sites are populated randomly using the desired concentration, see Section 5.1.

A full example of the lattice section is

```
lattice: # all fields here are mandatory
    name: bNaYF4
    N_uc: 40
    S_conc: 0
    A_conc: 0.3
    # unit cell
    # distances in Angstrom
    a: 5.9738
    b: 5.9738
    c: 3.5297
    # angles in degree
    alpha: 90
    beta: 90
    gamma: 120
    spacegroup: P-6
    # info about sites.
    sites_pos: [[0, 0, 0], [2/3, 1/3, 1/2]]
    sites_occ: [1, 1/2]
```

3.2.1 Optional values

The maximum distances for the normal and cooperative interactions restrict energy transfer to ions closer than the distances. The normal maximum distance (`d_max`) is not yet implemented; however, the cooperative one is, and it's strongly recommended to use it if cooperative interactions are present. The cooperative interactions grow approximately with the cube of the number of ions, the normal interactions grow with the square. If the values are not present, they are set to infinite.

```
lattice:
    ... all other fields ...
    d_max: 100.0
    d_max_coop: 25.0
```

3.3 States

This section determines the number of energy states per ion and their labels.

Four subsections are mandatory, two per ion type:

The ion labels is any text that describes the sensitizer or activator ion. It usually is the ion's chemical symbol.

The ion states labels is a list of text. Each item in the list labels an energy state; the first is the ground state, the rest should be in order of increasing energy. Usually the term symbols of the Dieke diagram are used.

```
states:
    sensitizer_ion_label: Yb
    sensitizer_states_labels: [GS, ES]
    activator_ion_label: Tm
    activator_states_labels: [3H6, 3F4, 3H5, 3H4, 3F3, 1G4, 1D2]
```

The ion and state labels will be used to define excitation (Section 3.4), decay (Section 3.5), branching ratios (Section 3.6), and energy transfer processes (Section 3.7).

3.4 Excitation

At the beginning of the simulation, all ions are in their ground state. An excitation source is necessary to pump them to an excited state. This source can be pulsed for dynamics or continuous for steady state simulations.

Any number of excitation subsections can be added to the file, and at least one has to be active (see below). A two-color experiment can be simulated by activating two excitations. Any number of excitations can be active.

An excitation block contains several mandatory subsections:

The label is a short unique text. It identifies the excitation and it is used in the naming of experimental data files (see Section 5.3.1), so short, informative names are encouraged.

active can be true or false and indicates whether the excitation will be used for the simulations.

The excitation power density in W cm^{-2}. For pulsed sources this has to be the instantaneous power density of the pulse. Usual values are in the order of 10^5–10^7 W cm^{-2} for pulsed excitations and 10^1–10^3 W cm^{-2} for continuous excitations.

The pulse width of the excitation source (in seconds) for pulsed excitations. For continuous excitations it can be omitted.

The process is the actual absorption transition. The format is `ion_label(state_label_i) -> ion_label(state_label_f)`. Where the ion and state labels were defined in Section 3.3. Both ion labels have to be present and equal; both state labels have to belong to the same ion type.

The degeneracy is the relative degeneracy of the initial and final states. The relative degeneracy of two trivalent lanthanide states with terms ${}^{2S_i+1}L_{iJ_i}$ and ${}^{2S_f+1}L_{fJ_f}$ is $g_{rel} = \frac{2J_i+1}{2J_f+1}$.

The pump rate is the absorption cross-section divided by the transition energy in units of cm^2 J^{-1}, that is $R_P = \frac{\sigma(\nu)}{h\nu}$. The pump rate multiplied by the power density is equal to the absorption probability, with units of s^{-1}.

