Efficient materials modelling on HPC
with QUANTUM ESPRESSO, Yambo and BigDFT

Hands-on session – Day 2

PHONONS FOR HPC AND GPUs

Laura Bellentani and Iurii Timrov

l.bellentani@cineca.it iurii.timrov@epfl.ch

INTERATOMIC FORCE CONSTANTS

Let us consider a unit cell with *Nat* atoms:

$s = 1 \ldots N_{at}$	index of an atom in the unit cell
$\alpha = x, y, z$	is the cartesian index
$\mathbf{R}$	is the point in the Bravais lattice, identifying the position of a given cell
$N_{\mathbf{R}}$	is the number of unit cells in the crystal
$\mathbf{u}_{s\alpha}(\mathbf{R}) \\ \downarrow \\ 3 \times N_{at}$	is the α component of the displacement of the s-th atom

Matrix of Interatomic Force Constants :

$$\boxed{C_{s\alpha,s'\beta}(\mathbf{R},\mathbf{R}') = C_{s\alpha,s'\beta}(\mathbf{R}-\mathbf{R}') = \frac{\partial^2 E_{tot}}{\partial \mathbf{u}_{s\alpha}(\mathbf{R})\partial \mathbf{u}_{s'\beta}(\mathbf{R}')}}$$

Main equations

SECULAR EQUATION

Normal mode frequencies, ω , and eigenvectors, $\tilde{\mathbf{u}}_{s\alpha}$ are determined by the secular equation:

$$\sum_{s',\beta} \tilde{D}_{s\alpha,s'\beta}(\mathbf{q})\, \tilde{\mathbf{u}}_{s'\beta}(\mathbf{q}) = \omega^2_{\mathbf{q}}\, \tilde{\mathbf{u}}_{s\alpha}(\mathbf{q})$$

where

Interatomic Force Constants (IFC)

$$\tilde{D}_{s\alpha,s'\beta}(\mathbf{q}) = \frac{1}{\sqrt{M_s M_{s'}}} \sum_{\mathbf{R},\mathbf{R}'} \boxed{\frac{\partial^2 E_{tot}}{\partial \mathbf{u}_{s\alpha}(\mathbf{R})\partial \mathbf{u}_{s'\beta}(\mathbf{R}')}} e^{i\mathbf{q}(\mathbf{R}'-\mathbf{R})}$$

is the ***dynamical matrix***.

<u>Diagonalization of the dynamical matrix gives phonon modes at **q**.</u>

Main equations

DENSITY FUNCTIONAL PERTURBATION THEORY

Sternheimer equation (solve_linter):

$$\big(\underset{\overset{\uparrow}{\texttt{h\_psi}}}{H^{\mathbf{k}+\mathbf{q}}_{SCF}} + \alpha P^{\mathbf{k}+\mathbf{q}}_v - \epsilon^{\mathbf{k}}_v\big)|\Delta\psi^{\mathbf{k}+\mathbf{q}}_v\rangle = -\underset{\overset{\uparrow}{\texttt{orthogonalize}}}{P^{\mathbf{k}+\mathbf{q}}_c} \Delta v^{\mathbf{q}}_{SCF}(\mathbf{r})\underset{\overset{\uparrow}{\texttt{apply\_dpot}}}{|}\psi^{\mathbf{k}}_v\rangle$$

$$\Delta v^{\mathbf{q}}_{SCF}(\mathbf{r}) = \Delta v^{\mathbf{q}}(\mathbf{r}) + e^2 \int \frac{\boxed{\Delta n^{\mathbf{q}}(\mathbf{r}')}}{|\mathbf{r}-\mathbf{r}'|} e^{-i\mathbf{q}\cdot(\mathbf{r}-\mathbf{r}')} d\mathbf{r}' + \left.\frac{dv_{xc}(n)}{dn}\right|_{n=n(\mathbf{r})} \Delta n^{\mathbf{q}}(\mathbf{r}).$$

$$\underset{\texttt{incdrhoscf}}{\boxed{\Delta n^{\mathbf{q}}_v(\mathbf{r})}} = 4\sum_{\mathbf{k}v} u^{\mathbf{k}*}_v(\mathbf{r}) \Delta u^{\mathbf{k}+\mathbf{q}}_v(\mathbf{r})$$

Main equations

DENSITY FUNCTIONAL PERTURBATION THEORY

Sternheimer equation:

$$\begin{tikzcd}
(H^{\mathbf{k}+\mathbf{q}}_{SCF} + \alpha P_v^{\mathbf{k}+\mathbf{q}} - \epsilon_v^{\mathbf{k}}) \boxed{|\Delta\psi_v^{\mathbf{k}+\mathbf{q}}\rangle} = -P_c^{\mathbf{k}+\mathbf{q}} \Delta v^{\mathbf{q}}_{SCF}(\mathbf{r}) |\psi_v^{\mathbf{k}}\rangle \arrow[d, Rightarrow] \\
C_{s\alpha,s'\beta}(\mathbf{R},\mathbf{R}') \arrow[d, Rightarrow] \\
\tilde{D}_{s\alpha,s'\beta}(\mathbf{q}) \arrow[d, Rightarrow] \\
\textbf{PHONONS}
\end{tikzcd}$$

Rev. Mod. Phys. **73**, 515 (2001).

Phys. Rev. B **43**, 7231 (1991).

HPC implementation

CALCULATIONS AVAILABLE FROM ph.x

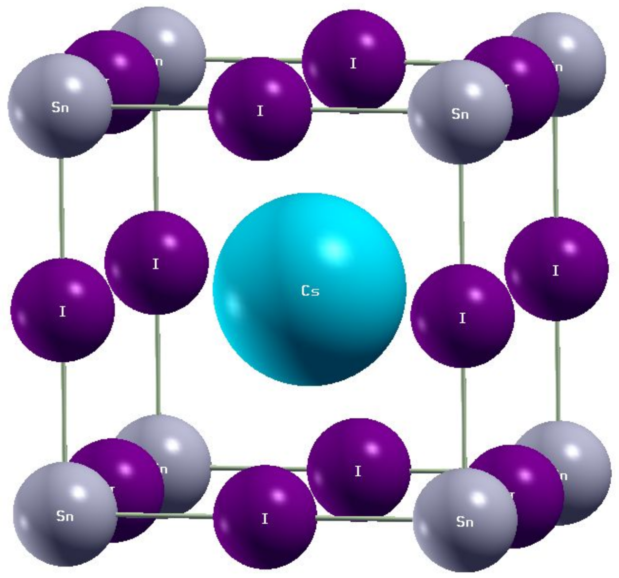
The phonon code works for a rather wide variety of systems and methods:

- ✓ Insulators (also polar insulators, with LO-TO splitting)
- ✓ Metals
- ✓ Magnetic systems at the scalar relativistic collinear level
- ✓ Spin-orbit coupling (fully relativistic approach)
- ✓ Electric field calculations: Born effective charges, dielectric tensor

Recent developments:

- ! Phonons for magnetic systems in the fully relativistic non-collinear approach
- ! Phonons within the DFT+U approach

EXERCISE ON VEGA : SIMULATION OF CnSnI3



- ✳ Experimentally metallic due to self-doping
- ✳ In DFT it is a semiconductor (polar material)
- ✳ 5 atoms in the primitive unit cell
- ✳ 3 x 5 = 15 phonon modes

EXERCISE ON VEGA : PHONON MODES AT GAMMA

Go to the directory with the input files:

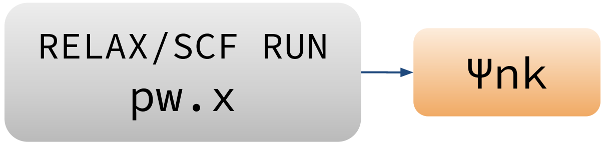
```
cd  ~/QE-2021/Day-2/example_ph/step*
```

In this directories you will find:

- `README.md` – File describing how to do the exercise
- `pw.CnSnI3.in` – Input file for the vc-relax/SCF calculation
- `ph.CnSnI3.in` – Input file for the phonon calculation at Γ
- `dyn.CnSnI3.in` – Input file to impose the acoustic sum rule
- submit.slurm – Script to submit the calculations
- `reference/` – Directory with the reference results

PWSCF SIMULATION, STEP 1

The Phonon workflow for modes at a single q point



The phonon workflow

PWSCF SIMULATION, STEP 1

DONE YESTERDAY!

1. `cd Day-2/exercise_ph/step1/`

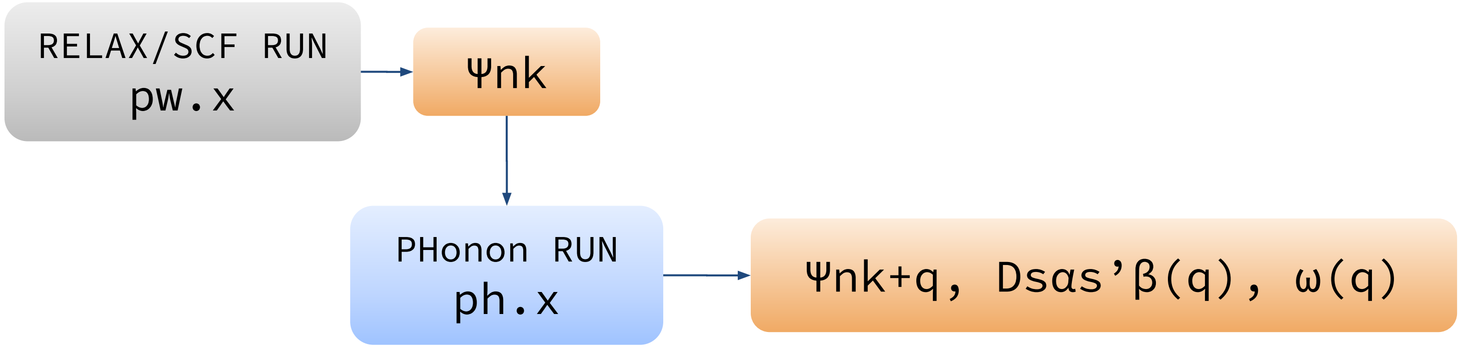
   Perform a vc-relax calculation for CnSnI3 using the `pw.x` *program.*

- Copy `../inputs/pw.CnSnI3.in` in the current folder and modify `&CONTROL` namelist to do a `vc-relax`
- Open `submit.slurm` and modify *npw* to use R&G on 4 MPIs : GPUs
- Submit the job file
- Copy the output directory `out/` in the folder of the next step

```
$ cat pw.CnSnI3.in
&CONTROL
        calculation = 'vc-relax'
        prefix = 'pwscf'
        outdir = './out'
/
&SYSTEM
   ecutwfc           = 80
   ecutrho           = 320
   occupations       = 'fixed'
   ntyp              = 3
   nat               = 5
   ibrav             = 0
/
&ELECTRONS
   conv_thr          = 1e-14
/
&IONS
/
&CELL
        press = 0
        press_conv_thr = 0.05
/

ATOMIC_SPECIES
Cs 132.90545196 Cs-nc-pbesol.upf
Sn 118.71 Sn-nc-pbesol.upf
I 126.90447 I-nc-pbesol.upf

K_POINTS automatic
8 8 8  1 1 1

CELL_PARAMETERS angstrom
6.1821206415142775  0.0000000000000000  0.0000000000000000
0.0000000000000000  6.1821206415142775  0.0000000000000000
0.0000000000000000  0.0000000000000000  6.1821206415142775

ATOMIC_POSITIONS angstrom
Cs    3.0910603207571383  3.0910603207571383  3.0910603207571383
Sn    0.0000000000000000  0.0000000000000000  0.0000000000000000
I     3.0910603207571383  0.0000000000000000  0.0000000000000000
I     0.0000000000000000  0.0000000000000000  3.0910603207571383
I     0.0000000000000000  3.0910603207571383  0.0000000000000000
```

PHONON CALCULATION, STEP 2

The Phonon workflow for modes at a single q point



The phonon workflow

PHONON CALCULATION, STEP 2

2. `cd Day-2/exercise_ph/step2/`

Perform a phonon calculation at Γ using the `ph.x` program.

