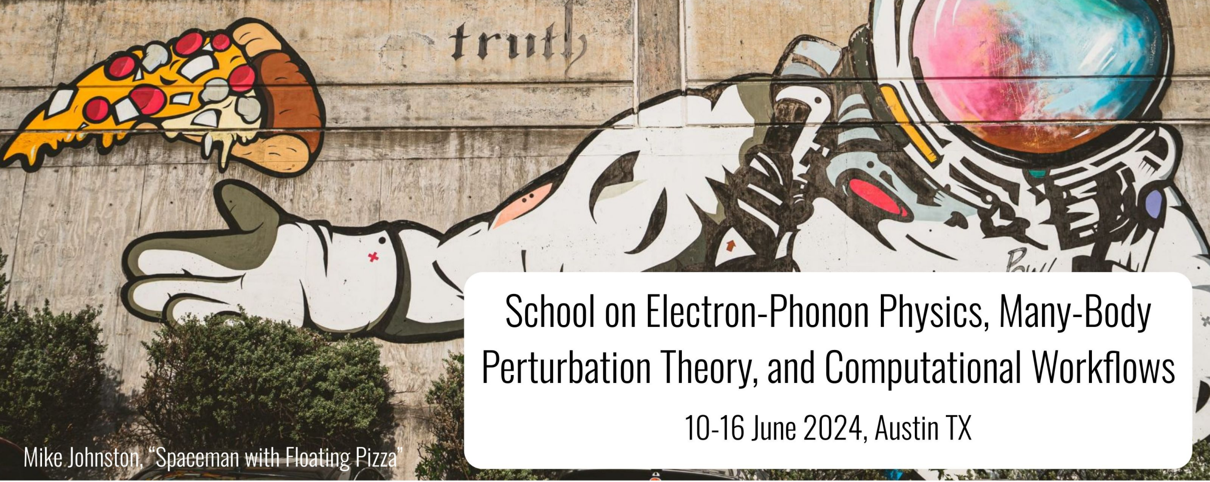


Mike Johnston, “Spaceman with Floating Pizza”

School on Electron-Phonon Physics, Many-Body Perturbation Theory, and Computational Workflows

10-16 June 2024, Austin TX

U.S. DEPARTMENT OF ENERGY

NSF

TACC
TEXAS ADVANCED COMPUTING CENTER

ODEN

Lecture Tue. 5

Introduction to EPW

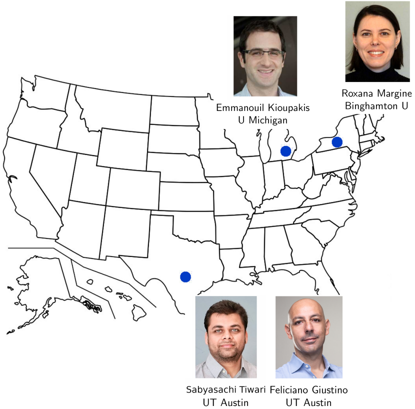
Sabyasachi Tiwari

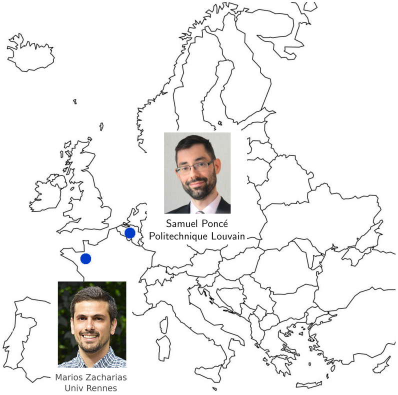
Oden Institute for Computational Engineering and Sciences
The University of Texas at Austin

Summary

- Computational flow in EPW
- Basic inputs for EPW
- Next in EPW
- Summary of exercises

EPW collaboration



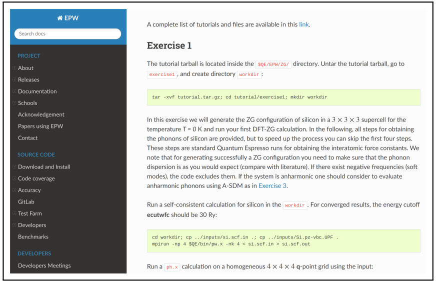


Developers

Kyle Bushick
Viet-Anh Ha
Samad Hajinazar
Jon Lafuente-Bartolomé
Hyungjun Lee
Johsua Leveillee
Chao Lian
Jae-Mo Lihm
Francesco Macheda
Hitoshi Mori
Hari Paudyal
Weng-Hong Sio
Aidan Thorn
Amanda Wang
Xiao Zhang

EPW resources

Docs



EPW

Search docs

PROJECT

- About
- Releases
- Documentation
- Schools
- Acknowledgement
- Papers using EPW
- Contact

SOURCE CODE

- Download and Install
- Code coverage
- Accuracy
- GitLab
- Test Farm
- Developers
- Benchmarks

DEVELOPERS

- Developers Meetings

A complete list of tutorials and files are available in this link.

