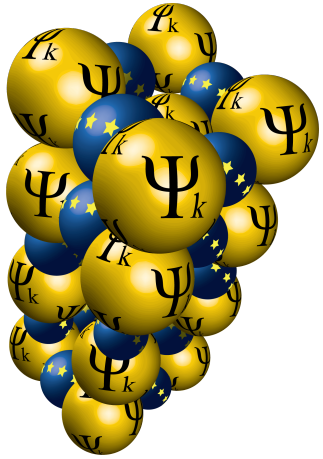
ICTP/Psi-k/CECAM School on Electron-Phonon Physics from First Principles

Trieste, 19-23 March 2018



Centre Européen de Calcul Atomique et Moléculaire

Lecture Wed.3

EPW in a nutshell

Samuel Poncé

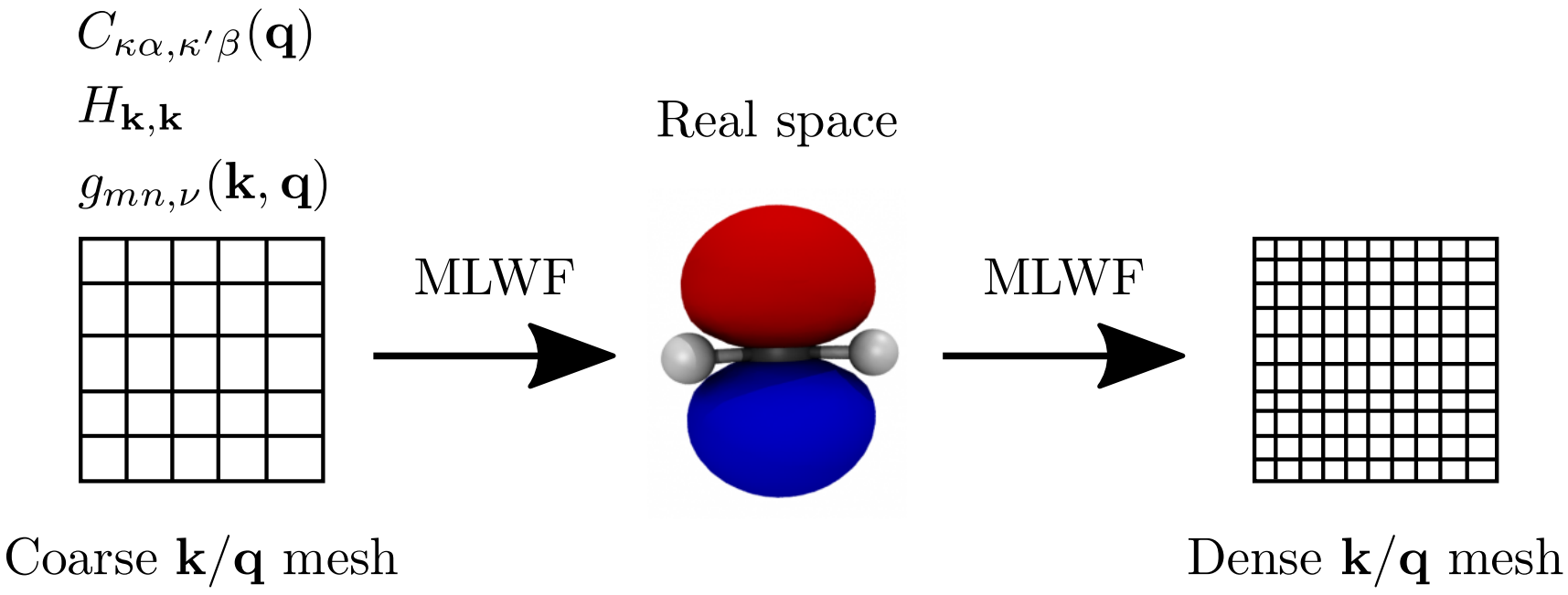
Department of Materials, University of Oxford

Lecture Summary

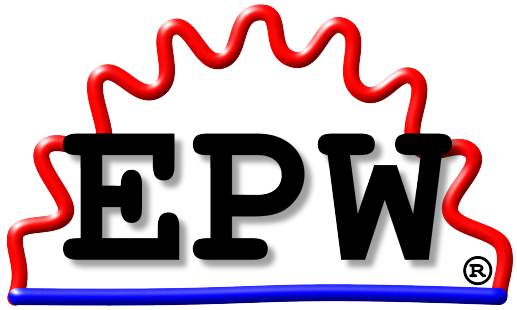
- Overview of the EPW software
- Structure of the code
- Technicalities and convergences parameters

What is EPW?

Electron-phonon Wannier (EPW) is a free GPL Fortran software, part of QE, that relies on MLWF to interpolate electron-phonon matrix elements.

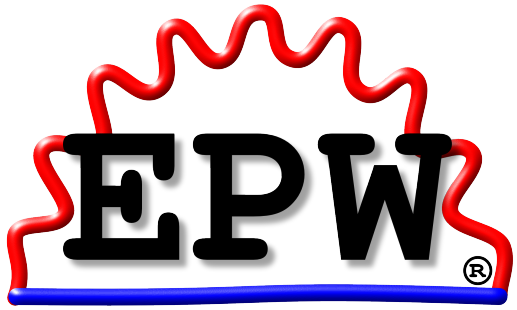


What can EPW do for you



- Interpolate el-ph matrix element on ultra dense momentum grids
- Compute electron and phonon linewidths, scattering rates & lifetimes
- Electron and phonon spectral functions
- Electron-phonon coupling strength
- Superconducting properties
- Transport properties (mobility & resistivity)