- Copy `../inputs/ph.CnSnI3.in` in the current folder and modify the `&inputph` namelist ; add coordinates of the Gamma point

```
$ cat ph.CnSnI3.in
&inputph
        prefix = 'pwscf'
        fildyn = 'harmdyn_support'
        amass(1) = 132.90545196
        amass(2) = 118.71
        amass(3) = 126.90447
        tr2_ph = 1d-16
        outdir = './out'
/
0.0 0.0 0.0
```

The same prefix as in the `pw.x` calculation

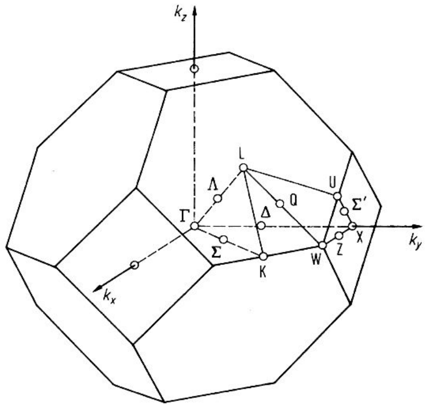
File storing the dynamical matrix

Atomic masses

Threshold for self consistency

Directory for temporary files

Coordinates of the q point **q** = 2*pi/a (0.0, 0.0, 0.0)



- Submit the job file `submit.slurm` to run on 1 MPI : GPU

The phonon workflow

DYNMAT MATRIX, STEP 2

- Check the number of k points

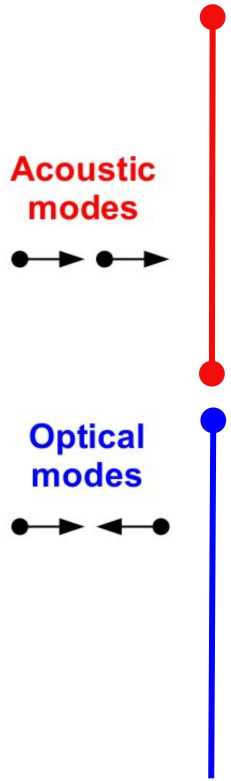
  `awk '/number of k/' ph.CnSnI3.out`

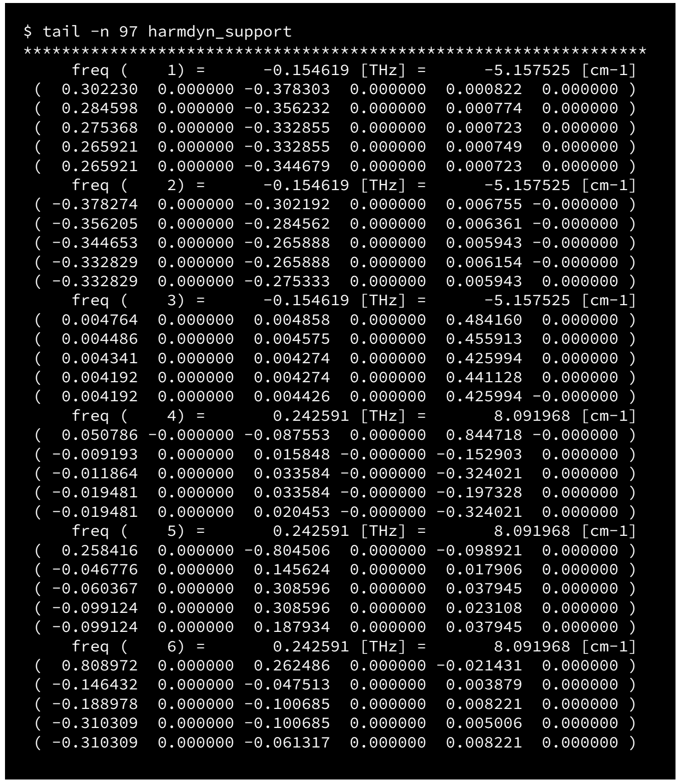
- Check the number of **irreducible representations**

  `awk '/irreducible/' ph.CnSnI3.out`

- Check the dynamical matrix in `harmdyn_`

  `tail -n 97 harmdyn_support`

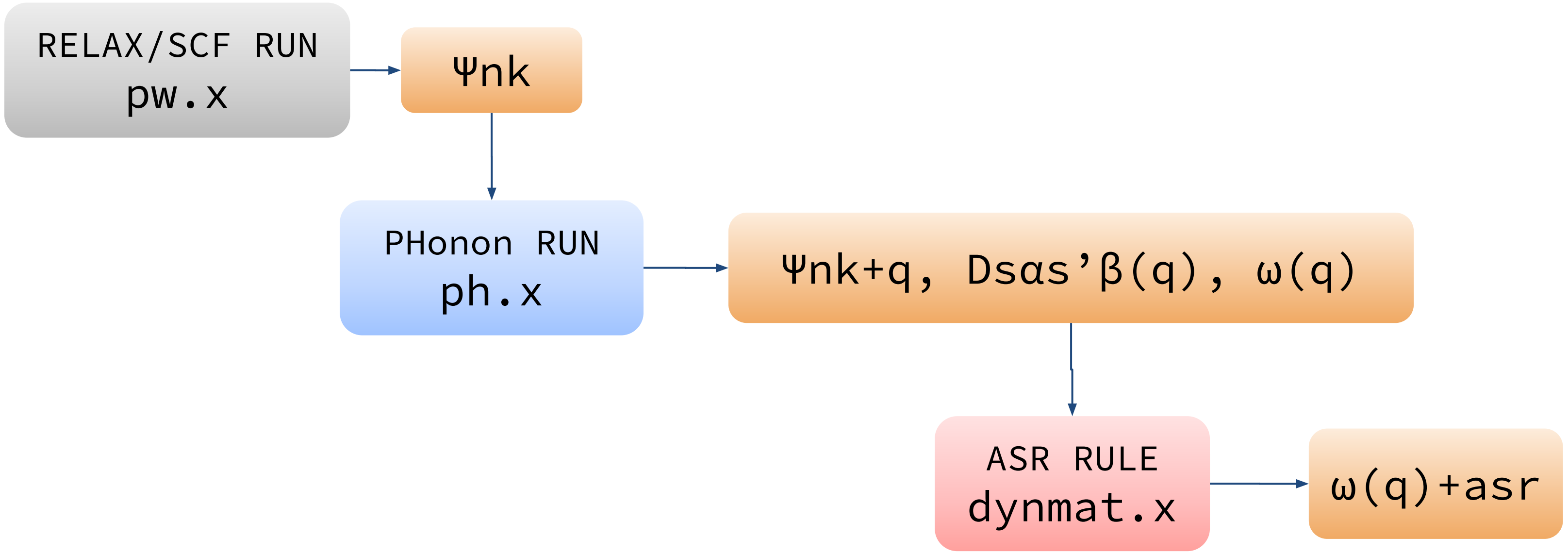




```
$ tail -n 97 harmdyn_support
******************************************************************
     freq (    1) =      -0.154619 [THz] =      -5.157525 [cm-1]
 (  0.302230  0.000000 -0.378303  0.000000  0.000822  0.000000 )
 (  0.284598  0.000000 -0.356232  0.000000  0.000774  0.000000 )
 (  0.275368  0.000000 -0.332855  0.000000  0.000723  0.000000 )
 (  0.265921  0.000000 -0.332855  0.000000  0.000749  0.000000 )
 (  0.265921  0.000000 -0.344679  0.000000  0.000723  0.000000 )
     freq (    2) =      -0.154619 [THz] =      -5.157525 [cm-1]
 ( -0.378274  0.000000 -0.302192  0.000000  0.006755 -0.000000 )
 ( -0.356205  0.000000 -0.284562  0.000000  0.006361 -0.000000 )
 ( -0.344653  0.000000 -0.265888  0.000000  0.005943 -0.000000 )
 ( -0.332829  0.000000 -0.265888  0.000000  0.006154 -0.000000 )
 ( -0.332829  0.000000 -0.275333  0.000000  0.005943  0.000000 )
     freq (    3) =      -0.154619 [THz] =      -5.157525 [cm-1]
 (  0.004764  0.000000  0.004858  0.000000  0.484160  0.000000 )
 (  0.004486  0.000000  0.004575  0.000000  0.455913  0.000000 )
 (  0.004341  0.000000  0.004274  0.000000  0.425994  0.000000 )
 (  0.004192  0.000000  0.004274  0.000000  0.441128  0.000000 )
 (  0.004192  0.000000  0.004426  0.000000  0.425994 -0.000000 )
     freq (    4) =       0.242591 [THz] =       8.091968 [cm-1]
 (  0.050786 -0.000000 -0.087553  0.000000  0.844718 -0.000000 )
 ( -0.009193  0.000000  0.015848 -0.000000 -0.152903  0.000000 )
 ( -0.011864  0.000000  0.033584 -0.000000 -0.324021  0.000000 )
 ( -0.019481  0.000000  0.033584 -0.000000 -0.197328  0.000000 )
 ( -0.019481  0.000000  0.020453 -0.000000 -0.324021  0.000000 )
     freq (    5) =       0.242591 [THz] =       8.091968 [cm-1]
 (  0.258416  0.000000 -0.804506  0.000000 -0.098921  0.000000 )
 ( -0.046776  0.000000  0.145624  0.000000  0.017906  0.000000 )
 ( -0.060367  0.000000  0.308596  0.000000  0.037945  0.000000 )
 ( -0.099124  0.000000  0.308596  0.000000  0.023108  0.000000 )
 ( -0.099124  0.000000  0.187934  0.000000  0.037945  0.000000 )
     freq (    6) =       0.242591 [THz] =       8.091968 [cm-1]
 (  0.808972  0.000000  0.262486  0.000000 -0.021431  0.000000 )
 ( -0.146432  0.000000 -0.047513  0.000000  0.003879  0.000000 )
 ( -0.188978  0.000000 -0.100685  0.000000  0.008221  0.000000 )
 ( -0.310309  0.000000 -0.100685  0.000000  0.005006  0.000000 )
 ( -0.310309  0.000000 -0.061317  0.000000  0.008221  0.000000 )
```

ACOUSTIC SUM RULE (ASR) RULE, STEP 3

The Phonon workflow for modes at a single q point



ACOUSTIC SUM RULE (ASR) RULE, STEP 3

3. `cd Day-2/exercise_ph/step3/`

*Apply the **Acoustic Sum Rule (ASR)** with* `dynmat.x`

Because of the numerical inaccuracies the interatomic force constants do not strictly satisfy the acoustic sum rule (ASR). ASR comes directly from the continuous translational invariance of the crystal. If we translate the whole solid by a uniform displacement, the forces acting on the atoms must be zero.

