Table of Contents

Phonon-limited mobility in graphene using the Boltzmann transport equation

Version: Q-2019.12-SP1

Downloads & Links

PDF version
`DynamicalMatrix`
`HamiltonianDerivatives`
`ElectronPhononCoupling`
`Mobility`
Basic QuantumATK Tutorial
QuantumATK Reference Manual

In this tutorial you will learn how to calculate the phonon-limited mobility in graphene. The mobility will be calculated using the Boltzmann transport equation (BTE) with the electronic structure, phonons and electron-phonon coupling calculated using density functional theory (DFT).

The mobility μ will be calculated using two different methods to calculate the relaxation times τ entering the BTE:

1. **Full angular (k,q)-dependence**: in this method, the full dependency of τ on the electron and phonon wave vectors k and q is taken into account, so that $\tau = \tau(\mathrm{k}, \mathrm{q})$. In the following, we will refer to this as the **(k,q)-dependent method**.
2. **Isotropic scattering rate**: in this method, only the energy dependence of τ is considered, so that $\tau = \tau(E)$, and $\tau(E)$ is assumed to vary isotropically in momentum-space. In the following, we will refer to this as the **E-dependent method**.

As it will be shown, the two methods give essentially the same results, but the second method is considerably faster than the first one.

In the theory section, you can read about the theoretical background. A more comprehensive description can also be found in the paper [1].

Geometry and electronic structure of graphene

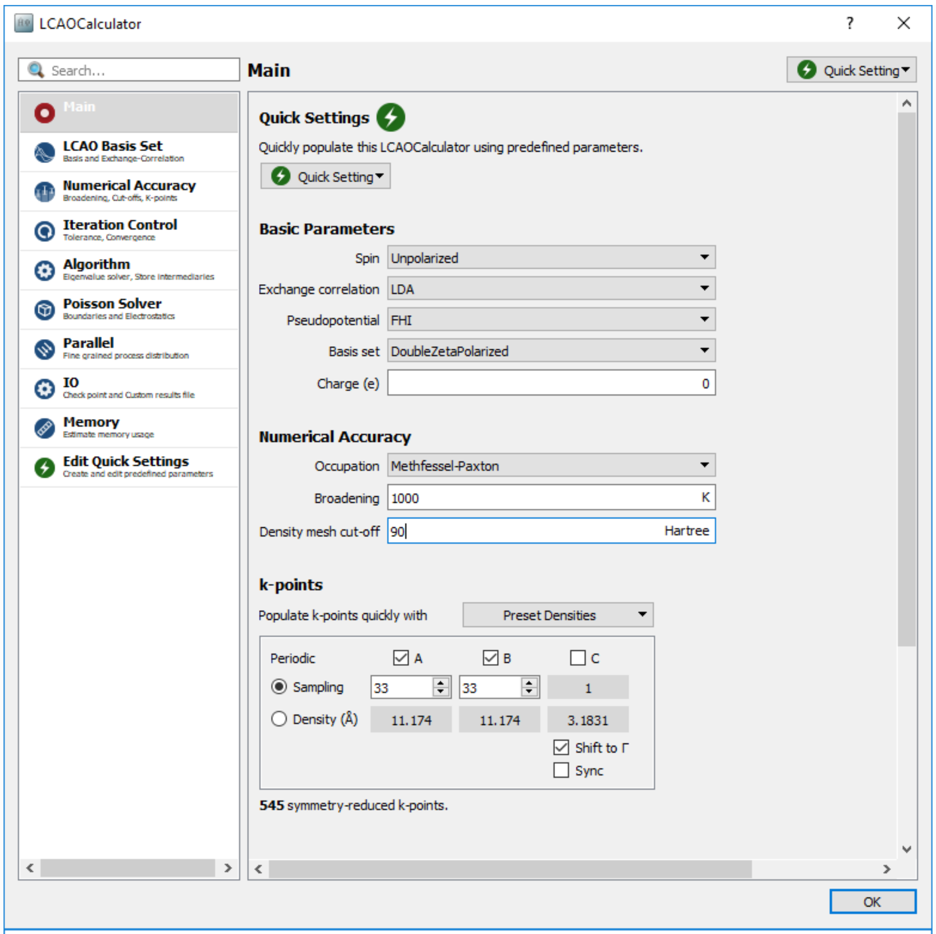
In the **Builder**, add a graphene configuration by clicking Add ‣ From Database and searching for '*Graphene*' in the structure database.

Next, you have to increase the vacuum gap above and below the graphene. Click on Bulk Tools ‣ Lattice parameters and set the lattice parameter along the C-direction to C = 20 Å.

Center the configuration by clicking on Coordinate Tools ‣ Center, and press *Apply*. Now click on the button to send the structure to the **Script Generator**. In the main panel of the **Script Generator**, set the **Results file** to `Graphene_relax.hdf5`.

To perform an accurate calculation of the relaxation times
τ and the mobility
μ, the first important step is to optimize the lattice parameters A and B and calculate the electronic band structure. In order to do this, add an **LCAOCalculator** block and in the *Main* section set the following parameters:

- Calculators ‣ LCAOCalculator
    - Set the **Exchange correlation** to *LDA*
    - Set the **Pseudopotential** to *FHI*
    - Set the **Occupation method** to *Methfessel-Paxton*
    - Set the **Density mesh cutoff** to *90 Ha*
    - Set the **k-point sampling** to:
        - k_A = 33
        - k_B = 33
        - k_C = 1
    - On the **Iteration Control** tab, Set the **Tolerance** to *0.000001*, i.e. 10^{-6}


LCAOCalculator
Search...
Main
Quick Setting
LCAO Basis Set
Basis and Exchange-Correlation
Numerical Accuracy
Broadening, Cut-offs, K-points
Iteration Control
Tolerance, Convergence
Algorithm
Eigenvalue solver, Store intermediaries
Poisson Solver
Boundaries and Electrostatics
Parallel
Fine grained process distribution
IO
Check point and Custom results file
Memory
Estimate memory usage
Edit Quick Settings
Create and edit predefined parameters
OK

Quick Settings

Quickly populate this LCAOCalculator using predefined parameters.

Basic Parameters

Spin	Unpolarized
Exchange correlation	LDA
Pseudopotential	FHI
Basis set	DoubleZetaPolarized
Charge (e)	0

Numerical Accuracy

Occupation	Methfessel-Paxton
Broadening	1000 K
Density mesh cut-off	90 Hartree

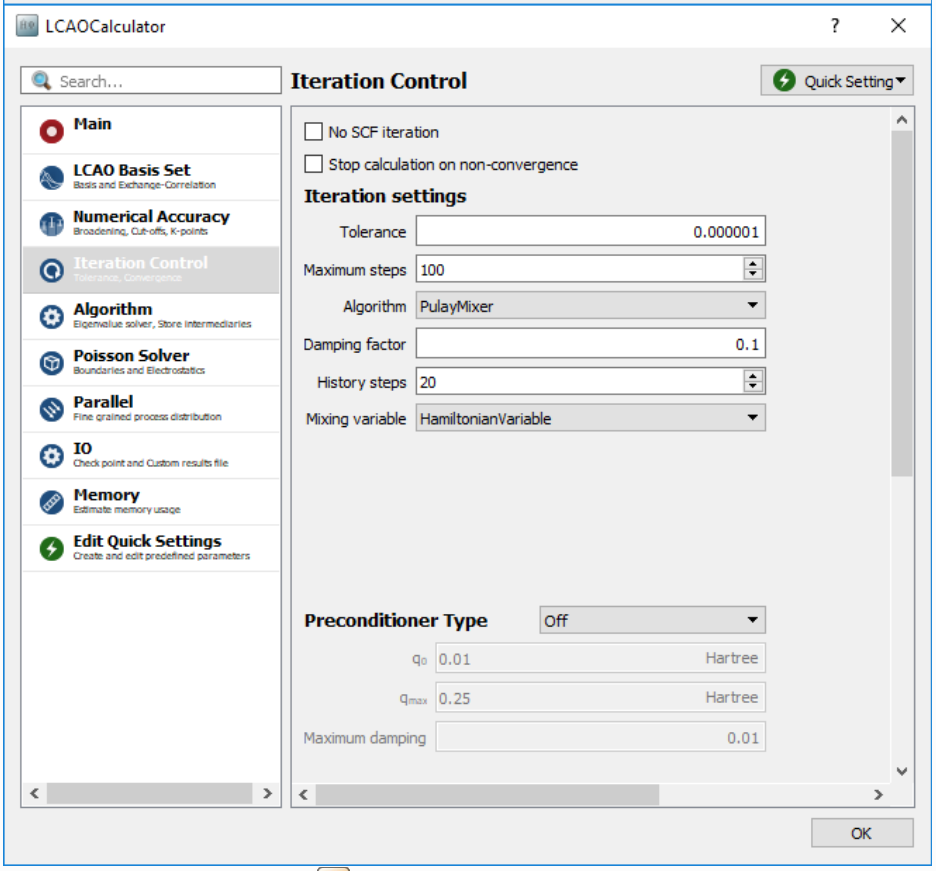
k-points

Populate k-points quickly with Preset Densities

Periodic	☑ A	☑ B	☐ C
● Sampling	33	33	1
○ Density (Å)	11.174	11.174	3.1831

☑ Shift to Γ
☐ Sync

545 symmetry-reduced k-points.