EPW specs

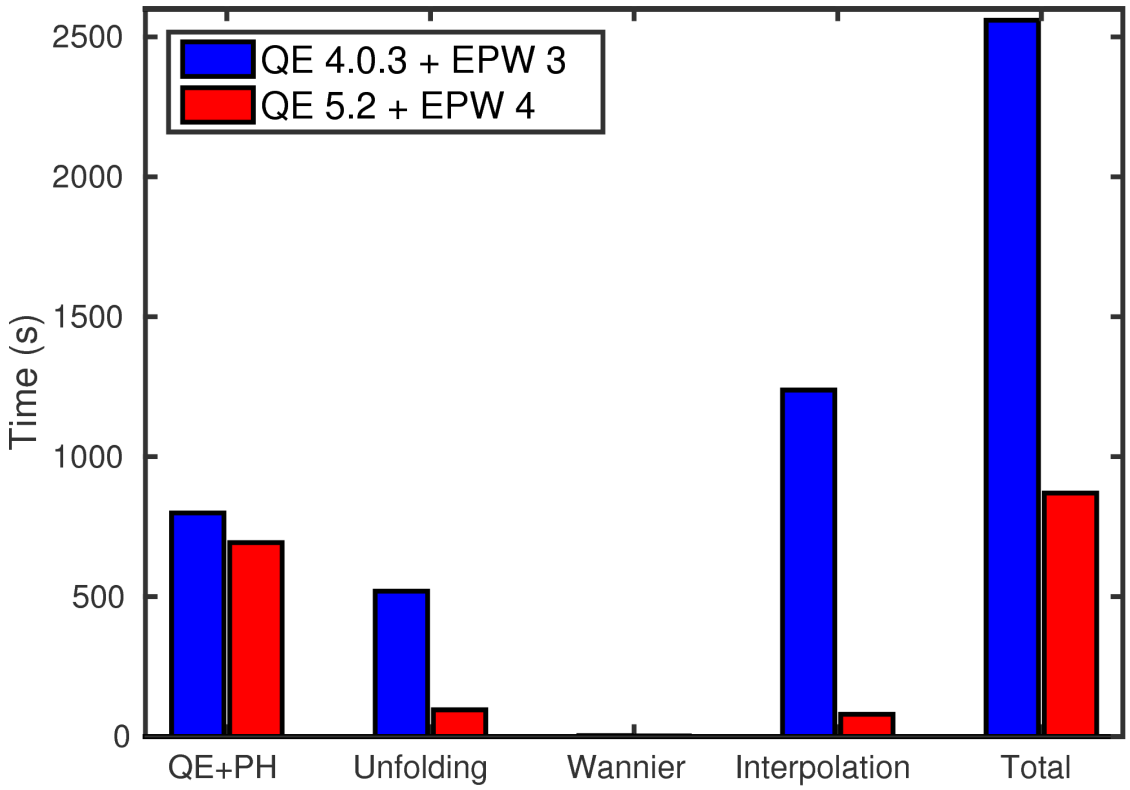


Features:

- Supports LDA/GGA functionals
- k/q-point parallelization (and bands)
- Supports spin-orbit coupling
- Supports time-reversal symmetry
- Polar divergence correctly interpolated
- Integrated into QE and rely on `Wannier90`
- MPI parallelization
- Has a test-farm for stability and portability of the code

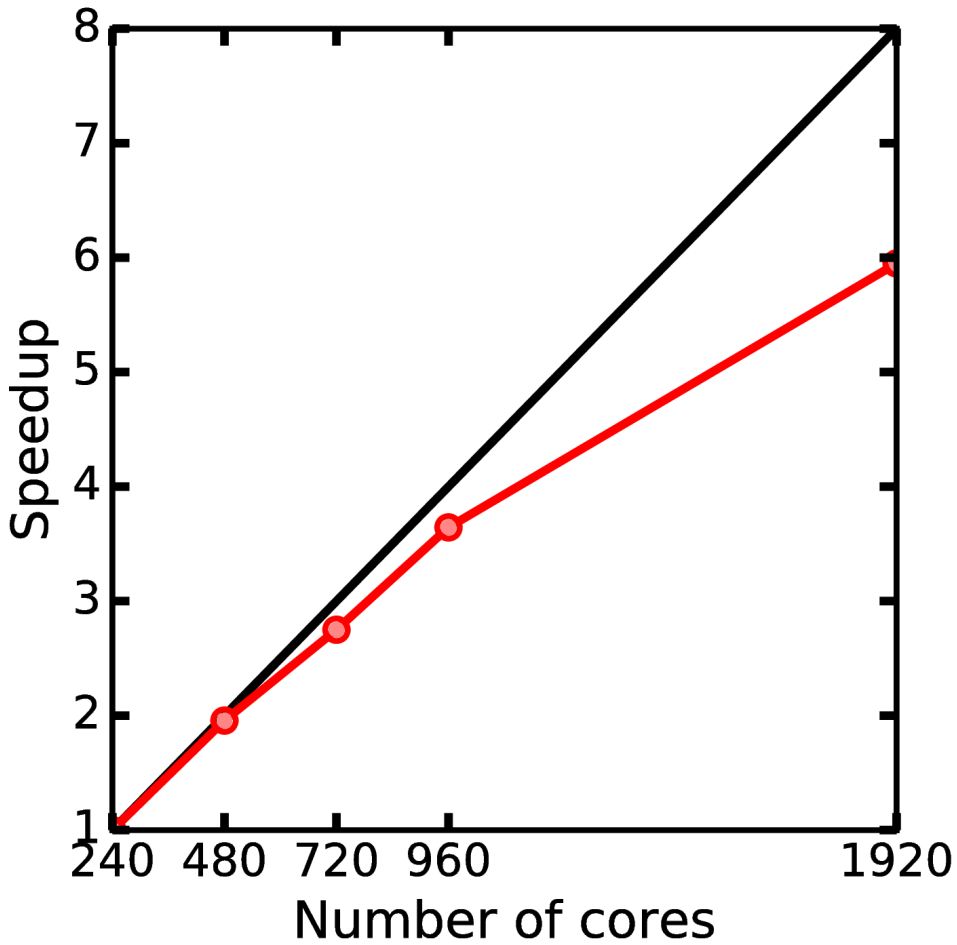
EPW speedup

Comparison of the time required to compute the electronic lifetime of SiC using EPW 3 and EPW 4, run on one processor.



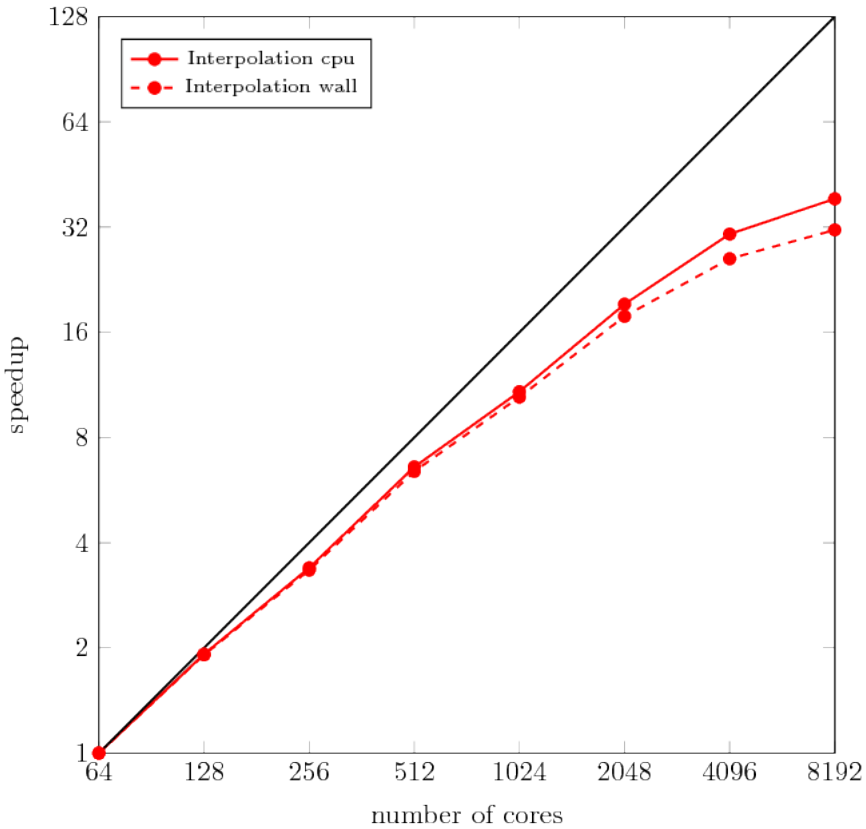
EPW scaling

Scalability of the interpolation part on ARCHER Cray XC30 for the polar w-GaN. The parallelization is done over k-points using MPI.