An example with two excitation blocks:

```
excitations:
    pulsed_1G4:
        active: True
        power_dens: 1e6
        t_pulse: 1e-8
        process: Tm(3H6) -> Tm(1G4)
        degeneracy: 13/9
        pump_rate: 9.3e-4
    CW_980nm:
        active: False
        power_dens: 1e2
        process: Yb(GS) -> Yb(ES)
        degeneracy: 4/3
        pump_rate: 4.4e-3
```

3.4.1 Excited state absorption

Excited state absorption (ESA) can be simulated by giving a list of transitions, degeneracies and pump rates in the corresponding fields, for example:

```
excitations:
    NIR_800:
        active: False
        power_dens: 1e5
        t_pulse: 1e-8
        process: [Tm(3H6)->Tm(3H4), Tm(3H5)->Tm(1G4)] # list
        degeneracy: [13/9, 11/9] # list
        pump_rate: [4.4e-3, 4e-3] # list
```

3.5 Decay

This section contains the decay lifetimes in seconds. There are two sections, one for the sensitizer and one for the activator. In both cases the decay lifetimes are input with `state_label: lifetime`:

```
sensitizer_decay:
    ES: 2.5e-3
activator_decay:
    3F4: 12e-3
    3H5: 25e-6
    3H4: 2e-3
    3F3: 2e-6
    1G4: 760e-6
    1D2: 67.5e-6
```

3.6 Branching ratios

These are two optional sections, one for the sensitizer and one for the activator.

The branching ratios describe the fraction of emission from an initial state to a final state. The format is `initial_state -> final_state: ratio`. The ratio is a number between zero and one. The branching ratio to the ground state does not need to be specified because it is calculated automatically from the other values.

```
sensitizer_branching_ratios: # can be empty or simply omitted
activator_branching_ratios:
    3H5->3F4: 0.4
    3H4->3F4: 0.3
    3H4->3H5: 0.1
    3F3->3H4: 0.999
    1G4->3F4: 0.15
    1G4->3H5: 0.16
    1G4->3H4: 0.04
    1G4->3F3: 0.001
    1D2->3F4: 0.43
```

3.7 Energy transfer

This optional section includes one or more energy transfer processes between ions in the simulation. Each energy transfer block refers to one process and contains several fields:

The label is a short unique text. It can also be used to restrict the optimization to certain processes (Section 3.8).

The process describes the ET interaction with the format `ion_label1(init_state) + ion_label2(init_state) -> ion_label1(end_state) + ion_label2(end_state)`. Where the ion and state labels were defined in Section 3.3.

The multipolarity of the interaction is an integer number, usually 6, 8 or 10.

The strength of the interaction given in units of $\mathrm{s}^{-1}\,\text{Å}^{\mathrm{n}}$, where n is the multipolarity. Optionally, an interaction strength for the average rate equation system can be given; this value is several orders of magnitude lower than the microscopic value. If it is not given and the average rate equations are solved, the normal strength will be used.

For example:

```
enery_transfer:
    CR50:
        process: Tm(1G4) + Tm(3H6) -> Tm(3H4) + Tm(3H5)
        multipolarity: 6
        strength: 4.3057e+09
        strength_avg: 8e+03 # optional, only for average system
    EM:
        process: Yb(ES) + Yb(GS) -> Yb(GS) + Yb(ES)
        multipolarity: 6
        strength: 4.50220614e+10
```

3.8 Optimization processes

An optional list of energy transfer labels. When optimizing the energy transfer parameters to the experimental data (see Section 5.4), by default all parameters will be used. This section restricts the optimization parameters to those in the list, for example

```
optimization_processes: [CR50]
```

will optimize only the parameter `CR50`, and not any other.

3.9 Optimization method

This optional section defines the method used to optimize the energy transfer parameters to the experimental data. Several options are available:

- SLSQP
- COBYLA
- L-BFGS-B
- basin_hopping
- brute_force

See the *Notes* section in the scipy minimize documentation. SLSQP (the default) or brute_force are recommended, as they seem to take the least time to arrive at the minimum.