Exercise 1

The tutorial tarball is located inside the `$QE/EPW/ZG/` directory. Untar the tutorial tarball, go to `exercise1`, and create directory `workdir`:

```
tar -xvf tutorial.tar.gz; cd tutorial/exercise1; mkdir workdir
```

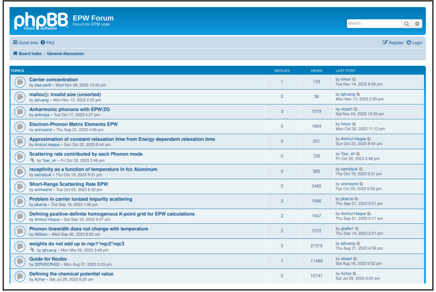
In this exercise we will generate the ZG configuration of silicon in a 3 × 3 × 3 supercell for the temperature $T = 0$ K and run your first DFT-ZG calculation. In the following, all steps for obtaining the phonons of silicon are provided, but to speed up the process you can skip the first four steps. These steps are standard Quantum Espresso runs for obtaining the interatomic force constants. We note that for generating successfully a ZG configuration you need to make sure that the phonon dispersion is as you would expect (compare with literature). If there exist negative frequencies (soft modes), the code excludes them. If the system is anharmonic one should consider to evaluate anharmonic phonons using A-SDM as in Exercise 3.

Run a self-consistent calculation for silicon in the `workdir`. For converged results, the energy cutoff **ecutwfc** should be 30 Ry:

```
cd workdir; cp ../inputs/si.scf.in .; cp ../inputs/Si.pz-vbc.UPF .
mpirun -np 4 $QE/bin/pw.x -nk 4 < si.scf.in > si.scf.out
```

Run a `ph.x` calculation on a homogeneous 4 × 4 × 4 **q**-point grid using the input:

Forum

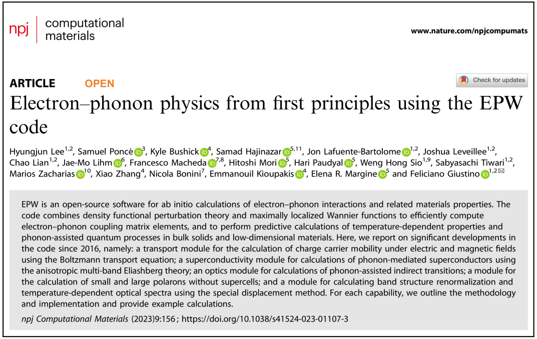


EPW Forum
Forum for EPW code

Board index ‹ General discussion

TOPICS	REPLIES	VIEWS	LAST POST
Carrier concentration by ilias-serifi » Wed Nov 08, 2023 12:00 pm	1	133	by hmori Tue Nov 14, 2023 8:45 pm
malloc(): invalid size (unsorted) by jqhuang » Mon Nov 13, 2023 2:55 pm	0	36	by jqhuang Mon Nov 13, 2023 2:55 pm
Anharmonic phonons with EPW/ZG by achintya » Tue Oct 17, 2023 4:27 pm	3	1019	by mzach Sat Nov 04, 2023 12:39 am
Electron-Phonon Matrix Elements EPW by animeshd » Thu Sep 21, 2023 4:56 pm	5	1904	by hmori Mon Oct 30, 2023 11:13 pm
Approximation of constant relaxation time from Energy dependent relaxation time by Aminul Haque » Sun Oct 22, 2023 8:44 am	0	201	by Aminul Haque Sun Oct 22, 2023 8:44 am
Scattering rate contributed by each Phonon mode by Tsar_sh » Fri Oct 20, 2023 2:46 pm	0	726	by Tsar_sh Fri Oct 20, 2023 2:46 pm
receptivity as a function of temperature in fcc Aluminum by samolyuk » Thu Oct 19, 2023 9:31 pm	0	585	by samolyuk Thu Oct 19, 2023 9:31 pm
Short-Range Scatteriing Rate EPW by animeshd » Tue Oct 03, 2023 6:32 pm	0	2485	by animeshd Tue Oct 03, 2023 6:32 pm
Problem in carrier ionized impurity scattering by pkarna » Tue Sep 19, 2023 1:46 pm	2	1566	by pkarna Thu Sep 21, 2023 6:31 pm
Defining positive-definite homogenous K-point grid for EPW calculations by Aminul Haque » Sat Sep 16, 2023 9:37 am	2	1547	by Aminul Haque Thu Sep 21, 2023 5:17 am
Phonon linewidth does not change with temperature by William » Wed Sep 06, 2023 8:55 am	2	1572	by gkafle1 Thu Sep 14, 2023 2:01 pm
weights do not add up to nqc1*nqc2*nqc3 by jqhuang » Mon Mar 06, 2023 3:48 pm	5	21319	by jqhuang Thu Aug 31, 2023 4:56 pm
Guide for Noobs by 22PhDCPH02 » Mon Aug 07, 2023 3:33 pm	1	11489	by stiwari Sat Aug 19, 2023 9:52 pm
Defining the chemical potential value by Azhar » Sat Jul 29, 2023 6:20 am	0	10747	by Azhar Sat Jul 29, 2023 6:20 am

Overview of work from 2016 to 2023



npj | computational materials

www.nature.com/npjcompumats

ARTICLE OPEN

Electron–phonon physics from first principles using the EPW code

Hyungjun Lee[1,2], Samuel Poncé[3], Kyle Bushick[4], Samad Hajinazar[5,11], Jon Lafuente-Bartolome[1,2], Joshua Leveillee[1,2], Chao Lian[1,2], Jae-Mo Lihm[6], Francesco Macheda[7,8], Hitoshi Mori[5], Hari Paudyal[5], Weng Hong Sio[1,9], Sabyasachi Tiwari[1,2], Marios Zacharias[10], Xiao Zhang[4], Nicola Bonini[7], Emmanouil Kioupakis[4], Elena R. Margine[5] and Feliciano Giustino[1,2]✉