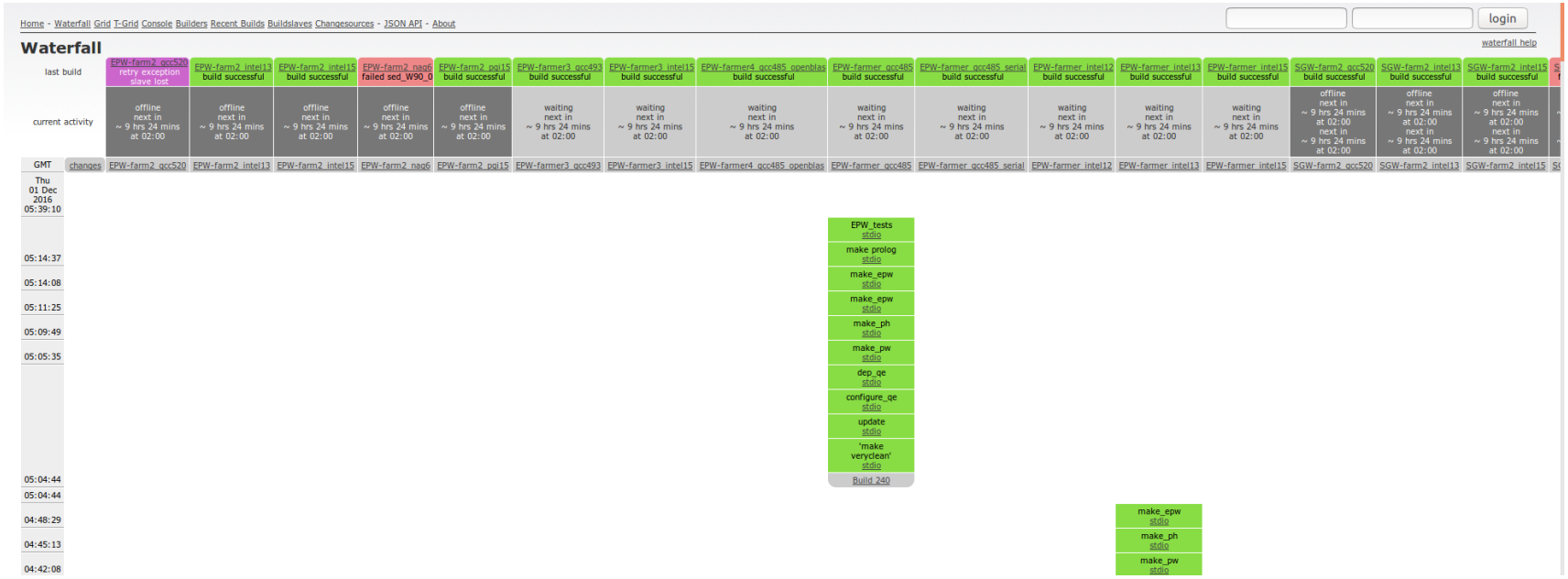
EPW scaling

Strong scaling of the interpolation part of EPW on CSD3 Xeon Phi for the polar SiC. The parallelization is done over k-points using MPI. The absolute time for the calculation was 6h01 at 64 cores and 9 min at 8192 cores.



Buildbot test-farm

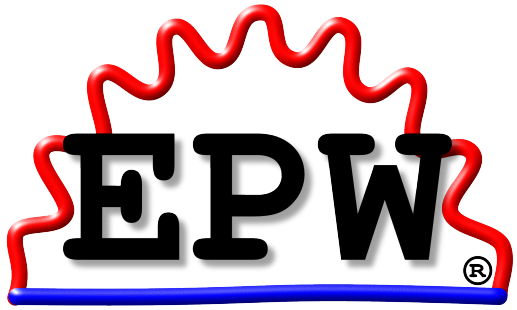
Buildbot is a continuous-integration testing software with automation of complex build systems, application deployment, and management of sophisticated software-release processes.



EPW specs

Limitations

- Norm-conserving psp only
- No magnetization
- No G-vector parallelization
- Nothing beyond LDA/GGA
- No +U (also not in `ph.x`)



Structure of the code

```
sponce@mauve:~/program/q-e/EPW$ ls
bin   epw.md   examples   Ford   License
   Makefile   README   src   tests
```

Structure of the code

```
sponce@mauve:~/program/q-e/EPW$ ls
bin   epw.md   examples   Ford   License
   Makefile   README   src   tests
```

- `bin`: Contains the epw.x soft link to the EPW executable.
- `examples`: Contains examples from the tutorials on [Link]
- `Ford`: Automatic documentation [Link]
- `src`: Contains all the EPW source files
- `tests`: Deprecated → replaced by `q-e/test-suite/epw_*`

Structure of the code

The `epw.f90` file:

```
1 CALL epw_readin
2 CALL allocate_epwq
3 CALL epw_setup
4 CALL wann_run()
5 CALL elphon_shuffle_wrap()
6 CALL deallocate_epw
7 CALL close_epw()
```

Structure of the code

The `epw.f90` file:

```
CALL epw_readin
CALL allocate_epwq
CALL epw_setup
CALL wann_run() <--
CALL elphon_shuffle_wrap()
CALL deallocate_epw
CALL close_epw()
```

Restart option:

- `wannierize = .false.`

Structure of the code

The `wannierize.f90` file:

```
! write the short input file for the wannier90 code
CALL write_winfil
! run the wannier90 code to create MLWFs
CALL pw2wan90epw
--> CALL setup_nnkp
--> CALL ylm_expansion
--> CALL compute_amn_para
--> CALL compute_mmn_para
```

Files created:

```
prefix.win ! w90 input file
prefix.wout ! w90 output file
prefix.nnkp ! Contains initial projections and the nearest
    neighbours of each k-points to compute M_mn(k,b) matrix
    elements
prefix.ukk ! Contains rotation matrix U(k) for interpolation
```

(Lecture Tue.2)

Structure of the code

The epw.f90 file:

```
CALL epw_readin
CALL allocate_epwq
CALL epw_setup
CALL wann_run()
CALL elphon_shuffle_wrap() <--
CALL deallocate_epw
CALL close_epw()
```

Restart option:

- kmaps = .false.
- epbwrite = .false.
- epbread = .true.

