Excercise sheet 7

The subject of this exercise sheet well be the simulation of optical spectra using time-dependent DFT. The TDDFT calculations will be performed with the YAMBO code, which can process the electronic structure calculated from Quantum Espresso.

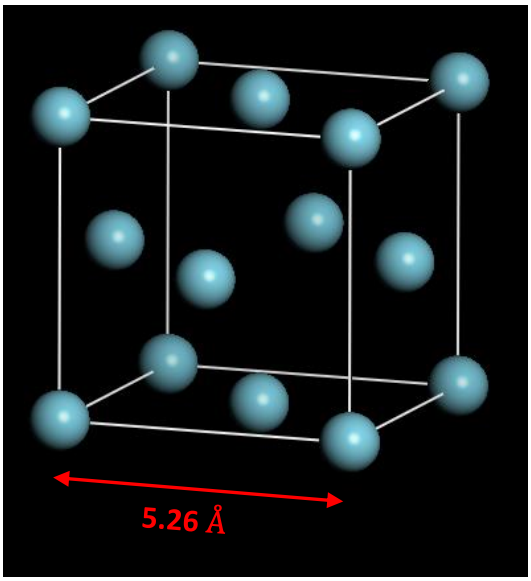
As most of the calculations are relatively heavy, it is strongly advised to run the simulations on the HPC. YAMBO itself only runs under Linux (or at least there is no Windows version available). Prepared YAMBO files for the use on the HPC can be found on StudOn (yambo.tar.gz). Ideally, simply put the folder with the YAMBO executables in your bin folder (next to the folder with the Quantum Espresso files). Like Quantum Espresso, YAMBO uses the math libraries provided by intel. For this reason, you need to load the intel64 module with

module load intel64

before you can run YAMBO on the HPC.

Exercise 1: TDDFT for absorption spectra

In this exercise, you will use TDDFT to simulate the absorption spectrum of solid Argon, a wide bandgap insulator. Solid Argon has an FCC structure with a cubic lattice constant of 5.26 Å.



(a) Optimize the geometry. Use a cutoff energy of 80 Ry and a 6x6x6 k-point grid. Download a normconserving Argon pseudopotential from http://www.pseudo-dojo.org/ (make sure that 'format' is set to 'upf') [1].
Then, calculate the electronic structure on a 20x20x20 k-point grid and using 30 bands. We will use the wavefunctions and band energies for the TDDFT calculation. For this, the k-point grid has to be uniform, i.e. it cannot be a band structure along the high symmetry paths.

In general, one wants to make sure that the electronic band gap is correct, for a better comparison with experiment and to make sure that the underestimation of the band gap from DFT does not somehow affect the predicted absorption spectra apart from a rigid red shift. We will do this here by using a so-called "scissor correction", which consists of shifting the conduction bands such that the electronic band gap of the material equals that from experiment or that from a more accurate method.

Quantum Espresso should have reported the band gaps (or at least the energies of the highest occupied and lowest unoccupied band) directly in the output files of the bandstructure calculation, otherwise you can calculate them from the reported band energies in those output

[1] A whole collection of normconserving and ultrasoft pseudopotentials and of PAWs for Quantum Espresso can be found at https://www.quantum-espresso.org/pseudopotentials . However, YAMBO can only use normconserving pseudopotentials at the moment.

files. The band gap of Argon is direct and located at the Brillouin zone center. LDA predicts a large band gap of about 8 eV, but this is significantly smaller than the experimentally measured electronic band gap of Argon, which has a value of about 14.3 eV.