$$\text{For each}\, \alpha, \beta \,\text{and}\, i\ :\ \sum_{\mathbf{L},\mathbf{j}} C_{\alpha i,\beta j}(\mathbf{R_L}) = 0$$

As a consequence, the frequencies of the acoustic modes must be zero. ASR can be imposed with `dynmat.x`

The phonon workflow

ACOUSTIC SUM RULE (ASR) RULE, STEP 3

3. `cd Day-2/exercise_ph/step3/`

*Apply the **Acoustic Sum Rule (ASR)** with* `dynmat.x`

- Copy `../inputs/dyn.CnSnI3.in` and add the 'crystal' ASR rule
- Copy `../step2/harmdyn_support` in the current folder
- Submit the job
- Check phonon modes with ASR rule applied in `dyn.CnSnI3.out`

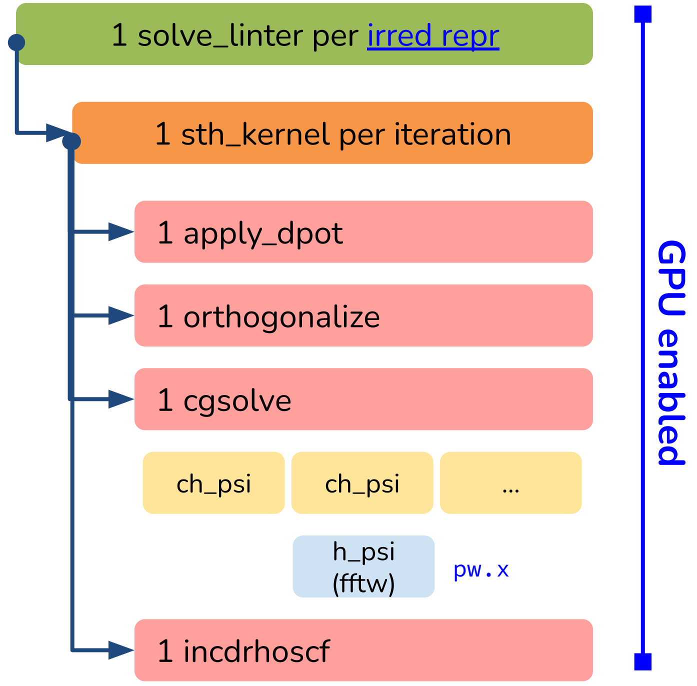
```
$ cat dyn.CnSnI3.in
&input
   fildyn = 'harmdyn_support',
   asr = 'crystal'
 /
```

The ASR rule to impose

File storing the dynamical matrix

The phonon workflow

PHONON ROUTINES RUNNING ON GPUs

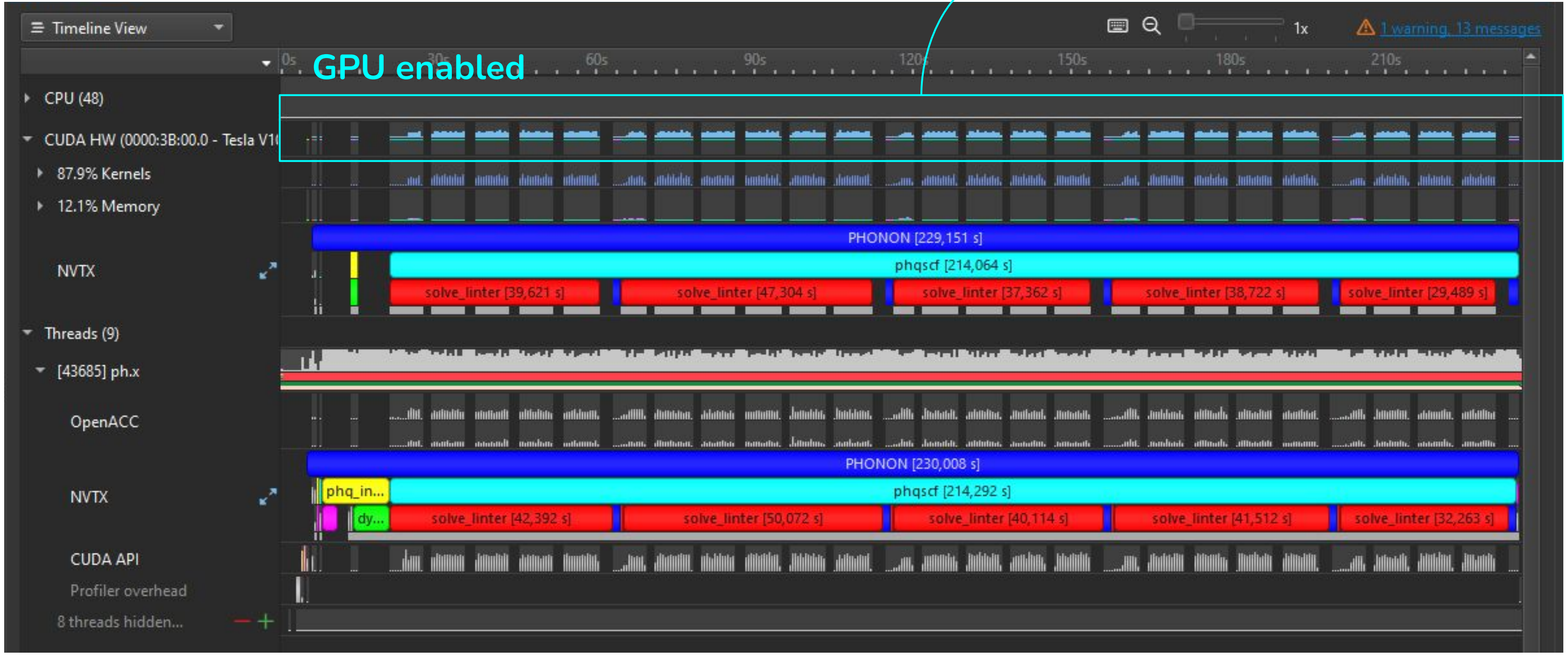


- GPU offloaded version currently available in the develop branch
- Most of the routines in the call path for NC pseudpotentials are GPU-enabled
- GPU offload based on OpenACC + CUDAFortran
- h_psi offload common to PWscf
- Offload of routines from LR_Modules/ exploited also in TDDFPT
- Check with tracing tools!

GPU implementation

NSIGHT SYSTEM TRACE OF PHONON

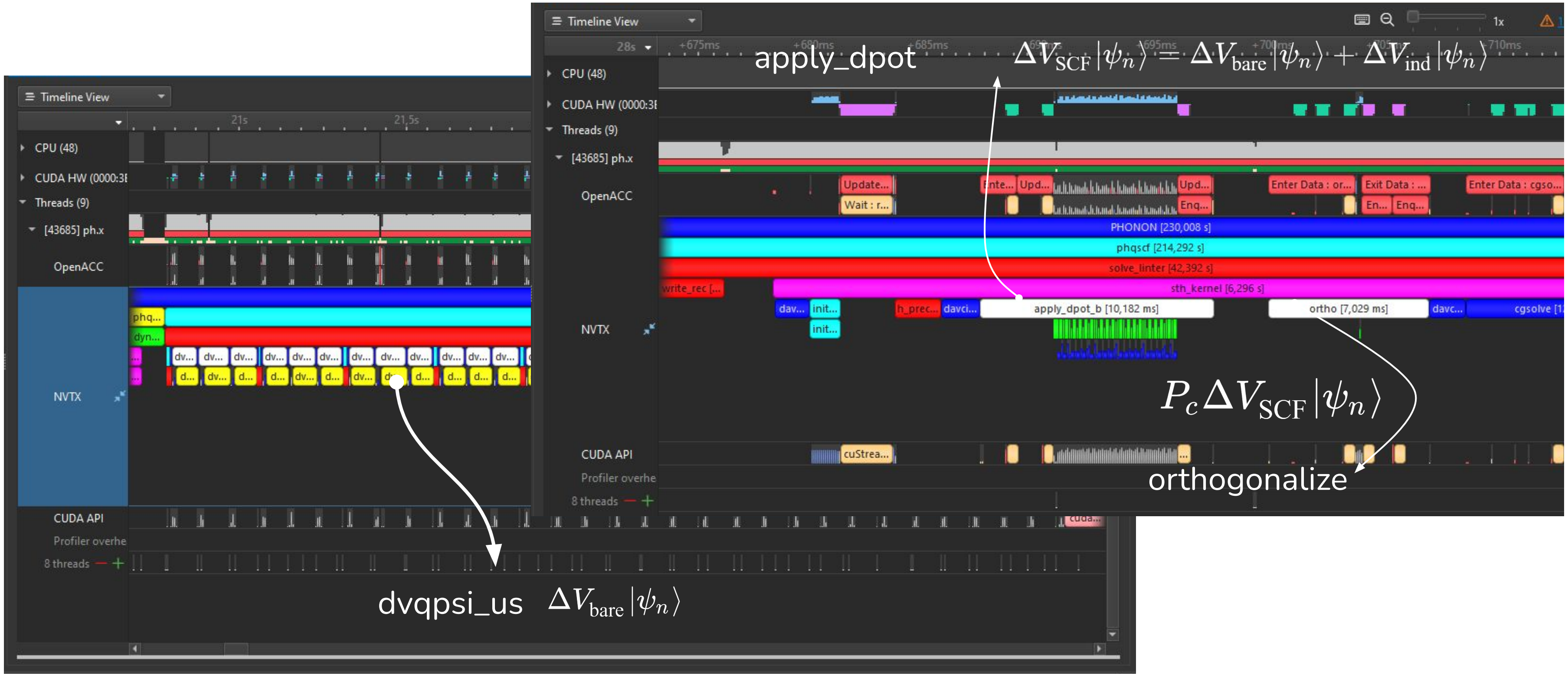
Kernels in `phqscf` offloaded to GPUs



`phqscf` is the main driver for phonon mode calculation (`trans=true`)

GPU implementation

NSIGHT SYSTEM TRACE OF PHONON



apply_dpot $\Delta V_{\mathrm{SCF}} |\psi_n\rangle = \Delta V_{\mathrm{bare}} |\psi_n\rangle + \Delta V_{\mathrm{ind}} |\psi_n\rangle$