LCAOCalculator
Search...
Iteration Control
Quick Setting
Main
LCAO Basis Set
Basis and Exchange-Correlation
Numerical Accuracy
Broadening, Cut-offs, K-points
Iteration Control
Tolerance, Convergence
Algorithm
Eigenvalue solver, Store intermediaries
Poisson Solver
Boundaries and Electrostatics
Parallel
Fine grained process distribution
IO
Check point and Custom results file
Memory
Estimate memory usage
Edit Quick Settings
Create and edit predefined parameters
OK

☐ No SCF iteration
☐ Stop calculation on non-convergence

Iteration settings

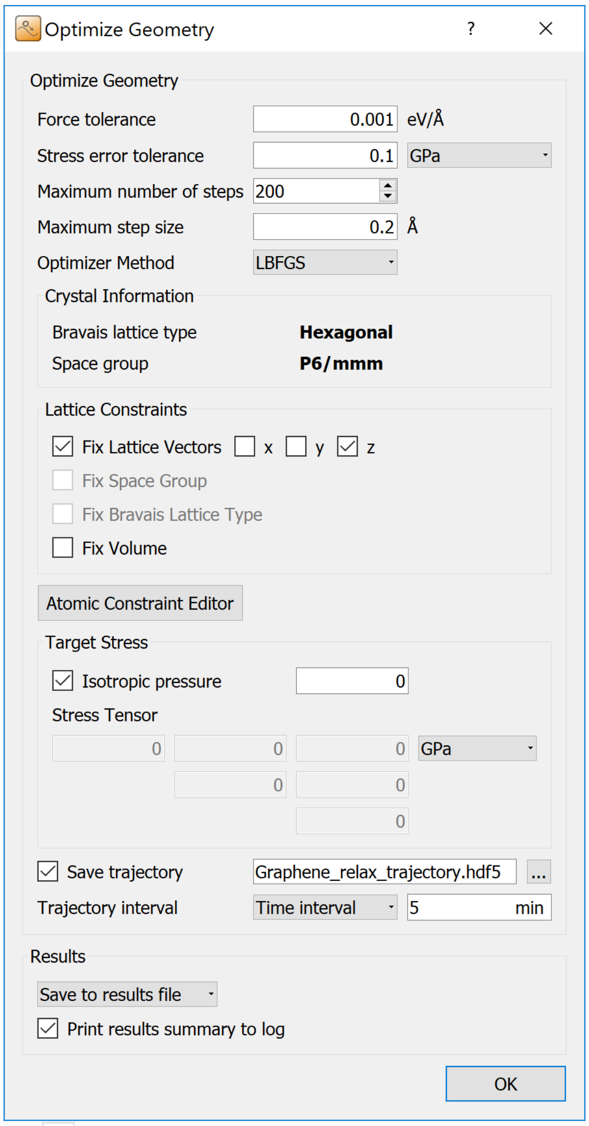
Tolerance	0.000001
Maximum steps	100
Algorithm	PulayMixer
Damping factor	0.1
History steps	20
Mixing variable	HamiltonianVariable

Preconditioner Type Off

q_0	0.01 Hartree
q_{max}	0.25 Hartree
Maximum damping	0.01

- Optimization ▸ OptimizeGeometry
  - Set the **Force tolerance** to *0.001 eV/Å*
  - Untick **Fix lattice vectors** in the

x and
y directions.


Optimize Geometry

Optimize Geometry

Force tolerance	0.001	eV/Å
Stress error tolerance	0.1	GPa
Maximum number of steps	200	
Maximum step size	0.2	Å
Optimizer Method	LBFGS	

Crystal Information

Bravais lattice type	**Hexagonal**
Space group	**P6/mmm**

Lattice Constraints

- ✓ Fix Lattice Vectors ☐ x ☐ y ✓ z
- ☐ Fix Space Group
- ☐ Fix Bravais Lattice Type
- ☐ Fix Volume

Atomic Constraint Editor

Target Stress

- ✓ Isotropic pressure 0

Stress Tensor

0	0	0	GPa
	0	0	
		0	

- ✓ Save trajectory Graphene_relax_trajectory.hdf5
- Trajectory interval: Time interval 5 min

Results

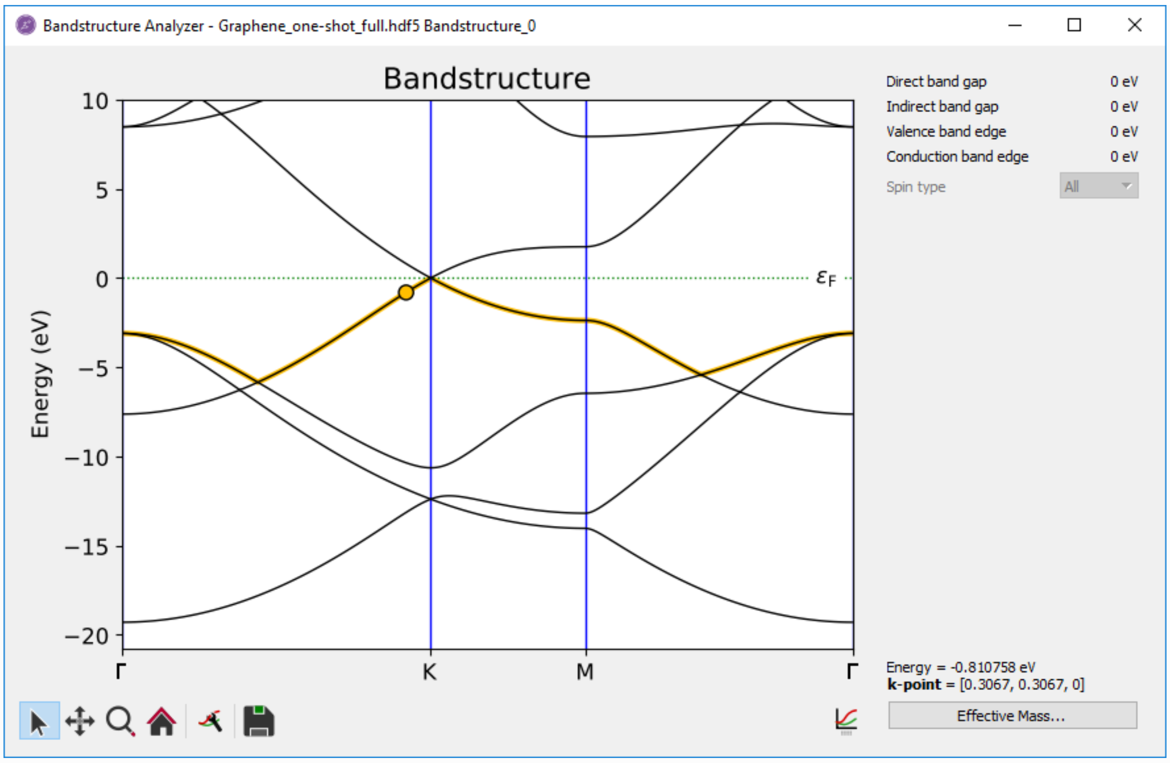
- Save to results file
- ✓ Print results summary to log

OK

- Analysis ‣ Bandstructure
    - Set the number of **Points per segment** to *100*
    - Set the **Brillouin zone route** to *G, K, M, G*

Now, send the script to the **Job manager**, save it as `Graphene_relax.py`, and click on the button to run the calculation.

When the calculation is done, click on the $\{\epsilon_i(\mathbf{k})\}$ **Bandstructure** object contained in the file `Graphene_relax.hdf5` on the **LabFloor** and use the **Bandstructure Analyzer** to visualize the band structure.



Direct band gap 0 eV
Indirect band gap 0 eV
Valence band edge 0 eV
Conduction band edge 0 eV
Spin type All

Energy = -0.810758 eV
k-point = [0.3067, 0.3067, 0]

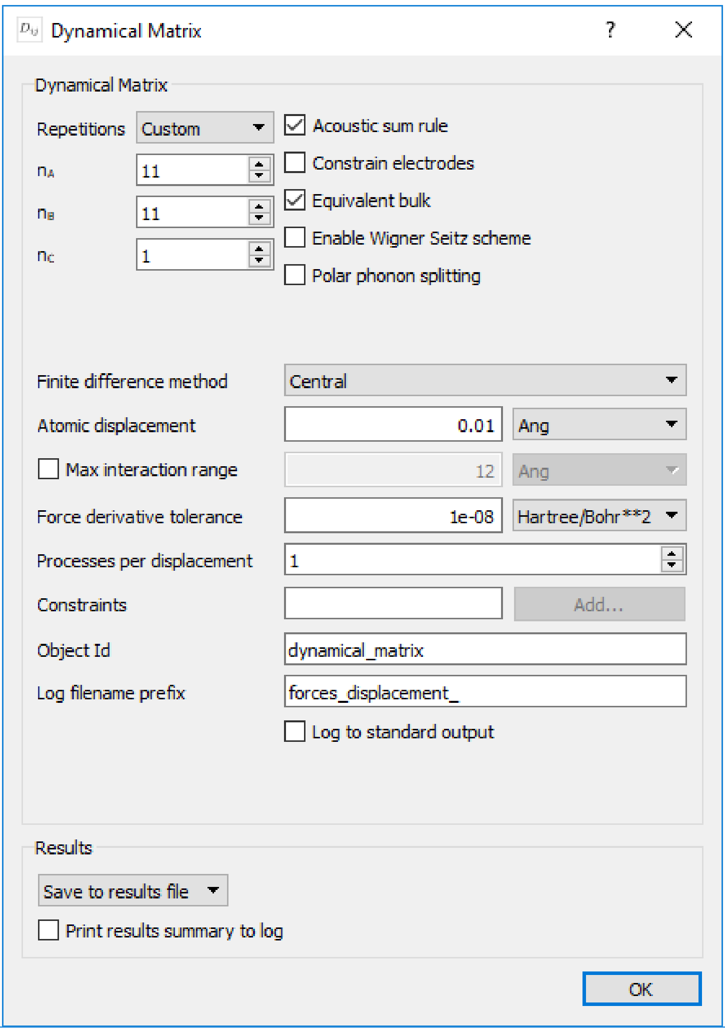
By placing the mouse cursor on top of a band, information about the band is shown. You see that the valence band (highlighted in yellow in the figure above) is number 3 and the conduction band is number 4. These are the two electronic bands relevant for the calculation of the mobility, and in the following we will concentrate on these two bands.

Phonons in Graphene

The next step is to calculate the dynamical matrix of graphene. In order to test the quality of the result you will also calculate the phonon band structure, which is based on the calculated dynamical matrix.