(b) Now we will perform the TDDFT calculation. First create a new folder "yambo", for example next to the folder where you calculated the electronic structure and copy the provided input files yambo.in_setup and yambo.in_tddft, as well as the job submission script j.submit.sh_input into it. In order to do the yambo calculations, we now need to convert the electronic structure to something yambo can work with. Go into the folder TMP_DIR/Ar.save written by the electronic structure calculation. Run the command

$PATH_TO_YAMBO/p2y -M

This will create a folder "SAVE" that contains a number of files with the calculated and converted wavefunctions and also information about the band energies. The command

$PATH_TO_YAMBO/p2y -M -H

shows you the different options that p2y accepts. "-M" and "-N" tell p2y to run in "serial mode", i.e. not to try and run on several CPUs (this does not work on the HPC, unless one uses a job submission script). P2Y also tests whether the supplied k-point grid is uniform. Careful here: if YAMBO thinks that the grid is not uniform, it will not abort. It will simply print a warning and proceed like normal, but then only using the Γ point for the calculation, which usually leads to bad results.

After the conversion finished, move the folder "SAVE" into the created "yambo" folder. As another preparatory step, yambo requires a quick 'setup' run. Run the command

$PATH_TO_YAMBO/yambo -M -F yambo.in_setup

This tells yambo to run a calculation using the input file yambo.in_setup (which, apart from some fancy header, only contains the line "setup"). Have a quick look at the output file r_setup created by the setup run. It lists the information that yambo extracted from converted Quantum Espresso data and is (mostly) self-explanatory. It can be quite useful to inspect this output file before the 'real' calculation to make sure that yambo did not 'misunderstand' the system in some way. For instance, one should have a look at the reported direct and indirect band gaps and check whether the values are as expected.

Now turn towards the second provided input file, yambo.in_tddft. This input file instructs yambo to calculate the optical properties by solving the Dyson equation of the polarizability within a TDDFT framework. You see that the input file contains a whole list of parameters with not exactly meaningful names. Fortunately, most of the input parameters are commented by default so that the user is able to understand the meaning.

They keyword `Chimod` allows the user to choose the level of electron-electron interaction and/or the kind of exchange-correlation kernel to be included in the calculation. "IP" corresponds to a calculation in the independent particle approximation, i.e. the Dyson equation is not solved, but χ is approximated by the non-interacting response function χ^0. "Hartree" tells yambo to set $f_{xc} = 0$ and to only use the Hartree kernel for the Dyson equation, i.e. do a calculation on the random phase approximation (RPA) level. "ALDA" tells yambo to use an exchange-correlation kernel in the adiabatic approximation, it here uses the exchange-correlation function that was used for the groundstate calculation. "PF" tells yambo to use a so-called "polarization functional" of the general form

$$f_{xc} \sim \frac{\alpha(\omega)}{q^2}$$

For example the bootstrap or JGM XC kernels. "BSfxc" is a state-of-the-art TDDFT method that should be very accurate for optical spectra but is also very computationally expensive (and the implementation apparently is not yet fully functional).

QpntsRXd defines a range of q-vectors that yambo will calculate the dielectric function and optical spectra for. Each of the q-vectors corresponds to one q-point (or k-point, they are the same grid for yambo) from the underlying electronic structure calculation. A range of 2 | 4 means that yambo will calculate the dielectric function for the second, third and fourth q-point/k-point in the list given in r_setup. Here, we are interested in the absorption spectra, i.e. we will only do calculations for the first q-point, the Γ point q=0 [2]. LongDrXd defines the polarization of the electric field perturbation that is used for the linear response approach. For inhomogeneous materials, particularly for 1D or 2D materials, the calculated spectra can depend significantly on this polarization direction and one should be a bit mindful that the polarization used in the calculation makes sense for the particular material, especially if one wants to compare with experiments.

XfnQP_E is used for an empirical correction of the electronic bandstructure: the first entry defines the scissor shift that is added to the conduction band energies. Add here the difference between the experimental band gap and the band gap from the LDA calculations. The second and third entry define a kind of bending value that can be used to stretch or quench the valence and conduction bands, i.e. allows on to somewhat change the dispersion of the bands in addition to the rigid scissor shift.