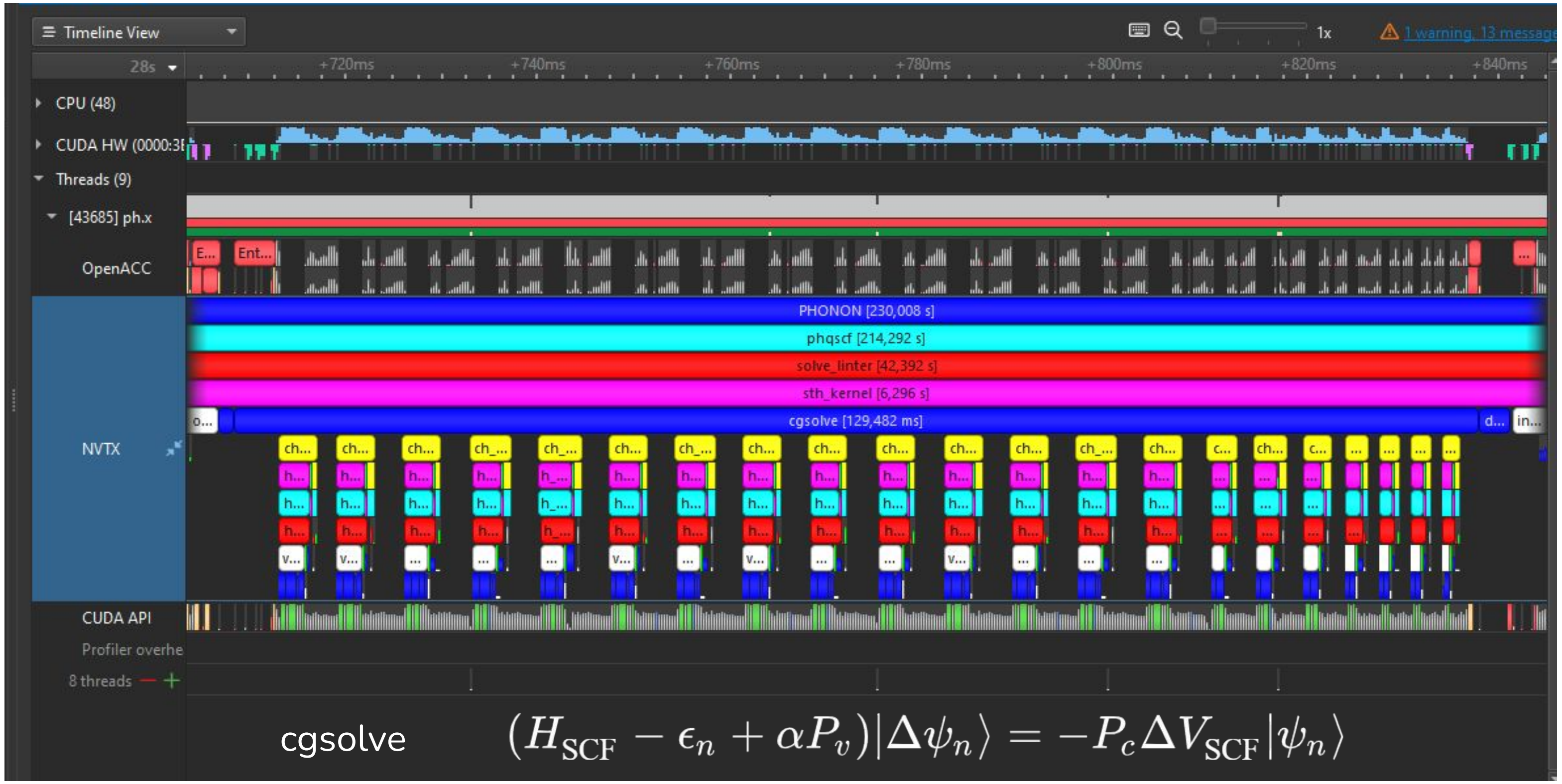
$P_c \Delta V_{\mathrm{SCF}} |\psi_n\rangle$

orthogonalize

dvqpsi_us $\Delta V_{\mathrm{bare}} |\psi_n\rangle$

GPU implementation

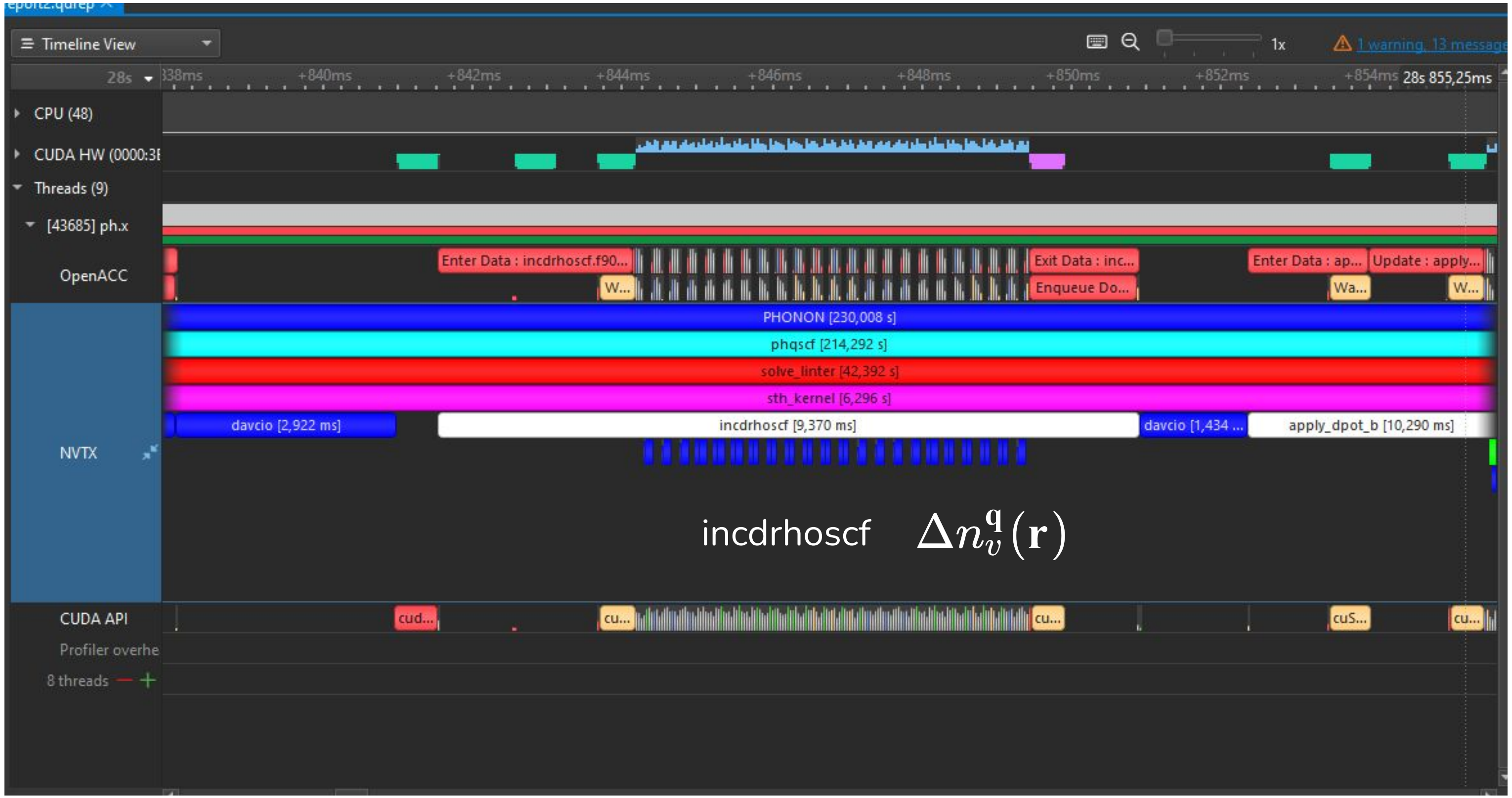
NSIGHT SYSTEM TRACE OF PHONON



cgsolve $$(H_{\mathrm{SCF}} - \epsilon_n + \alpha P_v)|\Delta\psi_n\rangle = -P_c \Delta V_{\mathrm{SCF}} |\psi_n\rangle$$

GPU implementation

NSIGHT SYSTEM TRACE OF PHONON



incdrhoscf $\Delta n_v^{\mathbf{q}}(\mathbf{r})$

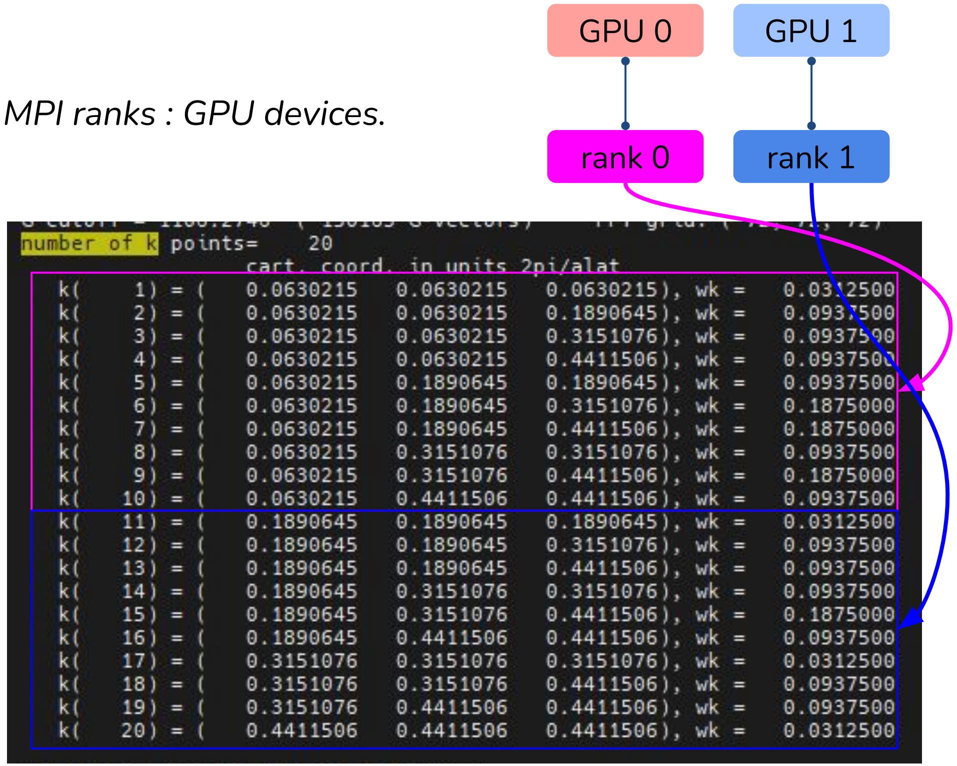
The phonon workflow

MULTI-GPU EXECUTION WITH POOLS, STEP 4

4. `cd Day-2/exercise_ph/step4/`

With pool parallelism we distribute k-points among MPI ranks : GPU devices.

```
mpirun -np N ph.x -nk npools ph.CnSnI3.in
```



```
number of k points=    20
                       cart. coord. in units 2pi/alat
   k(    1) = (   0.0630215   0.0630215   0.0630215), wk =   0.0312500
   k(    2) = (   0.0630215   0.0630215   0.1890645), wk =   0.0937500
   k(    3) = (   0.0630215   0.0630215   0.3151076), wk =   0.0937500
   k(    4) = (   0.0630215   0.0630215   0.4411506), wk =   0.0937500
   k(    5) = (   0.0630215   0.1890645   0.1890645), wk =   0.0937500
   k(    6) = (   0.0630215   0.1890645   0.3151076), wk =   0.1875000
   k(    7) = (   0.0630215   0.1890645   0.4411506), wk =   0.1875000
   k(    8) = (   0.0630215   0.3151076   0.3151076), wk =   0.0937500
   k(    9) = (   0.0630215   0.3151076   0.4411506), wk =   0.1875000
   k(   10) = (   0.0630215   0.4411506   0.4411506), wk =   0.0937500
   k(   11) = (   0.1890645   0.1890645   0.1890645), wk =   0.0312500
   k(   12) = (   0.1890645   0.1890645   0.3151076), wk =   0.0937500
   k(   13) = (   0.1890645   0.1890645   0.4411506), wk =   0.0937500
   k(   14) = (   0.1890645   0.3151076   0.3151076), wk =   0.0937500
   k(   15) = (   0.1890645   0.3151076   0.4411506), wk =   0.1875000
   k(   16) = (   0.1890645   0.4411506   0.4411506), wk =   0.0937500
   k(   17) = (   0.3151076   0.3151076   0.3151076), wk =   0.0312500
   k(   18) = (   0.3151076   0.3151076   0.4411506), wk =   0.0937500
   k(   19) = (   0.3151076   0.4411506   0.4411506), wk =   0.0937500
   k(   20) = (   0.4411506   0.4411506   0.4411506), wk =   0.0312500
```