EPW is an open-source software for ab initio calculations of electron–phonon interactions and related materials properties. The code combines density functional perturbation theory and maximally localized Wannier functions to efficiently compute electron–phonon coupling matrix elements, and to perform predictive calculations of temperature-dependent properties and phonon-assisted quantum processes in bulk solids and low-dimensional materials. Here, we report on significant developments in the code since 2016, namely: a transport module for the calculation of charge carrier mobility under electric and magnetic fields using the Boltzmann transport equation; a superconductivity module for calculations of phonon-mediated superconductors using the anisotropic multi-band Eliashberg theory; an optics module for calculations of phonon-assisted indirect transitions; a module for the calculation of small and large polarons without supercells; and a module for calculating band structure renormalization and temperature-dependent optical spectra using the special displacement method. For each capability, we outline the methodology and implementation and provide example calculations.

npj Computational Materials (2023)9:156 ; https://doi.org/10.1038/s41524-023-01107-3

Summary

- Computational flow in EPW
- Basic inputs for EPW
- Next in EPW
- Summary of exercises

Computational flow in EPW

The central quantity we calculate inside EPW is:

$$g_{mn\nu}(\mathbf{k}_{\text{fi}}, \mathbf{q}_{\text{fi}})$$

Mon. 1. Giustino,
Tue. 2. Giustino

[Phys. Rev. B 76, 165108 (2007), Rev. Mod. Phys. 89, 1 (2017)]

Computational flow in EPW

The central quantity we calculate inside EPW is:

$$g_{mn\nu}(\mathbf{k}_{\mathrm{fi}}, \mathbf{q}_{\mathrm{fi}}) \;\; \rightarrow$$

[Phys. Rev. B 76, 165108 (2007), Rev. Mod. Phys. 89, 1 (2017)]

Computational flow in EPW

The central quantity we calculate inside EPW is:

$$g_{mn\nu}(\mathbf{k}_{\text{fi}}, \mathbf{q}_{\text{fi}}) \;\rightarrow$$

Physical Properties

Computational flow in EPW

The central quantity we calculate inside EPW is:

$$g_{mn\nu}(\mathbf{k}_{\mathrm{fi}}, \mathbf{q}_{\mathrm{fi}}) \quad \rightarrow$$

Physical Properties

- Self-energies
  - Today: Lafuente-Bartolome

Computational flow in EPW

The central quantity we calculate inside EPW is:

$$g_{mn\nu}(\mathbf{k}_{\mathrm{fi}}, \mathbf{q}_{\mathrm{fi}}) \quad \rightarrow$$

Physical Properties

- Self-energies
  - Today: Lafuente-Bartolome
- Transport
  - Wed: 1. Poncé, 6. Ha

Computational flow in EPW

The central quantity we calculate inside EPW is:

$$g_{mn\nu}(\mathbf{k}_{\text{fi}}, \mathbf{q}_{\text{fi}}) \quad \rightarrow$$

Physical Properties

- Self-energies
  - Today: Lafuente-Bartolome
- Transport
  - Wed: 1. Poncé, 6. Ha
- Superconductivity
  - Wed: 2. Margine, 8. Mori

Computational flow in EPW

The central quantity we calculate inside EPW is:

$$g_{mn\nu}(\mathbf{k}_{\text{fi}}, \mathbf{q}_{\text{fi}}) \quad \rightarrow$$

Physical Properties

- Self-energies
  - Today: Lafuente-Bartolome
- Transport
  - Wed: 1. Poncé, 6. Ha
- Superconductivity
  - Wed: 2. Margine, 8. Mori
- Optics
  - Wed: 3. Kioupakis, Thu: 3. Zhang
  - Sat: 6. Tiwari

Computational flow in EPW

The central quantity we calculate inside EPW is:

$$g_{mn\nu}(\mathbf{k}_{\text{fi}}, \mathbf{q}_{\text{fi}}) \quad \rightarrow$$

Physical Properties

- Self-energies
  - Today: Lafuente-Bartolome
- Transport
  - Wed: 1. Poncé, 6. Ha
- Superconductivity
  - Wed: 2. Margine, 8. Mori
- Optics
  - Wed: 3. Kioupakis, Thu: 3. Zhang
  - Sat: 6. Tiwari
- Polarons
  - Thu: 2. Giustino, Fri: 5. Lafuente-Bartolome

Computational flow in EPW

The central quantity we calculate inside EPW is:

$$g_{mn\nu}(\mathbf{k}_{\text{fi}}, \mathbf{q}_{\text{fi}}) \quad \rightarrow$$

Physical Properties

- Self-energies
  - Today: Lafuente-Bartolome
- Transport
  - Wed: 1. Poncé, 6. Ha
- Superconductivity
  - Wed: 2. Margine, 8. Mori
- Optics
  - Wed: 3. Kioupakis, Thu: 3. Zhang
  - Sat: 6. Tiwari
- Polarons
  - Thu: 2. Giustino, Fri: 5. Lafuente-Bartolome
- WFPT
  - Sat: 4. Lihm

Computational flow in EPW

Fine-grid e-ph element:

$g_{mn\nu}(\mathbf{k}_{\text{fi}}, \mathbf{q}_{\text{fi}})$

Fine-grid

Computational flow in EPW

Fine-grid e-ph element:

$$\boxed{g_{mn\nu}(\mathbf{k}_{\text{co}}, \mathbf{q}_{\text{co}})} \rightarrow \qquad\qquad g_{mn\nu}(\mathbf{k}_{\text{fi}}, \mathbf{q}_{\text{fi}})$$