Open the **Script generator**, and create a new script:

- Add an Analysis from File block and select *BulkConfiguration_1* in `Graphene_relax.hdf5`.
- Add a Study Objects ‣ DynamicalMatrix D_{ij} block and modify the following settings:
  - Set **Repetitions** to *Custom*
  - Set the **Number of repetitions** to:
    - n_A = 11
    - n_B = 11
    - n_C = 1



Dynamical Matrix

Dynamical Matrix

Repetitions	Custom
n_A	11
n_B	11
n_C	1

- ✓ Acoustic sum rule
- Constrain electrodes
- ✓ Equivalent bulk
- Enable Wigner Seitz scheme
- Polar phonon splitting

Finite difference method	Central	
Atomic displacement	0.01	Ang
Max interaction range	12	Ang
Force derivative tolerance	1e-08	Hartree/Bohr**2
Processes per displacement	1	
Constraints		Add...
Object Id	dynamical_matrix	
Log filename prefix	forces_displacement_	

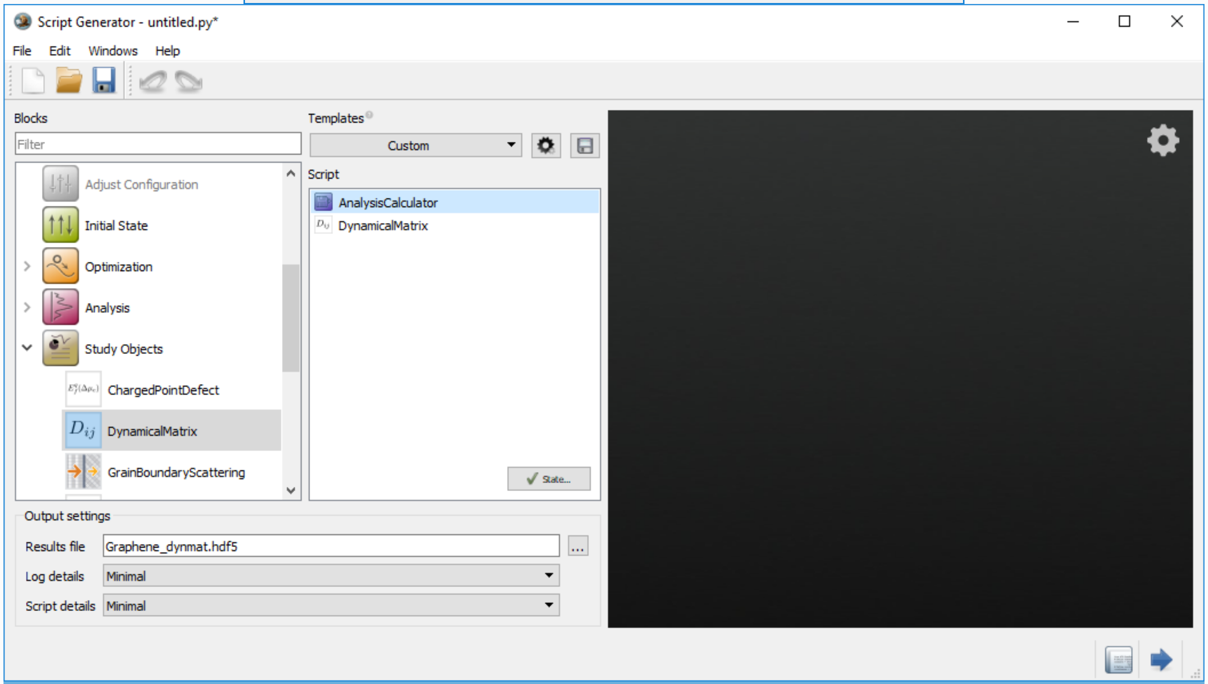
- Log to standard output

Results

Save to results file

- Print results summary to log

OK



Script Generator - untitled.py*

Templates: Custom

Script:
- AnalysisCalculator
- DynamicalMatrix

Output settings

Results file	Graphene_dynmat.hdf5
Log details	Minimal
Script details	Minimal

Note

In previous versions of **QuantumATK** it was necessary to manually scale down the number of k-points when doing a **DynamicalMatrix** or **HamiltonianDerivatives** calculation to take into account the repeated cell. From **QuantumATK-2019.03** and later versions the k-points are automatically scaled to maintain the same accuracy as the unit cell, and we can simply re-use the calculator settings.

Tip

Due to the large dimensions of the 11×11 graphene super cell (242 atoms), the calculation of the *DynamicalMatrix* object is rather time consuming. However, the calculation can be parallelized over the atomic displacements. Since there are two atoms in the graphene unit cell, and each is displaced in the x, y and z directions, there are in total 6 calculations to be performed. Maximum efficiency is obtained when the number of calculations times the value of the parameter *Processes per displacement* matches the total number of cores used for the calculations. In the present case, the calculation takes about 30 minutes if *Processes per displacement = 4* and the calculation is run on 24 cores.

- Add an Analysis ‣ PhononBandstructure block and set the following parameters:
  - Set the number of **Points per segment** to *100*
  - Set the **Brillouin zone route** to *G, M, K, G*

Finally, in the main panel of the **Script Generator** set the **Default output file** to `Graphene_dynmat.hdf5`, send the script to the **Job manager**, save it as `Graphene_dynmat.py` and click on the ▷ button to run the calculation.

Warning

Remember to make the file `Graphene_relax.hdf5` available for the script if you are running on a cluster. In the **Job Manager** you can use the **I/O** tab to transfer additional files together with a script. In this case, you may also need to modify the path in the script before submitting to the cluster, to make sure it points at the correct location of the file.

When the calculation is done, go back to the in the **LabFloor**, and inspect the $\{\omega_i(\mathbf{q})\}$ **PhononBandstructure** object contained in the file `Graphene_dynmat.hdf5` using **Compare data** or **PhononBandstructure Analyzer**. The calculated phonon band structure should match with that shown in the figure below.

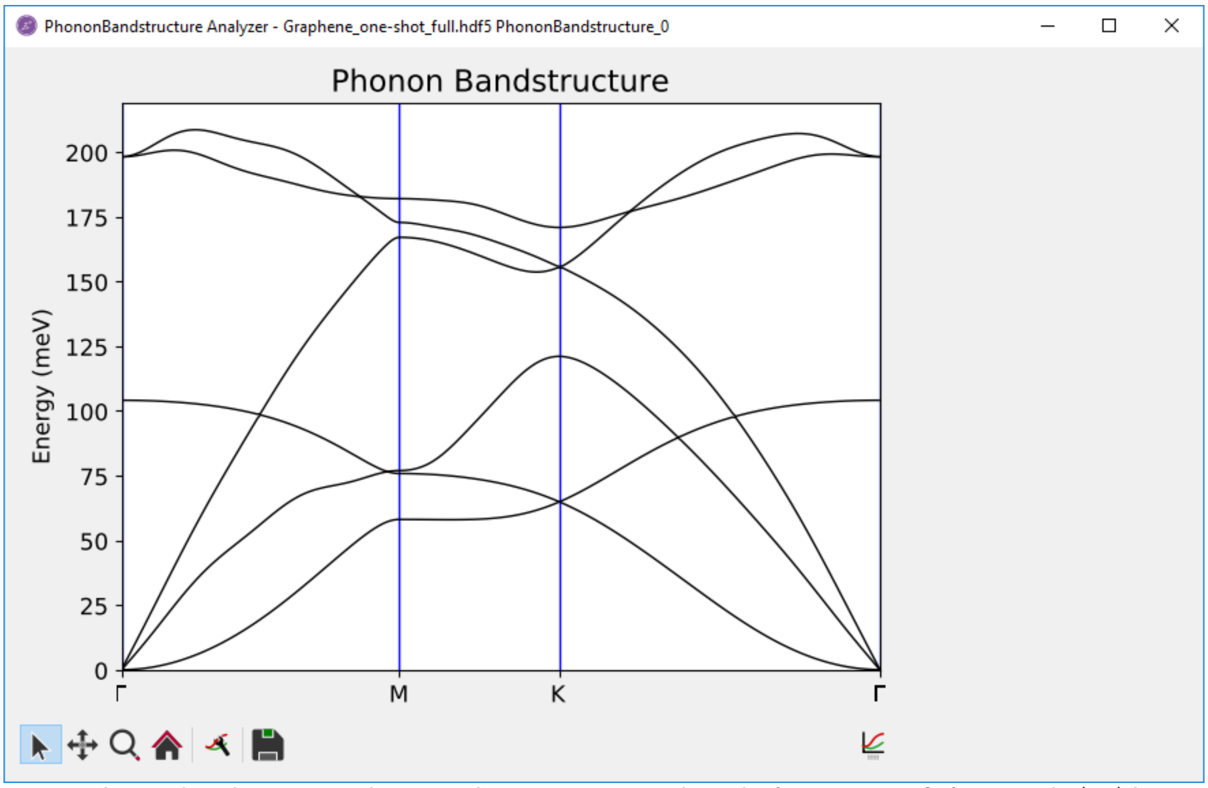


Fig. 29 Phonon bandstructure. There are three acoustic modes: The lowest, out-of-plane mode (ZA) has a $\propto \mathrm{q}^2$ for small
q, while the two next have linear
q-dependence with a constant velocity for small
q. The transverse acoustic (TA) mode is lower in energy than the longitudinal acoustic (LA) mode. ¶

Mobility of graphene

In the following, the procedure to calculate the electron mobility in graphene will be described. Provided that one has already calculated the electronic structure and dynamical matrix of the system, the following three steps are necessary to evaluate the mobility:

1. Calculation of the **Hamiltonian derivatives**
2. Calculation of the **Electron-phonon couplings**
3. Calculation of the **Mobility**

The two methods to calculate
μ described above differ in the way in which steps 2 and 3 are carried out. These two steps will therefore be described separately for each method. The procedure for the **(k, q)-dependent method** is outlined in sections 1, 2A and 3A, whereas the procedure for the **energy-dependent method** is outlined in in sections 1, 2B and 3B

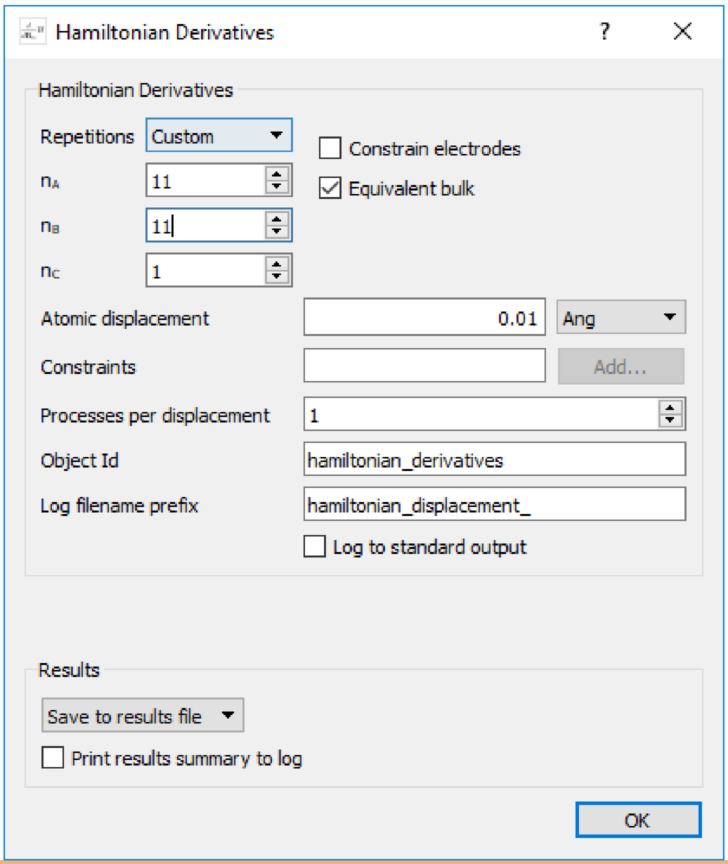
1. Hamiltonian derivatives

In order to calculate the electron-phonon coupling matrix, it is necessary to calculate the derivative of the Hamiltonian
$\partial \hat{H}/\partial R_{i,\alpha}$ with respect to the coordinate of the
i-th atom along the Cartesian direction
α.