The phonon workflow

MULTI-GPU EXECUTION WITH POOLS, STEP 4

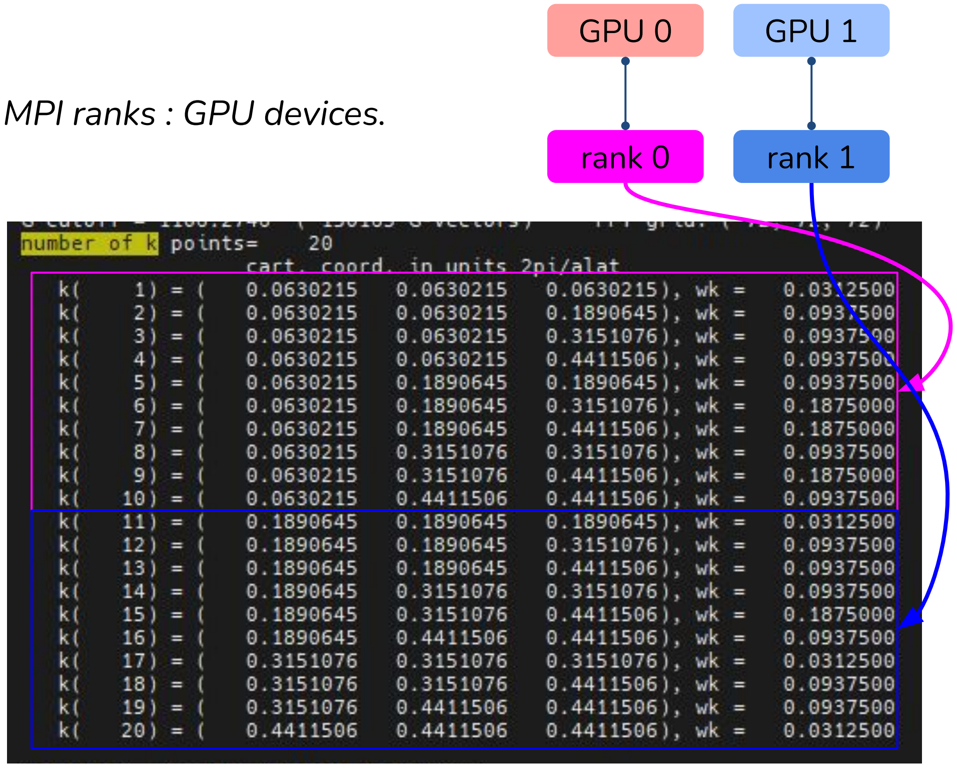
4. `cd Day-2/exercise_ph/step4/`

With pool parallelism we distribute k-points among MPI ranks : GPU devices.

```
mpirun -np N ph.x -nk npools ph.CnSnI3.in
```

- Copy the input file `../step2/ph.CnSnI3.in`
- Copy the folder `../step1/out`
- Modify *npools* in `submit.slurm` to use 2 pools : GPUs
- Submit the jobfile
- Check PHONON wall time

```
tail ph.CnSnI3.out
```



```
number of k points=    20
                       cart. coord. in units 2pi/alat
    k(    1) = (   0.0630215   0.0630215   0.0630215), wk =   0.0312500
    k(    2) = (   0.0630215   0.0630215   0.1890645), wk =   0.0937500
    k(    3) = (   0.0630215   0.0630215   0.3151076), wk =   0.0937500
    k(    4) = (   0.0630215   0.0630215   0.4411506), wk =   0.0937500
    k(    5) = (   0.0630215   0.1890645   0.1890645), wk =   0.0937500
    k(    6) = (   0.0630215   0.1890645   0.3151076), wk =   0.1875000
    k(    7) = (   0.0630215   0.1890645   0.4411506), wk =   0.1875000
    k(    8) = (   0.0630215   0.3151076   0.3151076), wk =   0.0937500
    k(    9) = (   0.0630215   0.3151076   0.4411506), wk =   0.1875000
    k(   10) = (   0.0630215   0.4411506   0.4411506), wk =   0.0937500
    k(   11) = (   0.1890645   0.1890645   0.1890645), wk =   0.0312500
    k(   12) = (   0.1890645   0.1890645   0.3151076), wk =   0.0937500
    k(   13) = (   0.1890645   0.1890645   0.4411506), wk =   0.0937500
    k(   14) = (   0.1890645   0.3151076   0.3151076), wk =   0.0937500
    k(   15) = (   0.1890645   0.3151076   0.4411506), wk =   0.1875000
    k(   16) = (   0.1890645   0.4411506   0.4411506), wk =   0.0937500
    k(   17) = (   0.3151076   0.3151076   0.3151076), wk =   0.0312500
    k(   18) = (   0.3151076   0.3151076   0.4411506), wk =   0.0937500
    k(   19) = (   0.3151076   0.4411506   0.4411506), wk =   0.0937500
    k(   20) = (   0.4411506   0.4411506   0.4411506), wk =   0.0312500
```

The phonon workflow

MULTI-GPU EXECUTION WITH IMAGES, STEP 5

5. `cd Day-2/exercise_ph/step5/`

With image parallelism at q point we distribute irreducible representations among MPI ranks : GPU devices.

```
mpirun -np N ph.x -ni nimages ph.CnSnI3.in
```

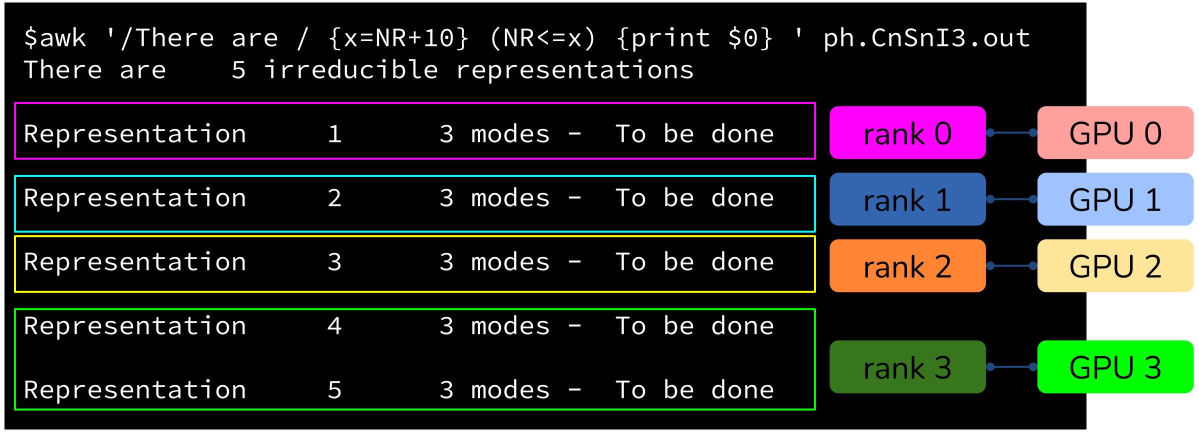
```
$awk '/There are / {x=NR+10} (NR<=x) {print $0} ' ph.CnSnI3.out
There are     5 irreducible representations

Representation     1       3 modes -   To be done

Representation     2       3 modes -   To be done

Representation     3       3 modes -   To be done

Representation     4       3 modes -   To be done

Representation     5       3 modes -   To be done
```

MULTI-GPU EXECUTION WITH IMAGES, STEP 5

5. `cd Day-2/exercise_ph/step5/`

With image parallelism at q point we distribute irreducible representations among MPI ranks : GPU devices.

mpirun -np N ph.x **-ni nimages** ph.CnSnI3.in



```
$awk '/There are / {x=NR+10} (NR<=x) {print $0} ' ph.CnSnI3.out
There are    5 irreducible representations

Representation     1       3 modes -  To be done

Representation     2       3 modes -  To be done

Representation     3       3 modes -  To be done

Representation     4       3 modes -  To be done

Representation     5       3 modes -  To be done
```

MULTI-GPU EXECUTION WITH IMAGES, STEP 5

5. `cd Day-2/exercise_ph/step5/`

With image parallelism at q point we distribute irreducible representations among MPI ranks : GPU devices.

```
mpirun -np N ph.x -ni nimages ph.CnSnI3.in
mpirun -np 1 ph.x -ni 1 ph.CnSnI3.recover.in
```

! A **recover run** is needed to collect the IFCs and diagonalize the dynamical matrix

- Copy the input file `../step2/ph.CnSnI3.in`
- Copy `ph.CnSnI3.in` as `ph.CnSnI3.recover.in` and add `recover=.true.` in `&inputph` of the latter
- Copy the `../step1/out` directory in the current folder
- Modify *nimages* in submit.slurm to distribute on 4 MPIs : GPUs
- Submit the jobfile

! *With image parallelism there is 1 output file for each image*

```
$ cat ph.CnSnI3.recover.in
&inputph
        fildyn = 'harmdyn_support'
        tr2_ph = 1d-16
        outdir = './out'
        recover = .true.
/
0.0 0.0 0.0
```

MULTI-GPU EXECUTION WITH IMAGES, STEP 5

5. `cd Day-2/exercise_ph/step5/`

With image parallelism at q point we distribute irreducible representations among MPI ranks : GPU devices.

```
mpirun -np N ph.x -ni nimages ph.CnSnI3.in

mpirun -np 1 ph.x -ni 1 ph.CnSnI3.recover.in
```