Coarse-grid

Fine-grid

Computational flow in EPW

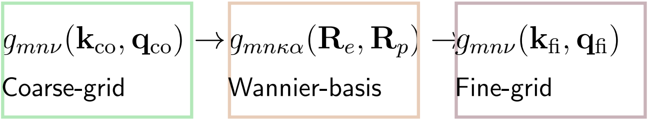
Fine-grid e-ph element:

$$g_{mn\nu}(\mathbf{k}_{\text{co}}, \mathbf{q}_{\text{co}}) \rightarrow g_{mn\kappa\alpha}(\mathbf{R}_e, \mathbf{R}_p) \rightarrow g_{mn\nu}(\mathbf{k}_{\text{fi}}, \mathbf{q}_{\text{fi}})$$

Coarse-grid Wannier-basis Fine-grid

Computational flow in EPW

Fine-grid e-ph element:



$g_{mn\nu}(\mathbf{k}_{\mathrm{co}}, \mathbf{q}_{\mathrm{co}}) \to g_{mn\kappa\alpha}(\mathbf{R}_e, \mathbf{R}_p) \to g_{mn\nu}(\mathbf{k}_{\mathrm{fi}}, \mathbf{q}_{\mathrm{fi}})$

Coarse-grid

Wannier-basis

Fine-grid

Computational flow in EPW

Fine-grid e-ph element:

$$\boxed{g_{mn\nu}(\mathbf{k}_{\mathrm{co}}, \mathbf{q}_{\mathrm{co}})}$$

Computational flow in EPW

Wavefunctions: QE/pw.x (nscf)

$$\boxed{g_{mn\nu}(\mathbf{k}_{\mathrm{co}}, \mathbf{q}_{\mathrm{co}})} = \langle u_{m\mathbf{k_{co}}+\mathbf{q_{co}}} | \Delta_{\mathbf{q_{co}}\nu} v^{\mathrm{KS}} | u_{n\mathbf{k_{co}}} \rangle$$

[Phys. Rev. B 76, 165108 (2007), Rev. Mod. Phys. 89, 1 (2017)]

Computational flow in EPW

Wavefunctions: QE/pw.x (nscf)

Polarization vectors and phonon frequencies: QE/ph.x

$$\boxed{g_{mn\nu}(\mathbf{k}_{\mathrm{co}},\mathbf{q}_{\mathrm{co}})} = \langle u_{m\mathbf{k_{co}}+\mathbf{q_{co}}} | \Delta_{\mathbf{q_{co}}\nu} v^{\mathrm{KS}} | u_{n\mathbf{k_{co}}} \rangle$$

$$\Delta_{\mathbf{q_{co}}\nu} v^{\mathrm{KS}} = \sum_{\kappa\alpha} \sqrt{\frac{\hbar}{2M_\kappa \omega_{\mathbf{q_{co}}\nu}}} e_{\kappa\alpha}(\mathbf{q_{co}}) \partial_{\kappa\alpha,\mathbf{q_{co}}} v^{\mathrm{KS}}$$

[Phys. Rev. B 76, 165108 (2007), Rev. Mod. Phys. 89, 1 (2017)]

Computational flow in EPW

Wavefunctions: QE/pw.x (nscf)

$$\boxed{g_{mn\nu}(\mathbf{k}_{\rm co},\mathbf{q}_{\rm co})} = \langle u_{m\mathbf{k_{co}}+\mathbf{q_{co}}} | \Delta_{\mathbf{q_{co}}\nu} v^{\rm KS} | u_{n\mathbf{k_{co}}} \rangle$$

Polarization vectors and phonon frequencies: QE/ph.x

$$\Delta_{\mathbf{q_{co}}\nu} v^{\rm KS} = \sum_{\kappa\alpha} \sqrt{\frac{\hbar}{2M_{\kappa}\omega_{\mathbf{q_{co}}\nu}}} e_{\kappa\alpha}(\mathbf{q_{co}}) \partial_{\kappa\alpha,\mathbf{q_{co}}} v^{\rm KS}$$

Perturbing potential: QE/ph.x
pattern representation

$$\partial_{\kappa\alpha,\mathbf{q_{co}}} v^{\rm KS} = e^{(-i\mathbf{q_{co}}.r)} \sum_{p} e^{(-i\mathbf{q_{co}}.\mathbf{R}_{\rm p})} \frac{\partial V^{\rm KS}(r)}{\partial \tau_{\kappa\alpha p}}$$

[Phys. Rev. B 76, 165108 (2007), Rev. Mod. Phys. 89, 1 (2017)]

Computational flow in EPW

Inputs to EPW

Wavefunctions: QE/pw.x (nscf)

Polarization vectors and phonon frequencies: QE/ph.x

Perturbing potential: QE/ph.x pattern representation

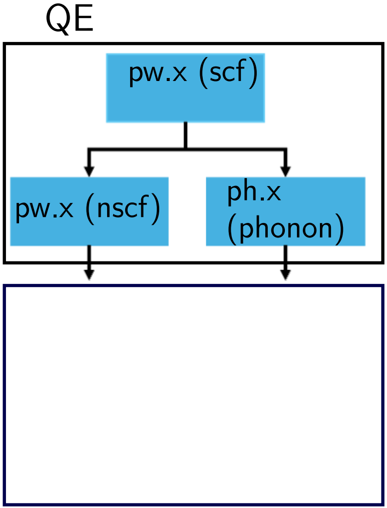
$$\boxed{g_{mn\nu}(\mathbf{k}_{\mathrm{co}}, \mathbf{q}_{\mathrm{co}})} = \langle u_{m\mathbf{k_{co}}+\mathbf{q_{co}}} | \Delta_{\mathbf{q_{co}}\nu} v^{\mathrm{KS}} | u_{n\mathbf{k_{co}}} \rangle$$

$$\Delta_{\mathbf{q_{co}}\nu} v^{\mathrm{KS}} = \sum_{\kappa\alpha} \sqrt{\frac{\hbar}{2M_\kappa \omega_{\mathbf{q_{co}}\nu}}} e_{\kappa\alpha}(\mathbf{q_{co}}) \partial_{\kappa\alpha,\mathbf{q_{co}}} v^{\mathrm{KS}}$$

