BTE-Barna examples

This document is intended to provide working examples for all of the BTE-Barna simulators. The provided examples are intended to be production ones, and thus the computational workload in some cases can be intensive. Before starting running the examples, it is necessary to create the materials database; those sections are tagged with [MANDATORY].

Contents

1 [MANDATORY] Previous step: almaBTE database

Directory: database
Purpose: The aim of this provide the needed information (structural data and phonon properties) for next steps.

We provide the necessary inputs to generate the database for: graphene, phosphorene, and hBN-encapsulated graphene. For converged results, we recommend the following parameters:

Graphene

q-mesh: $80 \times 80 \times 1$

broadening parameter: 1

thickness 0.335 nm

Phosphorene

q**-mesh:** $50 \times 50 \times 1$

broadening parameter: 1

thickness 0.533 nm

h-BN encapsulated graphene (aka sandwich)

q**-mesh:** $40 \times 40 \times 1$

broadening parameter: 1

thickness 1.001 nm

Be aware that the following examples assume those parameters, so if one chooses different ones, it would need to change input files and symbolic links accordingly.

The `VCAbuilder` input is given for each of those materials is provided. We recommend using our `VCAbuilder` implementation as it implements the symmetric adaptive smearing broadening.

2 [MANDATORY: `beRTAMC2D`] Propagator generation

Directory: propagator
Purpose: The aim of this is to provide an example of the use of `PropagatorBuilder`.

Here we indicate how to perform the calculation of the propagator for phosphorene at 300 K for a time step of 0.25 ps. Link the phosphorene `almaBTE` database into the folder, and then to perform the computation, run:

```
PropagatorBuilder phosphorene_50_50_1.h5
    FORCE_CONSTANTS_3RD 300.0 0.25 50 &> propagator_50_025.log
```

Be aware that in 50 threads the calculation takes around 36 h to complete. Therefore, this should only be done in the case one wants to run the `beRTAMC2D` example.

3 Phosphorene nanoribbons effective thermal conductivity

Directory: nanoribbons
Purpose: The aim of this is to provide an example of the use of `kappa_Tsweep_nanos`.

Inside this folder, one can find the inputs to generate the effective thermal conductivity for phosphorene nanoribbons in AC and ZZ directions for several widths. Before starting, link the phosphorene `almaBTE` database into the folder.

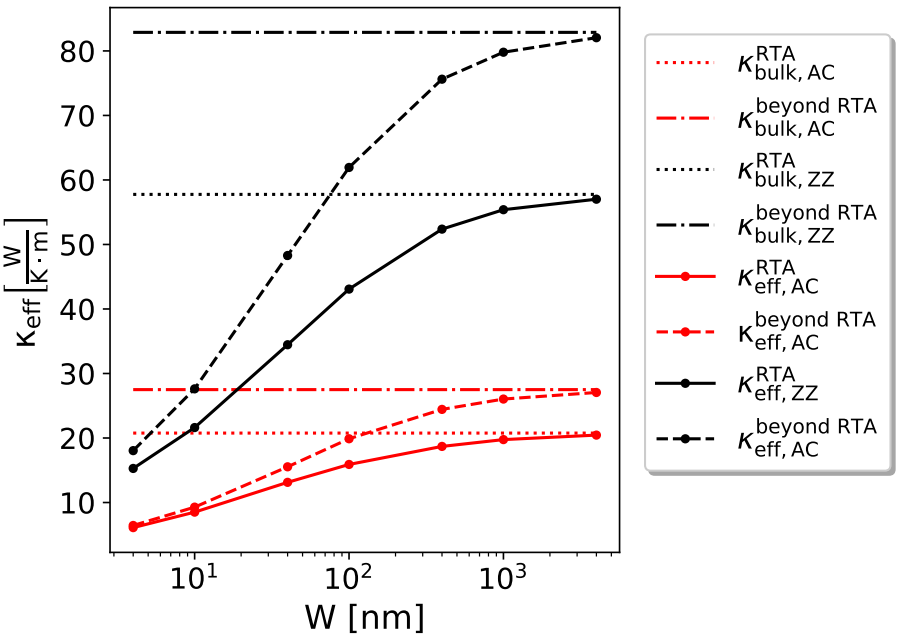

Figure 1: Effective thermal conductivity for AC and ZZ nanoribbons of different widths at 300 K obtained through the direct (RTA) and iterative (beyond the RTA) solution of the linerized-PBTE. Bulk values are provided as reference.

