

# CAP optimization manual

Mathis Boutrouelle

July 2022

## 1 Introduction

This algorithm aims to determine the optimal Complex Absorbing Potential for a specific system, so that it produces a transmission as close as possible from the exact system:

$$T_{CAP}(E, k) = T_{obj}(E, k)$$

It is done by minimizing the square of the difference between the exact and the CAP-computed transmission, with various k processing methods that can be chosen by the user. The python module *scipy.optimize* is used for that purpose.

## 2 Complex absorbing potential approach

In the Complex Absorbing Potential (CAP) approximation, the environment is represented by a finite part close to the system.

Under the NEGF (Non-Equilibrium Green Function) formalism, the Hamiltonian of the system is written as following:

$$H' = \begin{bmatrix} H_L + W_L & V_L^\dagger & 0 \\ V_L & H_D & V_R^\dagger \\ 0 & V_R & H_R + W_R \end{bmatrix}$$

Where  $W_{L/R}$  is the CAP matrix. We write

$$\mathbf{W}_{\mu,v} = \langle \mu | W | v \rangle = v_{\mu\nu} - iw_{\mu\nu}, \quad v_{\mu\nu} \in R, w_{\mu\nu} \in R^+$$

The imaginary part of the self-energy can be rewritten as

$$\Gamma_{L/R} = -2\Im[W_{L/R}]$$

We compute the Green function of the system

$$G(E, k) = [(E + i\eta)I - H'(k)]^{-1}, \quad \eta \ll 1$$

The transmission coefficient is defined as

$$T(E, k) = \text{Tr}[\mathbf{\Gamma}_L(E, k) \mathbf{G}(E, k) \mathbf{\Gamma}_R(E, k) \mathbf{G}^\dagger(E, k)]$$

$\Gamma$  must be positive-definite (i.e. with strictly positive eigenvalues) to ensure that the transmission is positive.

### 3 Content

- `job_opt` : job file to run the algorithm on the cluster
- `cap_opt.py` : python algorithm
- `input.csv` : user inputs for the algorithm
- `Tobj` : folder containing the exact transmission as a `.numpy` file
- `geom` : folder containing the geometry or hamiltonian of the device, left and right electrode, as `.xyz` or `.TSHS` files
- `res_opt` : folder containing the result of the optimization, as a `.numpy` file

### 4 Input

- ***Emin/Emax*** : minimum/maximum energy defining the optimization range.
- ***ne*** : number of energy points.
- ***kmin/kmax*** : minimum/maximum transverse periodic boundary condition number defining the optimization range.
- ***nk*** : number of k-points.
- ***eta*** : infinitesimal imaginary part of energy, for convergence of Green function calculation.
- ***axis*** : transport direction (0, 1 or 2).
- ***dev\_file/elec\_left\_file/elec\_right\_file*** : name of device, left electrode and right electrode file inside the `geom` folder. The device must contain the left and right electrode, with same orbital order than in the electrode files, and in this order: left - device - right.
  - if `.xyz` format, it is assumed to be a graphene-like system, therefore the hamiltonian is defined with the hopping elements that are written at the beginning of `cap_opt.py` file.
  - if `.TSHS` format, the hamiltonian is directly used as is all 3 files must be in the same format.
- ***nsc*** : number of super cells, list of 3 ints like "[1,3,1]". The supercell number in the transport direction must be 1. In the periodic boundary condition direction it is typically 3.
- ***Tobj\_file*** : name of exact transmission inside the `Tobj` folder. These are `.numpy` files with shape (ne,nk). If `Tobj_file` is set to `None`, the transmission will be recomputed.

- **kproc** : either "none", "avg" or "inde".
  - if "**none**" each k-point is considered in the least square objective function :  $\min_{CAP} \sum_E \sum_k (T_{CAP}(E, k) - T_{obj}(E, k))^2$
  - if "**avg**" the average of the transmission over k is considered in the least square objective function :  $\min_{CAP} 1/n_k^2 \sum_E [\sum_k (T_{CAP}(E, k) - T_{obj}(E, k))]^2$
  - if "**inde**" a different optimized CAP is computed for each k-point :  $\forall k, \min_{CAP(k)} \sum_E (T_{CAP(k)}(E, k) - T_{obj}(E, k))^2$
- **tol** : termination condition of the optimization algorithm, the gradient of the objective function must be inferior to this value.
- **save\_file** : name of the .npy file where the optimized CAP is saved, inside the res\_opt folder.
- **mode** : list or string of keywords, the possible keywords are "diag, real, hop, X0\_zero, sym, transym".
  - if "**diag**" the on-site potential of each orbital are optimised, the imaginary part is negative to satisfy the condition of definite-positiveness of  $\Gamma$ .  $w_{\mu\mu}$  is a variable, with bound  $w_{\mu\mu} > 0$ . By default,  $v_{\mu\mu} = 0$ , the CAP is pure imaginary.
  - if "**hop**" the hopping elements between orbitals are added to the optimization,  $W$  is now a symmetric matrix. This requires a new condition for  $\Gamma$  to be definite-positive. A simple and linear condition is enforcing the matrix to be diagonally dominant :  $\forall \mu, w_{\mu\mu} > \sum_{\nu \neq \mu} w_{\mu\nu}$ ,  $\forall \mu\nu, w_{\mu\nu} > 0$ . Thus, linear constraints are added to the optimization problem, resulting in much higher computation time. NOT COMPATIBLE WITH "real" AND "transym". WHEN "hop" IS ACTIVATED, "sym" MUST BE ACTIVATED ASWELL.
  - if "**real**" the elements of the CAP (diag or/and hop) have also a real part, independent from the imaginary part.  $v_{\mu\nu}$  is non-zero, subject to no bounds or constraints. This results in twice the number of variables to optimize so a multiplication of the computation time.
  - if "**X0\_zero**" the first guess of the optimization process is a zero matrix. If "X0\_zero" is not in mode, then a standard diagonal CAP is used as a first approximation :  $W_{\mu\mu} = -i \frac{\hbar^2}{2m} \left( \frac{2\pi}{r_1} \right)^2 \frac{4}{c^2} \left[ \frac{r_1^2}{(r_1 - r_{\mu\mu})^2} + \frac{r_1^2}{(r_1 + r_{\mu\mu})^2} - 2 \right]$ , where  $r_1$  is the length of the electrode along the transport direction,  $r_{\mu\mu}$  is the distance of orbital  $\mu$  to the device, projected onto the transport direction, and  $c = 2.62$ .
  - if "**sym**" the right and left electrodes are symmetric. The the same CAP are added to both electrodes:  $W_L = W_R$ . Use the '.mirror()' method in sisl to build the electrodes. This divides by two the number of variables and improves greatly the efficiency.

- if "**transym**" both electrodes are symmetric along the periodic boundary condition direction:  $W_{\mu\nu} = W_{\mu+no/2, \nu+no/2}$ . This divides by two the number of variables and improves greatly the efficiency.

## 5 Run

The algorithm can be run either by running directly "*python3 cap\_opt.py*" in a command shell, or by running "*bsub <job\_opt*" on the cluster. The lines in "*opt\_input.csv*" will be executed one after the other. After it is finished, all the information about the run can be found in *registre.csv* and the result in the *res\_opt* folder.