EnRngeXd, ETStpsXd, DmRngeXd define the output of the dielectric function: they tell yambo what energy window we are interested in (in the provided input file, we want yambo to write the dielectric function in the energy range 0-25 eV), how many points we want within this energy range, and the energy broadening of each excitation, respectively. The smaller DmRngeXd is, the pointier the individual peaks will be, but it will also require more k-point in the underlying electronic structure to not make the plots look to 'rugged', similar to DOS calculations.

BndsRnXd tells YAMBO, which bands from the available electronic structure it should use for the calculation. The density response function is calculated from a summation over occupied and unoccupied bands, i.e. the number of included bands is also a convergence parameter of sorts and should be chosen high enough that one includes all bands that contribute to the dielectric function in the energy window one is interested in. How large this number of bands is, depends then on the size of the energy window and the electronic structure of the material.

FxcGRLc and NGsBlkXd define cutoff energies for the planewave expansion of the polarizability, the Hartree kernel and the XC kernel, i.e. how many G and G' vectors are used to build the respective matrices. This corresponds to the inclusion of local field effects due to local interaction of light with the electron density in the calculation and can be very important for accurate results. We will come back to this point later and for now only consider the macroscopically averaged polarizability, i.e. we only use the single G-vector

[2] Quantum Espresso generates the k-points in a way that the Γ point is always the first one in the list. This, of course, does not have to be the case if the grid is not automatically generated by Quantum Espresso but supplied as a list by the user.

G=G'=0 (RL stands for 'reciprocal lattice vector).

Finally, the options marked with "PARALLEL" in the comments are, not surprisingly, aimed at controlling the parallelization of the calculation over many CPUs. In Quantum Espresso, the parallelization is automatically chosen [3], unless the user opts to manually control it. Unfortunately, YAMBO is less helpful in this respect and requires the user to explicitly state how many CPUs are used for which parallelization option. What options are available depends on the run level of the calculation. For TDDFT, we have 5 options: parallelization over g vectors (g), over q points in the Dyson equation (q), over k-points (for the summation over states in the calculation of the non-interaction response function, k), and over valence and conduction bands (also for the non-interacting response function, v and c). The options to be used are specified by the keyword `X_finite_q_ROLEs` for finite q and `X_q_0_ROLEs` specifically for the point q=0. The number of CPU groups for each of the options is specified by the keyword `X_finite_q_CPU.` Each of the nq q-point groups contains ng g vector groups. Each g vector group contains nk k-point groups and so on.
Altogether, the groups have to be chosen such that nq x ng x nk x nc x nv equals the number of CPUs used in the calculation. Typically, it appears to be best to focus on q-points and conduction band groups, and only then add valence band, g vector and k-point groups. In this calculation, however, we are only interested in the dielectric function for the point q=0, i.e. we cannot parallelize over q points. In general, the choice of the parallelization requires some testing to identify the options that lead to the fastest or least memory consuming calculations.

YAMBO allows to generate the input files automatically (even though the values for q-points of interest etc have to be entered manually). The above input file was generated with the command

yambo -o c -k alda -V par,qp

The possible options and runlevels can be displayed by running

yambo -H

Now run the calculations by submitting j.submit_yambo to the HPC. YAMBO will now write a number of output files documenting its progress: The file r_optics_chi _tddft has a similar purpose as r_setup and reports on the geometry, the used k- and q-points and the used computational parameters. In addition to that, YAMBO created on file per CPU in the folder LOG. The files contain the actual progress of the calculation for each CPU and provide a