Those input files are run for instance using 50 threads as:

```
kappa_Tsweep_nanos kxx_1000.xml 50 &> kxx_1000.log
```

After all the simulations are finished, one can execute `get_nanos.py` to obtain the effective thermal conductivity as a function of the nanoribbon width for the AC and ZZ direction at 300 K (see Fig. 1). Be aware, that results are re-scaled to take into account the difference between the DFT length in the out-of-plane direction and the real thickness of the 2D material.

4 Phosphorene bar heating: RTA, bRTA and Fourier comparison

Directory: example_bar
Purpose: The aim of this is to provide an example of how to use the Monte Carlo simulators and the post-processing tool `dist_reader`.

Before starting, link the phosphorene `almaBTE` database into the folders. Moreover, link the propagator into the *berta* directory.

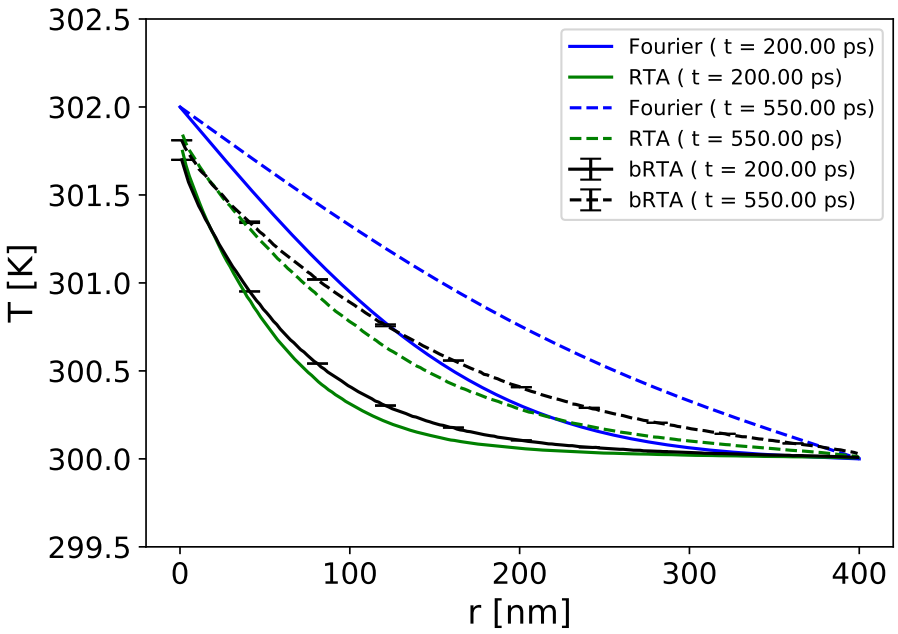

Figure 2: Temperature profiles obtained with the RTA (green), bRTA (black) and Fourier (blue) approaches as functions of position for an AC-phosphorene bar at 200 ps (solid) and 550 ps (dashed).

First, run `RTAMC2D` simulation within the *rta* directory:

```
RTAMC2D input.xml 50 1 &> rta.log
```
(1)

where the simulation is run in 50 threads a single time. This will generate the temperature and heat flux temperature for the selected time steps, as well as the spectral decomposition for some of the boxes.

Second, run `beRTAMC2D` simulation within *berta* directory:

```
beRTAMC2D input.xml 50 &> berta.log
```
(2)

this will dump energy deviations and fluxes to `berta.log` and the deviational distribution to `properties.msgpack.bin`. After this finishes, run `dist_reader` within the same folder, to obtain the spectral decompositions.

Finally, execute `fourier.py` and `fourier.jy.py` to obtain a comparative of the thermal profile between Fourier, RTA, and beyond the RTA (see Fig. 2). Additionally, you will obtain the deviational temperature and the heat flux in the ZZ direction spectral decomposition (see Figs. 3-4).

