DOCUMENTATION FOR

**First-principles simulation of electron transport and thermoelectric property of materials including electron-phonon scattering, defect scattering, and phonon drag**

Jiawei Zhou, Te-Huan Liu, Qichen Song, Qian Xu,

Zhiwei Ding, Bolin Liao, and Gang Chen

Department of Mechanical Engineering

Massachusetts Institute of Technology

Cambridge, MA 02139

September 3, 2020

Contents

1. Disclaimer
2. Software Structure
3. Input Parameter List
4. Installation
5. Example
   1. General Electron Transport Simulation (Si)
   2. Electron-Defect Scattering (PbTe)
   3. Phonon Drag (Si)
6. **Disclaimer**

This EPW code is a modified version of the EPW v4 from the open-source Quantum ESPRESSO suite (based on QE 5.4.0), and is released under GNU General Public License (v2). The source codes are published both at Materials Cloud (DOI: 10.24435/materialscloud:5a-7s) and GitHub (for comments and questions). The original EPW v4 is developed by S. Poncé, E.R. Margine, C. Verdi, and, F. Giustino, initially released inside Quantum ESPRESSO in 2016. This modified version is dedicated to the simulation of electron-phonon transport properties in quantum materials. Specifically, it calculates the electron-phonon and electron-defect scattering rates and uses them as inputs in Boltzmann transport equation to obtain transport properties (e.g. electrical conductivity, mobility, Seebeck coefficient, thermoelectric power factor, and electronic thermal conductivity).

As part of the Quantum ESPRESSO suite, please cite:

"P. Giannozzi et al., *J. Phys.:Condens. Matter* 21 395502 (2009); URL http://www.quantum-espresso.org",

in publications or presentations arising from this work. More details at http://www.quantum-espresso.org/quote.

Please also consider citing the EPW papers:

1) F. Giustino, M. L. Cohen, and S. G. Louie, *Phys. Rev. B* 76, 165108 (2007)

2) S. Poncé, E. R. Margine, C. Verdi, and F. Giustino, *Comput. Phys. Comm.* 209, 116 (2016)

For electron transport property calculations in materials with strong spin-orbit coupling or polar optical phonon scattering, please consider citing:

3) T.-H. Liu, J.W. Zhou, M.D. Li, Z.W. Ding, Q.C. Song, B.L. Liao, L. Fu, and G. Chen, *Proceedings of National Academy of Sciences*, 115, 879 (2018)

4) T.-H. Liu, J.W. Zhou, B.L. Liao, D.J. Singh, and G. Chen, *Phys. Rev. B*, 95, 075206 (2017)

For phonon drag calculations, please consider citing:

5) J.W. Zhou, B.L. Liao, B. Qiu, S. Huberman, K. Esfarjani, M.S. Dresselhaus and G. Chen, *Proceedings of National Academy of Sciences*, 112, 14777 (2015)

For questions related to the general usage of EPW and the functions of the original EPW code, please refer to the EPW website (<https://epw-code.org/>) and references there. For questions related to the modified EPW code, please address to *Issues* at our GitHub repository page. Comments to the code are also welcome.

Acknowledgement: The codes were developed over the years under the support of the following programs: S3TEC, an Energy Frontier Research Center funded by the U.S. Department of Energy, Office of Basic Energy Sciences under Award No. DE- SC0001299/DE-FG02-09ER46577 and the DARPA MATRIX program, under Grant HR0011-16-2-0041.

1. **Software Structure**

Under the /EPW-code folder, the software structure is

/code ==> Contains the EPW source code within the Quantum ESPRESSO package, together with a folder listing all added/modified source files.

* /code/change/ ==> All added/modified source files compared to the original QE and EPW code
* /code/espresso-5.4.0/ ==> Modified QE and EPW package

/example ==> Contains example files to illustrate transport property calculations in different cases (general electron transport, electron-defect scattering, and phonon drag simulation).