- Check the workfload for each image

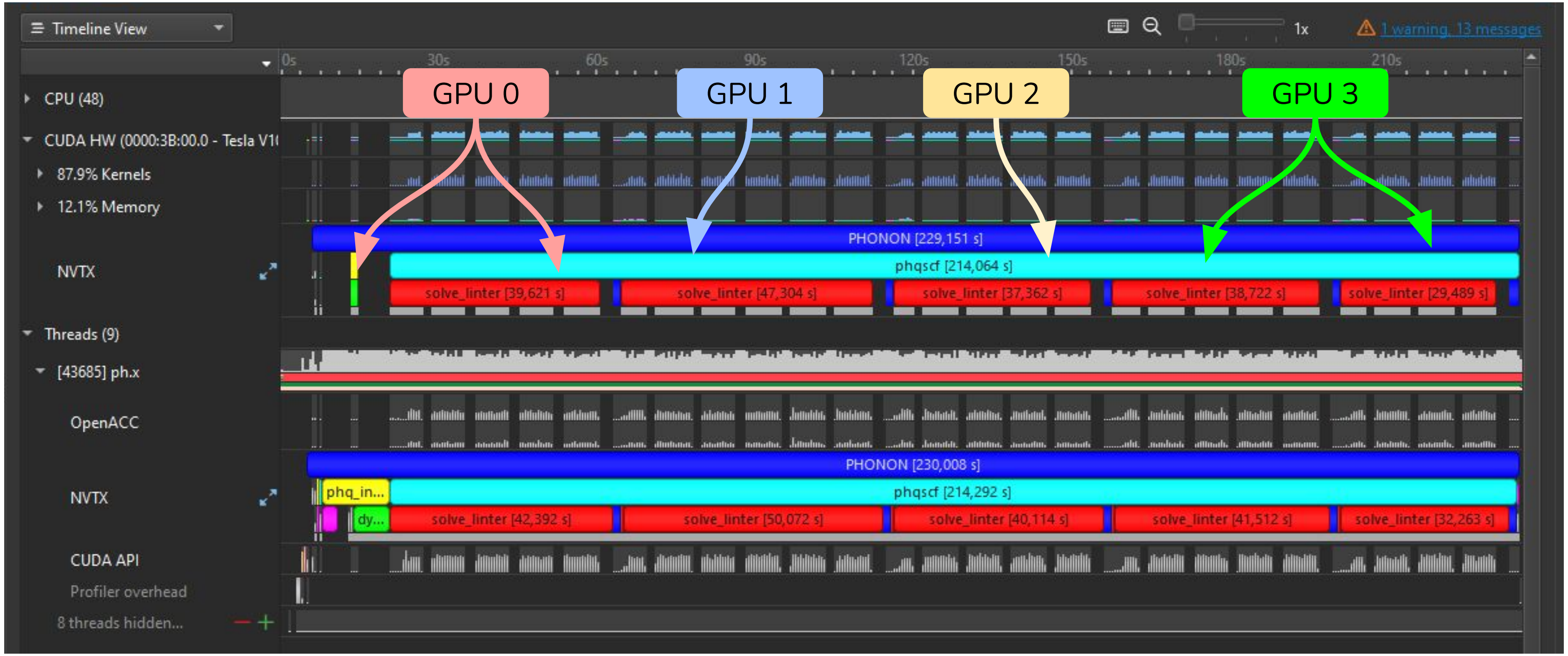
  ! *image 0 has an extra scf run to compute the part of the dyn matrix not depending upon the change of Bloch functions*

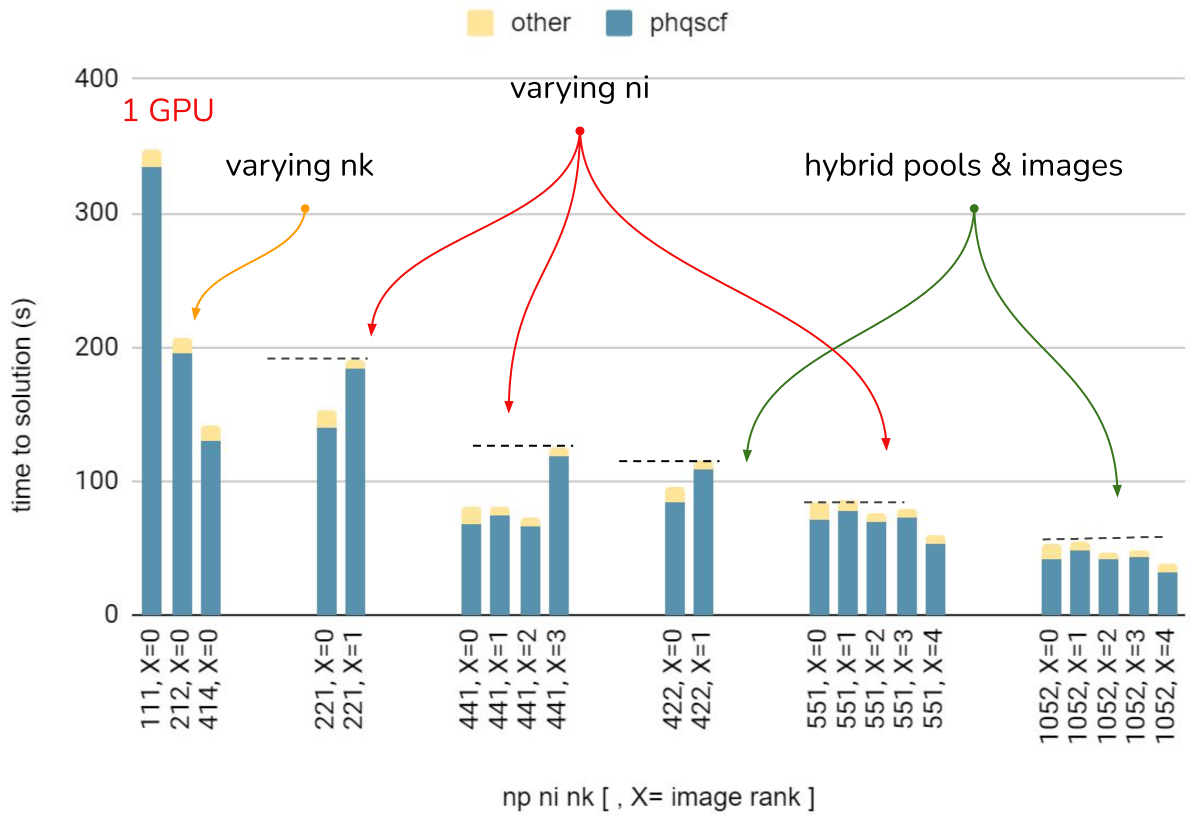
- Compare the wall times. Which image takes longer? Why ?

```
$ awk '/I am image/ {x=NR+3} (NR<=x) {print $0} ' out.*_0
I am image number     0 and my work is about    4 scf runs. I calculate:
q point number     1, representations:
0 1

I am image number     1 and my work is about    3 scf runs. I calculate:
q point number     1, representations:
2

I am image number     2 and my work is about    3 scf runs. I calculate:
q point number     1, representations:
3

I am image number     3 and my work is about    6 scf runs. I calculate:
q point number     1, representations:
4 5
```

The phonon workflow

MULTI-GPU EXECUTION WITH IMAGES, STEP 5




other
phqscf
400
300
200
100
0
time to solution (s)
1 GPU
varying nk
varying ni
hybrid pools & images
111, X=0
212, X=0
414, X=0
221, X=0
221, X=1
441, X=0
441, X=1
441, X=2
441, X=3
422, X=0
422, X=1
551, X=0
551, X=1
551, X=2
551, X=3
551, X=4
1052, X=0
1052, X=1
1052, X=2
1052, X=3
1052, X=4
np ni nk [, X= image rank]

CALCULATIONS AVAILABLE FROM ph.x

Find the input option for your calculation at QE/INPUT_PH

→ Single q calculation (trans = .true.) + ASR (TODAY)

✳ Dielectric constant (`epsil = .true.`), effective charges (`zeu = .true.`)

✳ LO-TO splitting in insulators and IR cross sections (`dynmat.x`)

✳ Raman cross sections (`lraman=.true.`)

✳ Phonon mode dispersion (`ldisp = .true.` , `ph.x + q2r.x + matdyn.x`)

✳ Electron-phonon interaction coefficients
(`electron_phonon='simple','interpolated',...`)

Efficient materials modelling on HPC
with QUANTUM ESPRESSO, Yambo and BigDFT

Hands-on session – Day 2

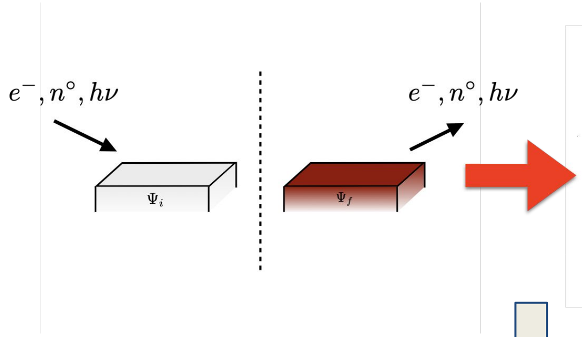
TDDFPT FOR HPC AND GPUs

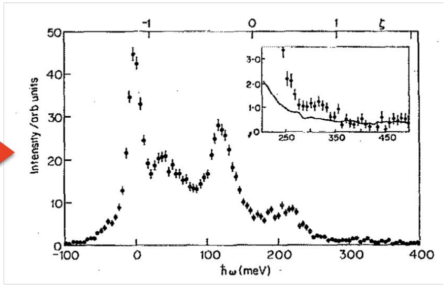
Tommaso Gorni and Oscar Baseggio

t.gorni@cineca.it obaseggi@sissa.it

Ab initio spectroscopy

Time-Dependent Density-Functional Perturbation Theory (TDDFpT)





- Optical absorption spectroscopy
- Electron energy loss spectroscopy (EELS)
- Inelastic X-ray scattering (IXS)
- Inelastic neutron scattering (INS)
- ...

Dynamical susceptibilities

$$\varphi_{\text{ext}}(t) \longrightarrow A(t) \approx A^\circ + A'(t)$$

$$A'(t) = \int \mathrm{d}t' \chi(t-t') \varphi_{\text{ext}}(t')$$

OUTLINE

1. How to compute the charge susceptibility with `turbo_eels.x`
2. How to compute the spin susceptibility with `turbo_magnon.x`

Time-Dependent Density-Functional Perturbation Theory

Kohn-Sham hamiltonian

Perturbing potential

$$(\hat{H}^\circ - \epsilon^\circ_{n,\mathbf{k}} - \hbar\omega)|\psi'_{n,\mathbf{k}+\mathbf{q}}(\omega)\rangle = -\hat{P}_c\left[\hat{V}'_{\mathrm{Hxc},\mathbf{q}}(\omega) + \hat{V}'_{\mathrm{ext},\mathbf{q}}(\omega)\right]|\psi^\circ_{n,\mathbf{k}}\rangle$$

$$(\hat{H}^\circ - \epsilon^\circ_{n,-\mathbf{k}} + \hbar\omega)|\psi'^*_{n,-\mathbf{k}-\mathbf{q}}(-\omega)\rangle = -\hat{P}_c\left[\hat{V}'_{\mathrm{Hxc},\mathbf{q}}(\omega) + \hat{V}'_{\mathrm{ext},\mathbf{q}}(\omega)\right]|\psi^{\circ *}_{n,-\mathbf{k}}\rangle$$

(2 X nbnd X nks) coupled equations

Resonant and anti-resonant response of the (n,k) state

Option 1: Sternheimer approach (PHonon-like)

- Invert the linear system **for each frequency** to build the response density matrix

$$\hat{\rho}'_{\mathbf{q}}(\omega) = \sum_{n,\mathbf{k}}^{\mathrm{occ.}}\left[|\psi'_{n,\mathbf{k}+\mathbf{q}}(\omega)\rangle\langle\psi^\circ_{n,\mathbf{k}}| + |\psi'^*_{n,-\mathbf{k}-\mathbf{q}}(-\omega)\rangle\langle\psi^{\circ *}_{n,-\mathbf{k}}|\right]$$

- Compute the dynamical susceptibility

$$n'(\mathbf{q},\omega) = \boxed{\chi(\mathbf{q},\mathbf{q},\omega)} V'_{\mathrm{ext},\mathbf{q}}(\omega) = \mathrm{Tr}\left[\hat{n}^\dagger_{\mathbf{q}}\hat{\rho}'_{\mathbf{q}}(\omega)\right]$$