Structure of the code

The `elphon_shuffle_wrap.f90` file:

```
1 ! compute coarse grid dipole matrix elements.
2 CALL compute_pmn_para
```

$\hat{v}_\alpha = i[\hat{H}, \hat{r}_\alpha]$ with matrix elements:

$$v_{mn\mathbf{k}\mathbf{k}'\alpha} = \langle \psi_{m\mathbf{k}'} | \hat{v}_\alpha | \psi_{n\mathbf{k}} \rangle = \langle \psi_{m\mathbf{k}'} | \hat{p}_\alpha + i[\hat{V}_{\mathrm{NL}}, \hat{r}_\alpha] | \psi_{n\mathbf{k}} \rangle,$$

where $\hat{p}_\alpha = -i\nabla_\alpha$ is the momentum operator. In the *local approximation* we neglect $\hat{V}_{\mathrm{NL}}$:

$$\tilde{v}_{mn\mathbf{k}} = \mathbf{k}\delta_{mn} + \sum_{\mathbf{G}} c_{m\mathbf{k}}(\mathbf{G})^* c_{n\mathbf{k}}(\mathbf{G})\mathbf{G}.$$

Structure of the code

The elphon_shuffle_wrap.f90 file:

```
! compute coarse grid dipole matrix elements.
CALL compute_pmn_para
CALL createkmap_pw2(xk_all,nkstot, xq0)
CALL createkmap ( xq )
```

Files created:

```
prefix.kmap  ! Store the index of k+q on the coarse k-grid.
prefix.kgmap ! G-vectors needed to fold the k+q grid into the
     k grid
```

Structure of the code

The elphon_shuffle_wrap.f90 file:

```
1  ! compute coarse grid dipole matrix elements.
2  CALL compute_pmn_para
3  CALL createkmap_pw2(xk_all,nkstot, xq0)
4  CALL createkmap ( xq )
5  ! Find crystal symmetry
6  CALL find_sym ( nat, tau, ityp, .false., m_loc )
7  ! Find symmetry for the specific q-point
8  CALL star_q2(xq, at, bg, nsym, s, invs, nq, sxq, isq, imq, .
       true., sym_smallq )
9  CALL elphon_shuffle ( iq_irr, nqc_irr, nqc, gmapsym, eigv,
       isym, xq0, .false. )
10 CALL rotate_epmat ( cz1, cz2, xq, nqc, lwin, lwinq )
```

Unfolding from the IBZ to full BZ

$$
\begin{aligned}
g_{mn,\nu}(\mathbf{k},\mathbf{q}) &= \frac{1}{\sqrt{2\omega_{\mathbf{q}\nu}}} \langle \psi_{m\mathbf{k}+\mathbf{q}}(\mathbf{r}) \big| \partial_{\mathbf{q}\nu} V^{\mathsf{scf}}(\mathbf{r}) \big| \psi_{n\mathbf{k}}(\mathbf{r}) \rangle \\
&= \frac{1}{\sqrt{2\omega_{\mathbf{q}\nu}}} \Big[\big\langle \psi_{m\mathbf{k}+\mathbf{q}}(\mathbf{r}) \big| \partial_{\mathbf{q}\nu} V^{\mathsf{el}}(\mathbf{r}) \big| \psi_{n\mathbf{k}}(\mathbf{r}) \big\rangle \\
&+ \sum_{\mathbf{G}\mathbf{G}'} \big\langle \psi_{m\mathbf{k}+\mathbf{q}}(\mathbf{r}) \big| \mathbf{k}+\mathbf{q}+\mathbf{G}(\mathbf{r}) \big\rangle \\
&\times \big\langle \mathbf{k}+\mathbf{q}+\mathbf{G}(\mathbf{r}) \big| \partial_{\mathbf{q}\nu} V^{\mathsf{ion}}(\mathbf{r}) \big| \mathbf{k}+\mathbf{G}'(\mathbf{r}) \big\rangle \big\langle \mathbf{k}+\mathbf{G}'(\mathbf{r}) \big| \psi_{n\mathbf{k}}(\mathbf{r}) \big\rangle \Big]
\end{aligned}
$$

Unfolding from the IBZ to full BZ

$$g_{mn,\nu}(\mathbf{k},\mathcal{S}\mathbf{q}) = \frac{1}{\sqrt{2\omega_{\mathbf{q}\nu}}}\Big[\big\langle\psi_{m\mathbf{k}+\mathcal{S}\mathbf{q}}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big|\partial_{\mathbf{q}\nu}V^{\mathsf{el}}(\mathbf{r})\big|\psi_{n\mathbf{k}}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big\rangle$$
$$+\sum_{\mathbf{G}\mathbf{G}'}\big\langle\psi_{m\mathbf{k}+\mathcal{S}\mathbf{q}}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big|\mathbf{k}+\mathcal{S}\mathbf{q}+\mathbf{G}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big\rangle$$
$$\times\big\langle\mathcal{S}^{-1}\mathbf{k}+\mathbf{q}+\mathbf{G}(\mathbf{r})\big|\partial_{\mathbf{q}\nu}V^{\mathsf{ion}}(\mathbf{r})\big|\mathcal{S}^{-1}\mathbf{k}+\mathbf{G}'(\mathbf{r})\big\rangle$$
$$\times\big\langle\mathbf{k}+\mathbf{G}'(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big|\psi_{n\mathbf{k}}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big\rangle\Big]$$

Unfolding from the IBZ to full BZ

$$g_{mn,\nu}(\mathbf{k},\mathcal{S}\mathbf{q}) = \frac{1}{\sqrt{2\omega_{\mathbf{q}\nu}}}\Big[\big\langle\psi_{m\mathbf{k}+\mathcal{S}\mathbf{q}}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big|\partial_{\mathbf{q}\nu}V^{\text{el}}(\mathbf{r})\big|\psi_{n\mathbf{k}}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big\rangle$$
$$+\sum_{\mathbf{G}\mathbf{G}'}\big\langle\psi_{m\mathbf{k}+\mathcal{S}\mathbf{q}}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big|\mathbf{k}+\mathcal{S}\mathbf{q}+\mathbf{G}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big\rangle$$
$$\times\big\langle\mathcal{S}^{-1}\mathbf{k}+\mathbf{q}+\mathbf{G}(\mathbf{r})\big|\partial_{\mathbf{q}\nu}V^{\text{ion}}(\mathbf{r})\big|\mathcal{S}^{-1}\mathbf{k}+\mathbf{G}'(\mathbf{r})\big\rangle$$
$$\times\big\langle\mathbf{k}+\mathbf{G}'(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big|\psi_{n\mathbf{k}}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big\rangle\Big]$$