A detailed explanation of the modifications in the source code can be found in the README file under /code/change/EPW/. Below we briefly list the new subroutines we have written and subroutines that have been modified. The following subroutines are new source files and added into the EPW package.

bte\_check.f90

bte\_driver.f90

bte\_export.f90

bte.f90

bte\_impurity.f90

bte\_io.f90

bte\_iteration.f90

bte\_phcheck.f90

bte\_symm.f90

bte\_transpt.f90

edos.f90

eimpbloch2wane.f90

eimpbloch2wanp.f90

eimpmat\_shuffle.f90

eimpwan2bloch2.f90

eimpwan2bloche.f90

ephwan2bloch2.f90

ephwan2bloche.f90

epw\_explore.f90

evcbloch2wan.f90

evcwan2bloch.f90

fermilocation.f90

interp\_thl.f90

para\_thl.f90

phdrag\_shuffle.f90

tetra.f90

velwan2bloch.f90

vgwan2bloch.f90

wannier\_lib.F90

wsweight.f90

The following source files in the original EPW version are modified.

allocate\_epwq.f90

bcast\_epw\_input.f90

constants\_epw.f90

dynbloch2wan.f90

dynwan2bloch.f90

elph2.f90

elphon\_shuffle\_wrap.f90

ephbloch2wanp.f90

ephwan2blochp.f90

ephwann\_shuffle.f90

epwcom.f90

epw.f90

epw\_readin.f90

fermiwindow.f90

io\_dyn\_mat2.f90

io\_epw.f90

loadumat.f90

nesting\_fn.f90

pw2wan90epw.f90

readmat\_shuffle2.f90

rigid\_epw.f90

selfen\_elec.f90

selfen\_phon.f90

spectral\_func.f90

wannierize.f90

write\_ephmat.f90

In addition, the following files for compilation are also modified accordingly.

make.depend

Makefile

1. **Input Parameter List**

Here list parameters added for transport property calculations.

|  |  |
| --- | --- |
| Parameter name | Function |
| bte | Controls the type of calculation, 0 for common electron transport calculation (including electron-defect scattering), 1 for phonon drag |
| phdrag | .true. for phonon drag |
| shengbte\_read | true, if phonon information are read from ShengBTE generated files |
| phwmax | Maximum phonon frequency (in cm-1) considered when performing transport calculation |
| phkmax | Maximum phonon wave vector (in reciprocal lattice vector) considered when performing phonon drag integration |
| eimp\_mode | Controls the type of impurity scattering, 0 for no electron-defect scattering, 8 for electron-defect scattering with momentum scattering rates |
| eimp\_ls\_mode | Controls how electron-defect scatterings are considered. 0: considers full defect potential; 1: considers only long-range Coulomb part; 2: considers only short-range part |
| dvimpsr | Determines if short-range defect potential has been calculated. For calculating electron-defect scattering first time, set to .false. |
| dvimpq | Determines if electron-defect scattering matrix has been calculated. For calculating electron-defect scattering first time, set to .false. |
| defectname | Filename of defect potential (dv\_tot.[defectname].dat) |
| imp\_charge | Charge of impurity, generally +1 if n-type, -1 if p-type |
| dielec | Dielectric constant, used for impurity scattering calculation |
| alloy\_pot | true, if alloy scattering is considered, alloy potential to be read from file |
| frac\_type | [X Y], alloying composition when considering alloy scattering |
| bte\_o | true, if electron-phonon scattering information to be written into files |
| egap\_rbm | Specified band gap (in eV; usually used if DFT calculated band gap is incorrect). This will shift the entire conduction band. |
| fsthick | Energy range when evaluating Boltzmann transport integral (in eV; 0.2 eV at room temperature is usually sufficient) |
| epthick | Extra energy range when counting final states (if the initial state is near/at the boundaries set by fsthick) |
| lpolar | Controls whether polar optical phonon scattering is considered. .true. if polar materials; .false. if non-polar materials. |
| screen\_polar | If true, considers the screening effect on the polar optical phonon scattering |
| eptemp(:) | Temperatures (K) to be calculated |
| epdope(:) | Doping levels (cm-3) to be calculated |
| smearing | Integration method for evaluating Fermi’s golden rule, usually ‘tetra’ is recommended |
| nptype | Type of electron transport, for one type, can specify either ‘n’ or ‘p’ |
| vg\_el, vg\_ph | Controls the method to calculate the group velocity of electrons and phonons. ‘matrix’ uses the diagonal terms of the dipole matrix, while ‘linear’ directly evaluates the group velocity by calculating d(energy)/d(k). |
| asr\_eph | Controls whether acoustic sum rule is employed for electron-phonon coupling matrices. Usually set to be .true. |
| ifc\_read, ph\_read, ephl\_read | Controls whether additional information (force constants etc) can be read from filles to save time. Usually .false. |
| save\_m\_mat, save\_m\_matw, save\_m\_ph,  save\_t\_el | Controls whether electron-phonon coupling matrices are read from files to save memory. For calculations with dense mesh, [save\_m\_matw] can be set to .true. |
| nkfdos1, nkfdos2, nkfdos3 | Electron mesh for density of states calculation |