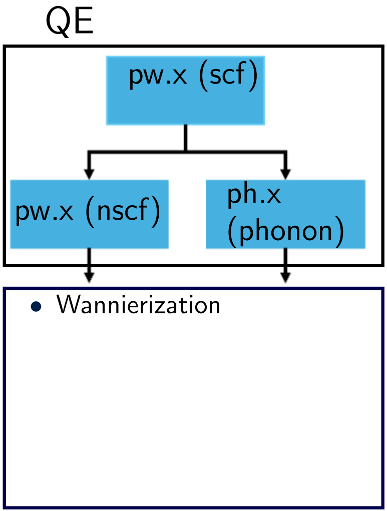
$$\partial_{\kappa\alpha,\mathbf{q_{co}}} v^{\mathrm{KS}} = e^{(-i\mathbf{q_{co}}.r)} \sum_p e^{(-i\mathbf{q_{co}}.\mathbf{R}_{\mathrm{p}})} \frac{\partial V^{\mathrm{KS}}(r)}{\partial \tau_{\kappa\alpha p}}$$

[Phys. Rev. B 76, 165108 (2007), Rev. Mod. Phys. 89, 1 (2017)]

Computational flow in EPW



Computational flow in EPW



QE

pw.x (scf)

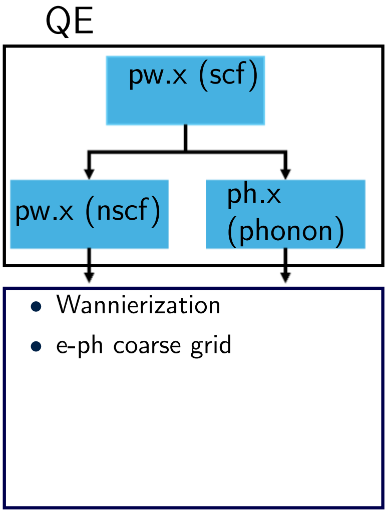
pw.x (nscf)

ph.x (phonon)

- Wannierization

$$w_m(r - \mathbf{R}_e) = \frac{1}{N} \sum_{n\mathbf{k}} e^{-i\mathbf{k}.\mathbf{R}_e} U_{mn\mathbf{k}} |\psi_{n\mathbf{k}}\rangle$$

Computational flow in EPW



- Wannierization
- e-ph coarse grid

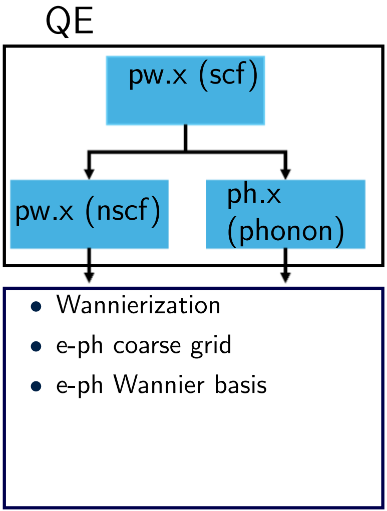
$$g_{mn\nu}(\mathbf{k}_{\mathrm{co}}, \mathbf{q}_{\mathrm{co}}) = \langle u_{m\mathbf{k}_{\mathbf{co}}+\mathbf{q}_{\mathbf{co}}} | \Delta_{\mathbf{q}_{\mathbf{co}}\nu} v^{\mathrm{KS}} | u_{n\mathbf{k}_{\mathbf{co}}} \rangle$$

$$\Delta_{\mathbf{q}_{\mathbf{co}}\nu} v^{\mathrm{KS}} = \sum_{\kappa\alpha} \sqrt{\frac{\hbar}{2M_{\kappa}\omega_{\mathbf{q}_{\mathbf{co}}\nu}}} e_{\kappa\alpha}(\mathbf{q}_{\mathbf{co}}) \partial_{\kappa\alpha,\mathbf{q}_{\mathbf{co}}} v^{\mathrm{KS}}$$

$$\partial_{\kappa\alpha,\mathbf{q}_{\mathbf{co}}} v^{\mathrm{KS}} = e^{(-i\mathbf{q}_{\mathbf{co}}.r)} \sum_{p} e^{(-i\mathbf{q}_{\mathbf{co}}.\mathbf{R}_{p})} \frac{\partial V^{\mathrm{KS}}(r)}{\partial \tau_{\kappa\alpha p}}$$

[Phys. Rev. B 76, 165108 (2007), Rev. Mod. Phys. 89, 1 (2017)]

Computational flow in EPW



QE

pw.x (scf)

pw.x (nscf)

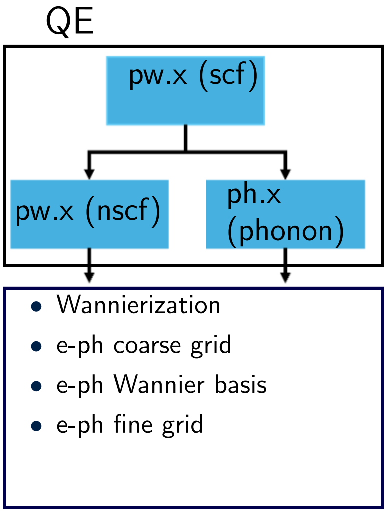
ph.x (phonon)