$$g_{mn,\nu}(\mathbf{k},-\mathcal{S}\mathbf{q}) = \frac{1}{\sqrt{2\omega_{\mathbf{q}\nu}}}\Big[\big\langle\psi_{m\mathbf{k}-\mathcal{S}\mathbf{q}}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big|\Big(\partial_{\mathbf{q}\nu}V^{\text{el}}(\mathbf{r})\Big)^{*}\big|\psi_{n\mathbf{k}}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big\rangle$$
$$+\sum_{\mathbf{G}\mathbf{G}'}\big\langle\psi_{m\mathbf{k}-\mathcal{S}\mathbf{q}}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big|\mathbf{k}-\mathcal{S}\mathbf{q}+\mathbf{G}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big\rangle$$
$$\times\big\langle\mathcal{S}^{-1}\mathbf{k}-\mathbf{q}+\mathbf{G}(\mathbf{r})\big|\partial_{-\mathbf{q}\nu}V^{\text{ion}}(\mathbf{r})\big|\mathcal{S}^{-1}\mathbf{k}+\mathbf{G}'(\mathbf{r})\big\rangle$$
$$\times\big\langle\mathbf{k}+\mathbf{G}'(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big|\psi_{n\mathbf{k}}(\{\mathcal{S}|\mathbf{v}\}\mathbf{r})\big\rangle\Big]$$

Structure of the code

The elphon_shuffle_wrap.f90 file:

```
! compute coarse grid dipole matrix elements.
CALL compute_pmn_para
CALL createkmap_pw2(xk_all,nkstot, xq0)
CALL createkmap ( xq )
! Find crystal symmetry
CALL find_sym ( nat, tau, ityp, .false., m_loc )
! Find symmetry for the specific q-point
CALL star_q2(xq, at, bg, nsym, s, invs, nq, sxq, isq, imq, .
    true., sym_smallq )
CALL elphon_shuffle ( iq_irr, nqc_irr, nqc, gmapsym, eigv,
    isym, xq0, .false. )
CALL rotate_epmat ( cz1, cz2, xq, nqc, lwin, lwinq )
CALL ephwann_shuffle ( nqc, xqc ) <--
```

Files created:

```
prefix.epbX  ! Contains unfolded matrix elements
```

Structure of the code

The epw.f90 file:

```
1 CALL epw_readin
2 CALL allocate_epwq
3 CALL epw_setup
4 CALL wann_run()
5 CALL elphon_shuffle_wrap()
6 --> CALL ephwann_shuffle ( nqc, xqc ) <--
7 CALL deallocate_epw
8 CALL close_epw()
```

Restart option:

- epwwrite = .false.
- epwread = .true.

Structure of the code

The ephwann_shuffle.f90 file:

```
CALL loadumat                ! Rotation matrix
CALL hambloch2wan(...)       ! Hamiltonian
CALL dmebloch2wan(...)       ! Dipole
CALL dynbloch2wan(...)       ! Dynamical matrix
CALL ephbloch2wane(...) ! Bloch el and Bloch ph -> Wannier el
     and Bloch ph
CALL ephbloch2wanp(...) ! Wannier el and Bloch ph -> Wannier
    el and Wannier ph
```

Files created (used for restart):

```
prefix.epmatwe1  ! Deleted when the run is finished
prefix.epmatwp1  ! Contains matrix element in real space
crystal.fmt      ! Formatted crystal information
dmedata.fmt      ! Formatted dipole (for velocities)
epwdata.fmt      ! Formatted eigenenergies, zstar, dielectric
    function, real-space Ham+dyn
```

From coarse Bloch space to localized real space

The Hamiltonian `CALL hambloch2wan(...)`

$$H_{mn}(\mathbf{R_p} - \mathbf{R'_p}) = \frac{1}{N_p} \sum_{m'n'\mathbf{k}} e^{-i\mathbf{k}\cdot(\mathbf{R}'_p - \mathbf{R}_p)} U^{\dagger}_{mm',\mathbf{k}+\mathbf{q}} H_{m'n'\mathbf{k}} U_{nn',\mathbf{k}}$$

The Dynamical matrix `CALL dynbloch2wan(...)`

$$D_{mn\kappa\alpha}(\mathbf{R_{p'}} - \mathbf{R'_{p'}}) = \frac{1}{N_{p'}} \sum_{\mathbf{q}\nu} e^{-i\mathbf{q}\cdot(\mathbf{R}'_{p'} - \mathbf{R}_{p'})} e^{*}_{\kappa\alpha,\nu}(\mathbf{q}) D_{\nu\kappa\alpha}(\mathbf{q}) e_{\kappa\alpha,\nu}(\mathbf{q})$$

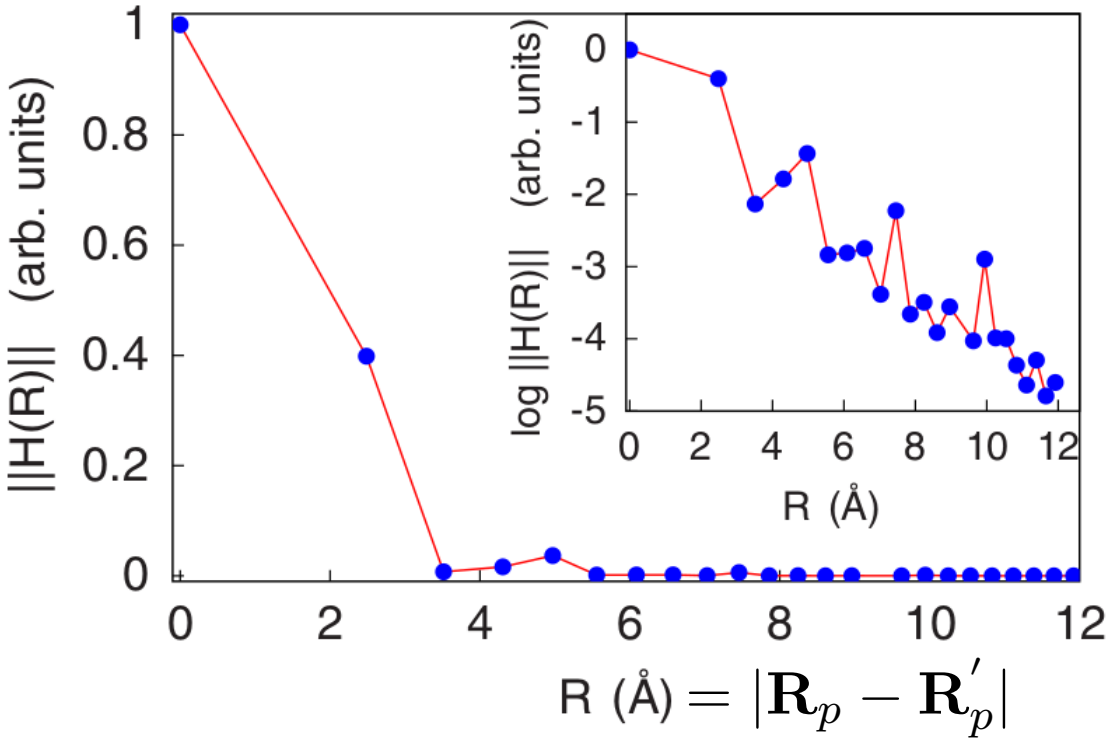
N_p, $N_{p'}$ are the number of BvK supercells for el and ph.