Open the **Script generator**, and modify the script as follows:

- Add an Analysis from File block and select *BulkConfiguration_1* in `Graphene_relax.hdf5`.
- Add a Study Objects ‣ HamiltonianDerivatives block and set the **Number of repetitions** to:

- $\mathrm{n_A}$ = 11
- $\mathrm{n_B}$ = 11
- $\mathrm{n_C}$ = 1


Hamiltonian Derivatives

Hamiltonian Derivatives

Repetitions	Custom
nA	11
nB	11
nC	1
Atomic displacement	0.01 Ang
Constraints	
Processes per displacement	1
Object Id	hamiltonian_derivatives
Log filename prefix	hamiltonian_displacement_

☐ Constrain electrodes
☑ Equivalent bulk
☐ Log to standard output

Results

Save to results file
☐ Print results summary to log

Warning

It is important that the number of repetitions matches the calculation of the dynamical matrix.

In the main panel of the **Script Generator**, set the **Results file** to `Graphene_dHdR.hdf5`, send the script to the **Job manager**, save it as `Graphene_dHdR.py` and click on the ▷ button to run the calculation.

Note

As for the **DynamicalMatrix**, the calculation of the **HamiltonianDerivatives** is rather time consuming, but can be parallelized over the atomic displacements. In the present case, the calculation takes around 1 hour if it is run on 24 cores with *Processes per displacement = 4* .

2A. Electron-Phonon couplings: (k,q)-dependent method

In order to calculate the lifetimes
$\tau(\mathrm{k},\mathrm{q})$ and the mobility
μ, we need to calculate the electron-phonon coupling matrix on a fine grid of
k- and
q-points.

Open the **Script generator**, and modify the script as follows:

- Add an Analysis from File block and select *BulkConfiguration_1* in `Graphene_relax.hdf5`.

- Add an Analysis ‣ ElectronPhononCoupling block.

You will notice that two additional blocks have been also added:

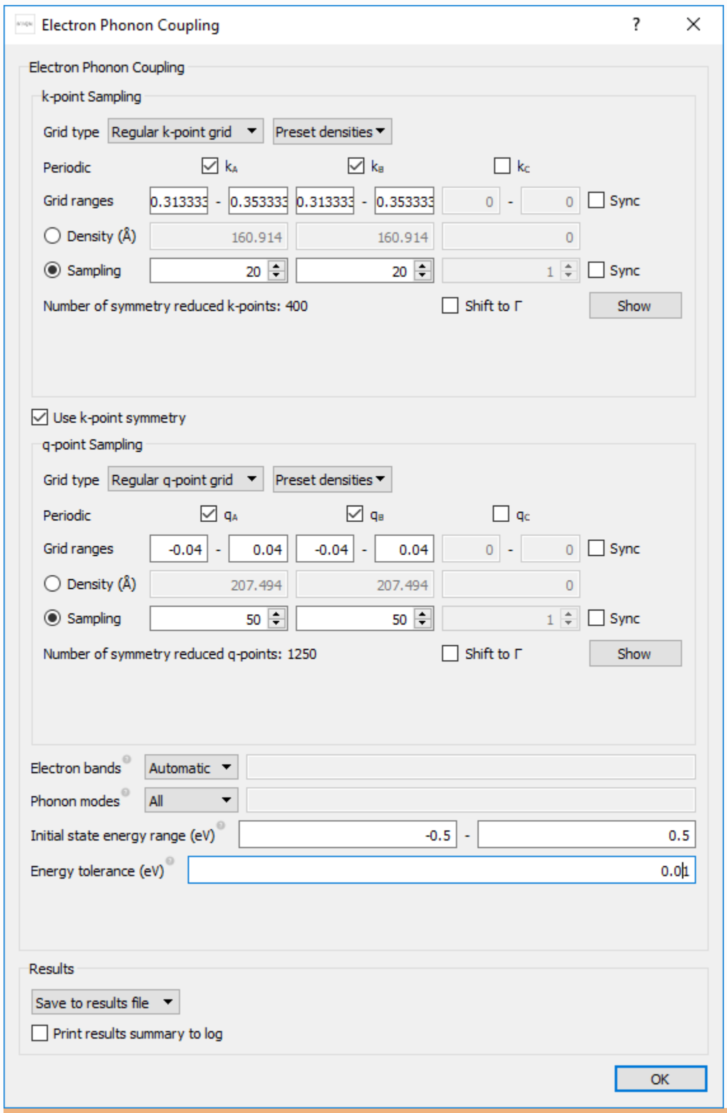
- Study Objects ‣ DynamicalMatrix D_{ij}
- Study Objects ‣ HamiltonianDerivatives $\frac{d}{d\mathbf{R}_\alpha}\hat{H}$

In the present case, both the dynamical matrix and the Hamiltonian derivatives have been calculated already and can be re-used. As they are study objects, this is automatically detected by QuantumATK if the provided filename and other input parameters are the same. Open each of them and change the repetitions to 11x11x1 and the filenames to `Graphene_dynmat.hdf5` and `Graphene_dHdR.hdf5`, respectively.

Now set the parameters in the Analysis ‣ ElectronPhononCoupling block as shown below. We will sample k-points in a small area around the K-point at [1/3, 1/3, 0], and q-points in a small area around the Γ-point at [0,0,0].

Tip

If you are unsure of the coordinates of a particular symmetry point you may use the built-in functionality of the `BravaisLattice` class, f.ex. like this: `k0=bulk_configuration.bravaisLattice().symmetryPoints()['K']`)



Electron Phonon Coupling

Electron Phonon Coupling

k-point Sampling

Grid type: Regular k-point grid; Preset densities

Periodic: [✓ kA] [✓ kB] kC

Grid ranges: 0.313333 - 0.353333, 0.313333 - 0.353333, 0 - 0; Sync

Density (Å): 160.914, 160.914, 0

[✓ Sampling]: 20, 20, 1; Sync

Number of symmetry reduced k-points: 400; Shift to Γ; Show

[✓ Use k-point symmetry]

q-point Sampling

Grid type: Regular q-point grid; Preset densities

Periodic: [✓ qA] [✓ qB] qC

Grid ranges: -0.04 - 0.04, -0.04 - 0.04, 0 - 0; Sync

Density (Å): 207.494, 207.494, 0

[✓ Sampling]: 50, 50, 1; Sync

Number of symmetry reduced q-points: 1250; Shift to Γ; Show

Electron bands: Automatic

Phonon modes: All

Initial state energy range (eV): -0.5 - 0.5

Energy tolerance (eV): 0.01

Results

Save to results file

Print results summary to log

OK

Warning

For production calculations, it is strongly recommended to always converge the sampling resolution in q- and k-space. The above settings are a result of such a study, with more information shown in the appendix: Convergence of q- and k-point sampling

In the main panel of the **Script Generator**, set the **Results file** to `Graphene_eph.hdf5`. Send the script to the **Job manager**, save it as `Graphene_eph.py` and click on the ▷ button to run the calculation.

Warning

For graphene, we need quite dense samplings in
k- and
q-space, and therefore we have chosen to sample very finely a small area around the relevant high-symmetry points. This still includes the full angular dependence, but neglects inter-valley scattering. We therefore recommend to sample the entire Brillouin zone if this could be an important effect.

Note

Sampling the
k- and
q-space using dense meshes means that the calculation might become very time consuming. Contemporary versions of QuantumATK automatically detect the relevant regions of
k- and
q-space, and do not compute matrix elements which will not contribute, considerably decreasing the computational time and memory requirements. The two parameters *energy_tolerance* and *initial_state_energy_range* govern the filters which reduce the number of calculated coupling elements. More information about these two parameters can be found on the manual page: `ElectronPhononCoupling`.

As QuantumATK parallelizes over
k- and
q-points, a high number of MPI processes can be used if sufficient memory is available. In the present case, the calculation took approximately 10 hours on a 16-core node on a cluster.