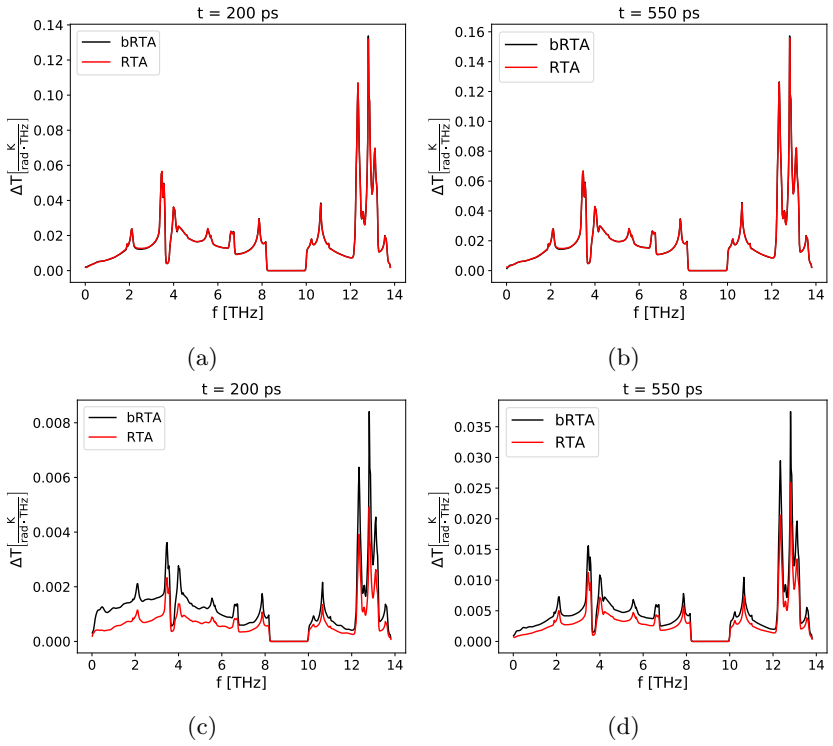

Figure 3: RTA (red) and bRTA (black) spectral decomposed temperature deviations for 400 nm ZZ-phosphorene bar at 15 nm (a and b) and 203 nm (c and d) from the hot edge, at times 200 ps and 550 ps.

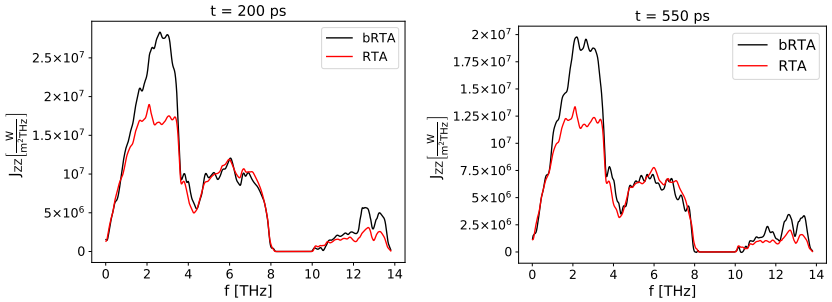

Figure 4: RTA (red) and bRTA (black) spectral decomposed heat flux in the ZZ-direction for 400 nm ZZ-phosphorene bar at 15 nm (a and b) and 203 nm (c and d) from the hot edge, at times 200 ps and 550 ps.

5 Interface structure: diffusive mismatch model vs localized diffusive mismatch model

Directory: interface
Purpose: The aim of this is to provide an example of how to use `RTAMC2D` to simulate structures with more than one material, using the two models implemented within `BTE-Barna`.

Here we simulate the temperature profile and fluxes of a heterojunction of graphene with hBN-encapsulated graphene. To do so, link the database *dmm* and *ldmm* folders, and then run the simulation in each of them using the following command:

```
RTAMC2D input.xml 25 10 &> rta.log
```
(3)

. The `input.xml` contain all the tags for the LDMM simulation, namely:

1. layer/material [string,optional]: name of material in which layer is localized.
2. layer/layer_name [string,optional]: name of layer.
3. layer/atoms [int,optional]: identity of atoms in the layer (follows the order of POSCAR).
4. layers_connection/layer_A [string,optional]: name of layer connected with layer_B (of same entry). There can be multiple entries of layer_connection if more than one connection is present.
5. layers_connection/layer_B [string,optional]: name of layer connected with layer_A (of same entry).

Execute the python executable within the folders to obtain the temperature profiles and fluxes using either DMM (Fig. 5) or LDMM (Fig. 6).