3.10 Power dependence

This section is mandatory only if a power dependence simulation is performed. The starting and ending power density values and the number of steps need to be given

```
power_dependence: [1e0, 1e7, 8]
```

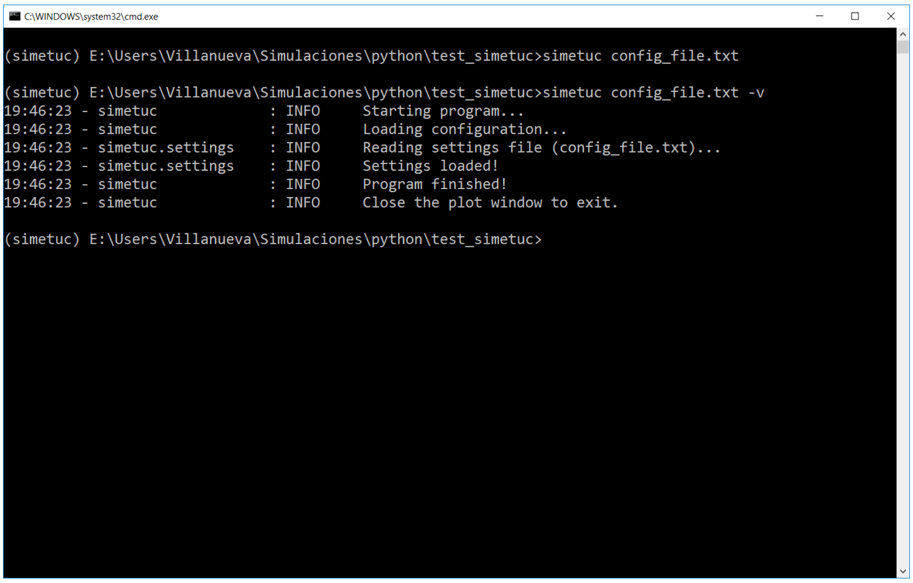
Note that it uses a logarithmic scale, so the above configuration will produce the power dependence at the following excitation power densities: 10^0, 10^1, 10^2, 10^3, 10^4, 10^5, 10^6, and 10^7 $\mathrm{W\,cm^{-2}}$. See also Section 5.6.

3.11 Concentration dependence

This section is mandatory only if a concentration dependence simulation is performed. It consists on a list of two lists; the first list gives the sensitizer concentrations at which the simulation will take place, the second gives the same for the activators.

```
concentration_dependence: [[0, 0.5], [0.1, 0.2, 0.3, 0.4, 0.5]]
```

Note that everything else will remain constant; therefore, if the number of unit cells is not high enough, at lower concentrations the quality of the simulation may be low. See also Section 5.7.



```
(simetuc) E:\Users\Villanueva\Simulaciones\python\test_simetuc>simetuc config_file.txt

(simetuc) E:\Users\Villanueva\Simulaciones\python\test_simetuc>simetuc config_file.txt -v
19:46:23 - simetuc              : INFO     Starting program...
19:46:23 - simetuc              : INFO     Loading configuration...
19:46:23 - simetuc.settings     : INFO     Reading settings file (config_file.txt)...
19:46:23 - simetuc.settings     : INFO     Settings loaded!
19:46:23 - simetuc              : INFO     Program finished!
19:46:23 - simetuc              : INFO     Close the plot window to exit.

(simetuc) E:\Users\Villanueva\Simulaciones\python\test_simetuc>
```

Figure 3.1: Example of a correct configuration file check with and without the verbose option.

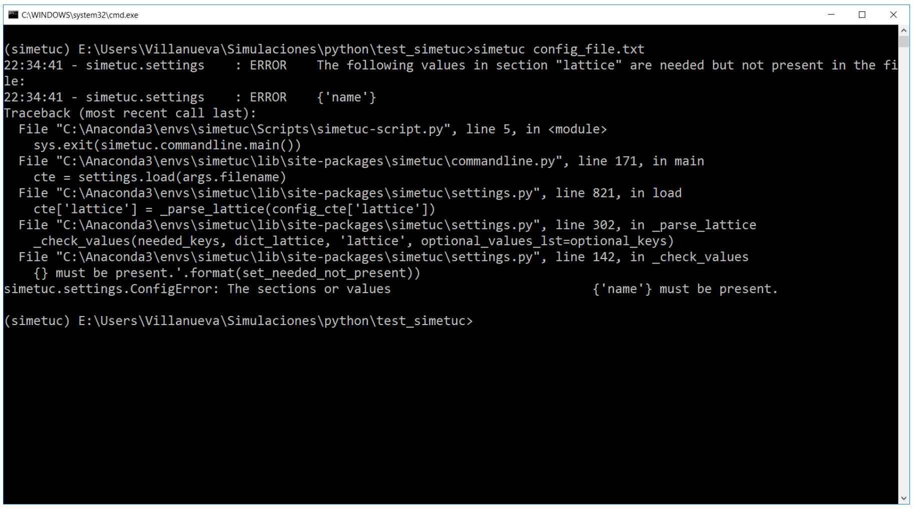
3.12 Checking the configuration file

To check that a configuration file is correct, execute:

```
simetuc config_file.cfg
```