[3] The main parallelization options in Quantum Espresso are parallelization over G-vectors (each CPU is allocated a share of the G-vectors in the system and does the calculation for this set of G vectors.) and parallelization over k-points (each CPU is allocated one or a few k-points and calculates the results for these k-points). The default is parallelization over G-vectors, which leads the smallest memory consumption and the best distribution of the computational load. It has, however, the disadvantage that it leads to more communication between the different CPUs, which can be slower than the actual calculation of a data package. This is especially the case if the calculation uses several server nodes. The parallelization over k-points has the advantage that for local exchange-correlations, the Kohn-Sham equations for different k-points are independent of each other, hence only a minimum of inter-CPU communication is required. It has the disadvantage, that each CPU knows needs to know all wavefunctions at the k-point, which can lead to a higher memory consumption, due to duplication of data. One can also combine both methods, for example using all CPUs of a node for the calculation of a subset of k-points and using G vector parallelization between the CPUs of a node, whole k-point parallelization is used between different nodes. This is usually beneficial, but requires some manual adjustments.

better insight into what the code is actually doing. Have a look at those output files while the calculation is running: YAMBO first calculates the transition dipole momenta (i.e. the optical matrix elements) for all transitions between occupied and unoccupied bands that were included in the calculation. The results will be used in the following step for the calculation of χ^0.

After χ^0 is calculated for all q-points of interest (here only q1=(0,0,0)), YAMBO solves the respective Dyson equations and write the calculated dielectric functions for each q-point into the files o.eps_q* and the EELS spectra into the files o.eel_q*, which can then be plotted and analysed. Each of the files contains five columns: the first column contains the energy axis. The second and third columns of the o.eps_* files contains the imaginary and the real parts of the frequency-dependent macroscopically averaged dielectric function ϵ_M. The frequency-dependent absorption coefficient can be calculated easily from these two columns, but in practice it is usually enough to analyse the peaks in the imaginary part of the dielectric function, which correspond to the excitation energies. The fourth and fifth column contain the dielectric function in the independent particle approximation, i.e. for the case $\chi = \chi^0$.

Plot the imaginary parts of ϵ_M both for ALDA and the IPA and compare the results with the experimental absorption spectra of Argon, for example the one found in Phys. Rev. Lett.37, 305 (1976). Also draw the experimental band gap of 14.3 eV into the plot and derive the exciton binding energies of real solid argon and the one from ALDA and IP by taking the energy difference between the band gap and the energy of the lowest peak maximum. ALDA and IP do not yield any noticeable exciton binding energy and the very prominent peak structure from experiment is completely missing.

(c) Now try a different fxc approximation, which should work better for excitons. We will use the relatively recently proposed RPA-Bootstrap method (RBO, Phys. Rev. Lett.114, 146402 (2015)) here, which is a simplification of the original bootstrap method and is implemented in YAMBO as a polarization functional. First, rename and store the ALDA results in a different folder. Then, modify the input file to use the RBO. Change `Chimod` from "ALDA" to "PF" and add another line with the keyword `PF_alpha="RBO"`.

Rerun the calculations and compare the results with the ones from (c). You should now have a prominent excitonic peak well below the electronic band gap, like in the experimental absorption spectrum. Calculate the new predicted exciton binding energy.

(d) So far, you did the calculations without local field effects, i.e. you only calculated and used $\chi(q = 0, G = 0, G' = 0)$. This speeds up the calculations a lot but neglects the effect that the components of χ for $G, G' \neq 0$ have on the results. For this reason, do a calculation including local field effects now. Store the results from (d) and open the input file again. All you have to do now is to modify `FxcGRLc` and `NGsBlkXd`, which control the number of G vectors included in the solution of the Dyson equation. You have two choices: you can manually specify the number of G vectors to be included (the file shells.dat lists all possible values in the second column), or you can define the number of G vectors through a cutoff energy. We will use the second option here. Replace "1 RL" for each for the keywords by "150 eV", which cause YAMBO to include all G vectors with a kinetic energy below 150 eV. Rerun the calculation and plot the results. You will see that the absorption spectrum was modified significantly through the inclusion of more G vectors, at the cost of a higher computational time. Recalculate the exciton binding energy and compare with the value from experiment.

Of course, a cutoff of 150 eV was just a random value and does not mean that the spectrum is converged yet. Rerun the calculation for a cutoff energy of 250 eV. Was the original cutoff energy of 150 eV already sufficient or not?