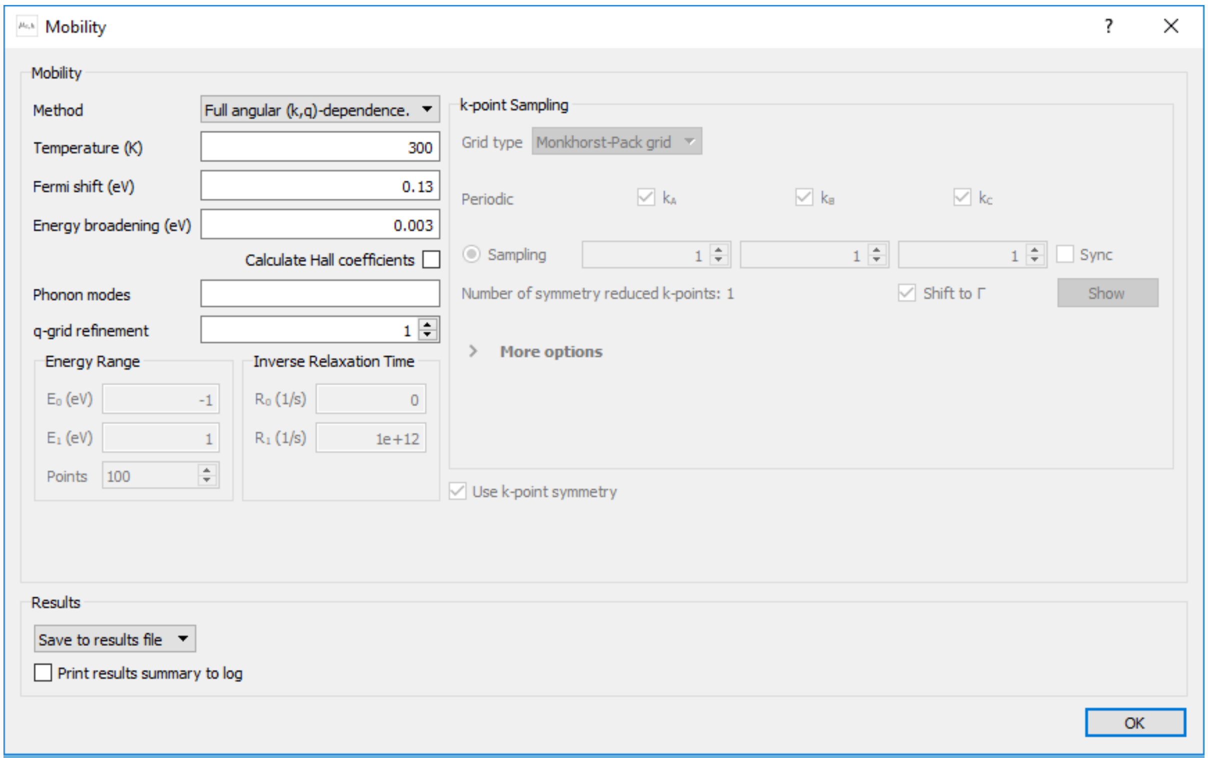
3A. Mobility: (k,q)-dependent method

Now open a new Script Generator window and add an Analysis from File block and then a Analysis ‣ Mobility $\mu_{e,h}$ block. In the Analysis from File block, select the file `Graphene_relax.hdf5` and load *BulkConfiguration_1* included in the file. Remove the ElectronPhononCoupling, DynamicalMatrix D_{ij} and HamiltonianDerivatives $\frac{d}{d\mathbf{R}_n}\mathcal{H}$ blocks, and replace them with a Custom blocks ‣ Code snippet block. This allows us to easily add arbitrary code to any script without editing the full script manually. Open the block, and write the following:

```
electron_phonon_coupling = nlread('Graphene_eph.hdf5', ElectronPhononCoupling)[0]
```

Set the parameters for the Analysis ‣ Mobility $\mu_{e,h}$ block as shown below:

- Leave the **Method** at *Full angular (k,q)-dependence*
- Set the **Fermi shift** to *0.13 eV*
- Untick *Calculate Hall coefficients*


Mobility

Mobility

Method: Full angular (k,q)-dependence.
Temperature (K): 300
Fermi shift (eV): 0.13
Energy broadening (eV): 0.003
Calculate Hall coefficients ☐
Phonon modes:
q-grid refinement: 1

Energy Range
E₀ (eV): -1
E₁ (eV): 1
Points: 100

Inverse Relaxation Time
R₀ (1/s): 0
R₁ (1/s): 1e+12

k-point Sampling
Grid type: Monkhorst-Pack grid
Periodic: ✓ kA ✓ kB ✓ kC
Sampling: 1 1 1 ☐ Sync
Number of symmetry reduced k-points: 1
✓ Shift to Γ
✓ Use k-point symmetry

Results
Save to results file
☐ Print results summary to log

Note

The Fermi shift of 0.13 eV corresponds to a carrier concentration of $n = 10^{12}\text{cm}^{-2}$

In the main panel of the **Script Generator**, set the **Results file** to `Graphene_mu-full.hdf5`, send the script to the **Job manager**, save it as `Graphene_mu-full.py` and click on the ▷ button to run the calculation. It should take no more than a few minutes on a laptop.

Note

You will get a notification that the Mobility block is invalid. You can ignore this warning, as NanoLab does not parse the *Code snippet* block and is thus unaware of the `ElectronPhononCoupling` object that will be loaded there.

Once the calculation is done, select the file `Graphene_mu-full.hdf5`, and on the **LabFloor**, select the $\mu_{e,h}$ **Mobility** object, and click on **Text Representation**. You should now see the following:

```
+------------------------------------------------------------------------------+
| Mobility Report                                                              |
| ---------------------------------------------------------------------------- |
| Input parameters:                                                            |
|     Temperature       = 300.00 K                                             |
|     Fermi level shift = 0.13 eV                                              |
|     Energy broadening = 0.0030 eV                                            |
|     q-grid refinement = 1                                                    |
|                                                                              |
+------------------------------------------------------------------------------+
| Trace of linear responce tensors:                                            |
+------------------------------------------------------------------------------+
|                                                                              |
| Electrons:                                                                   |
|                                                                              |
|     Mobility                 =  2.34e+05 cm^2/(V*s)                          |
|     Conductivity             =  4.19e+01 S/m                                 |
|     Seebeck coefficient      = -1.94e-05 V/K                                 |
|     Thermal conductivity     =  2.43e-04 W/(m*K)                             |
|     Carrier density (2D, xy) =  2.24e+06 cm^-2                               |
|                                                                              |
|                                                                              |
| Holes:                                                                       |
|                                                                              |
|     Mobility                 =  8.66e-04 cm^2/(V*s)                          |
|     Conductivity             =  1.19e-01 S/m                                 |
|     Seebeck coefficient      =  1.14e-03 V/K                                 |
|     Thermal conductivity     =  1.18e-05 W/(m*K)                             |
|     Carrier density (2D, xy) =  1.72e+12 cm^-2                               |
|                                                                              |
+------------------------------------------------------------------------------+
```

The calculated electron mobility, highlighted in yellow, is $2.34 \cdot 10^5 \ \mathrm{cm^2 V^{-1} s^{-1}}$, and agrees well with previously reported data at room temperature and $n = 10^{12}\mathrm{cm}^{-2}$ [2].

Note

Note that the carrier density listed here is significantly lower than $n = 10^{12}\mathrm{cm}^{-2}$. The carrier density is calculated independently from the mobility, and converges much more slowly than the mobility itself. In this tutorial, the focus is on achieving a converged value for the mobility - converging also the carrier density would require inclusion of more k-points and/or a larger region in k-space in the **ElectronPhononCoupling** calculation. Alternatively, you can calculate the carrier density with a finer sampling directly from the DensityOfStates object: carrier_density_test.py.

Tip

Starting from version Q-2019.12, QuantumATK NanoLab includes a Mobility Analyzer. This will be presented in a future update of this tutorial.

2B. Electron-Phonon couplings: energy-dependent method

We will now calculate the electron-phonon couplings to be used for the calculation of the energy-dependent relaxation times $\tau(E)$. We will assume that the relaxation times vary isotropically in k-space. This means that it will be sufficient to evaluate the electron-phonon coupling matrix for a number of

k-points along a line through one of the Dirac (K) points, thereby reducing significantly the computational workload of the calculation.

Open the **Script generator**, and modify the script as follows:

- Add an Analysis from File block and select *BulkConfiguration_1* in `Graphene_relax.hdf5`.
- Add an Analysis ‣ ElectronPhononCoupling block.

You will notice that two additional blocks have been also added:

- Study Objects ‣ DynamicalMatrix D_{ij}
- Study Objects ‣ HamiltonianDerivatives $\frac{d}{d\mathbf{R}_\alpha}\overline{H}$

In the present case, both the dynamical matrix and the Hamiltonian derivatives have been already calculated and can be re-used. As they are study objects, this is automatically detected by QuantumATK if the provided filename and other input parameters are the same. Open each of them and change the repetitions to 11x11x1 and the filenames to `Graphene_dynmat.hdf5` and `Graphene_dHdR.hdf5`, respectively.

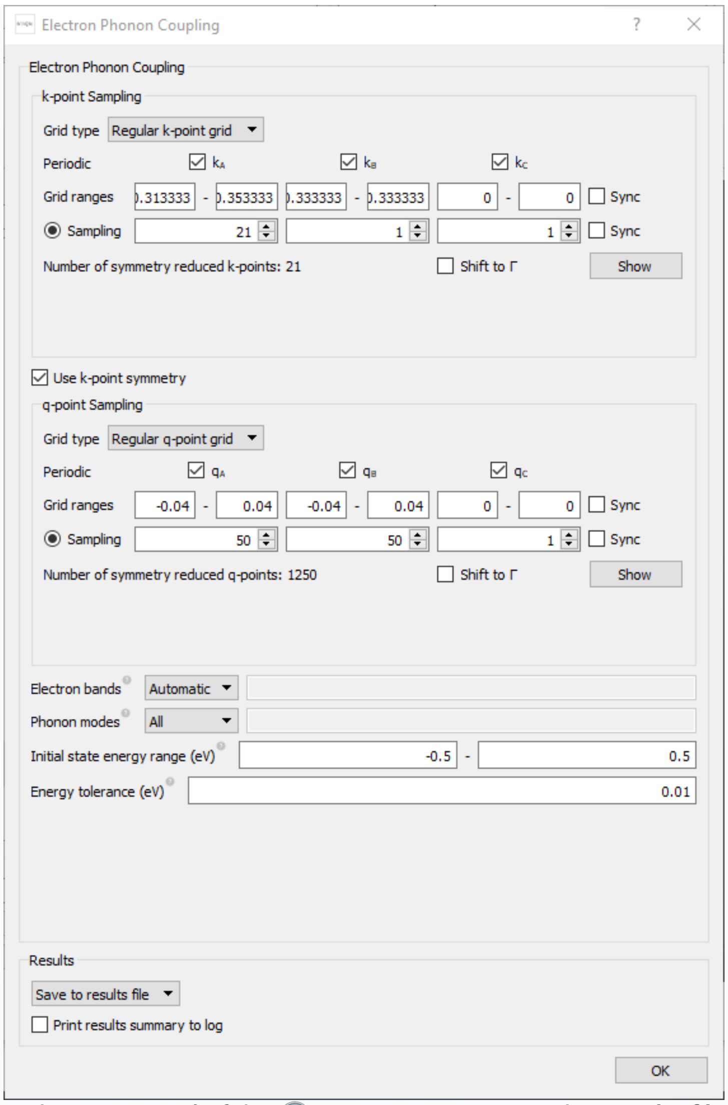
In the Analysis ‣ ElectronPhononCoupling block, modify the parameters as follows:

- In the **k-point sampling**, set the **Grid type** to *Regular k-point grid*, and the sampling parameters as shown in the figure below.