If no warnings or errors are displayed, the file is correct. Nonetheless, it is still possible that some optional values are missing but needed for a particular simulation type or have wrong values (e.g., negative distances or concentrations above 100%). In that case, this check will not warn the user, and only trying to perform the actual simulation will show if something is missing. In any case, the program never fails silently, that is, it always warns the user if there was a problem and how to fix it.

An example of a configuration file check that succeeded is shown in Figure 3.1. An example of a check failure is shown in Figure 3.2; in this case, the name or the lattice has been omitted from the file.



```
(simetuc) E:\Users\Villanueva\Simulaciones\python\test_simetuc>simetuc config_file.txt
22:34:41 - simetuc.settings    : ERROR    The following values in section "lattice" are needed but not present in the fi
le:
22:34:41 - simetuc.settings    : ERROR    {'name'}
Traceback (most recent call last):
  File "C:\Anaconda3\envs\simetuc\Scripts\simetuc-script.py", line 5, in <module>
    sys.exit(simetuc.commandline.main())
  File "C:\Anaconda3\envs\simetuc\lib\site-packages\simetuc\commandline.py", line 171, in main
    cte = settings.load(args.filename)
  File "C:\Anaconda3\envs\simetuc\lib\site-packages\simetuc\settings.py", line 821, in load
    cte['lattice'] = _parse_lattice(config_cte['lattice'])
  File "C:\Anaconda3\envs\simetuc\lib\site-packages\simetuc\settings.py", line 302, in _parse_lattice
    _check_values(needed_keys, dict_lattice, 'lattice', optional_values_lst=optional_keys)
  File "C:\Anaconda3\envs\simetuc\lib\site-packages\simetuc\settings.py", line 142, in _check_values
    {} must be present.'.format(set_needed_not_present))
simetuc.settings.ConfigError: The sections or values                           {'name'} must be present.

(simetuc) E:\Users\Villanueva\Simulaciones\python\test_simetuc>
```

Figure 3.2: Example of a wrong configuration file check. The two upper lines show the user-friendly error message; they can also be found in the error log (see Section 4.1.2).

Chapter 4

The logs and results folders

simetuc creates several subfolders during its usage; these subfolders are located under the current working directory (the directory from which *simetuc* is being used). Additionally, the experimental data must be saved in a folder with a specific format, see Section 5.3.1.

4.1 The log folder

Any operation of *simetuc* writes its status and current tasks on several log files, so the user can analyze what operations were performed, when, and whether they succeeded. The log files are stored in a subfolder called `logs`, which is created if it doesn't exist.

4.1.1 Normal log

This is the primary log. Here all parts of the program register their operations. For example, any operation begins with:

```
2017-01-10 19:13:51,227 - simetuc   : INFO    Starting program...
```

The log format is the following:

```
YYYY-MM-DD HH:MM:SS,mmm - simetuc[.part] : LEVEL message
```

The main program emits messages under the name *simetuc*, other parts of the program have names such as *simetuc.lattice* or *simetuc.simulations*. The log level can be DEBUG, INFO, WARNING, or ERROR.

4.1.2 Error log

Whenever an operation was requested, but couldn't be completed due to an user error or an internal bug, the errors are written to an error log. This is log remains empty if all operations succeeded.

4.1.3 Debug log

This log registers a lot of low level information that is usually not important, but can be relevant in the case of errors.

4.2 The results folder

All simulations produce some results that are stored in the subfolder results with the hdf5 format.

4.3 Other folders

The lattice folder, Section 5.1.1.

The experimental data folder, see Section 5.3.1.