1. **Installation**

The modified EPW code is contained within the ESPRESSO package (under /espresso-5.4.0/EPW), and is compatible with ESPRESSO-5.4.0. Within the /EPW folder, the software structure is

/bin ==> Contains the epw.x link to the EPW executable.

/src ==> Contains all the EPW source files.

To install EPW, first install relevant packages of the espresso program (including PW and PH). The make.sys file needs to be changed based on the specific environment being used. Then inside /espresso-5.4.0/EPW/src, run ./make, and this will compile EPW files (we use intel compiler with version number 1.0.080, and openmpi/intel compiler with version number 1.8.0 for parallelization). The EPW executable will be found in /espresso-5.4.0/EPW/bin.

A separate folder (under /example) is provided, which includes three examples of transport property calculations (general electron transport, electron-defect scattering, and phonon drag). More details of these examples are given below.

1. **Example**
   1. **General Electron Transport Simulation (Si)**

Our code computes general electron transport properties, including electrical conductivity, mobility, Seebeck coefficient, thermoelectric power factors, and electronic thermal conductivity. The code also takes into account the polar optical phonon scattering, and the screening effect on the polar scattering at high carrier densities. The code can compute the transport properties for materials with strong spin-orbit couplings. For general electron transport calculations, the following files are needed (folder /1-Si-mobility)

* Electron-phonon interaction files generated by DFPT on a coarse **q** (phonon wave vector) mesh, including
  + Phonon perturbed potential (\*.dvscf)
  + Phonon dynamical matrices (\*.dyn)
  + Other information files
* EPW input files, including
  + pw\_scf.in (electron band structure)
  + pw\_nscf.in (generate electron wavefunctions on a coarse mesh)
  + epw.in (EPW calculation)

An example for calculating electron mobility in silicon is given in /example/1-Si-mobility folder, with corresponding output files given in folder /1-Si-mobility/epw\_results/out/. To prepare for the EPW calculation, the electron-phonon information is obtained via the DFPT calculation using Quantum Espresso code (DFPT). In our silicon example, a 6x6x6 mesh has been used for phonons. To run the EPW calculation, three successive steps are performed. The bash script that runs these steps is provided (Si.epw). The first step computes the electronic ground state of the material (input pw\_scf.in), while the second step interpolates the band structure to a desired mesh for later EPW calculations (input pw\_nscf.in). The third step performs the EPW calculation (input epw.in).

If EPW calculations are run for the first time, the input file epw.in has to be used, where a few parameters are set such that certain information are written to files. For later calculations (for example to evaluate the transport properties at different carrier densities or temperatures), these files do not need to be calculated again. In such case, the input file epw.in2 can be used, with corresponding parameters (doping level or temperature) changed. A more detailed explanation on the method and the parameter list for the EPW calculations can be found in the EPW website (<https://epw-code.org/>).

The input parameters in the EPW input file (epw.in) are explained below (comments are separated from the script with ‘!’ label).