Exercise 2: Plasmon dispersion in Graphene

The topic of this exercise is calculating the plasmon dispersion of graphene.

(a) Use the provided input files and calculate the electronic structure of graphene on a 24x24x1 k-point grid and with 20 bands. There is another keyword in the input files: `force_symmorphic`. This tell QE to not use 'non-symmorphic' symmetry operations, which are, for example, screw axes, i.e. a rotation together with a translation. YAMBO cannot work with these and will complain if it finds that QE used non-symmorphic symmetry operations.

(b) Follow the steps from exercise 1 and the provided input files to prepare a TDDFT calculation for graphene based on the calculations from (a). Convert the Quantum Espresso data to a yambo-readable format, run a setup calculation and modify the input file. Include local field effects up to 50 eV, 20 bands and calculate the dielectric function for q-points 1-5. In this case, we parallelize over q-points, which speeds up the calculations a bit.

Apart from this, we have additional options in the input compared to exercise 1: Graphene is a two-dimensional material, which we model by a adding a vacuum layer in z-direction. Due to the long-range Coulomb interaction, the size of the vacuum layer can have a noticeable influence on the calculated optical properties. We apply a trick here, which is very useful for low-dimensional systems: instead of the full Coulomb interaction, we apply an auxilliary Coulomb interaction, which is truncated at the boundaries of the unit cell in z-direction, but acts like the coulomb potential in x- and y-direction. This is invoked by the run level `rim_cut` and is further specified with additional keywords. `CUTGeo= "box xy"` specifies that we want YAMBO to model our system in a box that is non-periodic in z direction and infinitely large in the x-y plane. The length of the box in z-direction is given in the `CUTBox` block, and chosen to be 28 bohr. It should be chosen to be equal to or slightly smaller than the lattice vector in z-direction. The box-truncated potential has a small problem: it features a singularity for q=0. YAMBO integrates this singularity using the random integration method (RIM), i.e. it samples a large number of random points q+G around the singularity and uses those to obtain the potential value at the singularity. The random sampling is defined through `RandQpts` and `RandGvec`, typically one needs order a million points for good results, `RandGvec` can be chosen small.

Run the calculations on the HPC (it takes a few hours to run them on a home computer). Before the calculation of χ^0, YAMBO now calculates the truncated potential and then each q-point group calculates its subset of the 5 q-vectors of interest. As you see when comparing the input files, the time required for the calculation of χ^0 can differ quite strongly on the q-point we are calculating χ^0 at. Use of symmetry operations makes the calculation of $\chi^0(q = 0)$ the fastest, while the required time increases for q-points that do not lie on high symmetry points or paths in the Brillouin zone. This is a reason why one should be careful with q-point parallelization, because it can lead to bad load balancing. A CPU that only does the q=0 or other high-symmetry q-points will be finished earlier than the other CPUs and then be idle until the slowest CPU is finished, thus effectively wasting computational resources.

In the calculated e.eel_* files, the second and the third column contain the imaginary and the real part of $1/\epsilon_M$, where the second column is the EEL spectrum we are looking for. Again, the fourth and fifth columns contain the quantities calculated with the non-interacting response function. Plot the EEL spectra for the four calculated 'finite-q' points. In principle, this would allow us to derive (a part of) the plasmon dispersion along the Gamma-M direction (i.e. between the center of the hexagonal Brillouin zone and the center of one of its faces) by extracting the peak energies from the EEL spectrum. However, this is easier said than done in this case, because the peaks look quite rugged and extracting the peak energies would carry a significant error. The problem is that the q-point grid we used was to not dense enough and hence the dielectric function was not converged with respect to the k-point grid used for the electronic structure. We could now do another calculation with a much denser grid and improve the spectrum in this way. However, the computational effort increases quadratically. In (c), will take a different route: the double grid method.