$\propto$ EELS cross section

Cost of static linear response X number of frequencies

Charge Fluctuations

Time-Dependent Density-Functional Perturbation Theory

Kohn-Sham hamiltonian

Perturbing potential

$$(\hat{H}^\circ - \epsilon^\circ_{n,\mathbf{k}} - \hbar\omega)|\psi'_{n,\mathbf{k}+\mathbf{q}}(\omega)\rangle = -\hat{P}_{\mathcal{C}}\left[\hat{V}'_{\mathrm{Hxc},\mathbf{q}}(\omega) + \hat{V}'_{\mathrm{ext},\mathbf{q}}(\omega)\right]|\psi^\circ_{n,\mathbf{k}}\rangle$$

$$(\hat{H}^\circ - \epsilon^\circ_{n,-\mathbf{k}} + \hbar\omega)|\psi'^{*}_{n,-\mathbf{k}-\mathbf{q}}(-\omega)\rangle = -\hat{P}_{\mathcal{C}}\left[\hat{V}'_{\mathrm{Hxc},\mathbf{q}}(\omega) + \hat{V}'_{\mathrm{ext},\mathbf{q}}(\omega)\right]|\psi^{\circ *}_{n,-\mathbf{k}}\rangle$$

(2 X nbnd X nks) coupled equations

Resonant and anti-resonant response of the (n,k) state

Option 2: Lanczos approach

- Recast as a unique linear problem

$$\left(\hbar\omega - \mathcal{L}_{\mathbf{q}}(\omega)\right) \cdot \hat{\rho}'_{\mathbf{q}}(\omega) = \left[\hat{V}'_{\mathrm{ext},\mathbf{q}}(\omega)\,,\hat{\rho}^\circ\right]$$

- The response density matrix can be represented as an array of response orbitals ('batch').

$$\hat{\rho}'_{\mathbf{q}}(\omega) \rightarrow \begin{pmatrix} \psi'_{n,\mathbf{k}+\mathbf{q}}(\omega) \\ \psi'^{*}_{n,-\mathbf{k}-\mathbf{q}}(-\omega) \end{pmatrix}_{n,\mathbf{k}}$$

batch size = (~~2~~ x npw x nbnd x nks) complex numbers

making use of time-reversal symmetry (standard batch rotation)

- The action of the Liouvillian on the batch costs roughly twice a static linear response step.

Time-Dependent Density-Functional Perturbation Theory

Option 2: Lanczos approach

For adiabatic xc kernels

$$\left(\hbar\omega - \mathcal{L}_{\mathbf{q}}(\cancel{\omega})\right) \cdot \hat{\rho}'_{\mathbf{q}}(\omega) = \left[\hat{V}'_{\mathrm{ext},\mathbf{q}}(\omega)\,,\hat{\rho}^{\circ}\right]$$

- Tridiagonalize the Liouvillian via Lanczos recursion (computationally intensive part).

$$\beta_{i+i}\mathbf{V}_{i+1} = \mathcal{L}_{\mathbf{q}}\mathbf{V}_i - \gamma_i\mathbf{V}_{i-1}$$
$$\gamma_{i+i}\mathbf{U}_{i+1} = \mathcal{L}^{\dagger}_{\mathbf{q}}\mathbf{U}_i - \beta_i\mathbf{U}_{i-1}$$

$$\mathcal{L}_{\mathbf{q}} \approx T^N_{\mathbf{q}} = \begin{pmatrix} 0 & \gamma_2 & 0 & \cdots & 0 \\ \beta_2 & 0 & \gamma_3 & 0 & 0 \\ 0 & \beta_3 & 0 & \ddots & 0 \\ \vdots & 0 & \ddots & \ddots & \gamma_N \\ 0 & 0 & 0 & \beta_N & 0 \end{pmatrix}$$

- Invert at **any desired frequency** at a negligible computation cost.

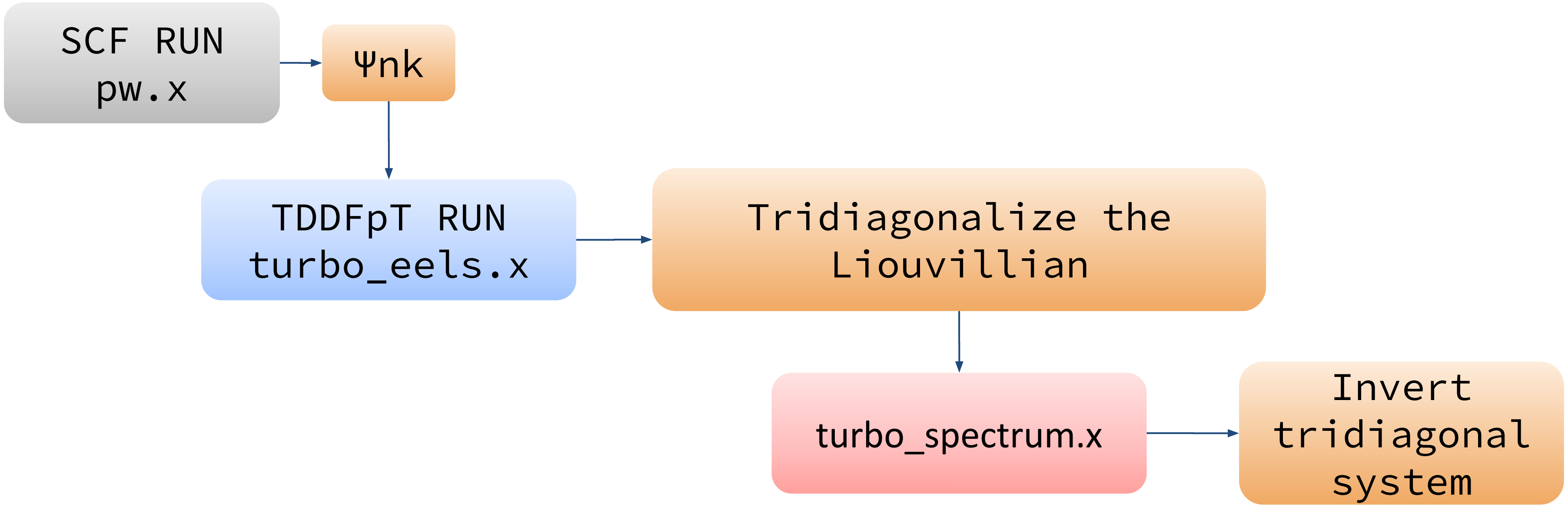
$$\chi(\mathbf{q},\mathbf{q},\omega) \approx \left\langle u\,,(\hbar\omega + i\eta - T^N_{\mathbf{q}})^{-1} v\right\rangle$$

Imaginary shift to regularize poles (broadening term)

Batches (npw x nbnd x nks) complex numbers

EXERCISE ON VEGA: TURBO_EELS.X

The turbo_eels workflow for modes at a single q point



EXERCISE ON VEGA: EELS IN BULK SILICON

Go to the directory with the input files:

```
cd  ~/…/…/example_eels
```

In this directory you will find:

- `README.md` – File describing how to do the exercise
- `pw.Si.scf.in` – Input file for the SCF ground-state calculation
- `turbo_eels.Si.tddfpt.in` – Input file for the EELS calculation
- `turbo_spectrum.Si.pp.in` – Input file for post processing calculation
- `reference` – Directory with the reference results

The eels workflow

PWSCF SIMULATION, STEP 1

1. `cd Day-2/example-eels`

   Perform a self-consistent field ground-state calculation for silicon using the `pw.x` program.

- Open and check `pw.Si.scf.in`
- Open and check `submit_pw.slurm`
- Submit the job file

```
&control
   calculation  = 'scf'
   restart_mode = 'from_scratch'
   prefix       = 'Si'
   pseudo_dir   = '../../pseudo'
   outdir       = './tempdir'
   verbosity    = 'high'
/
&system
   ibrav     = 2
   celldm(1) = 10.26
   nat       = 2
   ntyp      = 1
   ecutwfc   = 20.0
/
&electrons
   conv_thr =  1.0d-10
/
ATOMIC_SPECIES
 Si   28.08   Si.upf
ATOMIC_POSITIONS alat
 Si 0.00 0.00 0.00
 Si 0.25 0.25 0.25
K_POINTS automatic
 12 12 12 0 0 0
```

EELS CALCULATION, STEP 2

2. `cd Day-2/example-eels`

Perform a Lanczos coefficients calculation using the `turbo_eels.x` program.

- Open and check `turbo_eels.Si.tddfpt.in`

```
&lr_input
  prefix        = 'Si'
  outdir        = './tempdir'
  restart_step  = 250
  restart       = .false.
/
&lr_control
  itermax       = 2000
  q1            = 0.866
  q2            = 0.000
  q3            = 0.000
/
```

number of Lanczos iterations

value of the transferred momentum

- Submit the job file `submit_eels.slurm`

The eels workflow

SPECTRUM CALCULATION, STEP 3

3. `cd Day-2/example-eels`

Perform spectrum using the `turbo_spectrum.x` program.

- Open and check `turbo_spectrum.Si.pp.in`

```
&lr_input
  prefix        = 'Si'
  outdir        = './tmp'
  eels          = .true.
  units         = 1
  itermax0      = 2000
  itermax       = 10000
  extrapolation = 'osc'
  epsil         = 0.03675
  start         = 0.d0
  increment     = 0.05d0
  end           = 50.d0
/
```

units: the unit system used (1=eV)

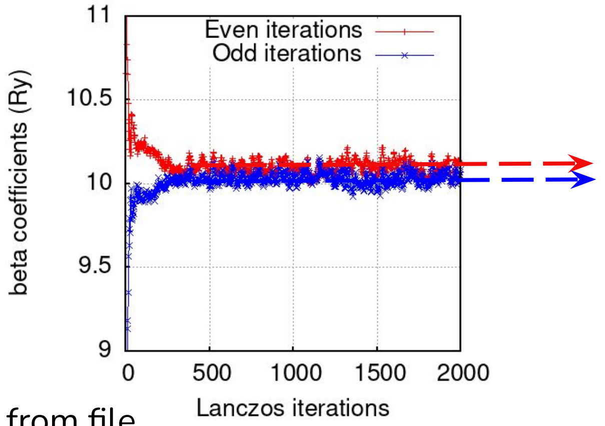
itermax0: number of Lanczos coefficients read from file

itermax: total number of Lanczos coefficients used (read + **extrapolated**)

extrapolation: extrapolation scheme

epsil: broadening term (Ry)

start, increment, end: spectrum energy grid (eV)



- Submit the job file `submit_spectrum.slurm` turbo_eels.x

IMPLEMENTED FEATURES

✓ Metals and insulators

! Non-magnetic systems only

✓ Spin-orbit coupling (fully relativistic approach)

✓ LDA, GGA functionals

✓ Norm-conserving and ultrasoft pseudopotentials

✓ Use of symmetries to reduce k-points (nks)

✓ R&G, k-point parallelization

✓ GPU offloading (new, work in progress)

Spin Fluctuations

Time-Dependent Density-Functional Perturbation Theory

Kohn-Sham hamiltonian

Spinor wavefunctions

Perturbing potential (magnetic field)

$$(\hat{H}^{\circ} - \epsilon^{\circ}_{n,\mathbf{k}} - \hbar\omega)|\psi'_{n,\mathbf{k}+\mathbf{q}}(\omega)\rangle = -\hat{P}_{\mathcal{C}}\left[\hat{V}'_{\mathrm{Hxc},\mathbf{q}}(\omega) + \hat{V}'_{\mathrm{ext},\mathbf{q}}(\omega)\right]|\psi^{\circ}_{n,\mathbf{k}}\rangle$$

$$(\hat{H}^{\circ} - \epsilon^{\circ}_{n,-\mathbf{k}} + \hbar\omega)\hat{T}|\psi'_{n,-\mathbf{k}-\mathbf{q}}(-\omega)\rangle = -\hat{P}^{+}_{\mathcal{C}}\left[\hat{V}'^{+}_{\mathrm{Hxc},\mathbf{q}}(\omega) + \hat{V}'^{+}_{\mathrm{ext},\mathbf{q}}(\omega)\right]\hat{T}|\psi^{\circ}_{n,-\mathbf{k}}\rangle$$

Time-reversal operator

$\hat{O}^{+}_{\mathbf{q}}(\omega) = \hat{T}\hat{O}_{-\mathbf{q}}(-\omega)\hat{T}^{-1}$

Resonant and anti-resonant response of the (n,k) state

- Recast as a unique linear problem $$\left(\hbar\omega - \mathcal{L}_{\mathbf{q}}(\omega)\right)\cdot\hat{\rho}'_{\mathbf{q}}(\omega) = \left[\hat{V}'_{\mathrm{ext},\mathbf{q}}(\omega)\,,\hat{\rho}^{\circ}\right]$$

- The response density matrix can be represented as an array of response orbitals ('batch'). $$\hat{\rho}'_{\mathbf{q}}(\omega) \rightarrow \begin{pmatrix} \psi'_{n,\mathbf{k}+\mathbf{q}}(\omega) \\ \hat{T}\psi'_{n,-\mathbf{k}-\mathbf{q}}(-\omega) \end{pmatrix}_{n,\mathbf{k}}$$

batch size = (4 x npw x nbnd x nks) complex numbers

- The action of the Liouvillian on the batch costs roughly twice a static linear response step.