- Wannierization
- e-ph coarse grid
- e-ph Wannier basis

$$g_{mn\kappa\alpha}(\mathbf{R}_p, \mathbf{R}_{p'}) = \langle w_m(r) | \frac{\partial V^{\mathrm{KS}}(r - \mathbf{R}_{p'})}{\partial \tau_{\kappa\alpha p}} | w_n(r - \mathbf{R}_p) \rangle$$

[Phys. Rev. B 76, 165108 (2007), Rev. Mod. Phys. 89, 1 (2017)]

Computational flow in EPW

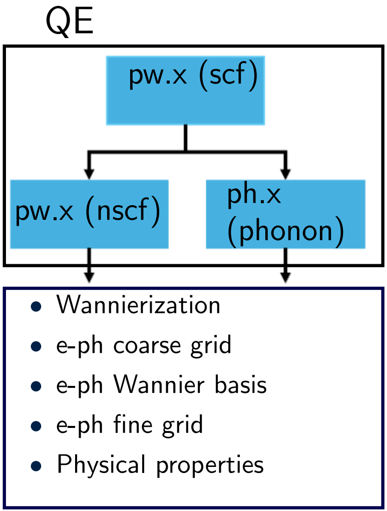


- Wannierization
- e-ph coarse grid
- e-ph Wannier basis
- e-ph fine grid

$$g_{mn\nu}(\mathbf{k}_{\text{fi}}, \mathbf{q}_{\text{fi}}) = \sqrt{\frac{\hbar}{2M_\kappa \omega_{\mathbf{q}_{\text{fi}}\nu}}} \sum_{\mathbf{R}_p \mathbf{R}_{p'}} e^{i(\mathbf{k}_{\text{fi}}\mathbf{R}_p + \mathbf{k}_{\text{fi}}\mathbf{R}_{p'})} \times [U_{m\mathbf{k}_{\text{fi}}+\mathbf{q}_{\text{fi}}} g_{mn\kappa\alpha}(\mathbf{R}_p, \mathbf{R}_{p'}).e_{\kappa\alpha,\nu}(\mathbf{q}_{\text{fi}}) U_{n\mathbf{k}_{\text{fi}}}]$$

[Phys. Rev. B 76, 165108 (2007), Rev. Mod. Phys. 89, 1 (2017)]

Computational flow in EPW



QE

pw.x (scf)

pw.x (nscf)

ph.x (phonon)

- Wannierization
- e-ph coarse grid
- e-ph Wannier basis
- e-ph fine grid
- Physical properties

Physical Properties

- Self-energies
  - Today: Lafuente-Bartolome
- Transport
  - Wed: 1. Poncé, 6. Ha
- Superconductivity
  - Wed: 2. Margine, 8. Mori
- Optics
  - Wed: 3. Kioupakis, Thu: 3. Zhang
  - Sat: 6. Tiwari
- Polarons
  - Thu: 2. Giustino, Fri: 5. Lafuente-Bartolome
- WFPT
  - Sat: 4. Lihm

Summary

- Computational flow in EPW
- Basic inputs for EPW
- Next in EPW
- Summary of exercises

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .true.
epwread = .false.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .true.
epwread = .false.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← I/O flags

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .true.
epwread = .false.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← I/O flags

← Control flags

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .true.
epwread = .false.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← I/O flags

← Control flags

← Wannier/electronic-structure flags

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .true.
epwread = .false.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← I/O flags

← Control flags

← Wannier/electronic-structure flags

← Coarse grid

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .true.
epwread = .false.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← Prefix which was used in scf calculation

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .true.
epwread = .false.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← Mass of ions and output directory

Basic inputs

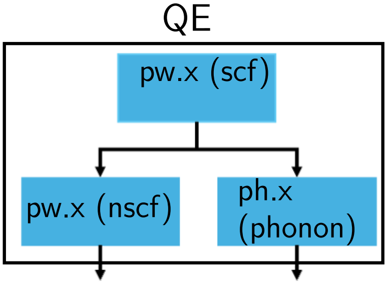
```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .true.
epwread = .false.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← Files where dvscf files are saved

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .true.
epwread = .false.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

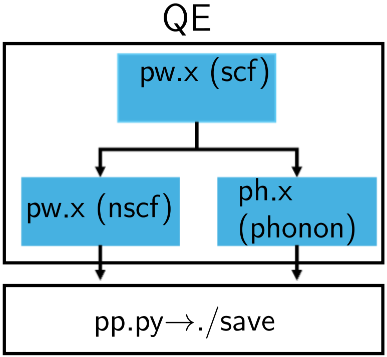
← Files where dvscf files are saved



Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .true.
epwread = .false.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

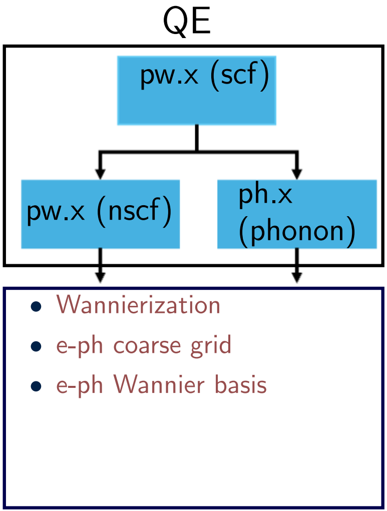
← Files where dvscf files are saved



Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .true.
epwread = .false.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← Calculation and restart



QE

pw.x (scf)

pw.x (nscf)

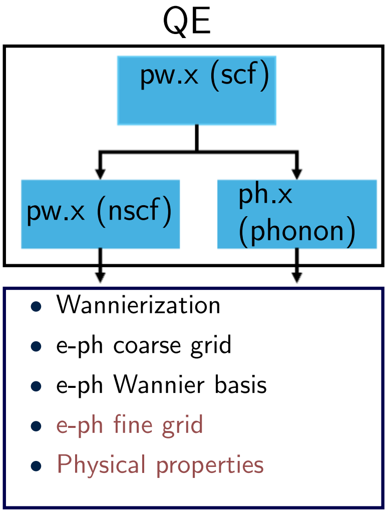
ph.x (phonon)

- Wannierization
- e-ph coarse grid
- e-ph Wannier basis

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .false.
epwread = .true.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

←
Calculation and restart



QE

pw.x (scf)

pw.x (nscf)

ph.x (phonon)

- Wannierization
- e-ph coarse grid
- e-ph Wannier basis
- e-ph fine grid
- Physical properties

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .false.
epwread = .true.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← Bands after Wannierization

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .false.
epwread = .true.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← Starting projections for Wannierization

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .false.
epwread = .true.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← To Wannierize or not

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .false.
epwread = .true.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .false.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← To Wannierize or not

Not needed for restart

Basic inputs