F. Giustino *et al.*, Phys. Rev. B **76**, 165108 (2007)

From coarse Bloch space to localized real space

Files created:

```
decay.H          ! Real-space decay of the Hamiltonian
decay.dynmat     ! Real-space decay of the dynamical matrix
```

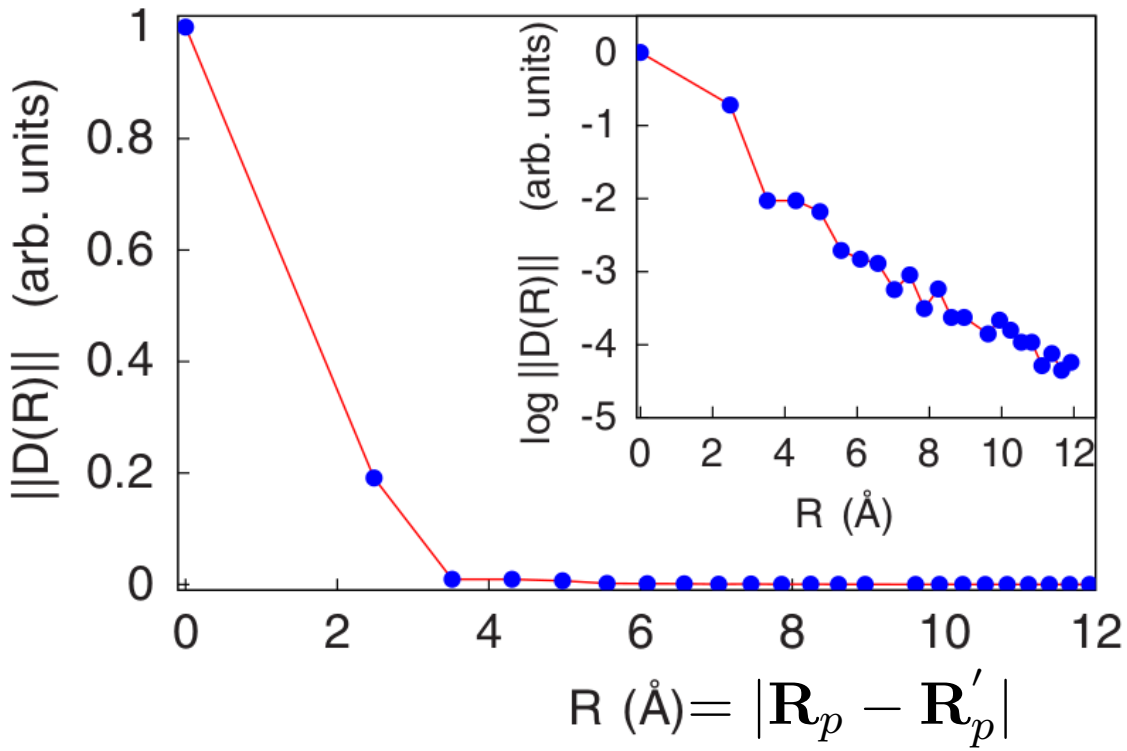


F. Giustino *et al.*, Phys. Rev. B **76**, 165108 (2007)

From coarse Bloch space to localized real space

Files created:

```
decay.H         ! Real-space decay of the Hamiltonian
decay.dynmat    ! Real-space decay of the dynamical matrix
```



F. Giustino *et al.*, Phys. Rev. B **76**, 165108 (2007)

From coarse Bloch space to localized real space

The electron-phonon matrix element

$$g_{nm\kappa\alpha}(\mathbf{R}_p, \mathbf{R}_{p'}) = \frac{1}{N_p N_{p'}} \sum_{\mathbf{k},\mathbf{q}} \sqrt{\frac{2M_\kappa \omega_{\mathbf{q}\nu}}{\hbar}} e^{-i(\mathbf{k}\cdot\mathbf{R}_p + \mathbf{q}\cdot\mathbf{R}_{p'})} \sum_{m'n'\nu} e^*_{\kappa\alpha,\nu}(\mathbf{q}) U^\dagger_{mm',\mathbf{k}+\mathbf{q}} g_{m'n'\nu}(\mathbf{k},\mathbf{q}) U_{n'n\mathbf{k}}$$

on the coarse grids (`prefix.epbX` files)

N_p, $N_{p'}$ are the number of BvK supercells for el and ph.

F. Giustino, Rev. Mod. Phys. **89**, 1 (2017)

From coarse Bloch space to localized real space

$$g_{nm\kappa\alpha}(\mathbf{R}_p, \mathbf{R}_{p'}) = \frac{1}{N_p N_{p'}} \sum_{\mathbf{k},\mathbf{q}} \sqrt{\frac{2M_\kappa \omega_{\mathbf{q}\nu}}{\hbar}} e^{-i(\mathbf{k}\cdot\mathbf{R}_p + \mathbf{q}\cdot\mathbf{R}_{p'})} \sum_{m'n'\nu} e^*_{\kappa\alpha,\nu}(\mathbf{q}) U^\dagger_{mm',\mathbf{k}+\mathbf{q}} g_{m'n'\nu}(\mathbf{k},\mathbf{q}) U_{n'n\mathbf{k}}$$

`CALL ephbloch2wane(...)`

$$g_{nm\nu}(\mathbf{R}_p, \mathbf{q}) = \frac{1}{N_p} \sum_{\mathbf{k}} e^{-i\mathbf{k}\cdot\mathbf{R}_p} \sum_{m'n'} U^\dagger_{mm',\mathbf{k}+\mathbf{q}} g_{m'n'\nu}(\mathbf{k},\mathbf{q}) U_{n'n\mathbf{k}}$$