  Note

  As you can see from the figure, the k-space is sampled only along a line!

- In the **q-point sampling**, set the **Grid type** to *Regular q-point grid*, and the sampling parameters as shown in the figure below.
- Set the **Energy tolerance** to *0.01* (eV) and the **Initial state energy range** to go from *-0.5* to *0.5*.


Electron Phonon Coupling

Electron Phonon Coupling

k-point Sampling

Grid type: Regular k-point grid

Periodic: [✓ kA] [✓ kB] [✓ kC]

Grid ranges: .313333 - .353333 | .333333 - .333333 | 0 - 0 | ☐ Sync

(●) Sampling: 21 | 1 | 1 | ☐ Sync

Number of symmetry reduced k-points: 21 | ☐ Shift to Γ | Show

[✓] Use k-point symmetry

q-point Sampling

Grid type: Regular q-point grid

Periodic: [✓ qA] [✓ qB] [✓ qC]

Grid ranges: -0.04 - 0.04 | -0.04 - 0.04 | 0 - 0 | ☐ Sync

(●) Sampling: 50 | 50 | 1 | ☐ Sync

Number of symmetry reduced q-points: 1250 | ☐ Shift to Γ | Show

Electron bands: Automatic

Phonon modes: All

Initial state energy range (eV): -0.5 - 0.5

Energy tolerance (eV): 0.01

Results

Save to results file

☐ Print results summary to log

OK

In the main panel of the **Script Generator**, set the **Results file** to `Graphene_eph-line.hdf5`, send the script to the **Job manager**, save it as `Graphene_eph-line.py` and click on the ▷ button to run the calculation.

Note

Since here we sample the
k-space only along a line, the calculation will in general be faster than the one described in section 2A. In the present case, the calculation took approximately 50 minutes on a 16-core node on a cluster, or about 1/10th of the full calculation in section 2A.

3B. Electron mobility: energy-dependent method

We will now use a two-step procedure to evaluate the room-temperature mobility
μ based on the energy-dependent relaxation times
$\tau(E)$:

- In the first step, the
  k- and
  q-dependent relaxation times
  $\tau(\mathrm{k}, \mathrm{q})$ are evaluated along the line outward from the K-point.
- In the second step, the values of
  $\tau(\mathrm{k}, \mathrm{q})$ are averaged in
  k-space to obtain the values of

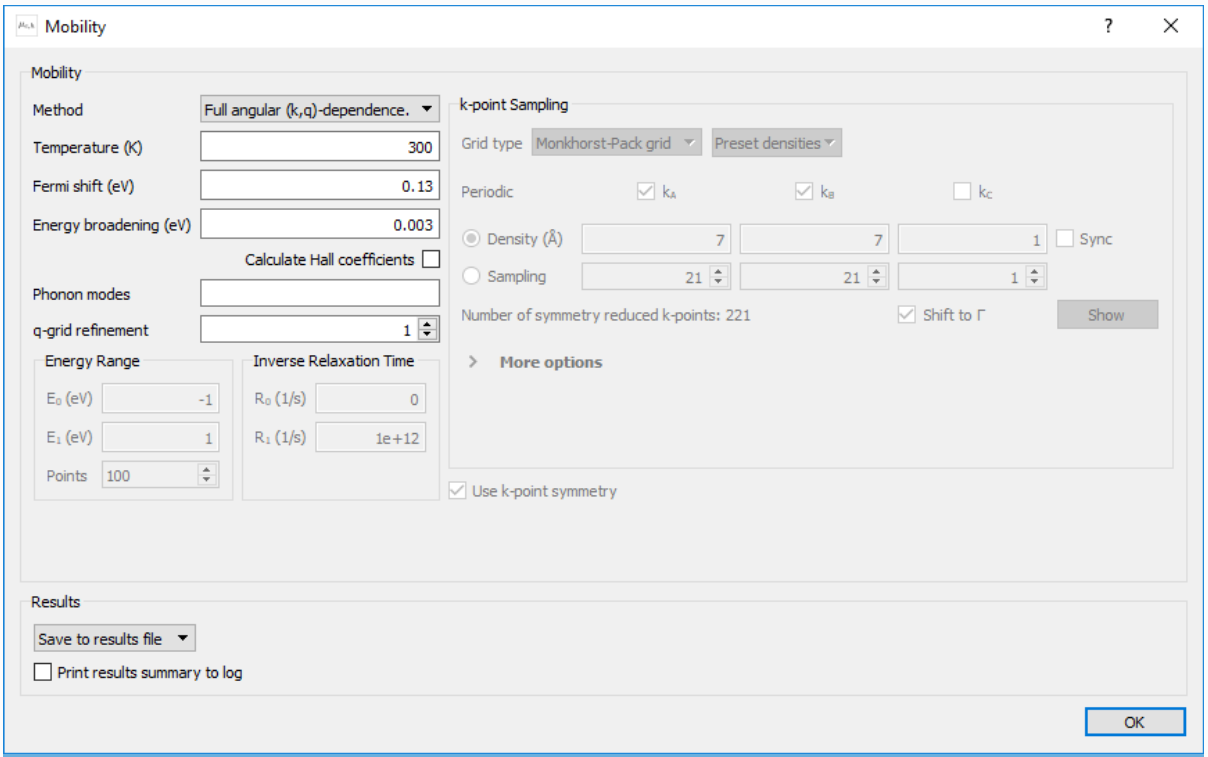
$\tau(E)$, which are then used to calculate the mobility
μ.

Now open a new Script Generator window and add an Analysis from File block and then a Analysis ‣ Mobility block. In the Analysis from File block, select the file `Graphene_relax.hdf5` and load *BulkConfiguration_1* included in the file. Remove the DynamicalMatrix and HamiltonianDerivatives objects and replace the Analysis ‣ ElectronPhononCoupling with a Custom blocks ‣ Code snippet block. Open the block, and write the following:

```
electron_phonon_coupling = nlread('Graphene_eph-line.hdf5', ElectronPhononCoupling)[0]
```

Finally, set the parameters in the Analysis ‣ Mobility block as shown below:

- Analysis ‣ Mobility
  - Set the **Method** to *Full angular (k,q)-dependence*
  - Set the **Fermi shift** to *0.13 eV*
  - Untick *Calculate Hall coefficients*



Mobility

Mobility

Method	Full angular (k,q)-dependence.
Temperature (K)	300
Fermi shift (eV)	0.13
Energy broadening (eV)	0.003
Calculate Hall coefficients	☐
Phonon modes	
q-grid refinement	1

Energy Range

E₀ (eV)	-1
E₁ (eV)	1
Points	100

Inverse Relaxation Time

R₀ (1/s)	0
R₁ (1/s)	1e+12

k-point Sampling

Grid type: Monkhorst-Pack grid | Preset densities

Periodic	✓ kA	✓ kB	☐ kC	
Density (Å)	7	7	1	☐ Sync
Sampling	21	21	1	

Number of symmetry reduced k-points: 221 ✓ Shift to Γ Show

More options

✓ Use k-point symmetry

Results

Save to results file

☐ Print results summary to log

OK

Note

Note that we choose a
k-line direction somewhat arbitrarily here. The point is that a low number of
k-points can be used to generate an energy dependent rate. An alternative to a line could be to coarsely sample the entire BZ to capture the anisotropy in
k-space, but with fewer points than needed in the full (
k,
q)-dependent method.

In the main panel of the **Script Generator**, set the **Results file** to `Graphene_mu-line-full.hdf5`, send the script to the **Job manager**,save it as `Graphene_mu-line-full.py` and click on the button to run the calculation. You will again see a warning that the script might be invalid, and you should ignore it this time as well.

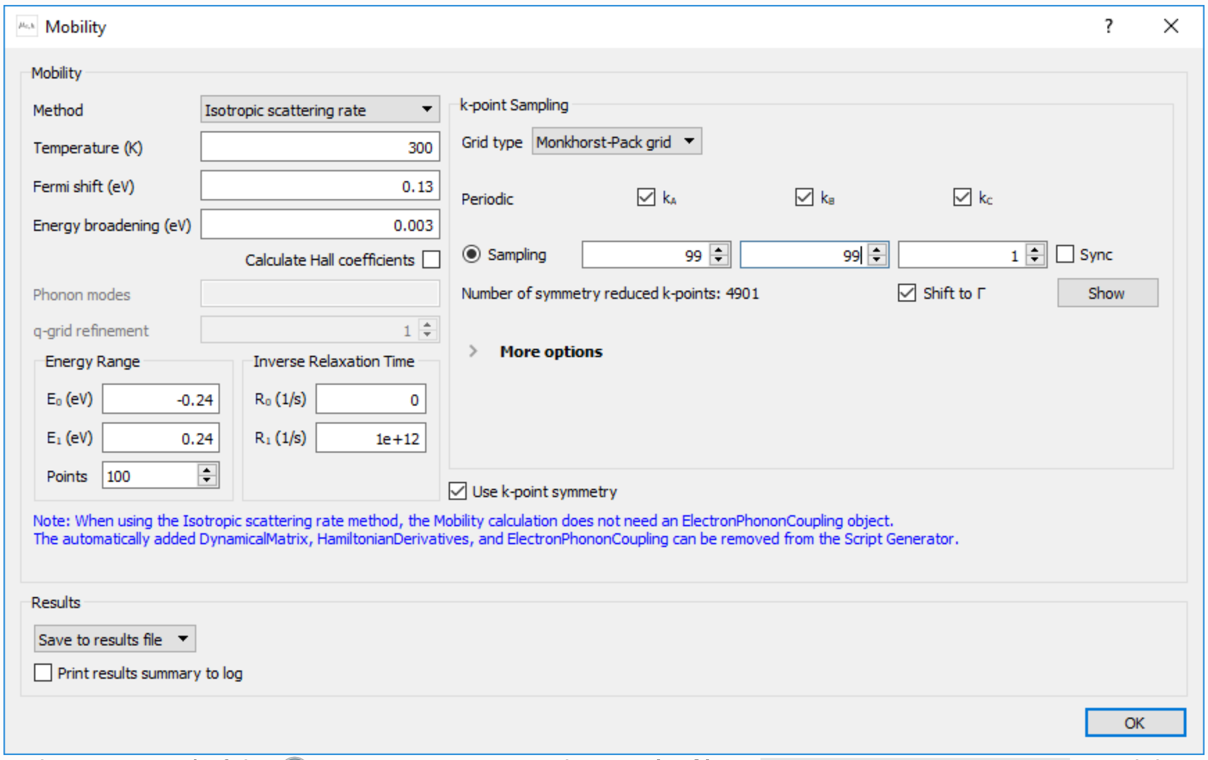
Once the calculation is done, you are ready to calculate the mobility using the isotropic scattering rate method. In this case, you will re-use the bulk configuration and the full angular (k, q) -dependent mobility from the `Graphene_relax.hdf5` and `Graphene_mu-line-full.hdf5` files.