Chapter 5

Simulations

5.1 Creating a lattice

5.1.1 The latticeData folder

5.2 Simulating the dynamics

5.3 Adding experimental data

5.3.1 The expData folder

5.4 Optimizing the ET parameters

5.5 Simulating the steady state

5.6 Simulating the power dependence of the steady state

5.7 Simulating the concentration dependence of the the steady state or of the dynamics

5.8 Microscopic or average rate equations

Chapter 6

Publications

This software has been described and used in these publications:

- Villanueva-Delgado, P.; Krämer, K. W. & Valiente, R. Simulating Energy Transfer and Upconversion in β-$NaYF_4$: Yb^{3+}, Tm^{3+}, *J. Phys. Chem. C*, **2015**, *119* (41), pp 23648–23657. DOI: 10.1021/acs.jpcc.5b06770.

- Villanueva-Delgado, P.; Krämer, K. W.; Valiente, R.; de Jong, M. & Meijerink, A. Modeling Blue to UV Upconversion in β-$NaYF_4$: Tm^{3+}, *Phys. Chem. Chem. Phys.*, **2016**, *18*, pp 27396-27404, DOI: 10.1039/C6CP04347J.

If you use this software in a scientific publication, please cite the appropriate articles above.

Chapter 7

Acknowledgments

The financial support of the EU FP7 ITN LUMINET (Grant agreement No. 316906) is gratefully acknowledged.

This work was started at the University of Cantabria under Prof. Rafael Valiente and continued at the University of Bern under PD Dr. Karl Krämer.

Chapter 8

License

Copyright Pedro Villanueva Delgado, 2016-2017.

Distributed under the terms of the MIT license, *simetuc* is free and open source software.