`CALL ephbloch2wanp(...)`

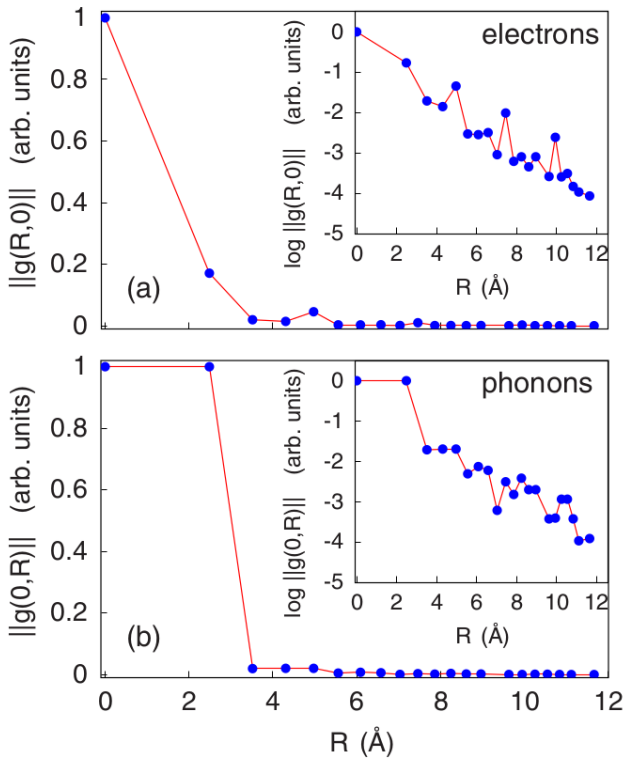
$$g_{nm\kappa\alpha}(\mathbf{R}_p, \mathbf{R}_{p'}) = \frac{1}{N_{p'}} \sum_{\mathbf{q}} \sqrt{\frac{2M_\kappa \omega_{\mathbf{q}\nu}}{\hbar}} e^{-i\mathbf{q}\cdot\mathbf{R}_{p'}} \sum_{\nu} e^*_{\kappa\alpha,\nu}(\mathbf{q}) g_{mn\nu}(\mathbf{R}_p, \mathbf{q})$$

F. Giustino, Rev. Mod. Phys. **89**, 1 (2017)

From coarse Bloch space to localized real space

Files created:

```
decay.epwane       ! Real-space decay of the electronic part
    of the el-ph
decay.epmat_wanep  ! Real-space decay of the el-ph
```



F. Giustino *et al.*, Phys. Rev. B **76**, 165108 (2007)

Structure of the code

The ephwann_shuffle.f90 file:

```
1  CALL dynwan2bloch(...)   ! Dynamical matrix
2  CALL ephwan2blochp(...) ! Wannier el and Wannier ph ->
       Wannier el and Bloch ph
3  CALL hamwan2bloch(...)   ! Hamiltonian
4  CALL dmewan2bloch(...)   ! Dipole
5  CALL ephwan2bloch(...)   ! Wannier el and Bloch ph -> Bloch el
       and Bloch ph
6  IF (prtgkk     ) CALL print_gkk( iq )
7  IF (phonselfen ) CALL selfen_phon_q( iq )
8  IF (elecselfen ) CALL selfen_elec_q( iq, first_cycle )
9  IF (plselfen   ) CALL selfen_pl_q( iq )
10 IF (nest_fn    ) CALL nesting_fn_q( iq )
11 IF (specfun_el ) CALL spectral_func_q( iq )
12 IF (specfun_ph ) CALL spectral_func_ph( iq )
13 IF (specfun_pl ) CALL spectral_func_pl_q( iq )
14 IF (scattering)  CALL scattering_rate_q( ... )
15 IF (.not. iterative_bte) CALL transport_coeffs (ef0,efcb)
```

Fan-Migdal electron self-energy

Fan-Migdal self-energy using Kohn-Sham states and DFPT phonons

$$\Sigma^{\mathrm{FM}}_{n\mathbf{k}}(\omega, T) = \frac{1}{\hbar}\sum_{m\nu}\int \frac{d\mathbf{q}}{\Omega_{\mathrm{BZ}}} |g_{mn\nu}(\mathbf{k},\mathbf{q})|^2$$
$$\times\left[\frac{1 - f_{m\mathbf{k}+\mathbf{q}}(T) + n_{\mathbf{q}\nu}(T)}{\omega - \varepsilon_{m\mathbf{k}+\mathbf{q}}/\hbar - \omega_{\mathbf{q}\nu} + i\eta} + \frac{f_{m\mathbf{k}+\mathbf{q}}(T) + n_{\mathbf{q}\nu}(T)}{\omega - \varepsilon_{m\mathbf{k}+\mathbf{q}}/\hbar + \omega_{\mathbf{q}\nu} + i\eta}\right]$$

(Lecture Wed.1)

Fan-Migdal electron self-energy

Fan-Migdal self-energy using Kohn-Sham states and DFPT phonons

Summation over all phonon branches and wavevectors

$$\Sigma^{\mathrm{FM}}_{n\mathbf{k}}(\omega, T) = \frac{1}{\hbar}\sum_{m\nu}\int \frac{d\mathbf{q}}{\Omega_{\mathrm{BZ}}} |g_{mn\nu}(\mathbf{k},\mathbf{q})|^2 \times \left[\frac{1 - f_{m\mathbf{k}+\mathbf{q}}(T) + n_{\mathbf{q}\nu}(T)}{\omega - \varepsilon_{m\mathbf{k}+\mathbf{q}}/\hbar - \omega_{\mathbf{q}\nu} + i\eta} + \frac{f_{m\mathbf{k}+\mathbf{q}}(T) + n_{\mathbf{q}\nu}(T)}{\omega - \varepsilon_{m\mathbf{k}+\mathbf{q}}/\hbar + \omega_{\mathbf{q}\nu} + i\eta}\right]$$

Extension to finite temperature

Dynamical structure on the scale of the phonon energy

(Lecture Wed.1)

Fan-Migdal phonon self-energy

$$\Pi_{\mathbf{q}\nu}(\omega, T) = 2 \sum_{mn} \int_{\mathrm{BZ}} \frac{d\mathbf{k}}{\Omega_{\mathrm{BZ}}} |g_{mn,\nu}(\mathbf{k}, \mathbf{q})|^2 \frac{f_{n\mathbf{k}}(T) - f_{m\mathbf{k}+\mathbf{q}}(T)}{\varepsilon_{n\mathbf{k}} - \varepsilon_{m\mathbf{k}+\mathbf{q}} - \omega - i\delta}$$

Structure of the code

Input variables:

```
elecselfen    = .true.          scattering  = .true.
nest_fn       = .true.          scattering_serta = .true.
phonselfen    = .true.          int_mob     = .true.
a2f           = .true.          carrier     = .true.
                                ncarrier    = 1E13
specfun_el    = .true.          nstemp      =   3
wmin_specfun  = -4              tempsmin    = 100
wmax_specfun  = 1               tempsmax    = 500
nw_specfun    = 20
```

Files created:

```
linewidth.phself    ! Im electron self-energy = linewidths
lambda.phself       ! Lambda phonon self-energy
linewidth.elself    ! Im electron self-energy = linewidths
specfun.elself      ! Electron spectral function
specfun_sup.elself  ! Supporting file for spectral function
specfun.phon        ! Phonon spectral function
specfun_sup.phon    ! Supporting file for spectral function
```