```
--
&inputepw
prefix = 'si'
amass(1) = 28.08
outdir = './'
dvscf_dir = './save'

elph = .true.
epwwrite = .false.
epwread = .true.

nbndsub = 8
proj(1) = 'Si : sp3'
wannierize = .true.

nk1 = 6
nk2 = 6
nk3 = 6
nq1 = 3
nq2 = 3
nq3 = 3
/
```

← Coarse grids used for nscf and ph calculations

Application based inputs

```
--
&inputepw
..
loptabs = .true.
scattering = .true.
temps = 300
...
/
```

Application based inputs

```
--
&inputepw
..
loptabs = .true.
scattering = .true.
temps = 300
...
/
```

← Learn in respective sessions

Application based inputs

```
--
&inputepw
..
loptabs = .true.
scattering = .true.
temps = 300
...
/
```

← Learn in respective sessions

Physical Properties

- Self-energies
  - Today: Lafuente-Bartolome
- Transport
  - Wed: 1. Poncé, 6. Ha
- Superconductivity
  - Wed: 2. Margine, 8. Mori
- Optics
  - Wed: 3. Kioupakis, Thu: 3. Zhang
  - Sat: 6. Tiwari
- Polarons
  - Thu: 2. Giustino, Fri: 5. Lafuente-Bartolome
- WFPT
  - Sat: 4. Lihm

Parallelization in EPW

- Current EPW version is only parallelized over k-grid

Parallelization in EPW

- Current EPW version is only parallelized over k-grid
  - mpirun -np <no of cores> <location of q-e bin>/epw.x -nk <no of cores> -in epw.in > epw.out

Parallelization in EPW

- Current EPW version is only parallelized over k-grid
  - mpirun -np <no of cores> <location of q-e bin>/epw.x -nk <no of cores> -in epw.in > epw.out
  - e.g., $6 \times 6 \times 6$ $\mathbf{k}$ grid: maximum 216 cores to utilize

Parallelization in EPW

- Current EPW version is only parallelized over k-grid
  - mpirun -np <no of cores> <location of q-e bin>/epw.x -nk <no of cores> -in epw.in > epw.out
  - e.g., $6 \times 6 \times 6$ $\mathbf{k}$ grid: maximum 216 cores to utilize
  - mpirun -np 216 <location of q-e bin>/epw.x -nk 216 -in epw.in > epw.out

EPW+ZG

- Special displacements for electron-phonon
- Special displacements anharmonicity
- Fri. 6. Zacharias

Marios Zacharias
Univ Rennes

Summary

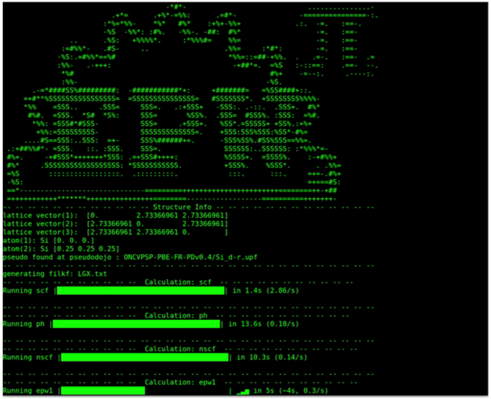
- Computational flow in EPW
- Basic inputs for EPW
- Next in EPW
- Summary of exercises

Next in EPW

In progress (beyond EPW v5.9):

- Exciton-phonon interactions
  - Fri. 4. Dai
- Many-body corrections to polarons
- Non-adiabatic superconductivity

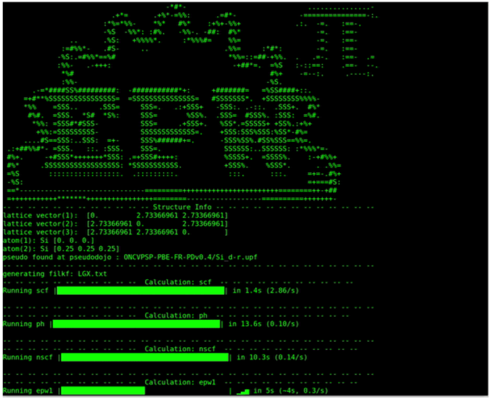
The EPWpy automation suite