Go back to the **Script generator**, and open the Custom ‣ Code Snippet block before the Mobility icon $\mu_{e,h}$. Change the contents to:

```
mobility_full = nlread('Graphene_mu-line-full.hdf5', Mobility)[0]
```

Then modify the Analysis ‣ Mobility $\mu_{e,h}$ block as follows:

- Analysis ‣ Mobility $\mu_{e,h}$
  - Set the **Method** to *Isotropic scattering rate*
  - Set the **Fermi shift** to *0.13 eV*
  - Untick *Calculate Hall coefficients*
  - Set the following values of the **Energy Range**:
    - E_0 = -0.24 eV
    - E_0 = 0.24 eV
    - Points = 100
  - Set the **k-point Sampling** to $99 \times 99 \times 1$


Mobility

Mobility

Method	Isotropic scattering rate
Temperature (K)	300
Fermi shift (eV)	0.13
Energy broadening (eV)	0.003
Calculate Hall coefficients	☐
Phonon modes	
q-grid refinement	1

Energy Range

E_0 (eV)	-0.24
E_1 (eV)	0.24
Points	100

Inverse Relaxation Time

R_0 (1/s)	0
R_1 (1/s)	1e+12

k-point Sampling

Grid type: Monkhorst-Pack grid

Periodic: ☑ k_A ☑ k_B ☑ k_C

◉ Sampling 99 99 1 ☐ Sync

Number of symmetry reduced k-points: 4901 ☑ Shift to Γ Show

More options

☑ Use k-point symmetry

Note: When using the Isotropic scattering rate method, the Mobility calculation does not need an ElectronPhononCoupling object.
The automatically added DynamicalMatrix, HamiltonianDerivatives, and ElectronPhononCoupling can be removed from the Script Generator.

Results

Save to results file

☐ Print results summary to log

OK

In the main panel of the **Script Generator**, set the **Results file** to `Graphene_mu-line-iso.hdf5`. Send the script to the **Editor**, change `mobility_object=None,` to `mobility_object=mobility_full,` and remove this line:

```
inverse_relaxation_time=numpy.linspace(0, 1e+12, 100)*Second**-1,
```

Send it to the **Job manager**, save it as `Graphene_mu-line-iso.py` and click on the ▷ button to run the calculation. This should take no more than a minute on a modern laptop.

Once the calculation is done, open the file `Graphene_mu-line-iso.hdf5`, select the $\mu_{e,h}$ **Mobility** object in the LabFloor and click on **Text Representation**.

```
+------------------------------------------------------------------------------+
| Mobility Report                                                              |
| ---------------------------------------------------------------------------- |
| Input parameters:                                                            |
|     Temperature        = 300.00 K                                            |
|     Fermi level shift = 0.13 eV                                              |
|     Energy broadening = 0.0030 eV                                            |
|     q-grid refinement = 1                                                    |
|                                                                              |
+------------------------------------------------------------------------------+
| Trace of linear responce tensors:                                            |
+------------------------------------------------------------------------------+
|                                                                              |
| Electrons:                                                                   |
|                                                                              |
|     Mobility                  =  2.53e+05 cm^2/(V*s)                         |
|     Conductivity              =  1.73e+07 S/m                                |
|     Seebeck coefficient       = -3.00e-05 V/K                                |
|     Thermal conductivity      =  9.31e+01 W/(m*K)                            |
|     Carrier density (2D, xy) =  8.55e+11 cm^-2                               |
|                                                                              |
|                                                                              |
| Holes:                                                                       |
|                                                                              |
|     Mobility                  =  1.89e+06 cm^2/(V*s)                         |
|     Conductivity              =  5.72e+04 S/m                                |
|     Seebeck coefficient       =  1.14e-03 V/K                                |
|     Thermal conductivity      =  5.74e+00 W/(m*K)                            |
|     Carrier density (2D, xy) =  3.78e+08 cm^-2                               |
|                                                                              |
+------------------------------------------------------------------------------+
```

The calculated value for the electron mobility, highlighted in yellow above, is $2.53 \cdot 10^5\ \mathrm{cm^2V^{-1}s^{-1}}$, in good agreement with the value obtained by using the full (k, q) -dependent method.

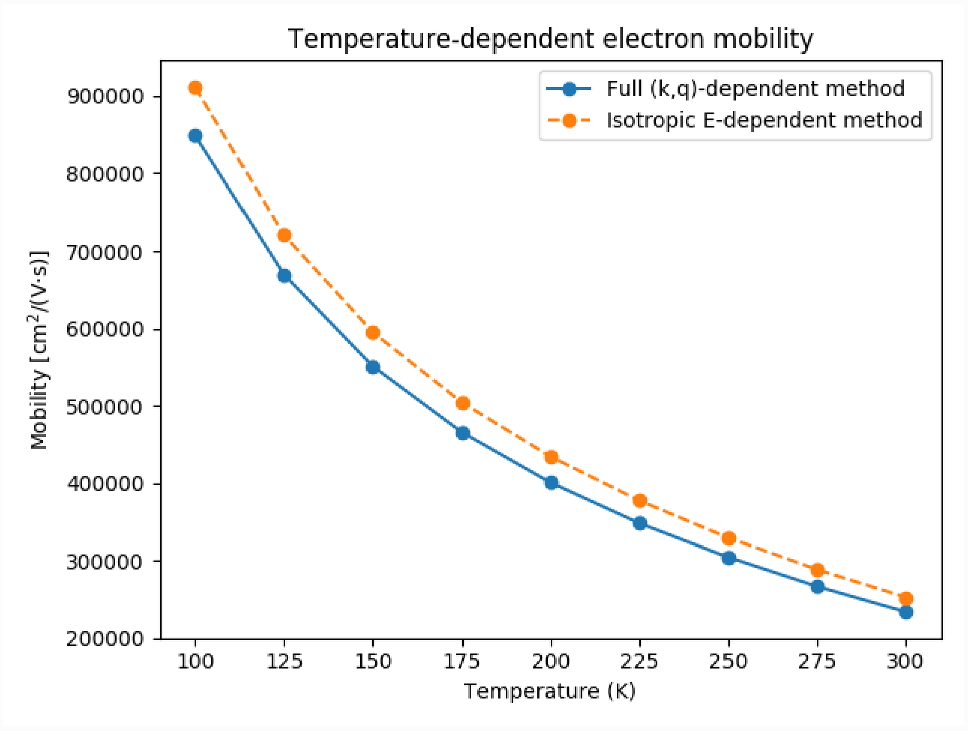
Temperature dependence of the mobility: (k,q)-dependent method vs. energy-dependent method

A more stringent test for the reliability of the energy-dependent method is to calculate the temperature dependence of the mobility in an energy range up to room temperature.

- In the **(k,q)-dependent method**, this can be done by simply modifying the value of the target temperature in the Mobility analysis object. temperature_dependence_full_mobility.py
- In the **energy-dependent method**, the two-step procedure must be repeated for each temperature, and the target temperature has to be set both when calculating the values of the k- and q-dependent relaxation times $\tau(\mathrm{k},\mathrm{q})$ along the line, and when calculating the energy-dependent relaxation times $\tau(E)$. temperature_dependence_isotropic_mobility.py

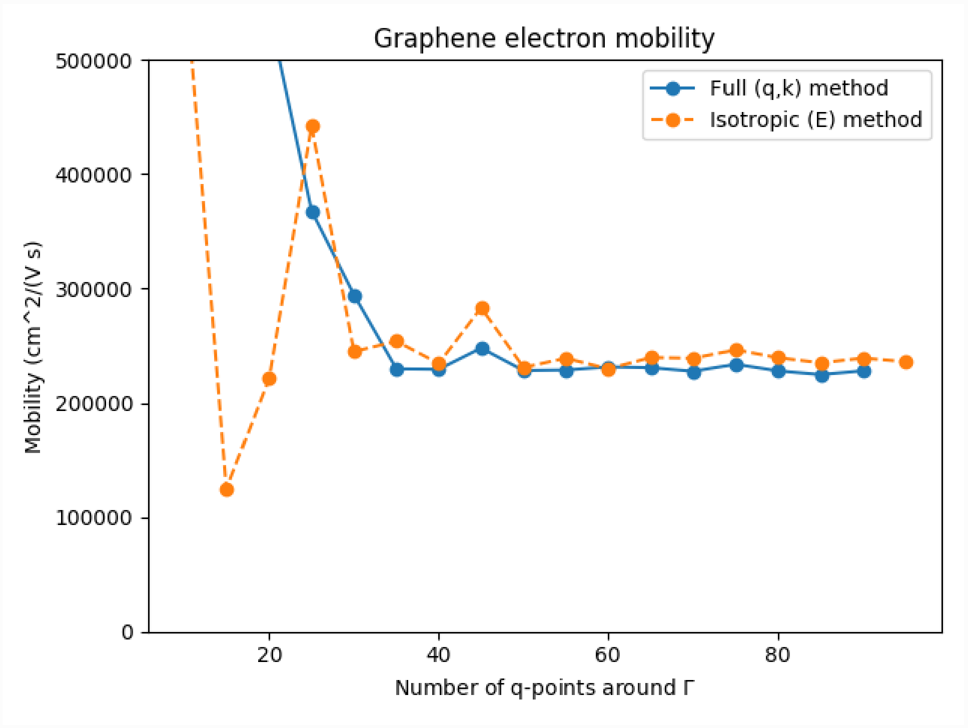
As it can be seen from the figure below, where the T-dependency of μ is calculated in the temperature range $100\mathrm{K} \leq T \leq 300\mathrm{K}$, both methods reproduce well the expected