EPW Si

&inputepw

! basic parameters, mesh, control, etc

prefix = 'Si'

outdir = 'output/' ! location where temporary files are written

dvscf\_dir = 'dvscf\_dyn' ! location where \*.dvscf and \*.dyn files can be found

amass(1) = 28.0855 ! atomic mass

iverbosity = 0 ! minimize writing information

nq1 = 6 ! nq1..3: phonon mesh number in the coarse mesh

nq2 = 6

nq3 = 6

nqf1 = 60 ! nqf1..3: phonon mesh number in the fine mesh

nqf2 = 60

nqf3 = 60

nk1 = 12 ! nk1..3: electron mesh number in the coarse mesh

nk2 = 12

nk3 = 12

nkf1 = 60 ! nkf1..3: electron mesh number in the fine mesh

nkf2 = 60

nkf3 = 60

elph = .true. ! calculate electron-phonon interaction, usually true

eig\_read = .false. ! if true, read band structure from files

kmaps = .false. ! if true, read kmaps from files (false if first time run)

epbwrite = .true. ! if true, read e-ph matrix from files (false if first time run)

epbread = .false. ! if true, read e-ph matrix from files (false if first time run)

epwwrite = .true.

epwread = .false.

wannierize = .true. ! if true, read wannier info from files (false if first time run)

edos\_read = .false. ! if true, read DOS from files (false if first time run)

etf\_mem = .true. ! EPW parameters (see EPW website for details)

parallel\_k = .true.

parallel\_q = .false.

elinterp = .true.

phinterp = .true.

tshuffle2 = .true.

tphases = .false.

elecselfen = .true.

phonselfen = .false.

a2f = .false.

! Wannierization parameters

mixing = 0.33 ! Wannierization parameters (See EPW website for details)

dis\_win\_min = -20.0

dis\_win\_max = 17.7

dis\_froz\_min = -20.0

dis\_froz\_max = 6.4

nbndsub = 8

spinors = .false.

num\_cg\_steps = 20

trial\_step = 0.5

guiding\_centres = .true.

num\_iter = 5000

dis\_num\_iter = 1500

proj(1) = 'f=0.0000,0.0000,0.0000:l=-3'

proj(2) = 'f=0.2500,0.2500,0.2500:l=-3'

wdata(1) = 'num\_print\_cycles = 200'

! for saving intermediate results

nbndskip = 0

ifc\_read = .false. ! if true, read interatomic force constant from files

ph\_read = .false. ! if true, read phonon info from files

ephl\_read = .false.

nkfdos1 = 240 ! electron mesh for DOS calculation

nkfdos2 = 240

nkfdos3 = 240

save\_m\_mat = .false.

save\_m\_matw = .false.

save\_m\_ph = .false.

save\_t\_el = .false.

vme = .false.

! transport parameters

bte = 0 ! bte = 0: electron transport calculation

bte\_o = .true.

egap\_rbm = 1.12 ! set the band gap by shifting conduction band

fsthick = 0.2 ! energy window for transport calculation (in eV)

degaussw = 0.5 ! Gaussian smearing parameter (not used)

eptemp(1) = 300.0 ! temperature (K)