Polar divergence

Input variables:

```
lpolar       = .true.
```

In polar materials, $g_{mn,\nu}(\mathbf{k},\mathbf{q})$diverge as $1/|\mathbf{q}|$ for $|\mathbf{q}| \to \mathbf{0}$
Split the electron-phonon matrix elements into a short- ($\mathcal{S}$) and a long-range ($\mathcal{L}$) contribution:

$$g_{mn,\nu}(\mathbf{k},\mathbf{q}) = g^{\mathcal{S}}_{mn,\nu}(\mathbf{k},\mathbf{q}) + g^{\mathcal{L}}_{mn,\nu}(\mathbf{k},\mathbf{q})$$

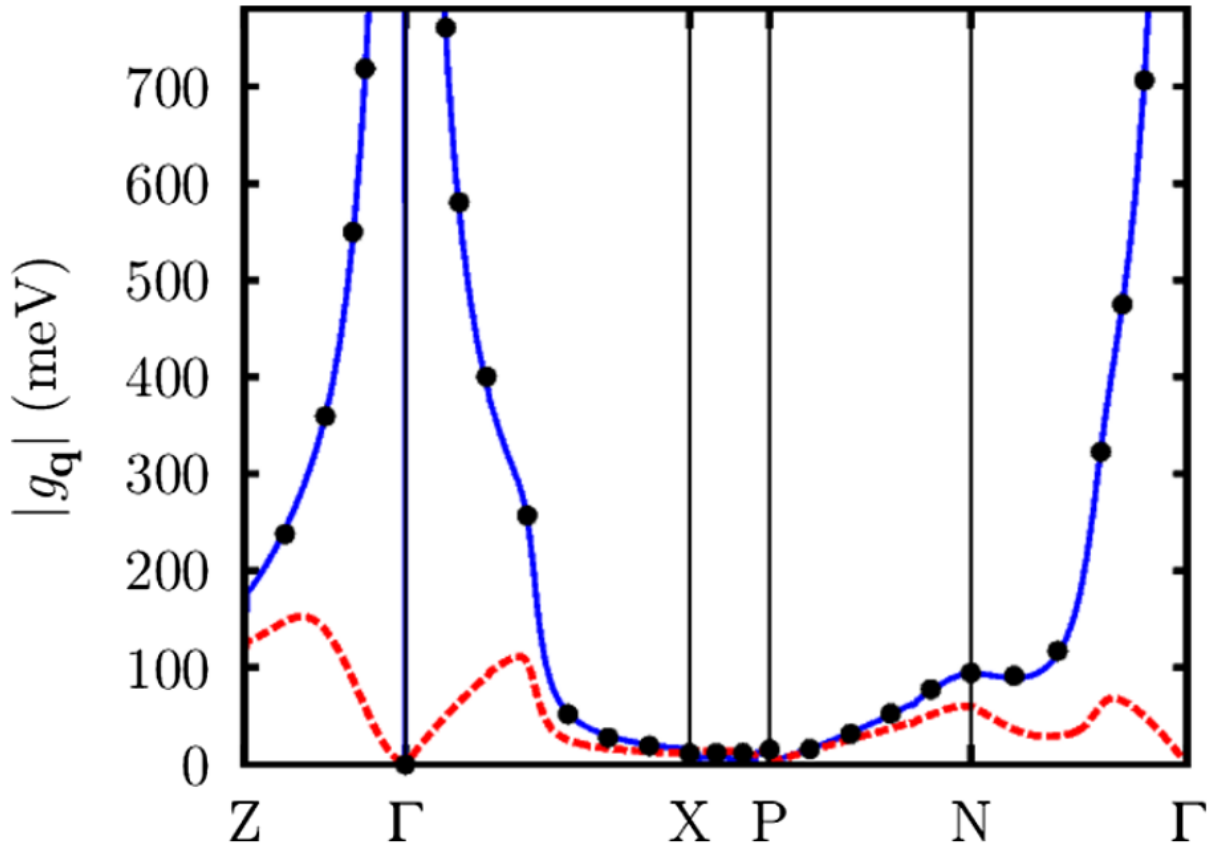
$$g^{\mathcal{L}}_{mn,\nu}(\mathbf{k},\mathbf{q}) = i \sum_{\kappa} \left(\frac{\hbar}{2M_\kappa \omega_{\mathbf{q}\nu}} \right)^{1/2} \times \sum_{\mathbf{G}\neq -\mathbf{q}} \frac{(\mathbf{q}+\mathbf{G})\cdot \mathbf{Z}^*{}_\kappa \cdot \mathbf{e}_{\kappa,\nu}(\mathbf{q})}{(\mathbf{q}+\mathbf{G})\cdot \boldsymbol{\varepsilon}^\infty \cdot (\mathbf{q}+\mathbf{G})} \left\langle \psi_{m\mathbf{k}+\mathbf{q}} \middle| e^{i(\mathbf{q}+\mathbf{G})\cdot(\mathbf{r}-\tau_\kappa)} \middle| \psi_{n\mathbf{k}} \right\rangle$$

S. Poncé *et al.*, J. Chem. Phys. **143**, 102813 (2015)

C. Verdi and F. Giustino, Phys. Rev. Lett. **115**, 176401 (2015)

Polar divergence

Wannier interpolation of the electron-phonon matrix elements for anatase TiO_2



C. Verdi and F. Giustino, Phys. Rev. Lett. **115**, 176401 (2015)

Crystal acoustic sum rule

Input variables:

```
lifc         = .true.
asr_typ      = 'simple'    ! freq. at Gamma for 3 acoustic modes = 0
asr_typ      = 'crystal'   ! Lagrangian approach
```

Requires to run `q2r.x` and to copy the `prefix.fc` file into `save/ifc.q2r` before `EPW` calculation

N. Mingo *et al.*, Phys. Rev. B **77**, 033418 (2008)

N. Mounet, PhD thesis, MIT (2005) [link]

Miscellaneous

Input variables:

```
fsthick      = 1.0 ! eV   ! Windows around the Fermi level.
    Some properties requires large values
eptemp       = 1   ! K    ! Temperature at which the
    calculation is performed. Not to be confused with
    tempsmin\max which is used for mobility and supersedes
    eptemp
degaussw     = 0.01 ! eV  ! Gaussian broadening for the delta
    functions. Should be small
degaussq     = 0.05 ! meV ! Gaussian broadening for the \
    lambda and a2f phonons
```

Convergence must be made for:

```
nk1,nk2,nk3
nq1,nq2,nq3
nkf1,nkf2,nkf3 or filkf
nqf1,nqf2,nqf3 or filqf
ecut
fsthick
```

References

- F. Giustino *et al.*, Phys. Rev. B **76**, 165108 (2007) [link]
- C. Verdi *et al.*, Phys. Rev. Lett. **115**, 176401 (2015) [link]
- S. Poncé *et al.*, Comput. Phys. Commun. **209**, 116 (2016) [link]
- F. Giustino, Rev. Mod. Phys. **89**, 1 (2017) [link]

More info

http://epw.org.uk

http://epwforum.uk

https://gitlab.com/QEF/q-e

Supplemental Slides