Spin Fluctuations

Time-Dependent Density-Functional Perturbation Theory

Lanczos approach

$$\left(\hbar\omega - \mathcal{L}_{\mathbf{q}}(\cancel{\omega})\right) \cdot \hat{\rho}'_{\mathbf{q}}(\omega) = \left[\hat{V}'_{\mathrm{ext},\mathbf{q}}(\omega)\,,\hat{\rho}^{\circ}\right]$$

For adiabatic xc kernels

- Tridiagonalize the Liouvillian via Lanczos recursion (computational intensive part).

$$\begin{aligned}\beta_{i+i}\mathbf{V}_{i+1} &= \mathcal{L}_{\mathbf{q}}\mathbf{V}_i - \alpha_i\mathbf{V}_i - \gamma_i\mathbf{V}_{i-1}\\ \gamma^*_{i+i}\mathbf{U}_{i+1} &= \mathcal{L}^{\dagger}_{\mathbf{q}}\mathbf{U}_i - \alpha_i\mathbf{U}_i - \beta_i\mathbf{U}_{i-1}\end{aligned}$$

$$\mathcal{L}_{\mathbf{q}} \approx T^N_{\mathbf{q}} = \begin{pmatrix} \alpha_1 & \gamma_2 & 0 & \cdots & 0 \\ \beta_2 & \alpha_2 & \gamma_3 & 0 & 0 \\ 0 & \beta_3 & \alpha_3 & \ddots & 0 \\ \vdots & 0 & \ddots & \ddots & \gamma_N \\ 0 & 0 & 0 & \beta_N & \alpha_N \end{pmatrix}$$

- Invert at **any desired frequency** at a negligible computation cost.

INS cross section $\propto$ anti-hermitian part of $\chi_{\lambda\mu}(\mathbf{q},\mathbf{q},\omega)$

Magnetization component

$$\chi_{\lambda\mu}(\mathbf{q},\mathbf{q},\omega) \approx \left\langle u^{\lambda}\,,(\hbar\omega + i\eta - T^N_{\mathbf{q}})^{-1} v^{\mu}\right\rangle$$

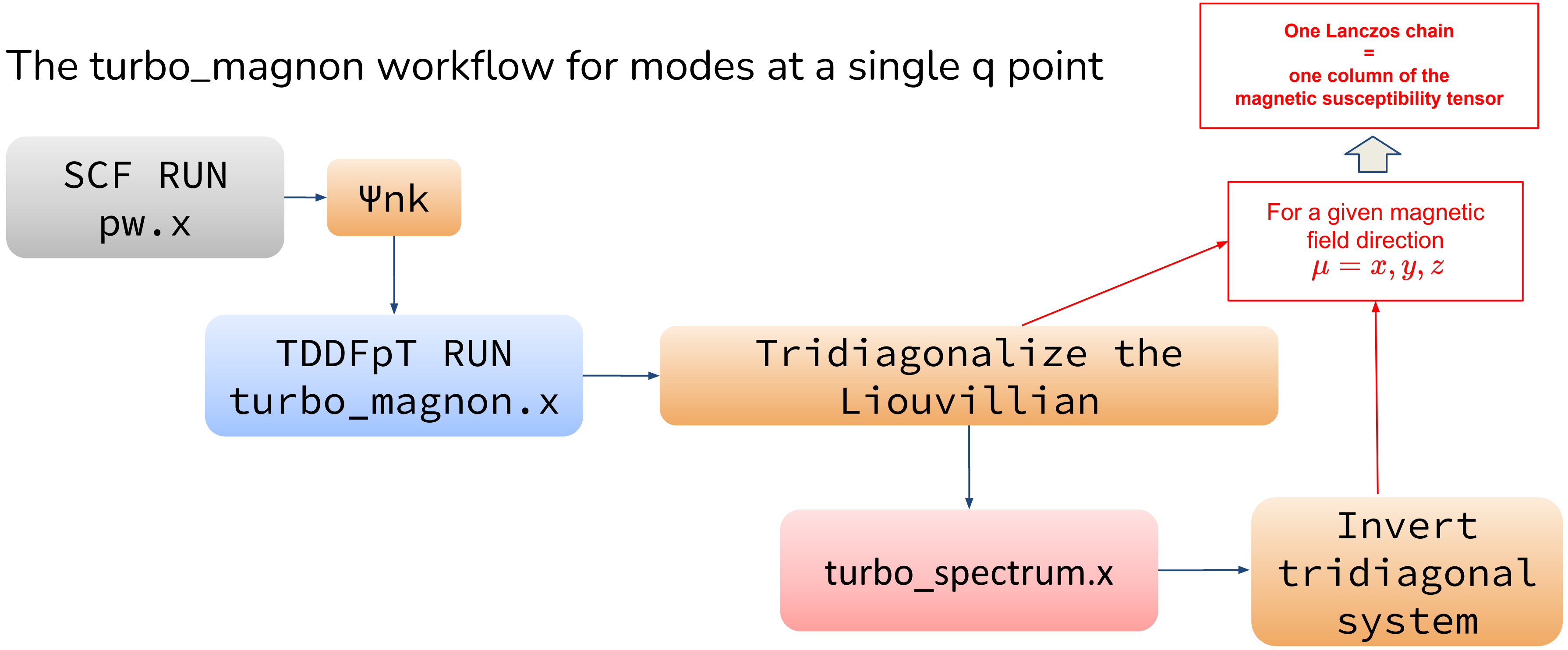
Magnetic field component

Imaginary shift to regularize poles (broadening term)

Batches (4 x npw x nbnd x nks) complex numbers

EXERCISE ON VEGA: TURBO_MAGNON.X

The turbo_magnon workflow for modes at a single q point


SCF RUN
pw.x
Ψnk
TDDFpT RUN
turbo_magnon.x
Tridiagonalize the Liouvillian
turbo_spectrum.x
Invert tridiagonal system
For a given magnetic field direction
$\mu = x, y, z$
One Lanczos chain
=
one column of the magnetic susceptibility tensor

HPC implementation

EXERCISE ON VEGA: MAGNONS IN BULK IRON

Go to the directory with the input files:

```
cd  ~/…/…/example_magnor
```

In this directory you will find:

- `README.md` – File describing how to do the exercise
- `pw.Fe.scf.in` – Input file for the SCF ground-state calculation
- `turbo_magnon.Fe.tddfpt.in` – Input file for the magnon calculation
- `turbo_spectrum.Fe.pp.in` – Input file for post processing calculation
- `reference` – Directory with the reference results

The magnon workflow

PWSCF SIMULATION, STEP 1

1. `cd Day-2/example-magnon`

   Perform a self-consistent field ground-state calculation for iron using the `pw.x` program.

- Open and check `pw.Fe.scf.in`
- Open and check `submit_pw.slurm`
- Submit the job file

```
 &control
    calculation  = 'scf'
    restart_mode = 'from_scratch'
    prefix       = 'Fe'
    outdir       = './tempdir'
    pseudo_dir   = '../../pseudo'
    verbosity    = 'high'
 /
 &system
    nosym           = .true.
    noinv           = .true.
    noncolin        = .true.
    lspinorb        = .false.
    ibrav           = 3
    celldm(1)       = 5.406
    nat             = 1
    ntyp            = 1
    ecutwfc         = 40
    occupations     = 'smearing'
    smearing        = 'gaussian'
    degauss         = 0.01
    starting_magnetization(1) = 0.15
 /
 &electrons
    mixing_beta     = 0.3
    conv_thr        = 1.d-12
 /
ATOMIC_SPECIES
Fe  55.85   Fe.pz-n-nc.UPF
ATOMIC_POSITIONS alat
Fe  0.00000000 0.00000000 0.00000000
K_POINTS automatic
 4 4 4 0 0 0
```

The magnon workflow

MAGNON CALCULATION, STEP 2

2. `cd Day-2/example-magnon`

Perform a Lanczos coefficients calculation using the `turbo_magnon.x` program.

- Open and check `turbo_magnon.Fe.tddfpt.in`

```
 &lr_input
   prefix       = 'Fe'
   outdir       = './tempdir'
   restart_step = 200
   restart      = .false.
 /
&lr_control
    itermax           = 5000
    q1                = 0.1d0
    q2                = 0.1d0
    q3                = 0.0d0
    pseudo_hermitian = .true.
    ipol              = 2
/
```

itermax: number of Lanczos iterations

q1, q2, q3: value of the transferred momentum

pseudo_hermitian: choose pseudo-Hermitian or non-Hermitian Lanczos algorithm

ipol: column of the dynamical magnetic susceptibility

- Submit the job file `submit_magnon.slurm`

The magnon workflow

SPECTRUM CALCULATION, STEP 3

3. `cd Day-2/example-magnon`

Perform spectrum using the `turbo_spectrum.x` program.

- Open and check `turbo_spectrum.Fe.pp.in`

```
&lr_input
  prefix        = 'Fe'
  outdir        = './tempdir'
  magnons       = .true.
  units         = 3
  itermax0      = 5000
  itermax       = 15000
  extrapolation = 'osc'
  epsil         = 1.0
  ipol          = 2
  start         = 0.d0
  increment     = 1.d0
  end           = 28.d0
/
```

flag for magnon calculation

the unit system used (3=meV)

number of Lanczos coefficients read from file

total number of Lanczos coefficients used (read + extrapolated)

extrapolation scheme

broadening term (meV)

element of the dynamical polarizability

spectrum energy grid (meV)

- Submit the job file `submit_spectrum.slurm`
- *grep chi_2_2 Femag.plot_chi.dat &> Chi_2_2.dat &   and use the script for gnuplot*

turbo_magnon.x

IMPLEMENTED FEATURES

✓ Metals and insulators

✓ Spin-orbit coupling (fully relativistic approach)

✓ LDA functionals

✓ Norm-conserving pseudopotentials only

! No symmetry is used (set noinv = .true. and nosym = .true. in the pw input)

✓ R&G, k-point parallelization

✓ GPU offloading (new, work in progress)