epdope(1) = -1.0d19 ! doping concentration (cm^-1), assume fully ionized

epdope(2) = -2.0d19

epdope(3) = -4.0d19

lpolar = .false. ! true, if polar optical phonon scattering is included

asr\_eph = .true. ! true if acoustic sum rule is applied to e-ph matrix

smearing = 'tetra' ! integration method for calculating scattering rates

vg\_el = 'matrix' ! method to calculate electron group velocity

vg\_ph = 'matrix' ! method to calculalte phonon group velocity

nptype = 'n' ! semiconductor doping type

/

&input\_explore

/

16 cartesian ! irreducible q point on coarse mesh

0.000000000000000E+00 0.000000000000000E+00 0.000000000000000E+00 1.0

-0.166666666666667E+00 0.166666666666667E+00 -0.166666666666667E+00 1.0

-0.333333333333333E+00 0.333333333333333E+00 -0.333333333333333E+00 1.0

0.500000000000000E+00 -0.500000000000000E+00 0.500000000000000E+00 1.0

0.000000000000000E+00 0.333333333333333E+00 0.000000000000000E+00 1.0

-0.166666666666667E+00 0.500000000000000E+00 -0.166666666666667E+00 1.0

0.666666666666667E+00 -0.333333333333333E+00 0.666666666666667E+00 1.0

0.500000000000000E+00 -0.166666666666667E+00 0.500000000000000E+00 1.0

0.333333333333333E+00 0.277555756156289E-16 0.333333333333333E+00 1.0

0.000000000000000E+00 0.666666666666667E+00 0.000000000000000E+00 1.0

0.833333333333333E+00 -0.166666666666667E+00 0.833333333333333E+00 1.0

0.666666666666667E+00 -0.555111512312578E-16 0.666666666666667E+00 1.0

0.000000000000000E+00 -0.100000000000000E+01 0.000000000000000E+00 1.0

0.666666666666667E+00 -0.333333333333333E+00 0.100000000000000E+01 1.0

0.500000000000000E+00 -0.166666666666667E+00 0.833333333333333E+00 1.0

-0.333333333333333E+00 -0.100000000000000E+01 0.000000000000000E+00 1.0

After the EPW calculation, a /BTE folder will be generated, which includes the transport property results at specified temperatures and doping concentrations. For example, transport results corresponding to the first temperature and doping concentration can be found in /BTE/T1\_N1. Within this folder, the following files include detailed transport information:

rate\_RTA.dat: electron scattering rates (third column in THz, second column is electron energy in eV)

velo.dat: electron velocity (last three columns, in m/s)

Besides, under the same folder with the input file, a bte.out file also includes all transport properties (the meaning and unit of each variable is indicated at the end of this file). In addition, an output file containing all scattering rates and electron energy is written (filename: Scat\_rate\_[smearing]\_[temperature]\_[doping]). Each line of this file is formated as follows.

*k point coordinates (irreducible), electron energy (eV), el-ph total(abs+emi or intra+inter) scattering rate (THz), el-ph intra valley scattering rate (THz), el-ph inter valley scattering rate (THz), el-ph abs scattering rate (THz), el-ph emi scattering rate (THz)*

When electron-defect scatterings are considered (see the following Section), additional columns are written out following the above outputs:

*eimp total (intra+inter) scattering rate (THz), eimp intra scattering rate (THz), eimp inter scattering rate (THz)*

If electron-alloy scatterings are considered, additional columns are further written out:

*alloy-el total (intra+inter) scattering rate (THz), alloy-el intra scattering rate (THz), alloy-el inter scattering rate (THz)*

* 1. **Example: Electron-Defect Scattering (PbTe)**

Our code can also simulate electron scatterings by defects, including ionized dopants and alloys. Besides the input files needed for general transport property calculation, an additional file including the defect potential is needed (folder /2-PbTe-defect)

1. Defect potential (dv\_tot.[defectname].dat)

In order to compute the defect potential, supercell calculations with and without the defect are performed, and the resulting total electron potential are subtracted to obtain the defect potential. An example defect potential is given in the /2-PbTe-defect/epw\_results/ folder for Bi dopant on Pb atom (dv\_tot.Bi\_Pb.n2.relaxed.dat). A MATLAB code is also provided in folder /2-PbTe-defect which reads DFT supercell potential files (\*.xsf) as input and outputs a defect potential file to be used by EPW. We note that because spin-orbit coupling is considered in PbTe, the Wannierization parameters are different from the case of silicon (specifically, spinors = true, see epw.in file for details). An example for electron-defect scattering in PbTe is given in the folder /2-PbTe-defect (with corresponding output files in /2-PbTe-defect/epw\_results/out folder).

To consider the electron-defect scattering in the transport calculation, the following parameters are given in the input file

eimp\_mode = 8

eimp\_ls\_mode = 2

defectname = [Defect\_name]

imp\_charge = [Charge]

dielec = [Dielectric constant]

dvimpsr = .false.

dvimpq = .false.

Variable [eimp\_ls\_mode] determines at which level electron-defect scatterings are considered. Variable [defectname] specifies the defect potential file name (dv\_tot.[defectname].dat). Variable [imp\_charge] specifies the defect charge (e.g. imp\_charge = +1 for positively charged dopant). Variable [dielec] specifies the material’s dielectric constant. Variables [dvimpsr] and [dvimpq] tell whether short-range defect potential and electron-defect scattering matrices have been calculated respectively. For running defect scattering for the first time, these variables are set to .false.. Once short-range defect potential and electron-defect scattering matrices are written to files, these variables can be set to .true. to avoid re-calculations.