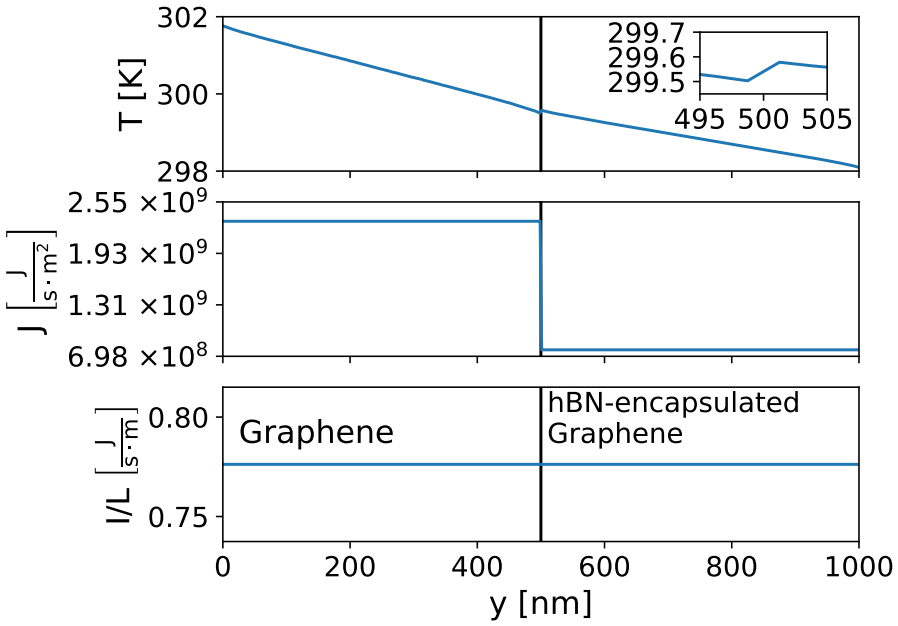

Figure 5: Thermal profile, flux and heat intensity per unit of length in graphene/hBN-encapsulated graphene obtained using the RTA and the traditional DMM to model interface scattering. Inset: Zoom of thermal profile at the interface.

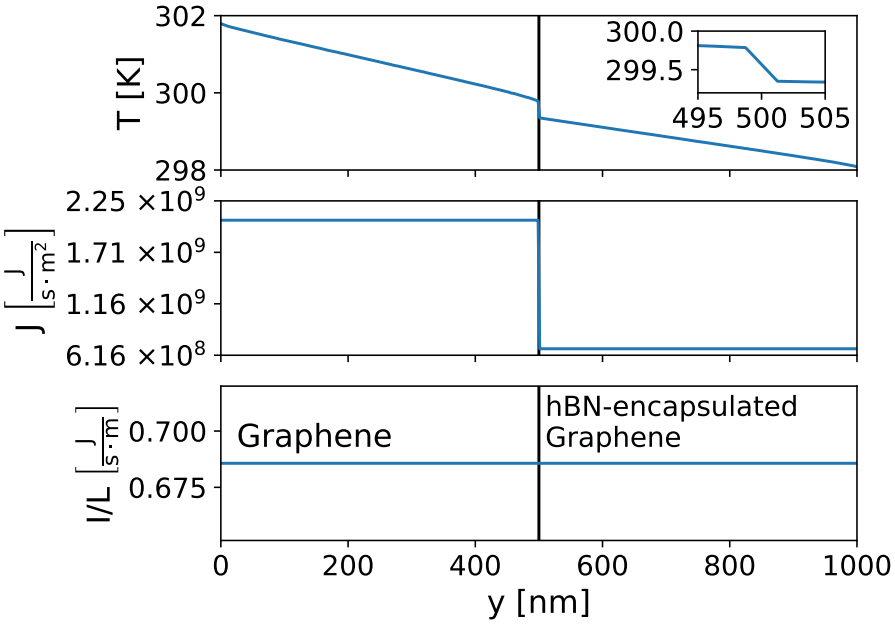

Figure 6: Thermal profile, flux and heat intensity per unit of length in graphene/hBN-encapsulated graphene obtained using the RTA and the LDMM to model interface scattering. Inset: Zoom of thermal profile at the interface