$1/T$ behavior of
μ. Script to create the plot: plot_temperature_dependence_mobility.py



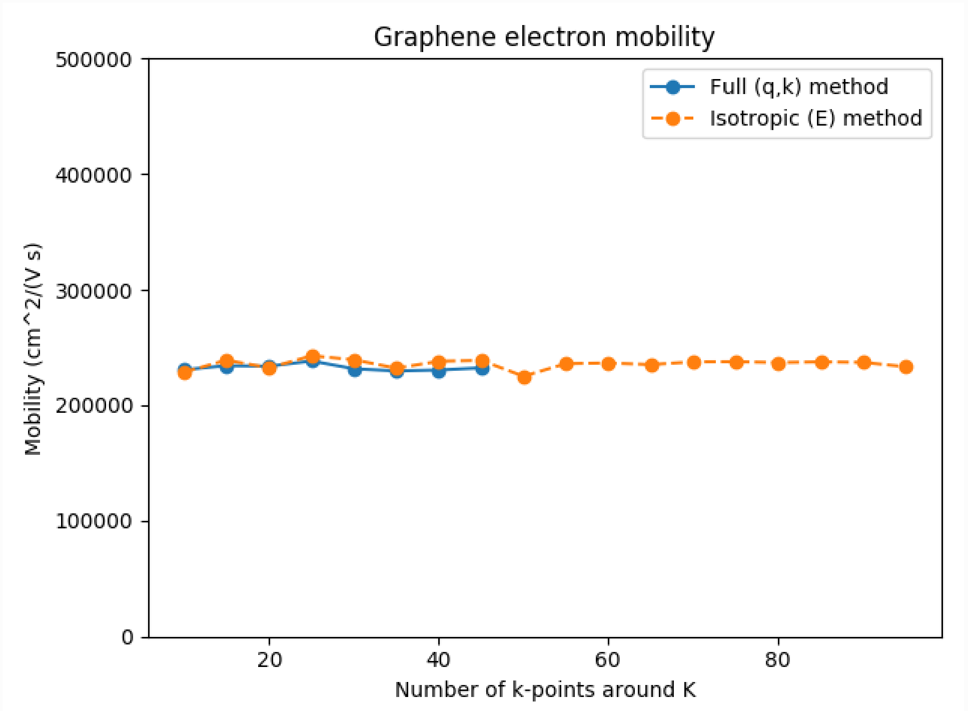
Convergence of q- and k-point sampling

In order to select the appropriate samplings in
k- and
q-space, we first studied the mobility as a function of the number of
q-points, for a fixed sampling of
k-points:



We see that the mobility is reasonably converged for 50
q-points and above. This corresponds to a density of at least 200
Å. We then study the mobility as a function of

k-points, and see that it converges very quickly in this case. We chose to use 20 k-points, corresponding to a density of about 160
Å.



Warning

This convergence study is only valid for this combination of system and computational settings. You should always study convergence for your system of interest and chosen computational model.

Theory section

The mobility
μ is related to the conductivity
σ via

(8)¶

$$\mu = \frac{\sigma}{ne},$$

where
n is the carrier density,
e is the electronic charge.

In this tutorial, you will calculate the conductivity using the semi-classical Boltzmann transport equation (BTE). In the case of zero magnetic field and a time-independent electric fields in the steady state limit the BTE simplifies to:

(9)¶

$$\frac{q\mathrm{E}}{\hbar} \cdot \nabla_{\mathrm{k}} f_{\mathrm{k}n} \approx \frac{q\mathrm{E}}{\hbar} \cdot \nabla_{\mathrm{k}} f^0_{\mathrm{k}n} = \frac{\partial f_{\mathrm{k}n}}{\partial t}\bigg|_{coll}$$

The right hand side (RHS), often called the collision integral, describes different sources of scattering and dissipation that drives the system towards steady state. The left hand-side is approximated to linear order in the electric field by changing to the equilibrium distribution.

The electron-phonon scattering is described by the collision integral. Most commonly this is approximated by a relaxation time:

$$\left.\frac{\partial f_{\mathrm{k}n}}{\partial t}\right|_{coll} = \frac{\delta f_{\mathrm{k}n}}{\tau_{\mathrm{k}n}}$$

describing quasielastic scattering on acoustic phonons (relaxation time approximation). We let
λ label the phonon modes,
n the electronic bands,
k the electron momentum. The transition rate from a state
$|\mathrm{k}n\rangle$ to
$|\mathrm{k}'n'\rangle$ is obtained from Fermi's golden rule (FGR):

$$P_{\mathrm{kk}'}^{\lambda nn'} = \frac{2\pi}{\hbar}|g_{\mathrm{kk}'}^{\lambda nn'}|^2[n_{\mathrm{q}}^{\lambda}\delta\left(\epsilon_{\mathrm{k}'n'} - \epsilon_{\mathrm{k}n} - \hbar\omega_{q\lambda}\right)\delta_{\mathrm{k}',\mathrm{k+q}}$$

$$+ \left(n_{-\mathrm{q}}^{\lambda} + 1\right)\delta\left(\epsilon_{\mathrm{k}'n'} - \epsilon_{\mathrm{k}n} + \hbar\omega_{-q\lambda}\right)\delta_{\mathrm{k}',\mathrm{k-q}}],$$

where the first and last terms in the square brackets describes phonon absorption and emission, respectively. The electron-phonon coupling matrix elements
$|g_{\mathrm{kk}'}^{\lambda nn'}|$ are in **ATK** calculated using the **ElectronPhononCoupling** analysis module.

The general electron-phonon collision integral is given by

(10)¶

$$\left.\frac{\partial f_{\mathrm{k}n}}{\partial t}\right|_{coll} = -\sum_{\mathrm{k}'n'}\left[f_{\mathrm{k}n}\left(1 - f_{\mathrm{k}'n'}\right)P_{\mathrm{kk}'}^{nn'} - f_{\mathrm{k}'n'}\left(1 - f_{\mathrm{k}n}\right)P_{\mathrm{k}'\mathrm{k}}^{n'n}\right]$$

Relaxation time approximation

In the relaxation-time approximation (RTA) one assumes a special simplified form of the RHS/collision integral:

(11)¶

$$\left.\frac{\partial f_{\mathrm{k}n}}{\partial t}\right|_{coll} = -\frac{\delta f_{\mathrm{k}n}}{\tau_{\mathrm{k}n}}$$

where
$\delta f_{\mathrm{k}n} = f_{\mathrm{k}n} - f_{\mathrm{k}n}^0$. The linearized BTE becomes:

$$q\mathrm{E}\cdot\mathrm{v}_{\mathrm{k}n}\frac{\partial f_{\mathrm{k}n}^0}{\partial\epsilon_{\mathrm{k}n}} = -\frac{\delta f_{\mathrm{k}n}}{\tau_{\mathrm{k}n}}.$$

Hereby we find the solution:

$$\delta f_{\mathrm{k}n} = -q\mathrm{E}\cdot\mathrm{v}_{\mathrm{k}n}\frac{\partial f_{\mathrm{k}n}^0}{\partial\epsilon_{\mathrm{k}n}}\tau_{\mathrm{k}n}.$$

The relaxation-time can be evaluated from the general collision integral, (10). However, the specific form in eqn. (11) only applies in the limit of quasielastic scattering (
$\omega_{q\lambda} \to 0$), in which case the collision integral, in eqn. (10), simplifies significantly:

$$\left.\frac{\partial f_{\mathrm{k}n}}{\partial t}\right|_{coll}^{RTA} = \sum_{\mathrm{k}'n'}\left(f_{\mathrm{k}'n'} - f_{\mathrm{k}n}\right)P_{\mathrm{kk}'}^{nn'} \equiv -\frac{\delta f_{\mathrm{k}n}}{\tau_{\mathrm{k}n}},$$

since
$P_{\mathrm{kk}'}^{nn'} = P_{\mathrm{k}'\mathrm{k}}^{n'n}$ in this limit.

We then obtain the expression for the scattering rate or inverse relaxation-time:

(12)¶

$$\frac{1}{\tau_{\mathrm{k}n}} = \sum_{\mathrm{k}'n'} \left(1 - \frac{\delta f_{\mathrm{k}'n'}}{\delta f_{\mathrm{k}n}}\right) P_{\mathrm{kk}'}^{nn'}$$

where we applied the detailed balance equation in this limit, $P_{\mathrm{kk}'}^{nn'}(f_{\mathrm{k}'n'}^{0} - f_{\mathrm{k}n}^{0}) = 0$, and assumed isotropic scattering.

To evaluate the relaxation time we introduce the scattering angle

$$cos(\theta_{\mathrm{kk}'}) = \frac{\mathrm{v}_{\mathrm{k}'n'} \cdot \mathrm{v}_{\mathrm{k}n}}{|\mathrm{v}_{\mathrm{k}'n'}||\mathrm{v}_{\mathrm{k}n}|}$$

where
$\mathrm{v}_{\mathrm{k}n}$ is the electron velocity. Following [1] we obtain

$$\frac{1}{\tau_{\mathrm{k}n}} == \sum_{\mathrm{k}'n'} \frac{(1 - f_{\mathrm{k}'n'}^{0})}{(1 - f_{\mathrm{k}n}^{0})} (1 - cos(\theta_{\mathrm{kk}'}))\, P_{\mathrm{kk}'}^{nn'}$$

Mobility

Once knowing the relaxation times one obtain the mobility as:

(13)¶

$$\mu = -2q \frac{\sum_{\mathrm{k}n} v_{\mathrm{k}n}^{2} \frac{\partial f_{\mathrm{k}n}^{0}}{\partial \epsilon_{\mathrm{k}n}} \tau_{\mathrm{k}n}}{\sum_{\mathrm{k}n} f_{\mathrm{k}n}^{0}}$$

where the factor 2 accounts for spin degeneracy.

References

[1] (1,2)
T. Gunst, T. Markussen, K. Stokbro, and M. Brandbyge. First-principles method for electron-phonon coupling and electron mobility: applications to two-dimensional materials. *Phys. Rev. B*, 93:035414, Jan 2016. doi:10.1103/PhysRevB.93.035414.

[2]
Kristen Kaasbjerg, Kristian S Thygesen, and Karsten W Jacobsen. Unraveling the acoustic electron-phonon interaction in graphene. *Physical Review B*, 85(16):165440, 2012.

Previous

Next