For charged defect scattering, eimp\_ls\_mode is set to 0, which adds up the long-range Coulomb potential and the short range defect potential when computing the electron-defect scattering matrix, (wave functions are obtained from PW calculations and interpolated onto fine mesh). The short range defect potential is extracted from the defect potential file, by subtracting the long range part based on the given impurity charge (imp\_charge) and dielectric constant (dielec). If one is only interested in the scattering by the long-range Coulomb part, eimp\_ls\_mode can be set to 1, which then assumes takes the form of screened Coulomb potential. In contrast, if one is interested in the scattering by the short range defect potential, eimp\_ls\_mode can be set to 2, which then assumes to be equal to the short range potential.

Alternatively, for evaluating charged defect scattering one can use the following parameter set

eimp\_mode = 2

defectname = [Defect\_name]

imp\_charge = [Charge]

dielec = [Dielectric constant]

which evaluates electron-charged defect scattering using Brooks-Herring model (ignores the short range potential, and assumes wavefunctions can be represented by plane waves). This can provide an approximate estimation for the charged defect scattering, but one should be cautious that the results could differ from the above configuration due to the assumption that wavefunctions are assumed as plane waves and short range perturbations are ignored.

For computing electron-alloying scattering in binary alloy, one can use the following parameter set (specifically, set alloy\_pot to true). One can still consider electron-dopant scattering at different doping concentrations in such case, but the dopant scattering is treated using Brooks-Herring model. The alloying scattering is considered by replacing the defect potential file with the alloy potential (obtained by supercell calculations where one host atom is replaced by an alloy atom, and subtracting the pristine supercell result). The [imp\_charge] in this case is 0 as the alloying atom usually has the same valence as the host atom and thus is not charged. In addition, the composition fractions need to be provided. For example, for Si0.8Ge0.2 alloy, frac\_type is 0.8 0.2.

eimp\_mode = 6

defectname = [Defect\_name]

imp\_charge = 0

dielec = [Dielectric constant]

alloy\_pot = .true.

frac\_type = [X\_composition] [Y\_composition]

dvimpsr = .false.

eimpbwrite = .true.

eimpbread = .false.

* 1. **Example: Phonon Drag** (Si)

Our code can also compute the phonon drag contribution to the Seebeck coefficient. For calculating phonon drag’s contribution, besides the input files needed for general transport property calculation (/dvscf\_dyn files can be copied from Example 1), additional files including phonon transport information are needed:

1. Phonon side

* BTE.qpoints\_full (phonon **q** mesh)
* CONTROL (phonon calculation parameters)
* espresso.ifc2 (harmonic force constants, inside folder /shengbte)
* FORCE\_CONSTANTS\_3RD (3rd-order anharmonic force constants, inside folder /shengbte)
* BTE.w (phonon lifetimes, inside folder /T300)

The code currently reads these phonon property files generated by the ShengBTE code (www.shengbte.org). To prepare for the phonon drag calculation using EPW, the phonon information in (a) are first obtained using ShengBTE code. An example for computing phonon drag in silicon is given in /3-Si-phdrag folder (with corresponding output files in folder /3-Si-phdrag/epw\_results/out). For more details in the theoretical formalism and simulation results on phonon drag calculation, please refer to

J.W. Zhou, B.L. Liao, B. Qiu, S. Huberman, K. Esfarjani, M.S. Dresselhaus and G. Chen, *Proceedings of National Academy of Sciences USA*, 112, 14777 (2015)

For phonon drag calculation, the following parameters need to be modified / added in the EPW input file (epw.in). Specifically, we have phdrag = .true. and bte = 1.

TO MODITY:

bte = 1

TO ADD:

phdrag = .true.

shengbte\_read = .true.

phwmax = 30 ! phonon frequency

phkmax = 0.2 ! phonon wavevector