```
-- -- -- -- -- -- -- -- -- -- -- -- -- Structure Info -- -- -- -- -- -- -- -- -- -- -- -- --
lattice vector(1):  [0.         2.73366961 2.73366961]
lattice vector(2):  [2.73366961 0.         2.73366961]
lattice vector(3):  [2.73366961 2.73366961 0.        ]
atom(1): Si [0. 0. 0.]
atom(2): Si [0.25 0.25 0.25]
pseudo found at pseudodojo : ONCVPSP-PBE-FR-PDv0.4/Si_d-r.upf
-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
generating filkf: LGX.txt
-- -- -- -- -- -- -- -- -- -- -- --  Calculation: scf  -- -- -- -- -- -- -- -- -- -- -- --
Running scf |████████████████████████████████| in 1.4s (2.86/s)

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- -- -- -- -- -- --  Calculation: ph  -- -- -- -- -- -- -- -- -- -- -- --
Running ph |████████████████████████████████| in 13.6s (0.10/s)

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- -- -- -- -- -- --  Calculation: nscf  -- -- -- -- -- -- -- -- -- -- -- --
Running nscf |████████████████████████████████| in 10.3s (0.14/s)

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- -- -- -- -- -- --  Calculation: epw1  -- -- -- -- -- -- -- -- -- -- -- --
Running epw1 |████████████████                | ▁▃▅ in 5s (~4s, 0.3/s)
```

- Python abstraction layer for EPW workflows
- High-level access to output data as objects
- Designed for HPC
- Developed under CI/CD
- Highly modular and extensible
- Intuitive for users and developers

The EPWpy automation suite



```
-- -- -- -- -- -- -- -- -- -- -- -- -- Structure Info -- -- -- -- -- -- -- -- -- -- -- -- -- --
lattice vector(1):  [0.         2.73366961 2.73366961]
lattice vector(2):  [2.73366961 0.         2.73366961]
lattice vector(3):  [2.73366961 2.73366961 0.        ]
atom(1): Si [0. 0. 0.]
atom(2): Si [0.25 0.25 0.25]
pseudo found at pseudodojo : ONCVPSP-PBE-FR-PDv0.4/Si_d-r.upf
-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
generating filkf: LGX.txt
-- -- -- -- -- -- -- -- -- -- -- --  Calculation: scf  -- -- -- -- -- -- -- -- -- -- -- --
Running scf |████████████████████████████████████████| in 1.4s (2.86/s)

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- -- -- -- -- -- --  Calculation: ph  -- -- -- -- -- -- -- -- -- -- -- --
Running ph |████████████████████████████████████████| in 13.6s (0.10/s)

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- -- -- -- -- -- --  Calculation: nscf  -- -- -- -- -- -- -- -- -- -- -- --
Running nscf |████████████████████████████████████████| in 10.3s (0.14/s)

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- -- -- -- -- -- --  Calculation: epw1  -- -- -- -- -- -- -- -- -- -- -- --
Running epw1 |████████████████████                    | ▁▃▅ in 5s (~4s, 0.3/s)
```

- Python abstraction layer for EPW workflows
- High-level access to output data as objects
- Designed for HPC
- Developed under CI/CD
- Highly modular and extensible
- Intuitive for users and developers

Sun. 2. Tiwari and Cucco

EPW resources

- Website: epw-code.org
- Forum: forum.epw-code.org/index.php
- References:
  - [1]. npj comput. mater., 9, 156 (2023)
  - [2]. Comput. Phys. Commun. 209, 116 (2016)
  - [3]. Phys. Rev. B 76, 165108 (2007)

Tutorial Tue. 5

Hands-On Intro: Running EPW

Jon Lafuente-Bartolome

University of the Basque Country

UPV/EHU

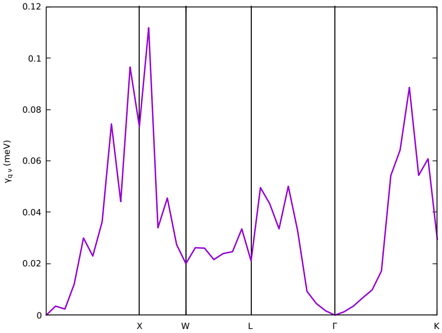
Summary

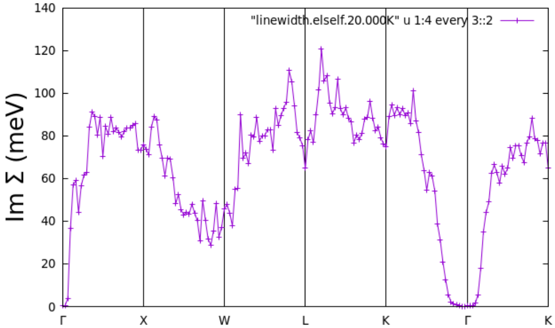
- Computational flow in EPW
- Basic inputs for EPW
- Next in EPW
- Summary of exercises

Summary of exercises

- **Exercise 1:** Phonon linewidths in Pb (metal)
- **Exercise 2:** Electron linewidths in SiC (polar semiconductor)

Main goal: To learn the basic usage of `EPW` and how to check the quality and convergence of the calculations.


$\gamma_{q\nu}$ (meV)
X
W
L
Γ
K


Im Σ (meV)
"linewidth.elself.20.000K" u 1:4 every 3::2
Γ
X
W
L
K
Γ
K