MIT License

Part II

Appendix

Chapter 9

Example configuration file

Listing 9.1: Exampe configuration file

```
# settings file
# USE SPACES AND NOT TABS

version: 1 # mandatory, only 1 is supported at the moment

# mandatory section
lattice:
# all fields here are mandatory
    name: bNaYF4
    N_uc: 40

    # concentration
    S_conc: 0
    A_conc: 0.3

    # unit cell
    # distances in Angstrom
    a: 5.9738
    b: 5.9738
    c: 3.5297
    # angles in degree
    alpha: 90
    beta: 90
    gamma: 120

    # the number is also ok for the spacegroup
    spacegroup: P-6

    # info about sites.
    # If there's only one site, use:
    # sites_pos: [0, 0, 0]
    # sites_occ: 1
    sites_pos: [[0, 0, 0], [2/3, 1/3, 1/2]]
    sites_occ: [1, 1/2]

    # optional
    # maximum distance of interaction for normal ET and for cooperative
    # if not present, both default to infinite
    d_max: 100.0
    # it’s strongly advised to keep this number low,
    # the number of coop interactions is very large (~num_atoms^3)
    d_max_coop: 25.0

# mandatory section
states:
# all fields here are mandatory,
# add any label if necessary
# i.e.: just "sensitizer_ion_label: label" on a line
# but don’t delete them.
# If you set the sensitizer concentration to zero,
# they will be ignored in all calculations
    sensitizer_ion_label: Yb
    sensitizer_states_labels: [GS, ES]
    activator_ion_label: Tm
    activator_states_labels: [3H6, 3F4, 3H5, 3H4, 3F3, 1G4, 1D2]

# mandatory section
excitations:
# the excitation label can be any text
# the t_pulse value is only mandatory for the dynamics
# at least one excitation must be present and active
    Vis_473:
        active: True
        power_dens: 1e6 # power density W/cm^2
        t_pulse: 1e-8 # pulse width, seconds
        process: Tm(3H6) -> Tm(1G4) # both ion labels are required
        degeneracy: 13/9 # initial_state_g/final_state_g
        pump_rate: 9.3e-4 # absorption cross-section/energy in cm2/J
    NIR_1470:
        active: False
        power_dens: 1e6
        t_pulse: 1e-8
        process: Tm(1G4) -> Tm(1D2)
        degeneracy: 9/5
        pump_rate: 2e-4
    NIR_980:
        active: False
        power_dens: 1e7
        t_pulse: 1e-8
        process: Yb(GS)->Yb(ES)
        degeneracy: 4/3
        pump_rate: 4.4e-3
    NIR_800: # ESA: list of processes, degeneracies and pump rates
        active: False
        power_dens: 1e2
        t_pulse: 1e-8
        process: [Tm(3H6)->Tm(3H4), Tm(3H5)->Tm(1G4)] # list
        degeneracy: [13/9, 11/9] # list
        pump_rate: [4.4e-3, 4e-3] # list

# mandatory section
sensitizer_decay:
# lifetimes in s
    ES: 2.5e-3

# mandatory section
activator_decay:
# lifetimes in s
    3F4: 12e-3
    3H5: 25e-6
    3H4: 2e-3
    3F3: 2e-6
    1G4: 760e-6
    1D2: 67.5e-6

# optional section
sensitizer_branching_ratios:

# optional section
activator_branching_ratios:
    3H5->3F4: 0.4
    3H4->3F4: 0.3
    3H4->3H5: 0.1
    3F3->3H4: 0.999
    1G4->3F4: 0.15
    1G4->3H5: 0.16
    1G4->3H4: 0.04
    1G4->3F3: 0.001
    1D2->3F4: 0.43

# optional section
enery_transfer:
# name:
#   process: ion_label(state_label) + ion_label(state_label) -> ion_label(state_label) + ion_label(state_label)
#   multipolarity, and strength (in s^(-1)*Angstrom^(multipolarity))
    CR50:
        process: Tm(1G4) + Tm(3H6) -> Tm(3H4) + Tm(3H5)
        multipolarity: 6
        strength: 4.3057e+09
        # strength_avg is used only for average rate equations
        # if it isn’t present and average equations are solved
        # the strength value will be used, which is probably too high
        strength_avg: 8e+03
    ETU53:
        process: Tm(1G4) + Tm(3H4) -> Tm(1D2) + Tm(3F4)
        multipolarity: 6
        strength: 2.5377e+08
        strength_avg: 4e+02
    ETU55:
        process: Tm(1G4) + Tm(1G4) -> Tm(1D2) + Tm(3F3)
        multipolarity: 6
        strength: 0 # 4.50220614e+7
    BackET:
        process: Tm(3H4) + Yb(GS) -> Tm(3H6) + Yb(ES)
        multipolarity: 6
        strength: 4.50220614e+3
    EM:
        process: Yb(ES) + Yb(GS) -> Yb(GS) + Yb(ES)
        multipolarity: 6
        strength: 4.50220614e+10
    ETU1:
        process: Yb(ES) + Tm(3H6) -> Yb(GS) + Tm(3H5)
        multipolarity: 6
        strength: 1e2
    ETU2:
        process: Yb(ES) + Tm(3F4) -> Yb(GS) + Tm(3F3)
        multipolarity: 6
        strength: 1e8
    ETU3:
        process: Yb(ES) + Tm(3H4) -> Yb(GS) + Tm(1G4)
        multipolarity: 6
        strength: 1e8
    ETU4:
        process: Yb(ES) + Tm(1G4) -> Yb(GS) + Tm(1D2)
        multipolarity: 6
        strength: 0 #1e4
    coop1:
        process: Yb(ES) + Yb(ES) + Tm(3H6) -> Yb(GS) + Yb(GS) + Tm(1G4)
        multipolarity: 8
        strength: 0 # 1e8

# this is optional: a list of enery_transfer labels to optimize.
# the fewer the number of parameters, the faster the optimization
optimization_processes: [CR50, ETU53]

# optional: method for optimization of ET parameters. It can be:
# SLSQP, COBYLA, L-BFGS-B, basin_hopping, or brute_force.
# SLSQP or brute_force are recommended.
optimize_method: SLSQP

# minimum and maximum excitation powers and the number of points
# to calculate the power dependence
# note: a logarithmic scale is used
# it’s only mandatory if you want to calculate the power dependence
power_dependence: [1e0, 1e7, 8]

# list of two lists:
# [[sensitizer concentrations], [activator concentrations]]
# to simulate for the concentration dependence
# it’s only mandatory if you want to calculate the concentration dependence
concentration_dependence: [[0], [0.1, 0.2, 0.3, 0.4, 0.5]]
```