(c) In the double-grid method, the contribution of each k-point on the 24x24x1 k-point grid to the polarizability will be calculated through integrating over its 'mini-BZ', i.e. small volume of points around each k-point that is closer to this k-point than to any other k-point in the lattice. To do this integration, we will calculate the electronic structure on a much finer 96x96x1 k-point grid, YAMBO will then allocate each point of the finer grid to one point in the coarse grid and then do a discrete integration over the mini-BZ around each point.

Calculate the electronic structure of graphene on a 96x96x1 k-point grid. After this is done, go to the TMOP_DIR/C.save folder and run p2y. Here, we are only interested in the band energies, not in the wavefunctions. You can thus run p2y as

$PATH_TO_P2Y/p2y -M -w

This will create a folder "SAVE" with one file inside, which contains the band energies. Now we need to import these energies into YAMBO and tell it to do use the double grid procedure. This is done with the program ypp, which is YAMBO's postprocessing code. An input file is already provided, generated by the command ypp-M -N -m . Put this file into the folder where you run the YAMBO calculation from (b). Open the file and make sure that the path in DbGd_DB1_paths points to the TMP_DIR/C.save folder where you just ran p2y (for some reason, ypp appears to need the path to be specified twice). The path in ypp.in assumes that your yambo folder is within the folder where you ran the electronic structure calculation.

Run $PATH_TO_YPP/ypp -M

This will automatically use the input file ypp.in, read the band energies from the newly generated SAVE folder and write a file ndb.Double_Grid into your yambo/SAVE folder. This file contains the necessary data for the double grid calculation. All you now have to do is rerun the YAMBO calculation from (c), the double grid will be automatically used. Make sure to save the files generated in (c) somewhere else.

Plot the newly generated EEL spectra. They are much smoother now and one can see a noticeable q-point dependence. The 96x96x1 k-points in the Brillouin zone was still not enough to completely converge the EEL spectrum. Also, while this procedure helps with the convergence of the optical spectra, we still only have access to the momentum transfers from the underlying 24x24x1 k-point grid. For this reason, 'real' simulations of the plasmon dispersion, which usually focusses on small momentum transfers, would require much denser k-point grids. In this recent study on graphene, for instance, the authors used a 192x192x1 k-point grid, which goes beyond the scope of this exercise.

(d) Now we have smooth EEL spectra for different q-vectors, but different peaks. Which of those peaks are actually plasmons? While EEL spectra particularly show plasmons, the peaks might also arise from "band transitions", i.e. excitations of single electrons into the conduction band instead of collective excitations of the electron gas. We can distinguish the two cases by looking at the imaginary and real parts of the dielectric function. A classical criterion for the classification of an excitation to be classified as a plasmon is that $Re[\epsilon]$ crosses zero at the excitation energy. If also $Im[\epsilon]$ is small, this gives rise to a high peak in the EEL spectrum. On the other hand, a single-electron transition is characterized by a high peak in $Im[\epsilon]$ at the excitation energy. Have a look at the dielectric function for q-point q2 [4]. $Re[\epsilon]$ crosses 0 twice, at energies of about 5.1 and 5.4 eV, which is close to the peak energy of the first sharp peak in the EEL spectrum. At the same time, $Im[\epsilon]$ is small after the second crossing. This suggests that the peak is indeed a plasmon, it is commonly known as π plasmon, because it arises from (collective) excitations between the π valence and the π* conduction band of graphene. Note that the high peak in $Im[\epsilon]$ around the first crossing suggests that a single-particle transition should somewhat contribute to the peak as well. The situation is far more complex for the broad peak in the EEL spectrum as $Re[\epsilon]$ does not cross 0 here, but only approaches it. Typically, the feature is attributed to a π+σ plasmon

[4] Graphene is a 2D material and the macroscopic dielectric function is ill defined in this case, one should actually use a 2D averaged dielectric function, see here. Still, the 3D dielectric function appears to work reasonably well for the purpose of the exercise.